XPS-Q8

Universal High-Performance Motion Controller/Driver

ESP COMPATIBLE

RoHS Compliant

Newport®

Experience | Solutions

Programmer's Manual

V1.4.x

Precision Motion – *Guaranteed*™

Preface

Confidentiality & Proprietary Rights

Reservation of Title

The Newport Programs and all materials furnished or produced in connection with them ("Related Materials") contain trade secrets of Newport and are for use only in the manner expressly permitted. Newport claims and reserves all rights and benefits afforded under law in the Programs provided by Newport Corporation.

Newport shall retain full ownership of Intellectual Property Rights in and to all development, process, align or assembly technologies developed and other derivative work that may be developed by Newport. Customer shall not challenge, or cause any third party to challenge, the rights of Newport.

Preservation of Secrecy and Confidentiality and Restrictions to Access

Customer shall protect the Newport Programs and Related Materials as trade secrets of Newport, and shall devote its best efforts to ensure that all its personnel protect the Newport Programs as trade secrets of Newport Corporation. Customer shall not at any time disclose Newport's trade secrets to any other person, firm, organization, or employee that does not need (consistent with Customer's right of use hereunder) to obtain access to the Newport Programs and Related Materials. These restrictions shall not apply to information (1) generally known to the public or obtainable from public sources; (2) readily apparent from the keyboard operations, visual display, or output reports of the Programs; (3) previously in the possession of Customer or subsequently developed or acquired without reliance on the Newport Programs; or (4) approved by Newport for release without restriction.

1791 Deere Ave.
Irvine, CA 92606, USA
(949) 863-3144

Table of Contents

XPS
Universal High-Performance
Motion Controller/Driver

1.0 TCP/IP Communication

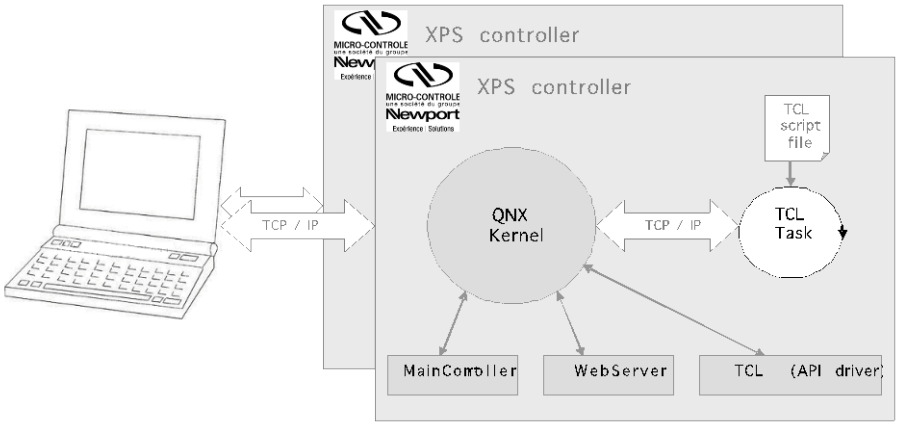
1.1 Description

XPS is based on a 10/100 Base-T Ethernet communication link with TCP/IP protocol and uses a web site approach for all software tools and an FTP server for file transfer. This makes the XPS controller independent of the user's operating system. When networked, Unix, Linux or Windows users can access the same controller from any place in the world for remote control, code development, file transfer or diagnostics. The completely object oriented approach of the XPS firmware with powerful, multi-parameter Function's (commands) is also much more consistent and intuitive to use than old-style mnemonic commands.

To connect to the XPS controller you must open a socket with the "OpenConnection()" Function. Communication through this socket is done by specifying the socket identifier (socketID) with the Function. The XPS allows up to 100 open sockets with 30 communicating simultaneously.

Each Function returns a completion or error message. In case of a successful completion, the return is 0 (zero). In case of an error, the returned error code can be used for diagnosing the problem with the Function ErrorStringGet().

The function call is blocked until a reply is sent by the XPS, or until the timeout value is reached. For running several processes in parallel (for instance getting the position while a stage is moving), several sockets can be used in parallel. When using the XPS controller with programming languages that do not support multiple sockets, the timeout value of the function can be set to a low value (20 ms).



1.2 Function Description

1.2.1 CloseAllOtherSockets

Name

CloseAllOtherSockets – Closes all sockets beside the one used.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check "Administrator" rights: ERR_NEED_ADMINISTRATOR_RIGHTS (-9)

Description

This function allows an administrator to close all sockets except the socket used to call this function.

All used sockets are closed. So, ERR_SOCKET_CLOSED_BY_ADMIN error is sent to each previously running function to close the used socket.

NOTE

Call the "Login" function to identify the user as "Administrator".

CAUTION

If some TCL scripts are in progress (after a "TCLScriptExecute" function or a "TCLScriptExecuteAndWait" function), do not use CloseAllOtherSockets to function to kill these TCL scripts. Use the "TCLScriptKill" function to stop the TCL execution and only after can you use the "CloseAllOtherSockets" function to close sockets.

Return

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_NEED_ADMINISTRATOR_RIGHTS (-107)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): No error

TCL

Prototype

CloseAllOtherSockets

Input parameters

– None

Output parameters

– None

Return

– Error	int	Function error code

C/C++

Prototype

int **CloseAllOtherSockets** ()

Input parameters

– None

Output parameters

– None

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **CloseAllOtherSockets** ()

Input parameters

– None

Output parameters

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **CloseAllOtherSockets** ()

Input parameters

– None

Return

– Error	int32	Function error code

Python

Prototype

[Error] **CloseAllOtherSockets** ()

Input parameters

– None

Return

– Error	int	Function error code

1.2.2 Login

Name

Login – Self-identification.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Check command format: ERR_WRONG_FORMAT (-7)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the user name and the password: ERR_WRONG_USERNAME_OR_PASSWORD (-106)

Description

This function allows a user to identify himself as "SuperUser", "Administrator" or "User".

The user account must be exited else ERR_WRONG_USERNAME_OR_PASSWORD (-106) is returned.

This function is not meant to be used from "terminal" web page.

NOTE

To add a new user account, you must use the XPS web site with "Administrator" rights. In the main menu, select "CONTROLLER CONFIGURATION" and go to the "Users management" page.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_USERNAME_OR_PASSWORD (-106)
- SUCCESS (0): no error

TCL

Prototype

Login SocketID UserName Password

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– UserName	string	User name
– Password	string	Password

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **Login** (int SocketID, char *UserName, char *Password)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– UserName	char *	User name
– Password	char *	Password

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **Login** (ByVal SocketID As Long, ByVal UserName As String, ByVal Password As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– UserName	string	User name
– Password	string	Password

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **Login** (int32 SocketID, cstring UserName, cstring Password)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– UserName	cstring	User name
– Password	cstring	Password

Return

– Function error code

Python

Prototype

integer **Login** (integer SocketID, string UserName, string Password)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– UserName	string	User name
– Password	string	Password

Return

– Function error code

1.2.3 OpenConnection

Name

OpenConnection – opens a socket to connect to the TCP server (local).

Input tests

- Check number of used sockets (Max = 100): if no free socket then the SocketID is set to -1

Description

This function opens a socket in a TCL script located in the "Scripts" directory of the XPS controller.

The TCP/IP communication is configured as:

Local Host Address = 127.0.0.1

IP Port = 5001

This function returns a socket identifier to use for each function call. The socket identifier is defined between 0 to 99. If the TCP/IP connection fails then the "SocketID" value is –1.

Return

- Socket identifier used in each function

TCL

Prototype

OpenConnection TimeOut SocketID

Input parameters

– TimeOut	floating point	Timeout in seconds used for each Function execution

Output parameters

– SocketID	int	Socket identifier used in each function

Return

– Error code

1.2.4 TCP_CloseSocket

Name

TCP_CloseSocket – Closes a socket.

Input tests

- Check socket identifier (Max = 100).
- Socket must be used.

Description

Closed the opened TCP/IP communication defined by the given socket identifier. If the socket is undefined or is not used, then nothing happens.

Return

– None

TCL

Prototype

TCP_CloseSocket SocketID

Input parameters

– SocketID	int	Socket identifier used in each function

Output parameters

– None

C/C++

Prototype

void **TCP_CloseSocket** (int SocketID)

Input parameters

– SocketID	int	Socket identifier used in each function

Output parameters

– None

Visual Basic

Prototype

TCP_CloseSocket (ByVal SocketID As Long)

Input parameters

– SocketID	long	Socket identifier used in each function

Output parameters

– None

Matlab

Prototype

TCP_CloseSocket (int32 SocketID)

Input parameters

–	SocketID	int32	Socket identifier used in each function

Python

Prototype

TCP_CloseSocket (integer SocketID)

Input parameters

–	SocketID	int	Socket identifier used in each function

1.2.5 TCP_ConnectToServer

Name

TCP_ConnectToServer – Configures TCP/IP communication and opens a socket.

Input tests

- Check number of used sockets (Max = 100): if no free socket then the SocketID is set to -1

Description

Configures the TCP/IP communication and opens a socket to connect TCP server.

This function returns a socket identifier to use for each function call. The socket identifier is defined between 0 to 99. If the TCP/IP connection failed then the "SocketID" value is –1.

NOTE

OpenConnection function is used when users are in local mode, it only needs the timeout and socket number to open the connection with the XPS controller. TCP_ConnectToServer function needs more information like the port number and the IP address. This function is called with the DLL.

Return

- Socket identifier used in each function

TCL

Prototype

TCP_ConnectToServer IP_Address IP_Port TimeOut SocketID

Input parameters

–	IP_Address	string	TCP IP address: 195.168.33.xxx or another
–	IP_Port	interger	TCP IP port: 5001 for XPS controller
–	TimeOut	floating point	Timeout in seconds used for each Function execution

Output parameters

– None

Return

–	SocketID	int	Socket identifier used in each function

C/C++

Prototype

int **TCP_ConnectToServer** (char * IP_Address, int IP_Port, double TimeOut)

Input parameters

– IP_Address	char *	TCP IP address: 195.168.33.xxx or another
– IP_Port	int	TCP IP port: 5001 for XPS controller
– TimeOut	double	Timeout in seconds used for each Function execution

Output parameters

– None

Return

– SocketID	int	Socket identifier used in each function

Visual Basic

Prototype

Long **TCP_ConnectToServer** (ByVal IP_Address As String, ByVal IP_Port As Long, ByVal TimeOut As Double)

Input parameters

– IP_AddressString	TCP IP address: 195.168.33.xxx or another	
– IP_Port	long	TCP IP port: 5001 for XPS controller
– TimeOut	double	Timeout in seconds used for each Function execution

Output parameters

– None

Return

– SocketID	long	Socket identifier used in each function

Matlab

Prototype

int32 **TCP_ConnectToServer** (cstring IP_Address, int32 IP_Port, double TimeOut)

Input parameters

– IP_Address	cstring	TCP IP address: 195.168.33.xxx or another
– IP_Port	int32	TCP IP port: 5001 for XPS controller
– TimeOut	double	Timeout in seconds used for each Function execution

Return

– SocketID	int32	Socket identifier used in each function

Python

Prototype

integer **TCP_ConnectToServer** (string IP_Address, integer IP_Port, double TimeOut)

Input parameters

– IP_Address	string	TCP IP address: 195.168.33.xxx or another
– IP_Port	int	TCP IP port: 5001 for XPS controller
– TimeOut	double	Timeout in seconds used for each Function execution

Return

– SocketID	int	Socket identifier used in each function

1.2.6 TCP_GetError

Name

TCP_GetError – Gets the last error about socket.

Input tests

- Check socket identifier (Max = 100).
- Socket must be used.

Description

Gets the last error from the socket defined by the given socket identifier. If the socket is undefined or is not used, the error description is blank.

Return

- None

TCL

Prototype

TCP_GetError SocketID

Input parameters

– SocketID	int	Socket identifier used in each function

Output parameters

– None

Return

– Last error description.

C/C++

Prototype

void **TCP_GetError** (int SocketID, char * ErrorString)

Input parameters

– SocketID	int	Socket identifier used in each function

Output parameters

– ErrorString	char *	Last error description

Visual Basic

Prototype

– **TCP_GetError** (ByVal SocketID As Long, ErrorString As String)

Input parameters

– SocketID	long	Socket identifier used in each function

Output parameters

– ErrorString	string	Last error description

Matlab

Prototype

[ErrorString] **TCP_GetError** (int32 SocketID)

Input parameters

–	SocketID	int32	Socket identifier used in each function

Return

–	ErrorString	cstring	Last error description

Python

Prototype

[ErrorString] **TCP_GetError** (integer SocketID)

Input parameters

–	SocketID	int	Socket identifier used in each function

Return

–	ErrorString	string	Last error description

1.2.7 TCP_SetTimeout

Name

TCP_SetTimeout – Configures the timeout for TCP/IP communication.

Input tests

- Check number of used sockets (Maximum number = 100).
- Socket must be used.
- Timeout value must be positive.

Description

Sets a new timeout value in seconds for the opened TCP/IP communication defined by a socket identifier.

If the timeout is less than 0.001, the timeout value defaults to 0.001.

If the socket is undefined or is not used then nothing happens.

Return

- None

TCL

Prototype

TCP_SetTimeout SocketID TimeOut

Input parameters

– SocketID	int	Socket identifier used in each function
– TimeOut	floating point	Timeout in seconds used for each Function execution

Output parameters

– None

Return

– TCL error (0 = success or 1 = syntax error)

C/C++

Prototype

void **TCP_SetTimeout** (int SocketID, double TimeOut)

Input parameters

– SocketID	int	Socket identifier used in each function
– TimeOut	double	Timeout in seconds used for each Function execution

Output parameters

– None

Return

– None

Visual Basic

Prototype

TCP_SetTimeout (ByVal SocketID As Long, ByVal TimeOut As Double)

Input parameters

– SocketID	long	Socket identifier used in each function
– TimeOut	double	Timeout in seconds used for each Function execution

Output parameters

– None

Return

– None

Matlab

Prototype

int32 **TCP_SetTimeout** (int32 SocketID, double TimeOut)

Input parameters

– SocketID	int32	Socket identifier used in each function
– TimeOut	double	Timeout in seconds used for each Function execution

Return

– None

Python

Prototype

integer **TCP_SetTimeout** (integer SocketID, double TimeOut)

Input parameters

– SocketID	int	Socket identifier used in each function
– TimeOut	double	Timeout in seconds used for each Function execution

Return

– None

2.0 XPS Features

2.1 General Features

2.1.1 ControllerMotionKernelMinMaxTimeLoadGet

Name

ControllerMotionKernelMinMaxTimeLoadGet – Returns controller motion kernel minimum and maximum time load.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function gets the minimum and the maximum value of controller motion kernel time load. It returns:

- Minimum Total Load Ratio
- Maximum Total Load Ratio
- Minimum Corrector Load Ratio
- Maximum Corrector Load Ratio
- Minimum Profiler Load Ratio
- Maximum Profiler Load Ratio
- Minimum servitudes Load Ratio
- Maximum servitudes Load Ratio

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

ControllerMotionKernelMinMaxTimeLoadGet SocketID
MinimumCPUTotalLoadRatio MaximumCPUTotalLoadRatio
MinimumCPUCorrectorLoadRatio MaximumCPUCorrectorLoadRatio
MinimumCPUProfilerLoadRatio MaximumCPUProfilerLoadRatio
MinimumCPUServitudesLoadRatio MaximumCPUServitudesLoadRatio

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– MinimumCPUTotalLoadRatio	double	Minimum controller motion kernel total CPU time load
– MaximumCPUTotalLoadRatio	double	Maximum controller motion kernel total CPU time load
– MinimumCPUCorrectorLoadRatio	double	Minimum controller motion kernel corrector CPU time load
– MaximumCPUCorrectorLoadRatio	double	Maximum controller motion kernel corrector CPU time load
– MinimumCPUProfilerLoadRatio	double	Minimum controller motion kernel profiler CPU time load
– MaximumCPUProfilerLoadRatio	double	Maximum controller motion kernel profiler CPU time load
– MinimumCPUServitudesLoadRatio	double	Minimum controller motion kernel servitudes CPU time load
– MaximumCPUServitudesLoadRatio	double	Maximum ontroller motion kernel servitudes CPU time load

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **ControllerMotionKernelMinMaxTimeLoadGet** (int SocketID, double * MinimumCPUTotalLoadRatio, double * MaximumCPUTotalLoadRatio, double * MinimumCPUCorrectorLoadRatio, double * MaximumCPUCorrectorLoadRatio, double * MinimumCPUProfilerLoadRatio, double * MaximumCPUProfilerLoadRatio, double * MinimumCPUServitudesLoadRatio, double * MaximumCPUServitudesLoadRatio)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function

Output parameters

– MinimumCPUTotalLoadRatio	double *	Minimum controller motion kernel total CPU time load
– MaximumCPUTotalLoadRatio	double *	Maximum controller motion kernel total CPU time load
– MinimumCPUCorrectorLoadRatio	double *	Minimum controller motion kernel corrector CPU time load
– MaximumCPUCorrectorLoadRatio	double *	Maximum controller motion kernel corrector CPU time load
– MinimumCPUProfilerLoadRatio	double *	Minimum controller motion kernel profiler CPU time load
– MaximumCPUProfilerLoadRatio	double *	Maximum controller motion kernel profiler CPU time load
– MinimumCPUServitudesLoadRatio	double *	Minimum controller motion kernel servitudes CPU time load
– MaximumCPUServitudesLoadRatio	double *	Maximum controller motion kernel servitudes CPU time load

Return

– Function error code

Visual Basic

Prototype

Long **ControllerMotionKernelMinMaxTimeLoadGet** (ByVal SocketID As Long, MinimumCPUTotalLoadRatio As Double, MaximumCPUTotalLoadRatio As Double, MinimumCPUCorrectorLoadRatio As Double, MaximumCPUCorrectorLoadRatio As Double, MinimumCPUProfilerLoadRatio As Double, MaximumCPUProfilerLoadRatio As Double, MinimumCPUServitudesLoadRatio As Double, MaximumCPUServitudesLoadRatio As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– MinimumCPUTotalLoadRatio	double	Minimum controller motion kernel total CPU time load
– MaximumCPUTotalLoadRatio	double	Maximum controller motion kernel total CPU time load
– MinimumCPUCorrectorLoadRatio	double	Minimum controller motion kernel corrector CPU time load
– MaximumCPUCorrectorLoadRatio	double	Maximum controller motion kernel corrector CPU time load
– MinimumCPUProfilerLoadRatio	double	Minimum controller motion kernel profiler CPU time load
– MaximumCPUProfilerLoadRatio	double	Maximum controller motion kernel profiler CPU time load
– MinimumCPUServitudesLoadRatio	double	Minimum controller motion kernel servitudes CPU time load
– MaximumCPUServitudesLoadRatio	double	Maximum controller motion kernel servitudes CPU time load

Return

– Function error code

Matlab

Prototype

[Error, MinimumCPUTotalLoadRatio, MaximumCPUTotalLoadRatio, MinimumCPUCorrectorLoadRatio, MaximumCPUCorrectorLoadRatio, MinimumCPUProfilerLoadRatio, MaximumCPUProfilerLoadRatio, MinimumCPUServitudesLoadRatio, MaximumCPUServitudesLoadRatio] **ControllerMotionKernelMinMaxTimeLoadGet** (int32 SocketID)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function

Return

– Error	int32	Function error code
– MinimumCPUTotalLoadRatio	double	Minimum controller motion kernel total CPU time load
– MaximumCPUTotalLoadRatio	double	Maximum controller motion kernel total CPU time load
– MinimumCPUCorrectorLoadRatio	double	Minimum controller motion kernel corrector CPU time load
– MaximumCPUCorrectorLoadRatio	double	Maximum controller motion kernel corrector CPU time load
– MinimumCPUProfilerLoadRatio	double	Minimum controller motion kernel profiler CPU time load
– MaximumCPUProfilerLoadRatio	double	Maximum controller motion kernel profiler CPU time load
– MinimumCPUServitudesLoadRatio	double	Minimum controller motion kernel servitudes CPU time load
– MaximumCPUServitudesLoadRatio	double	Maximum controller motion kernel servitudes CPU time load

Python

Prototype

[Error, MinimumCPUTotalLoadRatio, MaximumCPUTotalLoadRatio, MinimumCPUCorrectorLoadRatio, MaximumCPUCorrectorLoadRatio, MinimumCPUProfilerLoadRatio, MaximumCPUProfilerLoadRatio, MinimumCPUServitudesLoadRatio, MaximumCPUServitudesLoadRatio]
ControllerMotionKernelMinMaxTimeLoadGet (integer SocketID)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function

Return

– Error	int	Function error code
– MinimumCPUTotalLoadRatio	double	Minimum controller motion kernel total CPU time load
– MaximumCPUTotalLoadRatio	double	Maximum controller motion kernel total CPU time load
– MinimumCPUCorrectorLoadRatio	double	Minimum controller motion kernel corrector CPU time load
– MaximumCPUCorrectorLoadRatio	double	Maximum controller motion kernel corrector CPU time load
– MinimumCPUProfilerLoadRatio	double	Minimum controller motion kernel profiler CPU time load
– MaximumCPUProfilerLoadRatio	double	Maximum controller motion kernel profiler CPU time load
– MinimumCPUServitudesLoadRatio	double	Minimum controller motion kernel servitudes CPU time load
– MaximumCPUServitudesLoadRatio	double	Maximum controller motion kernel servitudes CPU time load

2.1.2 ControllerMotionKernelMinMaxTimeLoadReset

Name

ControllerMotionKernelMinMaxTimeLoadReset – Resets controller motion kernel minimum and maximum time load.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)

Description

This function Resets controller motion kernel minimum and maximum time load.

Errors

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- SUCCESS (0): no error

TCL

Prototype

ControllerMotionKernelMinMaxTimeLoadReset $SocketID

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **ControllerMotionKernelMinMaxTimeLoadReset** (int SocketID)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **ControllerMotionKernelMinMaxTimeLoadReset** (ByVal SocketID As Long)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– None

Return

– Function error code

Matlab

Prototype

[Error] **ControllerMotionKernelMinMaxTimeLoadReset** (int32 SocketID)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function

Return

–	Error	int32	Function error code

Python

Prototype

[Error] **ControllerMotionKernelMinMaxTimeLoadReset** (integer SocketID)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Return

Error	int	Function error code	

2.1.3 ControllerMotionKernelTimeLoadGet

Name

ControllerMotionKernelTimeLoadGet – Returns controller motion kernel time load.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)
- Valid group name: ERR_GROUP_NAME (-19)

Description

This function gets the last exact value of controller motion kernel time load (total , corrector, profier and servitudes calculation time).

CorrectorTimeLoad = CorrectorCalculationTime/CorrectorISRPeriod

ProfilerTimeLoad = ProfilerCalculationTime/CorrectorISRPeriod /ProfileGeneratorISRRatio

ServitudesTimeLoad = ServitudesCalculationTime/CorrectorISRPeriod /ServitudesISRRatio

TotalTimeLoad = CorrectorTimeLoad + ProfilerTimeLoad + ServitudesTimeLoad

---

NOTE

Refer to system.ref file to get CorrectorISRPeriod, ProfileGeneratorISRRatio and ServitudesISRRatio.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

ControllerMotionKernelTimeLoadGet SocketID CPUTotalLoadRatio CPUCorrectorLoadRatio CPUProfilerLoadRatio CPUServitudesLoadRatio

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

–	CPUTotalLoadRatio	double	Controller motion kernel total CPU time load
–	CPUCorrectorLoadRatio	double	Controller motion kernel corrector CPU time load
–	CPUProfilerLoadRatio	double	Controller motion kernel profiler CPU time load
–	CPUServitudesLoadRatio	double	Controller motion kernel servitudes CPU time load

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **ControllerMotionKernelTimeLoadGet** (int SocketID, double * CPUTotalLoadRatio, double * CPUCorrectorLoadRatio, double * CPUProfilerLoadRatio, double * CPUServitudesLoadRatio)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function

Output parameters

–	CPUTotalLoadRatio	double *	Controller motion kernel total CPU time load
–	CPUCorrectorLoadRatio	double *	Controller motion kernel corrector CPU time load
–	CPUProfilerLoadRatio	double *	Controller motion kernel profiler CPU time load
–	CPUServitudesLoadRatio	double *	Controller motion kernel servitudes CPU time load

Return

– Function error code

Visual Basic

Prototype

Long **ControllerMotionKernelTimeLoadGet** (ByVal SocketID As Long, CPUTotalLoadRatio As Double, CPUCorrectorLoadRatio As Double, CPUProfilerLoadRatio As Double, CPUServitudesLoadRatio As Double)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

–	CPUTotalLoadRatio	double	Controller motion kernel total CPU time load
–	CPUCorrectorLoadRatio	double	Controller motion kernel corrector CPU time load
–	CPUProfilerLoadRatio	double	Controller motion kernel profiler CPU time load
–	CPUServitudesLoadRatio	double	Controller motion kernel servitudes CPU time load

Return

–		Function error code

Matlab

Prototype

[Error, CPUTotalLoadRatio, CPUCorrectorLoadRatio, CPUProfilerLoadRatio, CPUServitudesLoadRatio] **ControllerMotionKernelTimeLoadGet** (int32 SocketID)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function

Return

–	Error	int32	Function error code
–	CPUTotalLoadRatio	double	Controller motion kernel total CPU time load
–	CPUCorrectorLoadRatio	double	Controller motion kernel corrector CPU time load
–	CPUProfilerLoadRatio	double	Controller motion kernel profiler CPU time load
–	CPUServitudesLoadRatio	double	Controller motion kernel servitudes CPU time load

Python

Prototype

[Error, CPUTotalLoadRatio, CPUCorrectorLoadRatio, CPUProfilerLoadRatio, CPUServitudesLoadRatio] **ControllerMotionKernelTimeLoadGet** (integer SocketID)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function

Return

–	Error	int	Function error code
–	CPUTotalLoadRatio	double	Controller motion kernel total CPU time load
–	CPUCorrectorLoadRatio	double	Controller motion kernel corrector CPU time load
–	CPUProfilerLoadRatio	double	Controller motion kernel profiler CPU time load
–	CPUServitudesLoadRatio	double	Controller motion kernel servitudes CPU time load

2.1.4 ControllerRTTimeGet

Name

ControllerRTTimeGet – Returns the controller corrector period and corrector calculation time.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)
- Valid group name: ERR_GROUP_NAME (-19)

Description

This function gets the last exact value of controller corrector period and of corrector calculation time.

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

The theoretical value of XPS controller corrector period is 0.125 ms (corresponding to 8 kHz controller corrector frequency).

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

ControllerRTTimeGet SocketID CurrentRTPeriod CurrentRTUsage

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– CurrentRTPeriod	double	Controller corrector period (seconds)
– CurrentRTUsage	double	Controller corrector calculation time (seconds)

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **ControllerRTTimeGet** (int SocketID, double * CurrentRTPeriod, double * CurrentRTUsage)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function

Output parameters

– CurrentRTPeriod	double *	Controller corrector period (seconds)
– CurrentRTUsage	double *	Controller corrector calculation time (seconds)

Return

– Function error code

Visual Basic

Prototype

Long **ControllerRTTimeGet** (ByVal SocketID As Long, CurrentRTPeriod As Double, CurrentRTUsage As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– CurrentRTPeriod	double	Controller corrector period (seconds)
– CurrentRTUsage	double	Controller corrector calculation time (seconds)

Return

– Function error code

Matlab

Prototype

[Error, CurrentRTPeriod, CurrentRTUsage] **ControllerRTTimeGet** (int32 SocketID)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function

Return

– Error	int32	Function error code
– CurrentRTPeriod	double	Controller corrector period (seconds)
– CurrentRTUsage	double	Controller corrector calculation time (seconds)

Python

Prototype

[Error, CurrentRTPeriod, CurrentRTUsage] **ControllerRTTimeGet** (integer SocketID)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function

Return

– Error	int	Function error code
– CurrentRTPeriod	double	Controller corrector period (seconds)
– CurrentRTUsage	double	Controller corrector calculation time (seconds)

2.1.5 ControllerSlaveStatusGet

Name

ControllerSlaveStatusGet – Returns the slave controller status code.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid output parameter type: ERR_WRONG_TYPE_INT (-15)
- Valid group name: ERR_GROUP_NAME (-19)

Description

This function, called from the master controller, gets status of its slave controller(s) . The slave controller status codes can take the following values:

0: All slave controllers are externally synchronized with the master controller.

1: The synchronization cable is not connected to any slave controller.

2: At least one slave controller is not externally synchronized with the master controller.

On the master controller, the description of the slave controller status code can be obtained with the "ControllerSlaveStatusStringGet" function.

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

ControllerSlaveStatusGet SocketID SlaveControllerStatus

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

–	SlaveControllerStatus	interger	Slave status of the controller.

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **ControllerSlaveStatusGet** (int SocketID, int * SlaveControllerStatus)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function

Output parameters

–	SlaveControllerStatus	int *	Slave status of the controller

Return

– Function error code

Visual Basic

Prototype

Long **ControllerSlaveStatusGet** (ByVal SocketID As Long, SlaveControllerStatus As Long)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

–	SlaveControllerStatus	long	Slave status of the controller

Return

– Function error code

Matlab

Prototype

[Error, SlaveControllerStatus] **ControllerSlaveStatusGet** (int32 SocketID)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function

Return

– Error	int32	Function error code
– SlaveControllerStatus	int32	Slave status of the controller

Python

Prototype

[Error, SlaveControllerStatus] **ControllerSlaveStatusGet** (integer SocketID)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function

Return

– Error	int	Function error code
– SlaveControllerStatus	int	Slave status of the controller

2.1.6 ControllerSlaveStatusStringGet

Name

ControllerSlaveStatusStringGet – Gets the slave controller status description from a slave controller status code.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check input parameter type: ERR_WRONG_TYPE_INT (-15)
- Check input parameter value: ERR_PARAMETER_OUT_OF_RANGE (-17)
- Check output parameter type: ERR_WRONG_TYPE_CHAR (-13)

Description

This function returns the slave controller status description corresponding to a slave controller status code.

If the slave status code is not referenced then the function returns ERR_PARAMETER_OUT_OF_RANGE (-17) error.

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

Error codes

- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13)
- ERR_WRONG_TYPE_INT (-15)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- SUCCESS (0): no error

TCL

Prototype

ControllerSlaveStatusStringGet $SocketID $SlaveControllerStatus SlaveControllerStatusString

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– SlaveControllerStatus	int	Slave controller status code

Output parameters

– SlaveControllerStatusString	string	Slave controller status description

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **ControllerSlaveStatusStringGet** (int SocketID, int SlaveControllerStatus, char * SlaveControllerStatusString)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– SlaveControllerStatus	int	Slave controller status code

Output parameters

– SlaveControllerStatusString	char *	Slave controller status description

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

Visual Basic

Prototype

Long **ControllerSlaveStatusStringGet** (ByVal SocketID As Long, SlaveControllerStatus As Integer, ByVal SlaveControllerStatusString As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– SlaveControllerStatus	int	Slave controller status code

Output parameters

– SlaveControllerStatusString	string	Slave controller status description

Return

– Error	long	TCL error code (0 = success or 1 = syntax error) or function error code

Matlab

Prototype

[Error, SlaveControllerStatusString] **ControllerSlaveStatusStringGet** (int32 SocketID, int32 SlaveControllerStatus)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– SlaveControllerStatus	int32	Slave controller status code

Return

– Error	int32	Function error code
– SlaveControllerStatusString	cstring	Slave controller status description

Python

Prototype

[Error, SlaveControllerStatusString] **ControllerSlaveStatusStringGet** (integer SocketID, integer SlaveControllerStatus)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	SlaveControllerStatus	int	Slave controller status code

Return

–	Error	int	Function error code
–	SlaveControllerStatusString	string	Slave controller status description

2.1.7 ControllerStatusGet

Name

ControllerStatusGet – Returns the controller status code.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid output parameter type: ERR_WRONG_TYPE_INT (-15)
- Valid group name: ERR_GROUP_NAME (-19)

Description

Returns the controller status code. The controller status codes are listed in the "Controller status list" § 03.

The description of the controller status code can be obtained with the "ControllerStatusStringGet" function.

The controller status flag is automatically reset after a controller status reading using the *ControllerStatusGet()* command.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

ControllerStatusGet SocketID ControllerStatus

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

–	ControllerStatus	interger	Status of the controller.

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **ControllerStatusGet** (int SocketID, int * ControllerStatus)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function

Output parameters

– ControllerStatus	int *	Status of the controller

Return

– Function error code

Visual Basic

Prototype

Long **ControllerStatusGet** (ByVal SocketID As Long, ControllerStatus As Long)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– ControllerStatus	long	Status of the controller

Return

– Function error code

Matlab

Prototype

[Error, ControllerStatus] **ControllerStatusGet** (int32 SocketID)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function

Return

– Error	int32	Function error code
– ControllerStatus	int32	Status of the controller

Python

Prototype

[Error, ControllerStatus] **ControllerStatusGet** (integer SocketID)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function

Return

– Error	int	Function error code
– ControllerStatus	int	Status of the controller

2.1.8 ControllerStatusRead

Name

ControllerStatusRead – Reads the controller status code.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid output parameter type: ERR_WRONG_TYPE_INT (-15)
- Valid group name: ERR_GROUP_NAME (-19)

Description

Returns the controller status code. The controller status codes are listed in the "Controller status list" § 03.

The description of the controller status code can be obtained with the "ControllerStatusStringGet" function.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

ControllerStatusRead SocketID ControllerStatus

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

–	ControllerStatus	int	Status of the controller.

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **ControllerStatusRead** (int SocketID, int *ControllerStatus)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function

Output parameters

–	ControllerStatus	int *	Status of the controller

Return

– Function error code

Visual Basic

Prototype

Long **ControllerStatusRead** (ByVal SocketID As Long, ControllerStatus As Long)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

–	ControllerStatus	long	Status of the controller

Return

– Function error code

Matlab

Prototype

[Error, ControllerStatus] **ControllerStatusRead** (int32 SocketID)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function

Return

–	Error	int32	Function error code
–	ControllerStatus	int32	Status of the controller

Python

Prototype

[Error, ControllerStatus] **ControllerStatusRead** (integer SocketID)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function

Return

– Error	int	Function error code
– ControllerStatus	int	Status of the controller

2.1.9 ControllerStatusStringGet

Name

ControllerStatusStringGet – Get the controller status description from a controller status code.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check input parameter type: ERR_WRONG_TYPE_INT (-15)
- Check output parameter type: ERR_WRONG_TYPE_CHAR (-13)

Description

This function returns the controller status description corresponding to a controller status code (see §0 controller status list).

If the status code is not referenced then the "Unknown controller status code" message will be returned.

Error codes

- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

ControllerStatusStringGet $SocketID $ControllerStatusCode ControllerStatusString

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– ControllerStatusCode	int	Controller status code

Output parameters

– ControllerStatusString	string	Controller status description

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **ControllerStatusStringGet** (int SocketID, int ControllerStatusCode, char * ControllerStatusString)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– ControllerStatusCode	int	Controller status code

Output parameters

– ControllerStatusString	char *	Controller status description

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **ControllerStatusStringGet** (ByVal SocketID As Long, ControllerStatusCode As Integer, ByVal ControllerStatusString As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– ControllerStatusCode	int	Controller status code

Output parameters

– ControllerStatusString	string	Controller status description

Return

– Error	long	Function error code

Matlab

Prototype

[Error, ControllerStatusString] **ControllerStatusStringGet** (int32 SocketID, int32 ControllerStatusCode)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– ControllerStatusCode	int32	Controller status code

Return

– Error	int32	Function error code
– ControllerStatusString	cstring	Controller status description

Python

Prototype

[Error, ControllerStatusString] **ControllerStatusStringGet** (integer SocketID, integer ControllerStatusCode)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– ControllerStatusCode	int	Controller status code

Return

– Error	int	Function error code
– ControllerStatusString	string	Controller status description

2.1.10 ControllerSynchronizeCorrectorISR

Name

ControllerSynchronizeCorrectorISR – Synchronize corrector ISR for master-slave controllers system.

Input testS

- Check command format: ERR_WRONG_FORMAT (-7)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check intput parameter type: ERR_WRONG_TYPE_CHAR (-13)

Description

This function sets the mode of corrector ISR synchronization between master controller and its slave controllers.

Possible synchronization mode *(ModeString*) values:

"*MasterWithEcho*": Turns a controller in the corrector ISR synchronization mode as Master. The master generates a synchronization signal (derived from its corrector ISR frequency) on a pin of the INHIBIT connector.

"*SlaveOnEcho*": Turns a controller in the corrector ISR synchronization mode as Slave. The slave receives a synchronization signal (on a pin of INHIBIT connector) coming from its master and uses it as its corrector frequency.

"*Master*": Return to local mode (each controller generates and uses its own corrector ISR frequency.

CAUTION

This function must be called on each controller (master and its slaves) in the following order: *Master first, then 1st, 2nd… slaves*.

Call this function only if every group is in the NOTINIT state.

If the controller has just rebooted, wait 300 seconds before calling this function (the necessary time to have the controller corrector ISR frequency stabilized).

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

Errors

- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13)
- SUCCESS (0): no error

TCL

Prototype

ControllerSynchronizeCorrectorISR $SocketID $ModeString

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
–	ModeString	string	Synchronization mode

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **ControllerSynchronizeCorrectorISR** (int SocketID, char *ModeString)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	ModeString	char *	Synchronization mode

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **ControllerSynchronizeCorrectorISR** (ByVal SocketID As Long, ByVal ModeString As String)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	ModeString	string	Synchronization mode

Output parameters

– None

Return

– Function error code

Matlab

Prototype

[Error] **ControllerSynchronizeCorrectorISR** (int32 SocketID, cstring ModeString)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– ModeString	cstring	Synchronization mode

Return

– Error	int32	Function error code

Python

Prototype

[Error] **ControllerSynchronizeCorrectorISR** (integer SocketID, string ModeString)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– ModeString	string	Synchronization mode

Return

– Error	int	Function error code

2.1.11 DoubleGlobalArrayGet

Name

DoubleGlobalArrayGet – Get the value of the global array of type "double".

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check input parameter type: ERR_WRONG_TYPE_INT (-15)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)
- Verify the index number [0:1000[: ERR_PARAMETER_OUT_OF_RANGE (-17)

Description

This function gets the variable value from the global array of type "double", related to a "Number" index. So, the first variable value from the global array is related to the index "0".

The returned value is returned in a double format.

NOTE

The number of data points in the global array of type "double" is limited to 1000.

Error codes

- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

DoubleGlobalArrayGet $SocketID $Number DoubleValue

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– Number	int	Index in the global array

Output parameters

– DoubleValue	floating point	Variable value

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **DoubleGlobalArrayGet** (int SocketID, int Number, double * DoubleValue)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– Number	int	Index in the global array

Output parameters

– DoubleValue	double *	Variable value

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **DoubleGlobalArrayGet** (ByVal SocketID As Long, Number As Integer, ByVal DoubleValue As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– Number	int	Index in the global array

Output parameters

– DoubleValue	double	Variable value

Return

– Error	long	Function error code

Matlab

Prototype

[Error, DoubleValue] **DoubleGlobalArrayGet** (int32 SocketID, int32 Number)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– Number	int32	Index in the global array

Return

– Error	int32	Function error code
– DoubleValue	doublePtr	Variable value

Python

Prototype

[Error, DoubleValue] **DoubleGlobalArrayGet** (integer SocketID, integer Number)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	Number	int	Index in the global array

Return

–	Error	int	Function error code
–	DoubleValue	doublePtr	Variable value

2.1.12 DoubleGlobalArraySet

Name

DoubleGlobalArraySet – Set a value for the global array of type "double".

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check input parameter type: ERR_WRONG_TYPE_INT (-15), ERR_WRONG_TYPE_DOUBLE (-14)
- Verify the index number [0:1000[: ERR_PARAMETER_OUT_OF_RANGE (-17)

Description

This function sets a new value in the global array located at the "Number" index and the new value is set in a double format.

NOTE

The first variable value from the global array is always located at index "0".

The number of data points in the global array is limited to 1000, so the last index is "999".

Error codes

- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

DoubleGlobalArraySet $SocketID $Number $DoubleValue

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– Number	int	Index in the global array
– DoubleValue	floating point	Variable value

Output parameters

– None

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **DoubleGlobalArraySet** (int SocketID, int Number, char * DoubleValue)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– Number	int	Index in the global array
– DoubleValue	double	Variable value

Output parameters

– None

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **DoubleGlobalArraySet** (ByVal SocketID As Long, Number As Integer, ByVal DoubleValue As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– Number	int	Index in the global array
– DoubleValue	double	Variable value

Output parameters

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **DoubleGlobalArraySet** (int32 SocketID, int32 Number, double DoubleValue)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– Number	int32	Index in the global array
– DoubleValue	double	Variable value

Return

– Error	int32	Function error code

Python

Prototype

[Error] **DoubleGlobalArraySet** (integer SocketID, integer Number, Double DoubleValue)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– Number	int	Index in the global array
– DoubleValue	double	Variable value

Return

– Error	int	Function error code

2.1.13 ErrorStringGet

Name

ErrorStringGet – Get the error description from a function error code.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check input parameter type: ERR_WRONG_TYPE_INT (-15)
- Check output parameter type: ERR_WRONG_TYPE_CHAR (-13)

Description

The function returns the error description corresponding to a function error code (see §0 Error list).

If the error code is not referenced then the "Unknown error code" message will be returned.

Error codes

- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

ErrorStringGet $SocketID $ErrorCode ErrorString

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– ErrorCode	int	Error code

Output parameters

– ErrorString	string	Error description

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **ErrorStringGet** (int SocketID, int ErrorCode, char *ErrorString)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– ErrorCode	int	Error code

Output parameters

– ErrorString	char *	Error description

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **ErrorStringGet** (ByVal SocketID As Long, ErrorCode As Integer, ByVal ErrorString As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– ErrorCode	int	Error code

Output parameters

– ErrorString	string	Error description

Return

– Error	long	Function error code

Matlab

Prototype

[Error, ErrorString] **ErrorStringGet** (int32 SocketID, int32 ErrorCode)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– ErrorCode	int32	Error code

Return

– Error	int32	Function error code
– ErrorString	cstring	Error description

Python

Prototype

[Error, ErrorString] **ErrorStringGet** (integer SocketID, integer ErrorCode)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– ErrorCode	int	Error code

Return

– Error	int	Function error code
– ErrorString	string	Error description

2.1.14 ElapsedTimeGet

Name

ElapsedTimeGet – Get the elapsed time since the controller was power on.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function returns the time in seconds that elapsed since the controller was power on.

Error codes

- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

ElapsedTimeGet $SocketID $ErrorCode ElapsedTime

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– ErrorString	double	Elapsed time (seconds)

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **ElapsedTimeGet** (int SocketID, double * ElapsedTime)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– ElapsedTime	double *	Elapsed time (seconds)

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **ElapsedTimeGet** (ByVal SocketID As Long, ErrorCode As Integer, ByVal ElapsedTime As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– ElapsedTime	double	Elapsed time (seconds)

Return

– Error	long	Function error code

Matlab

Prototype

[Error, ElapsedTime] **ElapsedTimeGet** (int32 SocketID)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function

Return

– Error	int32	Function error code
– ElapsedTime	double	Elapsed time (seconds)

Python

Prototype

[Error, ElapsedTime] **ElapsedTimeGet** (integer SocketID)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Return

– Error	int	Function error code
– ElapsedTime	double	Elapsed time (seconds)

2.1.15 FirmwareVersionGet

Name

FirmwareVersionGet – Gets the version of the firmware inside the controller.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check output parameter type: ERR_WRONG_TYPE_CHAR (-13)

Description

This function gets the controller name and the firmware version.

Example of returned version string:

"XPS-Q8 Firmware V1.0.0"

Controller name is **XPS-Q8**

Firmware version is **V1.0.0**

Error codes

- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13)
- SUCCESS (0): no error

TCL

Prototype

FirmwareVersionGet $SocketID Version

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– Version	string	Controller version

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **FirmwareVersionGet** (int SocketID, char * Version)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– Version	char *	Controller version

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **FirmwareVersionGet** (ByVal SocketID As Long, ByVal ErrorString As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– Version	string	Controller version

Return

– Error	long	Function error code

Matlab

Prototype

[Error, Version] **FirmwareVersionGet** (int32 SocketID)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function

Return

– Error	int32	Function error code
– Version	cstring	Controller version

Python

Prototype

[Error, Version] **FirmwareVersionGet** (integer SocketID)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Return

– Error	int	Function error code
– Version	string	Controller version

2.1.16 GroupStatusStringGet

Name

GroupStatusStringGet – Get the group state description from a group state code.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check input parameter type: ERR_WRONG_TYPE_INT (-15)
- Check output parameter type: ERR_WRONG_TYPE_CHAR (-13)

Description

This function returns the group state description corresponding to a group state code (see § 0 Group state list).

If the group state code is not referenced then the "Error: undefined status" message will be returned.

Error codes

- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

GroupStatusStringGet $SocketID $GroupStatusCode GroupStatusString

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupStatusCode	int	Group status code

Output parameters

– GroupStatusString	string	Group status description

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **GroupStatusStringGet** (int SocketID, int GroupStatusCode, char * GroupStatusString)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupStatusCode	int	Group status code

Output parameters

– GroupStatusString	char *	Group status description
– Return		
– Error	int	Function error code

Visual Basic

Prototype

Long **GroupStatusStringGet** (ByVal SocketID As Long, GroupStatusCode As Integer, ByVal GroupStatusString As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupStatusCode	int	Group status code

Output parameters

– GroupStatusString	string	Group status description
– Return		
– Error	long	Function error code

Matlab

Prototype

[Error, GroupStatusString] **GroupStatusStringGet** (int32 SocketID, int32 GroupStatusCode)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupStatusCode	int32	Group status code

Return

– Error	int32	Function error code
– GroupStatusString	cstring	Group status description

Python

Prototype

[Error, GroupStatusString] **GroupStatusStringGet** (integer SocketID, integer GroupStatusCode)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupStatusCode	int	Group status code

Return

–	Error	int	Function error code
–	GroupStatusString	string	Group status description

2.1.17 GlobalArrayGet

Name

GlobalArrayGet – Get a value from the global array.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check input parameter type: ERR_WRONG_TYPE_INT (-15)
- Check output parameter type: ERR_WRONG_TYPE_CHAR (-13)
- Verify the index number [0:100[: ERR_PARAMETER_OUT_OF_RANGE (-17)

Description

This function gets the variable value from the global array, related to a "Number" index. So, the first variable value from the global array is referenced to the index "0".

The returned value is returned in a string.

NOTE

The number of data points in the global array is limited to 100.

Error codes

- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

GlobalArrayGet $SocketID $Number StringValue

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– Number	int	Index in the global array

Output parameters

– StringValue	string	Variable value

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **GlobalArrayGet** (int SocketID, int Number, char * StringValue)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– Number	int	Index in the global array

Output parameters

– StringValue	char *	Variable value

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **GlobalArrayGet** (ByVal SocketID As Long, Number As Integer, ByVal StringValue As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– Number	int	Index in the global array

Output parameters

– StringValue	string	Variable value

Return

– Error	long	Function error code

Matlab

Prototype

[Error, StringValue] **GlobalArrayGet** (int32 SocketID, int32 Number)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– Number	int32	Index in the global array

Return

– Error	int32	Function error code
– StringValue	cstring	Variable value

Python

Prototype

[Error, StringValue] **GlobalArrayGet** (integer SocketID, integer Number)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– Number	int	Index in the global array

Return

– Error	int	Function error code
– StringValue	string	Variable value

2.1.18 GlobalArraySet

Name

GlobalArraySet – Set the value of the global array.

Input tests

Check command format: ERR_WRONG_FORMAT (-7)

Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)

Check input parameter type: ERR_WRONG_TYPE_INT (-15), ERR_WRONG_TYPE_CHAR (-13)

- Verify the index number [0:100[: ERR_PARAMETER_OUT_OF_RANGE (-17)

Description

This function sets a new value in the global array related to the "Number" index and the new value is set to a string.

NOTE

The first variable value of the global array is always referenced to the index "0".

The number of data points in the global array is limited to 100, so the last index is "99".

Error codes

- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

GlobalArraySet $SocketID $Number $StringValue

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– Number	int	Index in the global array
– StringValue	string	Variable value

Output parameters

– None

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **GlobalArraySet** (int SocketID, int Number, char * StringValue)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– Number	int	Index in the global array
– StringValue	char *	Variable value

Output parameters

– None

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **GlobalArraySet** (ByVal SocketID As Long, Number As Integer, ByVal StringValue As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– Number	int	Index in the global array
– StringValue	string	Variable value

Output parameters

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **GlobalArraySet** (int32 SocketID, int32 Number, cstring StringValue)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– Number	int32	Index in the global array
– StringValue	cstring	Variable value

Return

– Error	int32	Function error code

Python

Prototype

[Error] **GlobalArraySet** (integer SocketID, integer Number, string StringValue)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– Number	int	Index in the global array
– StringValue	string	Variable value

Return

– Error	int	Function error code

2.1.19 InstallerVersionGet

Name

InstallerVersionGet – Gets the version of the used installer to upgrade the controller.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check output parameter type: ERR_WRONG_TYPE_CHAR (-13)

Description

This function gets the controller name and the installer version.

Example of returned version string:

"XPS-Q8 Standard Installer Pack Number 30016"

Controller name is **XPS-Q8**

Installer version is **30016**

Error codes

- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13)
- SUCCESS (0): no error

TCL

Prototype

InstallerVersionGet $SocketID Version

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– Version	string	Installer version

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **InstallerVersionGet** (int SocketID, char * Version)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

–	Version	char *	Installer version

Return

–	Error	int	Function error code

Visual Basic

Prototype

Long **InstallerVersionGet** (ByVal SocketID As Long, ByVal ErrorString As String)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

–	Version	string	Installer version

Return

–	Error	long	Function error code

Matlab

Prototype

[Error, Version] **InstallerVersionGet** (int32 SocketID)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function

Return

–	Error	int32	Function error code
–	Version	cstring	Installer version

Python

Prototype

[Error, Version] **InstallerVersionGet** (integer SocketID)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Return

–	Error	int	Function error code
–	Version	string	Installer version

2.1.20 KillAll

Name

KillAll – Kills all groups.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)

Description

This function kills and resets all groups.

This function resets all analog and digital I/O also.

The following sequence of steps is performed by the KillAll command.

1) An "emergency stop" is done if the group state is defined as:
   HOMING
   REFERENCING
   MOVING
   JOGGING
   ANALOG_TRACKING
2) The motor is turned off, the motion done is stopped and the control loop is stopped.
3) "ERR_EMERGENCY_SIGNAL" is returned by each function that is in progress, and where the group state is:
   MOTOR_INIT
   ENCODER_CALIBRATING
   HOMING
   REFERENCING
   MOVING
   TRAJECTORY
   ERR_EMERGENCY_SIGNAL
4) At end, the group state is not initialized "NOTINIT" for all groups.

Error codes

- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

KIllAll $SocketID

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– None

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **KillAll** (int SocketID)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– None

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **KillAll** (ByVal SocketID As Long)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **KillAll** (int32 SocketID)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function

Return

Python

Prototype

[Error] **KillAll** (integer SocketID)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Return

–	Error	int	Function error code

2.1.21 Reboot

Name

Reboot – Reboots the controller.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)

Description

This function reboots the controller.

Notes that this function is not a hardware reboot (power off/on), it is a firmware reboot.

NOTE

If an FTP client is connected, this function is not allowed and ERR_NOT_ALLOWED_ACTION (-22) is returned.

Error codes

- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

Reboot $SocketID

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– None

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **Reboot** (int SocketID)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– None

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **Reboot** (ByVal SocketID As Long)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **Reboot** (int32 SocketID)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function

Return

– Error	int32	Function error code

Python

Prototype

[Error] **Reboot** (integer SocketID)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Return

– Error	int	Function error code

2.1.22 RestartApplication

Name

RestartApplication – Restarts the controller's application and avoids hardware reboot.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)

Description

This function allows restarting controller applications without hardware reboot.

Error codes

- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

RestartApplication $SocketID

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– None

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **RestartApplication** (int SocketID)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– None

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **RestartApplication** (ByVal SocketID As Long)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **RestartApplication** (int32 SocketID)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function

Return

– Error	int32	Function error code

Python

Prototype

[Error] **RestartApplication** (integer SocketID)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Return

– Error	int	Function error code

2.1.23 TimerGet

Name

TimerGet – Gets the number of frequency ticks for the selected timer.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check input parameter mnemonic: ERR_WRONG_TYPE (-10)
- Check output parameter type: ERR_WRONG_TYPE_INT (-15)

Description

This function returns the number of frequency ticks configured for the selected timer.

The "TimerName" can be defined as:

- Timer1
- Timer2
- Timer3
- Timer4
- Timer5

The "FrequencyTicks" defines the frequency of the timer:

One frequency tick represents a corrector period => 0.125 ms => 8 khz

N frequency ticks represents N corrector periods => N * 0.125 ms => $\frac{8}{N}$ kHz

NOTE

The NULL "FrequencyTicks" (=0) means that the timer is disabled.

Error codes

- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE (-10)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

TimerGet $SocketID $TimerName FrequencyTicks

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– TimerName	string	Name of timer

Output parameters

– FrequencyTicks	int	Number of frequency ticks

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **TimerGet** (int SocketID, char *TimerName, int* FrequencyTicks)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– TimerName	char *	Name of timer

Output parameters

– FrequencyTicks	int *	Number of frequency ticks

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **TimerGet** (ByVal SocketID As Long, ByVal TimerName As String, FrequencyTicks As Integer)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– TimerName	string	Name of timer

Output parameters

– FrequencyTicks	int	Number of frequency ticks

Return

– Error	long	Function error code

Matlab

Prototype

[Error, FrequencyTicks] **TimerGet** (int32 SocketID, cstring TimerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– TimerName	cstring	Name of timer

Return

– Error	int32	Function error code
– FrequencyTicks	int32	Number of frequency ticks

Python

Prototype

[Error, FrequencyTicks] **TimerGet** (integer SocketID, string TimerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– TimerName	string	Name of timer

Return

– Error	int	Function error code
– FrequencyTicks	int	Number of frequency ticks

2.1.24 TimerSet

Name

TimerSet – Sets the number of frequency ticks for the selected timer.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check input parameter mnemonic: ERR_WRONG_TYPE (-10)
- Check output parameter type: ERR_WRONG_TYPE_INT (-15)

Description

This function sets the number of frequency ticks for the selected timer to activate it.

The "TimerName" can be defined as:

- Timer1
- Timer2
- Timer3
- Timer4
- Timer5

The "FrequencyTicks" allows to defined the frequency of the timer:

One frequency tick represents a corrector period => 0.125 ms => 8 khz

N frequency ticks represents N corrector periods => N * 0.125 ms => $\frac{8}{N}$ kHz

NOTE

If the "FrequencyTicks" is null (0) then the timer is disabled.

Error codes

- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE (-10)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

TimerSet $SocketID $TimerName $FrequencyTicks

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– TimerName	string	Name of timer
– FrequencyTicks	int	Number of frequency ticks

Output parameters

– None

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **TimerSet** (int SocketID, char *TimerName, int FrequencyTicks)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– TimerName	char *	Name of timer
– FrequencyTicks	int	Number of frequency ticks

Output parameters

– None

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **TimerSet** (ByVal SocketID As Long, ByVal TimerName As String, ByVal FrequencyTicks As Integer)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– TimerName	string	Name of timer
– FrequencyTicks	int	Number of frequency ticks

Output parameters

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **TimerSet** (int32 SocketID, cstring TimerName, int32 FrequencyTicks)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– TimerName	cstring	Name of timer
– FrequencyTicks	int32	Number of frequency ticks

Return

– Error	int32	Function error code

Python

Prototype

[Error, FrequencyTicks] **TimerSet** (integer SocketID, string TimerName, integer FrequencyTicks)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– TimerName	string	Name of timer
– FrequencyTicks	int	Number of frequency ticks

Return

– Error	int	Function error code

2.2 Positioner

2.2.1 Description

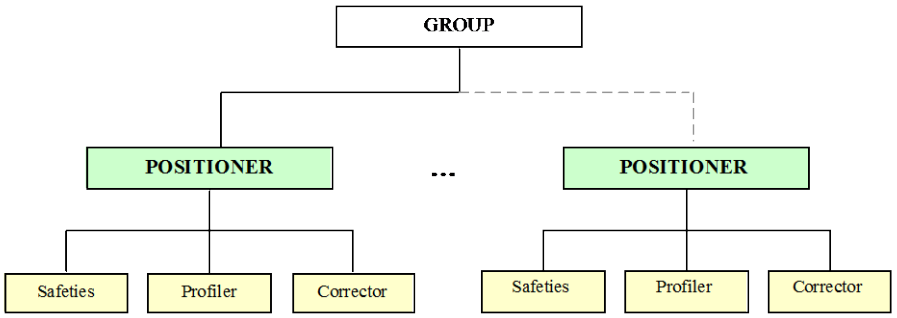
Positioner objects are used to define all motion specific configuration parameters.

The positioner includes a mapping correction: X = f (X)

The positioner includes the SGamma profile.

The maximum number of positioners is limited to **8**.

2.2.2 Object Structure



To use a **positioner**, it must belong to a motion group. Positioners are defined by **full positioner name**. The full positioner name is composed of the group name and the positioner name seperated by a period (.).

Example:

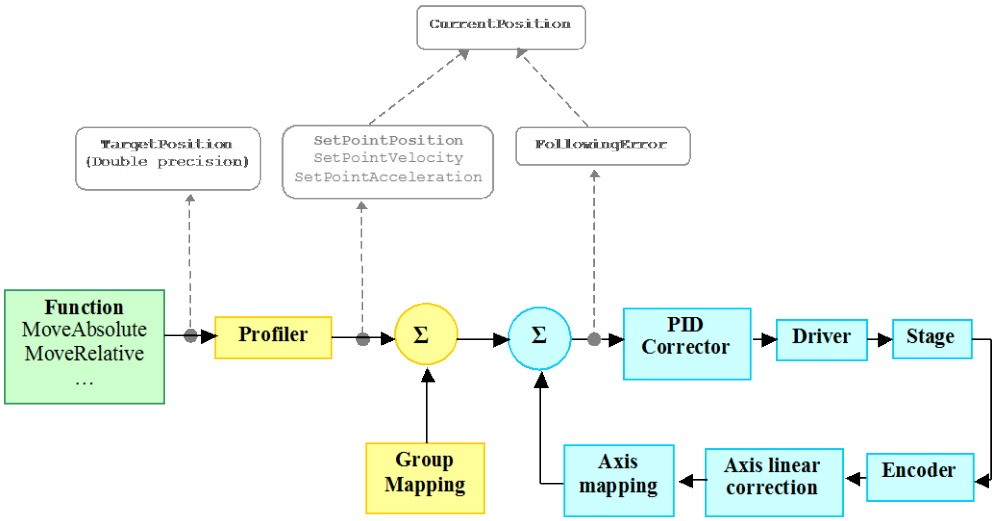
GroupName.PositionerName

2.2.3 Definition of the Different Positions for a Positioner

For each positioner, three different positions can be called:

1. The **SetpointPosition** is the profiler position. This is the position where the positioner should be according to ideal theoretical motion profile.
2. The **CurrentPosition** is the encoder position of the stage after mapping corrections. This is the actual position of the positioner
3. The **TargetPosition** is the final target position commanded by the user.

The difference between the SetpointPosition and the CurrentPosition is called the following error.



For instance, during a motion from the position 0 (units) to 100 (units), we could have the following result:

SetpointPosition = 50

CurrentPosition = 49.998 (FollowingError = 50 – 49.998 = 0.002 unit)

TargetPosition = 100.

2.2.4 Function Description

2.2.4.1 PositionerAccelerationAutoScaling

Name

PositionerAccelerationAutoScaling – Auto-scaling process for determining the stage scaling acceleration.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)
- Positioner must not be a "Secondary Positioner": ERR_WRONG_OBJECT_TYPE (-8)
- Check positioner name: ERR_POSITIONER_NAME (-18)
- Check group type: ERR_WRONG_OBJECT_TYPE (-8)
- Control loop type must be "PIDFFAcceleration": ERR_UNCOMPATIBLE (-24)
- Group status must be "NOT_INITIALIZED": ERR_NOT_ALLOWED_ACTION (-22)

Description

The function executes an auto-scaling process and returns the calculated scaling acceleration. The selected group must be in "NOTINIT" state, else ERR_NOT_ALLOWED_ACTION (-22) is returned.

This function works only if the positioner control loop type is "PIDFFAcceleration" (acceleration control), else it returns ERR_UNCOMPATIBLE error.

This function checks the positioner error. If an error is detected, the hardware status register is reset (motor on) and the positioner error is cleared before checking it again. If a positioner error is present, the motor is turned off, ERR_POSITIONER_ERROR (-5) is returned and the group status becomes "NOTINIT".

If there is no positioner error then the master-slave error is cleared, the encoder is preset (update encoder position) and the user travel limits are checked. If a travel limit error is detected then the motor is turned off, ERR_TRAVEL_LIMITS (-35) is returned and the group status becomes "NOTINIT".

If no error, the motor will be initialized in the case of stage acceleration control. If motor initialization fails then the error ERR_MOTOR_INITIALIZATION_ERROR (-50) is returned and the group status becomes "NOTINIT".

If motor initialization is successful, the positions are preset, the motion is enabled (the motor is powered) permitting the process of auto-scaling. If the motion cannot be enabled, ERR_NOT_ALLOWED_ACTION (-22) is returned.

If the auto-scaling fails ERR_SCALING_CALIBRATION (-105) is returned or if the motion becomes disabled then ERR_EMERGENCY_SIGNAL (-26) is returned.

The auto-scaling process is executed in 5 periods. At the end of each period, the auto-tuning process estimates the auto-tuning quality by calculating the noise/signal ratio. If the noise/signal ratio is very close to zero (it means no oscillation) ERR_RELAY_FEEDBACK_TEST_NO_OSCILLATION (-101) is returned. Else, where if the noise ratio > MaximumNoiseRatio (normally between 0.1 and 0.2, exact

value defined in system.ref) then
ERR_RELAY_FEEDBACK_TEST_SIGNAL_NOISY (-102) is returned.

If the number of acquired data points (minimum = 9) or the number of acquired signal periods (minimum = 5) is not enough for a good estimate then
ERR_SIGNAL_POINTS_NOT_ENOUGH (-103) is returned.

At end of this function, the new value of scaling acceleration is returned and the group status becomes "NOTINIT" once again.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_MOTOR_INITIALIZATION_ERROR (-50)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_POSITIONER_ERROR (-5)
- ERR_POSITIONER_NAME (-18)
- ERR_RELAY_FEEDBACK_TEST_NO_OSCILLATION (-101)
- ERR_RELAY_FEEDBACK_TEST_SIGNAL_NOISY (-102)
- ERR_SCALING_CALIBRATION (-105)
- ERR_SIGNAL_POINTS_NOT_ENOUGH (-103)
- ERR_TRAVEL_LIMITS (-35)
- ERR_UNCOMPATIBLE (-24)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerAccelerationAutoScaling $SocketID $PositionerName Scaling

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– PositionerName	string	Name of a positioner

Output parameters

– Scaling	floating point	Calculated scaling acceleration value

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **PositionerAccelerationAutoScaling** (int SocketID, char * PositionerName, double * Scaling)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– PositionerName	char *	Name of a positioner

Output parameters

– Scaling	double *	Calculated scaling acceleration value

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerAccelerationAutoScaling** (ByVal SocketID As Long, ByVal PositionerName As String, Scaling As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– PositionerName	string	Name of a positioner

Output parameters

– Scaling	double	Calculated scaling acceleration value

Return

– Error	long	Function error code

Matlab

Prototype

[Error, Scaling] **PositionerAccelerationAutoScaling** (int32 SocketID, cstring PositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– PositionerName	cstring	Name of a positioner

Return

– Error	int32	Function error code
– Scaling	double	Calculated scaling acceleration value

Python

Prototype

[Error, Scaling] **PositionerAccelerationAutoScaling** (integer SocketID, string PositionerName, string Password)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– PositionerName	string	Name of a positioner

Return

– Error	int	Function error code
– Scaling	double	Calculated scaling acceleration value

2.2.4.2 PositionerAnalogTrackingPositionParametersGet

Name

PositionerAnalogTrackingPositionParametersGet – Gets the parameters of the current tracking position mode.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check input parameter: ERR_WRONG_OBJECT_TYPE (-8), ERR_POSITIONER_NAME (-18)
- Check output parameter type: ERR_WRONG_TYPE_CHAR (-13), ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function returns the current analog input name, the current offset and the current scale used by analog tracking position mode. For a more thorough description of the analog tracking mode, please refer to the XPS Motion Tutorial section Motion/Analog tracking.

NOTE

"velocity" and "acceleration" define the maximum velocity and acceleration used in the position tracking mode.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerAnalogTrackingPositionParametersGet $SocketID $FullPositionerName GPIOName Offset Scale velocity acceleration

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– GPIOName	string	Analog input name (ADC)
– Offset	double	Offset (volts)
– Scale	double	Scale (Units/Volts)
– Velocity	double	Velocity (Units/s)
– Acceleration	double	Acceleration (Units/s²)

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerAnalogTrackingPositionParametersGet** (int SocketID, char FullPositionerName [250], char *GPIOName, double * Offset, double * Scale, double * velocity, double * acceleration)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name

Output parameters

– GPIOName	char *	Analog input name (ADC)
– Offset	double *	Offset in volts
– Scale	double *	Scale (Units/Volts)
– Velocity	double *	Velocity (Units/s)
– Acceleration	double *	Acceleration (Units/s²)

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerAnalogTrackingPositionParametersGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal GPIOName As String, Offset As Double, Scale As Double, velocity As Double, acceleration As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– GPIOName	string	Analog input name (ADC)
– Offset	double	Offset in volts
– Scale	double	Scale (Units/Volts)
– Velocity	double	Velocity (Units/s)
– Acceleration	double	Acceleration (Units/s²)

Return

– Error	long	Function error code

Matlab

Prototype

[Error, GPIOName, Offset, Scale, velocity, acceleration]
PositionerAnalogTrackingPositionParametersGet (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– GPIOName	cstring	Analog input name (ADC)
– Offset	double	Offset in volts
– Scale	double	Scale (Units/Volts)
– Velocity	double	Velocity (Units/s)
– Acceleration	double	Acceleration (Units/s²)

Python

Prototype

[Error, GPIOName, Offset, Scale, velocity, acceleration]
PositionerAnalogTrackingPositionParametersGet (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code
– GPIOName	string	Analog input name (ADC)
– Offset	double	Offset in volts
– Scale	double	Scale (Units/Volts)
– Velocity	double	Velocity (Units/s)
– Acceleration	double	Acceleration (Units/s²)

2.2.4.3 PositionerAnalogTrackingPositionParametersSet

Name

PositionerAnalogTrackingPositionParametersSet – Sets the parameters of the current tracking position mode.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check input parameter: ERR_POSITIONER_NAME (-18)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)
- Check Positioner and GPIO type (ADC): ERR_WRONG_OBJECT_TYPE (-8)
- Check velocity and acceleration: ERR_PARAMETER_OUT_OF_RANGE (-17)

Description

This function modifies the analog input name, the offset and the scale used by the analog tracking position mode. To use this function, the group state must be READY else ERR_NOT_ALLOWED_ACTION is returned.

The "Offset" and the "Scale" parameters are used to calculate the target tracking position:

TrackingPosition = InitialPosition + (AnalogValue - **Offset**) * **Scale**

The "velocity" and "acceleration" parameters define the maximum velocity and acceleration used in the position tracking mode.

NOTE

The parameters of analog tracking position mode can be reset if the "GPIOName" parameter is blank.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerAnalogTrackingPositionParametersSet $SocketID $FullPositionerName $GPIOName $Offset $Scale $velocity $acceleration

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– GPIOName	string	Analog input name (ADC)
– Offset	double	Offset (volts)
– Scale	double	Scale (Units/Volts)
– Velocity	double	Velocity (Units/s)
– Acceleration	double	Acceleration (Units/s²)

Output parameters

– None

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerAnalogTrackingPositionParametersSet** (int SocketID, char FullPositionerName [250], char *GPIOName, double Offset, double Scale, double velocity, double acceleration)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name
– GPIOName	char *	Analog input name (ADC)
– Offset	double	Offset in volts
– Scale	double	Scale (Units/Volts)
– Velocity	double	Velocity (Units/s)
– Acceleration	double	Acceleration (Units/s²)

Output parameters

– None

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerAnalogTrackingPositionParametersSet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal GPIOName As String, ByVal Offset As Double, ByVal Scale As Double, ByVal velocity As Double, ByVal acceleration As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– GPIOName	string	Analog input name (ADC)
– Offset	double	Offset in volts
– Scale	double	Scale (Units/Volts)
– Velocity	double	Velocity (Units/s)
– Acceleration	double	Acceleration (Units/s²)

Output parameters

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **PositionerAnalogTrackingPositionParametersSet** (int32 SocketID, cstring FullPositionerName, cstring GPIOName, double Offset, double Scale, double velocity, double acceleration)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name
– GPIOName	cstring	Analog input name (ADC)
– Offset	double	Offset in volts
– Scale	double	Scale (Units/Volts)
– Velocity	double	Velocity (Units/s)
– Acceleration	double	Acceleration (Units/s²)

Return

– Error	int32	Function error code

Python

Prototype

[Error] **PositionerAnalogTrackingPositionParametersSet** (integer SocketID, string FullPositionerName, string GPIOName, double Offset, double Scale, double velocity, double acceleration)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– GPIOName	string	Analog input name (ADC)
– Offset	double	Offset in volts
– Scale	double	Scale (Units/Volts)
– Velocity	double	Velocity (Units/s)
– Acceleration	double	Acceleration (Units/s²)

Return

– Error	int	Function error code

2.2.4.4 PositionerAnalogTrackingVelocityParametersGet

Name

PositionerAnalogTrackingVelocityParametersGet – Gets the parameters of the current tracking velocity mode.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check input parameter: ERR_WRONG_OBJECT_TYPE (-8), ERR_POSITIONER_NAME (-18)
- Check output parameter type: ERR_WRONG_TYPE_CHAR (-13), ERR_WRONG_TYPE_DOUBLE (-14), ERR_WRONG_TYPE_INT (-15)

Description

This function returns the analog input name, the offset, the scale, the deadband threshold and the order used by analog traking velocity mode. For a more thorough description of the analog tracking mode, please refer to the XPS Motion Tutorial section Motion/Analog tracking.

NOTE

"velocity" and "acceleration" define the maximum velocity and acceleration used in the velocity tracking mode.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13)
- ERR_WRONG_TYPE_DOUBLE (-14)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

PositionerAnalogTrackingVelocityParametersGet $SocketID $FullPositionerName GPIOName Offset Scale DeadBandThreshold Order velocity acceleration

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– GPIOName	string	Analog input name (ADC)
– Offset	double	Offset (volts)
– Scale	double	Scale (Units/Volts)
– DeadBandThreshold	double	Dead band threshold (Volts)
– Order	int	Order (No unit)
– Velocity	double	Velocity (Units/s)
– Acceleration	double	Acceleration (Units/s²)

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerAnalogTrackingVelocityParametersGet** (int SocketID, char FullPositionerName [250], char *GPIOName, double * Offset, double * Scale, double * DeadBandThreshold, int * Order, double * velocity, double * acceleration)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name

Output parameters

– GPIOName	char *	Analog input name (ADC)
– Offset	double *	Offset in volts
– Scale	double *	Scale (Units/Volts)
– DeadBandThreshold	double *	Dead band threshold (Volts)
– Order	integer *	Order (No unit)
– Velocity	double *	Velocity (Units/s)
– Acceleration	double *	Acceleration (Units/s²)

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerAnalogTrackingVelocityParametersGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal GPIOName As String, Offset As Double, Scale As Double, DeadBandThreshold As Double, Order As Integer, velocity As Double, acceleration As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– GPIOName	string	Analog input name (ADC)
– Offset	double	Offset in volts
– Scale	double	Scale (Units/Volts)
– DeadBandThreshold	double	Dead band threshold (Volts)
– Order	int	Order (No unit)
– Velocity	double	Velocity (Units/s)
– Acceleration	double	Acceleration (Units/s²)

Return

– Error	long	Function error code

Matlab

Prototype

[Error, GPIOName, Offset, Scale, DeadBandThreshold, Order , velocity, acceleration] **PositionerAnalogTrackingVelocityParametersGet** (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– GPIOName	cstring	Analog input name (ADC)
– Offset	double	Offset in volts
– Scale	double	Scale (Units/Volts)
– DeadBandThreshold	double	Dead band threshold (Volts)
– Order	int32	Order (No unit)
– Velocity	double	Velocity (Units/s)
– Acceleration	double	Acceleration (Units/s²)

Python

Prototype

[Error, GPIOName, Offset, Scale, DeadBandThreshold, Order , velocity, acceleration] **PositionerAnalogTrackingVelocityParametersGet** (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code
– GPIOName	string	Analog input name (ADC)
– Offset	double	Offset in volts
– Scale	double	Scale (Units/Volts)
– DeadBandThreshold	double	Dead band threshold (Volts)
– Order	int	Order (No unit)
– Velocity	double	Velocity (Units/s)
– Acceleration	double	Acceleration (Units/s²)

2.2.4.5 PositionerAnalogTrackingVelocityParametersSet

Name

PositionerAnalogTrackingVelocityParametersSet – Sets the parameters of the current tracking velocity mode.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check Positioner: ERR_POSITIONER_NAME (-18), ERR_WRONG_OBJECT_TYPE (-8)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14), ERR_WRONG_TYPE_INT (-15)
- Check GPIO type (ADC): ERR_WRONG_OBJECT_TYPE (-8)
- Check velocity and acceleration: ERR_PARAMETER_OUT_OF_RANGE (-17)

Description

This function allows modifying the GPIO name, offset, scale, deadband threshold and the order used by analog tracking velocity mode. To use this function the group state must be READY else ERR_NOT_ALLOWED_ACTION (-22) is returned.

The target tracking velocity is defined as follows:

InputValue = GPIOAnalogInput – ***Offset***

MaxADCAmplitude = 10/GPIOAnalogGain

if (InputValue >= 0) then

  InputValue = InputValue - ***DeadBandThreshold***

  if (InputValue < 0) then InputValue = 0

else

  InputValue = AnalogInputValue + ***DeadBandThreshold***

  if (InputValue > 0) then InputValue = 0

OutputValue = (|InputValue|/MaxADCAmplitude)^{Order}

TrackingVelocity = Sign(InputValue) * OutputValue * ***Scale*** * MaxADCAmplitude

The "velocity" and "acceleration" define the maximum velocity and acceleration used in the velocity tracking mode.

NOTE

The analog tracking velocity mode can be reset if the "GPIOName" parameter is empty.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

PositionerAnalogTrackingVelocityParametersSet $SocketID $FullPositionerName $GPIOName $Offset $Scale $DeadBandThreshold $Order $velocity $acceleration

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– GPIOName	string	Analog input name (ADC)
– Offset	double	Offset (volts)
– Scale	double	Scale (Units/Volts)
– DeadBandThreshold	double	Dead band threshold (Volts)
– Order	int	Order (No unit)
– Velocity	double	Velocity (Units/s)
– Acceleration	double	Acceleration (Units/s²)

Output parameters

– None

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerAnalogTrackingVelocityParametersSet** (int SocketID, char FullPositionerName [250], char *GPIOName, double Offset, double Scale, double DeadBandThreshold, int Order, double velocity, double acceleration)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name
– GPIOName	char *	Analog input name (ADC)
– Offset	double	Offset in volts
– DeadBandThreshold	double	Dead band threshold (Volts)
– Order	int	Order (No unit)
– Scale	double	Scale (Units/Volts)
– Velocity	double	Velocity (Units/s)
– Acceleration	double	Acceleration (Units/s²)

Output parameters

– None

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerAnalogTrackingVelocityParametersSet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal GPIOName As String, ByVal Offset As Double, ByVal Scale As Double, ByVal DeadBandThreshold As Double, ByVal Order As Integer, ByVal velocity As Double, ByVal acceleration As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– GPIOName	string	Analog input name (ADC)
– Offset	double	Offset in volts
– DeadBandThreshold	double	Dead band threshold (Volts)
– Order	int	Order (No unit)
– Scale	double	Scale (Units/Volts)
– Velocity	double	Velocity (Units/s)
– Acceleration	double	Acceleration (Units/s²)

Output parameters

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **PositionerAnalogTrackingVelocityParametersSet** (int32 SocketID, cstring FullPositionerName, cstring GPIOName, double Offset, double Scale, double DeadBandThreshold, int32 Order, double velocity, double acceleration)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name
– GPIOName	cstring	Analog input name (ADC)
– Offset	double	Offset in volts
– DeadBandThreshold	double	Dead band threshold (Volts)
– Order	int32	Order (No unit)
– Scale	double	Scale (Units/Volts)
– Velocity	double	Velocity (Units/s)
– Acceleration	double	Acceleration (Units/s²)

Return

– Error	int32	Function error code

Python

Prototype

[Error] **PositionerAnalogTrackingVelocityParametersSet** (integer SocketID, string FullPositionerName, string GPIOName, double Offset, double Scale, double DeadBandThreshold, integer Order, double velocity, double acceleration)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– GPIOName	string	Analog input name (ADC)
– Offset	double	Offset in volts
– DeadBandThreshold	double	Dead band threshold (Volts)
– Order	int	Order (No unit)
– Scale	double	Scale (Units/Volts)
– Velocity	double	Velocity (Units/s)
– Acceleration	double	Acceleration (Units/s²)

Return

– Error	int	Function error code

2.2.4.6 PositionerBacklashDisable

Name

PositionerBacklashDisable – Disables the backlash compensation.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type (must not be a secondary positioner): ERR_WRONG_OBJECT_TYPE (-8)

Description

This function disables the backlash compensation. For a more thorough description of the backlash compensation, please refer to the XPS Motion Tutorial section Compensation/Backlash compensation.

In the "stages.ini" file the parameter "Backlash" will enable or disable this feature as follows:

Backlash = 0 —> Disable backlash

Backlash > 0 —> Enable backlash

NOTE

The backlash compensation is not allowed with a secondary positioner (gantry mode).

The backlash must be disabled to execute a trajectory, to use the jog mode or to use the analog tracking mode.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_OBJECT_TYPE (-8)
- SUCCESS (0): no error

TCL

Prototype

PositionerBacklashDisable SocketID FullPositionerName

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name (maximum size = 250)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **PositionerBacklashDisable** (int SocketID, char * FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– FullPositionerName	char *	Positioner name (maximum size = 250)

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **PositionerBacklashDisable** (ByVal SocketID As Long, ByVal FullPositionerName As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name (maximum size = 250)

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **PositionerBacklashDisable** (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– FullPositionerName	cstring	Positioner name (maximum size = 250)

Return

– Function error code

Python

Prototype

integer **PositionerBacklashDisable** (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– FullPositionerName	string	Positioner name (maximum size = 250)

Return

– Function error code

2.2.4.7 PositionerBacklashEnable

Name

PositionerBacklashEnable – Enables the backlash compensation.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Group status must be "NOTINIT": ERR_NOT_ALLOWED_ACTION (-22)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type (must not be a secondary positioner): ERR_WRONG_OBJECT_TYPE (-8)

Description

This function enables the backlash compensation defined in the "stages.ini" file or defined by the "PositionerBacklashSet" function. If the backlash compensation value is null then this function will have not effect, and backlash compensation will remain disabled. For a more thorough description of the backlash compensation, please refer to the XPS Motion Tutorial section Compensation/Backlash compensation.

The group state must be NOTINIT to enable the backlash compensation. If it is not the case then ERR_NOT_ALLOWED_ACTION (-22) is returned.

In the "stages.ini" file the parameter "Backlash" allows the user to enable or disable the backlash compensation.

Backlash = 0 —> Disable backlash

Backlash > 0 —> Enable backlash

NOTE

The backlash must be disabled to execute a trajectory, to use the jog mode or to use the analog tracking mode.

CAUTION

It is not possible to use backlash compensation with positioners that have a "HomeSearchSequenceType" defined as "CurrentPositionAsHome" or that have a "PositionerMappingFileName" defined in the stages.ini file.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_OBJECT_TYPE (-8)
- SUCCESS (0): no error

TCL

Prototype

PositionerBacklashEnable SocketID FullPositionerName

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name (maximum size = 250)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **PositionerBacklashEnable** (int SocketID, char * FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– FullPositionerName	char *	Positioner name (maximum size = 250)

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **PositionerBacklashEnable** (ByVal SocketID As Long, ByVal FullPositionerName As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name (maximum size = 250)

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **PositionerBacklashEnable** (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– FullPositionerName	cstring	Positioner name (maximum size = 250)

Return

– Function error code

Python

Prototype

integer **PositionerBacklashEnable** (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– FullPositionerName	string	Positioner name (maximum size = 250)

Return

– Function error code

2.2.4.8 PositionerBacklashGet

Name

PositionerBacklashGet – Gets the backlash compensation value.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check output parameter type: ERR_WRONG_TYPE_CHAR (-13), ERR_WRONG_TYPE_DOUBLE (-14)
- Check the positioner type (must not be a secondary positioner): ERR_WRONG_OBJECT_TYPE (-8)

Description

This function returns the backlash compensation value, defined in the "stages.ini" file or defined by the "PositionerBacklashSet" function, and the backlash status ("Enable" or "Disable"). For a more thorough description of the backlash compensation, please refer to the XPS Motion Tutorial section Compensation/Backlash compensation.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerBacklashGet $SocketID $FullPositionerName BacklashValue Status

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name

Output parameters

–	BacklashValue	double	Backlash compensation value (units)
–	StatusStringBacklash status ("Enable" or "Disable")		

Return

–	Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerBacklashGet** (int SocketID, char FullPositionerName [250], double * BacklashValue, char * Status)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name

Output parameters

– BacklashValue	double *	Backlash compensation value (units)
– Status	char *	Backlash status ("Enable" or "Disable")

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerBacklashGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, BacklashValue As Double, ByVal Status As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– BacklashValue	double	Backlash compensation value (units)
– Status	string	Backlash status ("Enable" or "Disable")

Return

– Error	long	Function error code

Matlab

Prototype

[Error, BacklashValue, Status] **PositionerBacklashGet** (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– BacklashValue	double	Backlash compensation value (units)
– Status	cstring	Backlash status ("Enable" or "Disable")

Python

Prototype

[Error, BacklashValue, Status] **PositionerBacklashGet** (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code
– BacklashValue	double	Backlash compensation value (units)
– Status	string	Backlash status ("Enable" or "Disable")

2.2.4.9 PositionerBacklashSet

Name

PositionerBacklashSet – Sets the backlash compensation value.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type (must not be a secondary positioner): ERR_WRONG_OBJECT_TYPE (-8)
- Check input parameter: ERR_WRONG_TYPE_DOUBLE (-14)
- The "BacklashValue" must be positive: ERR_PARAMETER_OUT_OF_RANGE (-17)

Description

This function changes the backlash compensation value. For a more thorough description of the backlash compensation, please refer to the XPS Motion Tutorial section Compensation/Backlash compensation.

NOTE

This function can be used only if a backlash compensation is defined in the "stages.ini" file (Backlash > 0) else ERR_NOT_ALLOWED_ACTION (-22) is returned.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerBacklashSet $SocketID $FullPositionerName $BacklashValue

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– BacklashValue	double	Backlash compensation value (units)

Output parameters

– None

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerBacklashSet** (int SocketID, char FullPositionerName [250], double BacklashValue)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name
– BacklashValue	double	Backlash compensation value (units)

Output parameters

– None

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerBacklashSet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal BacklashValue As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– BacklashValue	double	Backlash compensation value (units)

Output parameters

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **PositionerBacklashSet** (int32 SocketID, cstring FullPositionerName, double BacklashValue)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name
– BacklashValue	double	Backlash compensation value (units)

Return

– Error	int32	Function error code

Python

Prototype

[Error] **PositionerBacklashSet** (integer SocketID, string FullPositionerName, double BacklashValue)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– BacklashValue	double	Backlash compensation value (units)

Return

– Error	int	Function error code

2.2.4.10 PositionerCompensatedPCOAbort

Name

PositionerCompensatedPCOAbort – Abort the CIE08 compensated PCO pulses generation.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Check the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type: ERR_WRONG_OBJECT_TYPE (-8)
- Check the position encoder ("AquadB" or "AnalogInterpolated"): ERR_UNCOMPATIBLE (-24)
- Check the CIE board supports this function: ERR_HARDWARE_FUNCTION_NOT_SUPPORTED (-115)
- Check CIE08CompensatedPCOMode = Enabled (*system.ini*) : ERR_NOT_ALLOWED_MODE_DISABLED (-121)

Description

This function aborts the CIE08 compensated PCO pulses generation. The pulses generation is stopped immediately; no more pulses will be generated even if the scanning positioner continues to move across the predefined firing positions. To stop the scanning move, use *GroupMoveAbort()* function.

NOTE

- **The function works only when the CIE08 compensated PCO mode configuration is enabled (*system.ini* : CIE08CompensatedPCOMode = Enabled).**
- **This function can be used only with a position encoder ("AquadB" or "AnalogInterpolated"), elsewhere ERR_UNCOMPATIBLE (-24) error is returned.**

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_UNCOMPATIBLE (-24)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_HARDWARE_FUNCTION_NOT_SUPPORTED (-115)
- ERR_NOT_ALLOWED_MODE_DISABLED (-121)
- SUCCESS (0): no error

TCL

Prototype

PositionerCompensatedPCOAbort $SocketID $FullPositionerName

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– None

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCompensatedPCOAbort** (int SocketID, char FullPositionerName[250])

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name

Output parameters

– None

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCompensatedPCOAbort** (ByVal SocketID As Long, ByVal FullPositionerName As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **PositionerCompensatedPCOAbort** (int32 SocketID, cstring FullPositionerName)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	cstring	Positioner name

Return

–	Error	int32	Function error code

Python

Prototype

[Error] **PositionerCompensatedPCOAbort** (integer SocketID, string FullPositionerName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name

Return

–	Error	int	Function error code

2.2.4.11 PositionerCompensatedPCOCurrentStatusGet

Name

PositionerCompensatedPCOCurrentStatusGet – Get current status of CIE08 compensated PCO pulses generation.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Check the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type: ERR_WRONG_OBJECT_TYPE (-8)
- Check the position encoder ("AquadB" or "AnalogInterpolated"): ERR_UNCOMPATIBLE (-24)
- Check the CIE board supports this function: ERR_HARDWARE_FUNCTION_NOT_SUPPORTED (-115)
- Check CIE08CompensatedPCOMode = Enabled (*system.ini*) : ERR_NOT_ALLOWED_MODE_DISABLED (-121)

Description

This function gets the current status of CIE08 compensated PCO pulses generation.

Status possible values :

0 : Pulses generation inactive (*idle, no error*)

1 : Pulses generation activated (*running*)

-1 : Pulses generation aborted with errors.

NOTE

- **The function works only when the CIE08 compensated PCO mode configuration is enabled (*system.ini* : CIE08CompensatedPCOMode = Enabled).**
- **This function can be used only with a position encoder ("AquadB" or "AnalogInterpolated"), otherwise ERR_UNCOMPATIBLE (-24) error is returned.**

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_UNCOMPATIBLE (-24)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_HARDWARE_FUNCTION_NOT_SUPPORTED (-115)
- ERR_NOT_ALLOWED_MODE_DISABLED (-121)
- SUCCESS (0): no error

TCL

Prototype

PositionerCompensatedPCOCurrentStatusGet $SocketID $FullPositionerName Status

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name

Output parameters

–	Status	int	Mode status

Return

–	Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCompensatedPCOCurrentStatusGet** (int SocketID, char FullPositionerName[250], int * Status)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	char *	Positioner name

Output parameters

–	Status	int *	Mode status

Return

–	Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCompensatedPCOCurrentStatusGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, Status As Long)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name

Output parameters

–	Status	long	Mode status

Return

–	Error	long	Function error code

Matlab

Prototype

[Error, Status] **PositionerCompensatedPCOCurrentStatusGet** (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– Status	int32	Mode status

Python

Prototype

[Error, Status] **PositionerCompensatedPCOCurrentStatusGet** (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code
– Status	int	Mode status

2.2.4.12 PositionerCompensatedPCOEnable

Name

PositionerCompensatedPCOEnable – Activate the CIE08 compensated PCO pulses generation.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Check the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type: ERR_WRONG_OBJECT_TYPE (-8)
- Check the position encoder ("AquadB" or "AnalogInterpolated"): ERR_UNCOMPATIBLE (-24)
- Check the CIE board supports this function: ERR_HARDWARE_FUNCTION_NOT_SUPPORTED (-115)
- Check CIE08CompensatedPCOMode = Enabled (*system.ini*) : ERR_NOT_ALLOWED_MODE_DISABLED (-121)
- Check if data have been prepared by PositionerCompensatedPCOPrepare() : ERR_CHECK_DATA_INCORRECT (-122)
- Check current position is out and at the good side of scanning zone (*left of scanning zone if ScanDirection positive, right of scanning zone if ScanDirection negative*) : ERR_NOT_ALLOWED_ACTION (-22)
- Check CIE08 compensated PCO mode is running : ERR_NOT_ALLOWED_ACTION (-22)

Description

This function activates the CIE08 compensated PCO pulses generation *(status becomes running (value 1))*. The pulses will be generated when the scanning positioner will move across the predefined positions. When the last pulse is generated, the CIE08 compensated PCO mode will become inactive (*status becomes inactive (value 0))*. To get status of the CIE08 compensated PCO pulses generation, use *PositionerCompensatedPCOCurrentStatusGet()* function.

Note that only the scanning positioner positions are used to fire pulses: if you prepare a set of positions at a given location and then enable the pulses generation and start the move from a different location, the pulses could be generated but their accuracy will be impacted by the mapping difference between the two locations.

This function must be used after the firing pulses data preparation with the *PositionerCompensatedPCOPrepare*(), elsewhere the function fails and the error ERR_CHECK_DATA_INCORRECT (-122) will be returned.

NOTE

The PCO pulses generation depends on the ScanVelocity and the pulse settling time (*set by PositionerPositionComparePulseParametersSet()*)

Valid settings are shown in the table below:

	PCO encoder frequency (kHz)			
Pulse settling time (µs)	25	50	125	> 500
0.075			OK	OK
1		OK	OK	
4	OK	OK		
12	OK			

How to determine the PCO encoder frequency :

- For AquadB encoder :
  PCO encoder frequency = Velocity / EncoderResolution
- For analog interpolated encoder :
  *PCO encoder frequency = Velocity * HardInterpolatorFactor / EncoderScalePitch*

Example: XML310 stage (EncoderScalePitch=0.004 mm, HardInterpolatorFactor=200).
With ScanVelocity=10mm/s => PCO encoder frequency = 10*200/0.004 = 500 kHz

NOTE

- **The function works only when the CIE08 compensated PCO mode configuration is enabled (*system.ini* : CIE08CompensatedPCOMode = Enabled).**
- **This function can be used only with a position encoder ("AquadB" or "AnalogInterpolated"), otherwise ERR_UNCOMPATIBLE (-24) error is returned.**

Error codes

- ERR_FATAL_INIT (-20)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_IN_INITIALIZATION (-21)
- ERR_UNCOMPATIBLE (-24)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_HARDWARE_FUNCTION_NOT_SUPPORTED (-115)
- ERR_NOT_ALLOWED_MODE_DISABLED (-121)
- ERR_CHECK_DATA_INCORRECT (-122)
- SUCCESS (0): no error

TCL

Prototype

PositionerCompensatedPCOEnable $SocketID $FullPositionerName

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameter

– None

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCompensatedPCOEnable** (int SocketID, char FullPositionerName[250])

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name

Output parameter

– None

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCompensatedPCOEnable** (ByVal SocketID As Long, ByVal FullPositionerName As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **PositionerCompensatedPCOEnable** (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code

Python

Prototype

[Error] **PositionerCompensatedPCOEnable** (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code

2.2.4.13 PositionerCompensatedPCOFromFile

Name

PositionerCompensatedPCOFromFile – Read firing positions from a data file to controller's memory.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Check the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type: ERR_WRONG_OBJECT_TYPE (-8)
- Check the position encoder ("AquadB" or "AnalogInterpolated"): ERR_UNCOMPATIBLE (-24)
- Check the CIE board supports this function: ERR_HARDWARE_FUNCTION_NOT_SUPPORTED (-115)
- Check CIE08CompensatedPCOMode = Enabled (*system.ini*) : ERR_NOT_ALLOWED_MODE_DISABLED (-121)
- Check data file exists : ERR_READ_FILE (-61)
- Check data from file (must be $Position_i > Position_{i-1}$, $Width_i < Position_{i+1} - Position_i$) : ERR_CHECK_DATA_INCORRECT (-122)

Description

This function reads firing positions from a data file to the controller's memory.

The data file contains lines of data, formatted as follows:

$Position_i$<Space or Tabulation>$Width_i$<CRLF or LF>

Example :

$Position_1 Width_1$

$Position_2 Width_2$

… …

$Position_N Width_N$

Data conditions : $Position_i > Position_{i-1}$, $Width_i < Position_{i+1} - Position_i$

NOTE

- **$Position_i$ (i=1..N) are the offset values relative to the scanning positioner start position that is defined in the PositionerCompensatedPCOPrepare().**
- **The function works only when the CIE08 compensated PCO mode configuration is enabled (*system.ini* : CIE08CompensatedPCOMode = Enabled).**
- **This function can be used only with a position encoder ("AquadB" or "AnalogInterpolated"), otherwise ERR_UNCOMPATIBLE (-24) error is returned.**

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_UNCOMPATIBLE (-24)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_HARDWARE_FUNCTION_NOT_SUPPORTED (-115)
- ERR_NOT_ALLOWED_MODE_DISABLED (-121)
- ERR_READ_FILE (-61)
- ERR_CHECK_DATA_INCORRECT (-122)
- SUCCESS (0): no error

TCL

Prototype

PositionerCompensatedPCOFromFile $SocketID $FullPositionerName $DataFileName

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– DataFileName	string	Data file name

Output parameter

– None

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCompensatedPCOFromFile** (int SocketID, char FullPositionerName[250] , char DataFileName[250])

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name
– DataFileName	char *	Data file name

Output parameter

– None

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCompensatedPCOFromFile** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal DataFileName As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– DataFileName	string	Data file name

Output parameter

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **PositionerCompensatedPCOFromFile** (int32 SocketID, cstring FullPositionerName, cstring DataFileName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name
– DataFileName	cstring	Data file name

Return

– Error	int32	Function error code

Python

Prototype

[Error] **PositionerCompensatedPCOFromFile** (integer SocketID, string FullPositionerName, string DataFileName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name
–	DataFileName	string	Data file name

Return

–	Error	int	Function error code

2.2.4.14 PositionerCompensatedPCOLoadToMemory

Name

PositionerCompensatedPCOLoadToMemory – Append firing positions to controller's memory..

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Check the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type: ERR_WRONG_OBJECT_TYPE (-8)
- Check the position encoder ("AquadB" or "AnalogInterpolated"): ERR_UNCOMPATIBLE (-24)
- Check the CIE board supports this function: ERR_HARDWARE_FUNCTION_NOT_SUPPORTED (-115)
- Check CIE08CompensatedPCOMode = Enabled (*system.ini*) : ERR_NOT_ALLOWED_MODE_DISABLED (-121)
- Check data lines (must be $Position_i > Position_{i-1}$, $Width_i < Position_{i+1} - Position_i$) : ERR_CHECK_DATA_INCORRECT (-122)

Description

This function appends firing positions to controller's memory from *DataLines* parameter.

To reset the controller's memory, the *PositionerCompensatedPCOMemoryReset()* function is provided.

The data line format must be :

$Position_i$<Space or Tabulation>$Width_i$<CRLF, LF or ;>

Example :

$Position_1 Width_1$

$Position_2 Width_2$

… …

$Position_N Width_N$

Or :

$Position_1 Width_1$;$Position_2 Width_2$; …;$Position_N Width_N$

Data conditions : $Position_i > Position_{i-1}$, $Width_i < Position_{i+1} - Position_i$

Example : Send PositionerCompensatedLoadToMemory (XY.X,0 0.1;1 0.1;2 0.1;3 0.1)

NOTE

- **Position$_i$ (i=1..N) are the offset values relative to the scanning positioner start position that is defined in the PositionerCompensatedPCOPrepare().**
- **The function works only when the CIE08 compensated PCO mode configuration is enabled (*system.ini* : CIE08CompensatedPCOMode = Enabled).**
- **This function can be used only with a position encoder ("AquadB" or "AnalogInterpolated"), otherwise ERR_UNCOMPATIBLE (-24) error is returned.**

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_UNCOMPATIBLE (-24)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_HARDWARE_FUNCTION_NOT_SUPPORTED (-115)
- ERR_NOT_ALLOWED_MODE_DISABLED (-121)
- ERR_CHECK_DATA_INCORRECT (-122)
- SUCCESS (0): no error

TCL

Prototype

PositionerCompensatedPCOLoadToMemory $SocketID $FullPositionerName $DataLines

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name
–	DataLines	string	Some data lines

Output parameter

– None

Return

–	Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCompensatedPCOLoadToMemory** (int SocketID, char FullPositionerName[250] , char DataLines[500])

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name
– DataLines	char *	Some data lines

Output parameters

– None

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCompensatedPCOLoadToMemory** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal DataLines As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– DataLines	string	Some data lines

Output parameter

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **PositionerCompensatedPCOLoadToMemory** (int32 SocketID, cstring FullPositionerName, cstring DataLines)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name
– DataLines	cstring	Some data lines

Return

– Error	int32	Function error code

Python

Prototype

[Error] **PositionerCompensatedPCOLoadToMemory** (integer SocketID, string FullPositionerName, string DataLines)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– DataLines	string	Some data lines

Return

– Error	int	Function error code

2.2.4.15 PositionerCompensatedPCOMemoryReset

Name

PositionerCompensatedPCOMemoryReset – Reset CIE08 compensated PCO data memory.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Check the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type: ERR_WRONG_OBJECT_TYPE (-8)
- Check the position encoder ("AquadB" or "AnalogInterpolated"): ERR_UNCOMPATIBLE (-24)
- Check the CIE board supports this function: ERR_HARDWARE_FUNCTION_NOT_SUPPORTED (-115)
- Check CIE08CompensatedPCOMode = Enabled (*system.ini*) : ERR_NOT_ALLOWED_MODE_DISABLED (-121)

Description

This function resets the CIE08 compensated PCO data memory. This function is useful to remove the data that was previously entered with the *PositionerCompensatedPCOLoadToMemory()* function.

NOTE

- **The function works only when the CIE08 compensated PCO mode configuration is enabled (*system.ini* : CIE08CompensatedPCOMode = Enabled).**
- **This function can be used only with a position encoder ("AquadB" or "AnalogInterpolated"), otherwise ERR_UNCOMPATIBLE (-24) error is returned.**

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_UNCOMPATIBLE (-24)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_HARDWARE_FUNCTION_NOT_SUPPORTED (-115)
- ERR_NOT_ALLOWED_MODE_DISABLED (-121)
- SUCCESS (0): no error

TCL

Prototype

PositionerCompensatedPCOMemoryReset $SocketID $FullPositionerName

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name

Output parameter

– None

Return

–	Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCompensatedPCOMemoryReset** (int SocketID, char FullPositionerName[250])

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	char *	Positioner name

Output parameter

– None

Return

–	Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCompensatedPCOMemoryReset** (ByVal SocketID As Long, ByVal FullPositionerName As String)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name

Output parameter

– None

Return

–	Error	long	Function error code

Matlab

Prototype

[Error] **PositionerCompensatedPCOMemoryReset** (int32 SocketID, cstring FullPositionerName)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	cstring	Positioner name

Return

–	Error	int32	Function error code

Python

Prototype

[Error] **PositionerCompensatedPCOMemoryReset** (integer SocketID, string FullPositionerName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name

Return

–	Error	int	Function error code

2.2.4.16 PositionerCompensatedPCOPrepare

Name

PositionerCompensatedPCOPrepare – Prepare data for CIE08 compensated PCO pulses generation.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Check the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type: ERR_WRONG_OBJECT_TYPE (-8)
- Check the position encoder ("AquadB" or "AnalogInterpolated"): ERR_UNCOMPATIBLE (-24)
- Check the CIE board supports this function: ERR_HARDWARE_FUNCTION_NOT_SUPPORTED (-115)
- Check CIE08CompensatedPCOMode = Enabled (*system.ini*) : ERR_NOT_ALLOWED_MODE_DISABLED (-121)
- Check data have been set, loaded or read from file to buffer (DataNumber must > 0 and < CIE08CompensatedPCOMaximumDataNumber (*system.ini)*) : ERR_CHECK_DATA_INCORRECT (-122)
- Check scanning zone exceed stage travel limits : ERR_TRAVEL_LIMITS (-35)
- Check input parameter values (ScanDirection value must equal to 1 *(positive direction)* or -1 *(negative direction)*) : ERR_PARAMETER_OUT_OF_RANGE (-17)

Description

This function calculates the firing for absolute positions*, in user's coordinate system* and converts them to firing absolute raw PCO positions, *in encoder's coordinate system*.

When mappings are enabled, the correction between user's coordinate system position and raw encoder position will be different at each location. For this reason, the prepare function must know the location (*positions of all positioners in the scanning group*) where the scan will be done.

This function must be called before the use of *PositionerCompensatedPCOEnable()* function.

Parameters :

- ScanDirection : Scan direction,(value : 1 (*positive*) or -1 (*negative*)).
- StartPosition1 : Group 1st positioner start position.
- StartPosition2 : Group 2nd positioner start position
- StartPosition3 : Group 3rd positioner start position
- …… etc ……

NOTE

- **The function works only when the CIE08 compensated PCO mode configuration is enabled (*system.ini* : CIE08CompensatedPCOMode = Enabled)..**
- **This function can be used only with a position encoder ("AquadB" or "AnalogInterpolated"), otherwise ERR_UNCOMPATIBLE (-24) error is returned.**

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_UNCOMPATIBLE (-24)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_HARDWARE_FUNCTION_NOT_SUPPORTED (-115)
- ERR_NOT_ALLOWED_MODE_DISABLED (-121)
- ERR_CHECK_DATA_INCORRECT (-122)
- ERR_TRAVEL_LIMITS (-35)
- SUCCESS (0): no error

TCL

Prototype

PositionerCompensatedPCOPrepare $SocketID $FullPositionerName $ScanDirection $StartPosition1 $StartPosition2 $StartPosition3 …

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function	
– FullPositionerName	string	Positioner name	
– ScanDirection	int	Scan direction (1 or -1)	
– StartPosition1	double	Group 1st positioner start position (units)	
– StartPosition2	double	Group 2nd positioner start position (units)	
– StartPosition3	double	Group 3rd positioner start position (units)	
–	…….	…….	…….

Output parameter

– None

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCompensatedPCOPrepare** (int SocketID, char FullPositionerName[250], int ScanDirection, double StartPosition1, double StartPosition2, double StartPosition3, …)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function	
– FullPositionerName	char *	Positioner name	
– ScanDirection	int	Scan direction (1 or -1)	
– StartPosition1	double	Group 1st positioner start position (units)	
– StartPosition2	double	Group 2nd positioner start position (units)	
– StartPosition3	double	Group 3rd positioner start position (units)	
–	…….	…….	…….

Output parameter

– None

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCompensatedPCOPrepare** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal ScanDirection As Double, ByVal StartPosition1 As Double, ByVal StartPosition2 As Double, ByVal StartPosition3 As Double, …)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function	
– FullPositionerName	string	Positioner name	
– ScanDirection	long	Scan direction (1 or -1)	
– StartPosition1	double	Group 1st positioner start position (units)	
– StartPosition2	double	Group 2nd positioner start position (units)	
– StartPosition3	double	Group 3rd positioner start position (units)	
–	…….	…….	…….

Output parameter

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **PositionerCompensatedPCOPrepare** (int32 SocketID, cstring FullPositionerName, int32 ScanDirection, double StartPosition1, double StartPosition2, double StartPosition3, …)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function	
– FullPositionerName	cstring	Positioner name	
– ScanDirection	int32	Scan direction (1 or -1)	
– StartPosition1	double	Group 1st positioner start position (units)	
– StartPosition2	double	Group 2nd positioner start position (units)	
– StartPosition3	double	Group 3rd positioner start position (units)	
–	…….	…….	…….

Return

– Error	int32	Function error code

Python

Prototype

[Error] **PositionerCompensatedPCOPrepare** (integer SocketID, string FullPositionerName, integer ScanDirection, double StartPosition1, double StartPosition2, double StartPosition3, …)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function	
– FullPositionerName	string	Positioner name	
– ScanDirection	int32	Scan direction (1 or -1)	
– StartPosition1	double	Group 1st positioner start position (units)	
– StartPosition2	double	Group 2nd positioner start position (units)	
– StartPosition3	double	Group 3rd positioner start position (units)	
–	…….	…….	…….

Return

– Error	int	Function error code

2.2.4.17 PositionerCompensatedPCOSet

Name

PositionerCompensatedPCOSet – Calculate a set of evenly spaced firing positions to the controller's memory.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Check the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type: ERR_WRONG_OBJECT_TYPE (-8)
- Check input parameter values: ERR_PARAMETER_OUT_OF_RANGE (-17)
- (Must be: Start < Stop, Distance > 0, Width > 0, Width < Distance, Width < Stop-Start)
- Check the position encoder ("AquadB" or "AnalogInterpolated"): ERR_UNCOMPATIBLE (-24)
- Check the CIE board supports this function: ERR_HARDWARE_FUNCTION_NOT_SUPPORTED (-115)
- Check CIE08CompensatedPCOMode = Enabled (*system.ini*) : ERR_NOT_ALLOWED_MODE_DISABLED (-121)
- Check DataNumber : NData = integer((Stop-Start)/Distance) + 1
- If NData > CIE08CompensatedPCOMaximumDataNumber (*system.ini*): ERR_CHECK_DATA_INCORRECT (-122)

Description

This function calculates a set of evenly spaced firing positions to the controller's memory.

Parameters :

- Start: Relative distance to the start position where PCO pulses start.
- Stop : Relative distance to the start position where PCO pulses stop.
- Distance : Step of pulses (distance between two consecutive pulses)
- Width : Width of the pulse enable signal at the firing positions.

Example : Send PositionerCompensatedPCOSet (XY.X, 0, 3, 1, 0.1)

NOTE

- **Start and Stop are the offset values relative to the scanning positioner start position that is defined in the PositionerCompensatedPCOPrepare().**
- **The function works only when the CIE08 compensated PCO mode configuration is enabled (*system.ini* : CIE08CompensatedPCOMode = Enabled).**
- **This function can be used only with a position encoder ("AquadB" or "AnalogInterpolated"), otherwise ERR_UNCOMPATIBLE (-24) error is returned.**

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_UNCOMPATIBLE (-24)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_HARDWARE_FUNCTION_NOT_SUPPORTED (-115)
- ERR_NOT_ALLOWED_MODE_DISABLED (-121)
- ERR_CHECK_DATA_INCORRECT (-122)
- SUCCESS (0): no error

TCL

Prototype

PositionerCompensatedPCOSet $SocketID $FullPositionerName $Start $Stop $Distance $Width

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– Start	double	Start position (units)
– Stop	double	Stop position (units)
– Distance	double	Distance between two consecutive pulses (units)
– Width	double	Width of pulse enable signal (units)

Output parameter

– None

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCompensatedPCOSet** (int SocketID, char FullPositionerName[250] , double Start, double Stop, double Distance, double Width)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name
– Start	double	Start position (units)
– Stop	double	Stop position (units)
– Distance	double	Distance between two consecutive pulses (units)
– Width	double	Width of pulse enable signal (units)

Output parameter

– None

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCompensatedPCOSet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal Start As Double, ByVal Stop As Double, ByVal Distance As Double, ByVal Width As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– Start	double	Start position (units)
– Stop	double	Stop position (units)
– Distance	double	Distance between two consecutive pulses (units)
– Width	double	Width of pulse enable signal (units)

Output parameter

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **PositionerCompensatedPCOSet** (int32 SocketID, cstring FullPositionerName, double Start, double Stop, double Distance, double Width)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name
– Start	double	Start position (units)
– Stop	double	Stop position (units)
– Distance	double	Distance between two consecutive pulses (units)
– Width	double	Width of pulse enable signal (units)

Return

– Error	int32	Function error code

Python

Prototype

[Error] **PositionerCompensatedPCOSet** (integer SocketID, string FullPositionerName, double Start, double Stop, double Distance, double Width)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– Start	double	Start position (units)
– Stop	double	Stop position (units)
– Distance	double	Distance between two consecutive pulses (units)
– Width	double	Width of pulse enable signal (units)

Return

– Error	int	Function error code

2.2.4.18 PositionerCompensationFrequencyNotchsGet

Name

PositionerCompensationFrequencyNotchsGet – Gets pre-feedforward compensation notch filters parameters.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)
- Check corrector type: ERR_WRONG_OBJECT_TYPE (-8)

Description

This functions returns the *CompensationSystemPreFeedForward* frequency notch filters parameters. These notch filters allow the user to reduce external perturbations such as base motion or floor vibrations. Note that the *CompensationSystemPreFeedForward* feature is available for all corrector types (acceleration, velocity, voltage or position) functioning *in closed loop configuration*.

- NotchFrequency1
- NotchsBandwidth1
- NotchsGain1
- NotchFrequency2
- NotchsBandwidth2
- NotchsGain2
- NotchFrequency3
- NotchsBandwidth3
- NotchsGain3

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerCompensationFrequencyNotchsGet $SocketID $FullPositionerName NotchFrequency1 NotchBandwidth1 NotchGain1 NotchFrequency2 NotchBandwidth2 NotchGain2 NotchFrequency3 NotchBandwidth3 NotchGain3

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– NotchFrequency1	double	Notch frequency for filter #1 (Hz)
– NotchBandwidth1	double	Notch bandwidth for filter #1 (Hz)
– NotchGain1	double	Notch gain for filter #1
– NotchFrequency2	double	Notch frequency for filter #2 (Hz)
– NotchBandwidth2	double	Notch bandwidth for filter #2 (Hz)
– NotchGain2	double	Notch gain for filter #2
– NotchFrequency3	double	Notch frequency for filter #3 (Hz)
– NotchBandwidth3	double	Notch bandwidth for filter #3 (Hz)
– NotchGain3	double	Notch gain for filter #3

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCompensationFrequencyNotchsGet** (int SocketID, char FullPositionerName [250], double* NotchFrequency1, double* NotchBandwidth1, double* NotchGain1, double* NotchFrequency2, double* NotchBandwidth2, double* NotchGain2, double* NotchFrequency3, double* NotchBandwidth3, double* NotchGain3)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name

Output parameters

– NotchFrequency1	double *	Notch frequency for filter #1 (Hz)
– NotchBandwidth1	double *	Notch bandwidth for filter #1 (Hz)
– NotchGain1	double *	Notch gain for filter #1
– NotchFrequency2	double *	Notch frequency for filter #2 (Hz)
– NotchBandwidth2	double *	Notch bandwidth for filter #2 (Hz)
– NotchGain2	double *	Notch gain for filter #2
– NotchFrequency3	double *	Notch frequency for filter #3 (Hz)
– NotchBandwidth3	double *	Notch bandwidth for filter #3 (Hz)
– NotchGain3	double *	Notch gain for filter #3

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCompensationFrequencyNotchsGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, NotchFrequency1 As Double, NotchBandwidth1 As Double, NotchGain1 As Double, NotchFrequency2 As Double, NotchBandwidth2 As Double, NotchGain2 As Double, NotchFrequency3 As Double, NotchBandwidth3 As Double, NotchGain3 As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– NotchFrequency1	double	Notch frequency for filter #1 (Hz)
– NotchBandwidth1	double	Notch bandwidth for filter #1 (Hz)
– NotchGain1	double	Notch gain for filter #1
– NotchFrequency2	double	Notch frequency for filter #2 (Hz)
– NotchBandwidth2	double	Notch bandwidth for filter #2 (Hz)
– NotchGain2	double	Notch gain for filter #2
– NotchFrequency3	double	Notch frequency for filter #3 (Hz)
– NotchBandwidth3	double	Notch bandwidth for filter #3 (Hz)
– NotchGain3	double	Notch gain for filter #3

Return

– Error	long	Function error code

Matlab

Prototype

[Error, NotchFrequency1, NotchBandwidth1, NotchGain1, NotchFrequency2, NotchBandwidth2, NotchGain2, NotchFrequency3, NotchBandwidth3, NotchGain3] **PositionerCompensationFrequencyNotchsGet** (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– NotchFrequency1	double	Notch frequency for filter #1 (Hz)
– NotchBandwidth1	double	Notch bandwidth for filter #1 (Hz)
– NotchGain1	double	Notch gain for filter #1
– NotchFrequency2	double	Notch frequency for filter #2 (Hz)
– NotchBandwidth2	double	Notch bandwidth for filter #2 (Hz)
– NotchGain2	double	Notch gain for filter #2
– NotchFrequency3	double	Notch frequency for filter #3 (Hz)
– NotchBandwidth3	double	Notch bandwidth for filter #3 (Hz)
– NotchGain3	double	Notch gain for filter #3

Python

Prototype

[Error, NotchFrequency1, NotchBandwidth1, NotchGain1, NotchFrequency2, NotchBandwidth2, NotchGain2, NotchFrequency3, NotchBandwidth3, NotchGain3] **PositionerCompensationFrequencyNotchsGet** (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code
– NotchFrequency1	double	Notch frequency for filter #1 (Hz)
– NotchBandwidth1	double	Notch bandwidth for filter #1 (Hz)
– NotchGain1	double	Notch gain for filter #1
– NotchFrequency2	double	Notch frequency for filter #2 (Hz)
– NotchBandwidth2	double	Notch bandwidth for filter #2 (Hz)
– NotchGain2	double	Notch gain for filter #2
– NotchFrequency3	double	Notch frequency for filter #3 (Hz)
– NotchBandwidth3	double	Notch bandwidth for filter #3 (Hz)
– NotchGain3	double	Notch gain for filter #3

2.2.4.19 PositionerCompensationFrequencyNotchsSet

Name

PositionerCompensationFrequencyNotchsSet – Sets pre-feedforward compensation notch filters parameters.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)
- Check correcor type: ERR_WRONG_OBJECT_TYPE (-8)
- Check parameter values: ERR_PARAMETER_OUT_OF_RANGE (-17)
- NotchFrequency $\in \left[0 : \frac{0.5}{\text{CorrectorISRPeriod}} \right]$
- NotchBandwidth $\in \left[0 : \frac{0.5}{\text{CorrectorISRPeriod}} \right]$

NOTE

Refer to *system.ref* file to get CorrectorISRPeriod value.

This function can be used only with the XPS-Q8 Precision Platform controller.

Description

This functions sets the *CompensationSystemPreFeedForward* frequency notch filters parameters. These notch filters all the user to reduce external perturbations such as base motion or floor vibrations. Note that the *CompensationSystemPreFeedForward* feature is available for all corrector types (acceleration, velocity, voltage or position) functioning *in closed loop configuration*.

NotchFrequency1

NotchsBandwidth1

NotchsGain1

NotchFrequency2

NotchsBandwidth2

NotchsGain2

NotchFrequency3

NotchsBandwidth3

NotchsGain3

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerCompensationFrequencyNotchsSet $SocketID $FullPositionerName $NotchFrequency1 $NotchBandwidth1 $NotchGain1 $NotchFrequency2 $NotchBandwidth2 $NotchGain2 $NotchFrequency3 $NotchBandwidth3 $NotchGain3

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– NotchFrequency1	double	Notch frequency for filter #1 (Hz)
– NotchBandwidth1	double	Notch bandwidth for filter #1 (Hz)
– NotchGain1	double	Notch gain for filter #1
– NotchFrequency2	double	Notch frequency for filter #2 (Hz)
– NotchBandwidth2	double	Notch bandwidth for filter #2 (Hz)
– NotchGain2	double	Notch gain for filter #2
– NotchFrequency3	double	Notch frequency for filter #3 (Hz)
– NotchBandwidth3	double	Notch bandwidth for filter #3 (Hz)
– NotchGain3	double	Notch gain for filter #3

Output parameters

– None

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCompensationFrequencyNotchsSet** (int SocketID, char FullPositionerName [250], double NotchFrequency1, double NotchBandwidth1, double NotchGain1, double NotchFrequency2, double NotchBandwidth2, double NotchGain2, double NotchFrequency3, double NotchBandwidth3, double NotchGain3)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name
– NotchFrequency1	double	Notch frequency for filter #1 (Hz)
– NotchBandwidth1	double	Notch bandwidth for filter #1 (Hz)
– NotchGain1	double	Notch gain for filter #1
– NotchFrequency2	double	Notch frequency for filter #2 (Hz)
– NotchBandwidth2	double	Notch bandwidth for filter #2 (Hz)
– NotchGain2	double	Notch gain for filter #2
– NotchFrequency3	double	Notch frequency for filter #3 (Hz)
– NotchBandwidth3	double	Notch bandwidth for filter #3 (Hz)
– NotchGain3	double	Notch gain for filter #3

Output parameters

– None

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCompensationFrequencyNotchsSet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal NotchFrequency1 As Double, ByVal NotchBandwidth1 As Double, ByVal NotchGain1 As Double, ByVal NotchFrequency2 As Double, ByVal NotchBandwidth2 As Double, ByVal NotchGain2 As Double, ByVal NotchFrequency3 As Double, ByVal NotchBandwidth3 As Double, ByVal NotchGain3 As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– NotchFrequency1	double	Notch frequency for filter #1 (Hz)
– NotchBandwidth1	double	Notch bandwidth for filter #1 (Hz)
– NotchGain1	double	Notch gain for filter #1
– NotchFrequency2	double	Notch frequency for filter #2 (Hz)
– NotchBandwidth2	double	Notch bandwidth for filter #2 (Hz)
– NotchGain2	double	Notch gain for filter #2
– NotchFrequency3	double	Notch frequency for filter #3 (Hz)
– NotchBandwidth3	double	Notch bandwidth for filter #3 (Hz)
– NotchGain3	double	Notch gain for filter #3

Output parameters

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **PositionerCompensationFrequencyNotchsSet** (int32 SocketID, cstring FullPositionerName, double NotchFrequency1, double NotchBandwidth1, double NotchGain1, double NotchFrequency2, double NotchBandwidth2, double NotchGain2, double NotchFrequency3, double NotchBandwidth3, double NotchGain3)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name
– NotchFrequency1	double	Notch frequency for filter #1 (Hz)
– NotchBandwidth1	double	Notch bandwidth for filter #1 (Hz)
– NotchGain1	double	Notch gain for filter #1
– NotchFrequency2	double	Notch frequency for filter #2 (Hz)
– NotchBandwidth2	double	Notch bandwidth for filter #2 (Hz)
– NotchGain2	double	Notch gain for filter #2
– NotchFrequency3	double	Notch frequency for filter #3 (Hz)
– NotchBandwidth3	double	Notch bandwidth for filter #3 (Hz)
– NotchGain3	double	Notch gain for filter #3

Return

– Error	int32	Function error code

Python

Prototype

[Error] **PositionerCompensationFrequencyNotchsSet** (integer SocketID, string FullPositionerName, double NotchFrequency1, double NotchBandwidth1, double NotchGain1, double NotchFrequency2, double NotchBandwidth2, double NotchGain2, double NotchFrequency3, double NotchBandwidth3, double NotchGain3)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– NotchFrequency1	double	Notch frequency for filter #1 (Hz)
– NotchBandwidth1	double	Notch bandwidth for filter #1 (Hz)
– NotchGain1	double	Notch gain for filter #1
– NotchFrequency2	double	Notch frequency for filter #2 (Hz)
– NotchBandwidth2	double	Notch bandwidth for filter #2 (Hz)
– NotchGain2	double	Notch gain for filter #2
– NotchFrequency3	double	Notch frequency for filter #3 (Hz)
– NotchBandwidth3	double	Notch bandwidth for filter #3 (Hz)
– NotchGain3	double	Notch gain for filter #3

Return

– Error	int	Function error code

2.2.4.20 PositionerCompensationLowPassTwoFilterGet

Name

PositionerCompensationLowPassTwoFilterGet – Gets the post-feedforward compensation second order lowpass filter parameters.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)
- Check corrector type: ERR_WRONG_OBJECT_TYPE (-8)

Description

This functions returns the system compensation parameters defined for the post-feedforward compensation second order low-pass filter.

CutOffFrequency

NOTE

This function can be used only with the XPS-Q8 Precision Platform controller.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerCompensationLowPassTwoFilterGet $SocketID $FullPositionerName CutOffFrequency

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name

Output parameters

–	CutOffFrequency	double	Second order filter cut-off frequency (Herz)

Return

–	Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCompensationLowPassTwoFilterGet** (int SocketID, char FullPositionerName [250], double* CutOffFrequency)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	char *	Positioner name

Output parameters

–	CutOffFrequency	double *	Second order filter cut-off frequency (Herz)

Return

–	Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCompensationLowPassTwoFilterGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, CutOffFrequency As Double)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name

Output parameters

–	CutOffFrequency	double	Second order filter cut-off frequency (Herz)

Return

–	Error	long	Function error code

Matlab

Prototype

[Error, CutOffFrequency] **PositionerCompensationLowPassTwoFilterGet** (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– CutOffFrequency	double	Second order filter cut-off frequency (Herz)

Python

Prototype

[Error, CutOffFrequency] **PositionerCompensationLowPassTwoFilterGet** (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code
– CutOffFrequency	double	Second order filter cut-off frequency (Herz)

2.2.4.21 PositionerCompensationLowPassTwoFilterSet

Name

PositionerCompensationLowPassTwoFilterSet – Sets the post-feedforward compensation second order lowpass filter parameters.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)
- Check correcor type: ERR_WRONG_OBJECT_TYPE (-8)
- Check parameter values: ERR_PARAMETER_OUT_OF_RANGE (-17)
- CutOffFrequency $\in \left[0 : \frac{0.5}{\text{CorrectorISRPeriod}} \right]$

NOTE

Refer to *system.ref* file to get CorrectorISRPeriod value.

Description

This function configures the parameters defined for the post-feedforward compensation second order low-pass filter.

CutOffFrequency

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

If the "CutOffFrequency" value = 0 then the second order low-pass filter is not activated.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerCompensationLowPassTwoFilterSet $SocketID $FullPositionerName $CutOffFrequency

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– CutOffFrequency	double	Second order filter cut-off frequency (Herz)

Output parameter

– None

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCompensationLowPassTwoFilterSet** (int SocketID, char FullPositionerName [250], double CutOffFrequency)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name
– CutOffFrequency	double	Second order filter cut-off frequency (Herz)

Output parameter

– None

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCompensationLowPassTwoFilterSet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal CutOffFrequency As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– CutOffFrequency	double	Second order filter cut-off frequency (Herz)

Output parameters

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **PositionerCompensationLowPassTwoFilterSet** (int32 SocketID, cstring FullPositionerName, double CutOffFrequency)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name
– CutOffFrequency	double	Second order filter cut-off frequency (Herz)

Return

– Error	int32	Function error code

Python

Prototype

[Error] **PositionerCompensationLowPassTwoFilterSet** (integer SocketID, string FullPositionerName, double CutOffFrequency)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– CutOffFrequency	double	Second order filter cut-off frequency (Herz)

Return

– Error	int	Function error code

2.2.4.22 PositionerCompensationNotchModeFiltersGet

Name

PositionerCompensationNotchModeFiltersGet – Gets the post-feedforward compensation notch mode filter parameters.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)
- Check corrector type: ERR_WRONG_OBJECT_TYPE (-8)

Description

This functions returns the system compensation parameters defined for two post-feedforward compensation notch mode filters.

First notch mode filter parameters:

- NotchModeFr1
- NotchModeFa1
- NotchModeZr1
- NotchModeZa1

Second notch mode filter parameters:

- NotchModeFr2
- NotchModeFa2
- NotchModeZr2
- NotchModeZa2

NOTE

This function can be used only with the XPS-Q8 Precision Platform controller.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerCompensationNotchModeFiltersGet $SocketID $FullPositionerName NotchModeFr1 NotchModeFa1 NotchModeZr1 NotchModeZa1 NotchModeFr2 NotchModeFa2 NotchModeZr2 NotchModeZa2

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– NotchModeFr1	double	Resonance frequency (Herz) for notch mode filter #1
– NotchModeFa1	double	Anti-resonance frequency (Herz) for notch mode filter #1
– NotchModeZr1	double	Resonance damping factor for notch mode filter #1
– NotchModeZa1	double	Anti-resonance damping factor for notch mode filter #1
– NotchModeFr2	double	Resonance frequency (Herz) for notch mode filter #2
– NotchModeFa2	double	Anti-resonance frequency (Herz) for notch mode filter #2
– NotchModeZr2	double	Resonance damping factor for notch mode filter #2
– NotchModeZa2	double	Anti-resonance damping factor for notch mode filter #2

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCompensationNotchModeFiltersGet** (int SocketID, char FullPositionerName [250], double* NotchModeFr1,double* NotchModeFa1, double* NotchModeZr1, double* NotchModeZa1, double* NotchModeFr2,double* NotchModeFa2, double* NotchModeZr2, double* NotchModeZa2)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name

Output parameters

– NotchModeFr1	double *	Resonance frequency (Herz) for notch mode filter #1
– NotchModeFa1	double *	Anti-resonance frequency (Herz) for notch mode filter #1
– NotchModeZr1	double *	Resonance damping factor for notch mode filter #1
– NotchModeZa1	double *	Anti-resonance damping factor for notch mode filter #1
– NotchModeFr2	double *	Resonance frequency (Herz) for notch mode filter #2
– NotchModeFa2	double *	Anti-resonance frequency (Herz) for notch mode filter #2
– NotchModeZr2	double *	Resonance damping factor for notch mode filter #2
– NotchModeZa2	double *	Anti-resonance damping factor for notch mode filter #2

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCompensationNotchModeFiltersGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, NotchModeFr1 As Double, NotchModeFa1 As Double, NotchModeZr1 As Double, NotchModeZa1 As Double, NotchModeFr2 As Double, NotchModeFa2 As Double, NotchModeZr2 As Double, NotchModeZa2 As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– NotchModeFr1	double	Resonance frequency (Herz) for notch mode filter #1
– NotchModeFa1	double	Anti-resonance frequency (Herz) for notch mode filter #1
– NotchModeZr1	double	Resonance damping factor for notch mode filter #1
– NotchModeZa1	double	Anti-resonance damping factor for notch mode filter #1
– NotchModeFr2	double	Resonance frequency (Herz) for notch mode filter #2
– NotchModeFa2	double	Anti-resonance frequency (Herz) for notch mode filter #2
– NotchModeZr2	double	Resonance damping factor for notch mode filter #2
– NotchModeZa2	double	Anti-resonance damping factor for notch mode filter #2

Return

– Error	long	Function error code

Matlab

Prototype

[Error, NotchModeFr1, NotchModeFa1, NotchModeZr1, NotchModeZa1, NotchModeFr2, NotchModeFa2, NotchModeZr2, NotchModeZa2] **PositionerCompensationNotchModeFiltersGet** (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– NotchModeFr1	double	Resonance frequency (Herz) for notch mode filter #1
– NotchModeFa1	double	Anti-resonance frequency (Herz) for notch mode filter #1
– NotchModeZr1	double	Resonance damping factor for notch mode filter #1
– NotchModeZa1	double	Anti-resonance damping factor for notch mode filter #1
– NotchModeFr2	double	Resonance frequency (Herz) for notch mode filter #2
– NotchModeFa2	double	Anti-resonance frequency (Herz) for notch mode filter #2
– NotchModeZr2	double	Resonance damping factor for notch mode filter #2
– NotchModeZa2	double	Anti-resonance damping factor for notch mode filter #2

Python

Prototype

[Error, NotchModeFr1, NotchModeFa1, NotchModeZr1, NotchModeZa1, NotchModeFr2, NotchModeFa2, NotchModeZr2, NotchModeZa2] **PositionerCompensationNotchModeFiltersGet** (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code
– NotchModeFr1	double	Resonance frequency (Herz) for notch mode filter #1
– NotchModeFa1	double	Anti-resonance frequency (Herz) for notch mode filter #1
– NotchModeZr1	double	Resonance damping factor for notch mode filter #1
– NotchModeZa1	double	Anti-resonance damping factor for notch mode filter #1
– NotchModeFr2	double	Resonance frequency (Herz) for notch mode filter #2
– NotchModeFa2	double	Anti-resonance frequency (Herz) for notch mode filter #2
– NotchModeZr2	double	Resonance damping factor for notch mode filter #2
– NotchModeZa2	double	Anti-resonance damping factor for notch mode filter #2

2.2.4.23 PositionerCompensationNotchModeFiltersSet

Name

PositionerCompensationNotchModeFiltersSet – Sets the post-feedforward compensation notch mode filter parameters.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)
- Check corrector type: ERR_WRONG_OBJECT_TYPE (-8)
- Check parameter values: ERR_PARAMETER_OUT_OF_RANGE (-17)
- NotchModeFr $\in \left[0 : \frac{0.5}{\text{CorrectorISRPeriod}} \right]$
- NotchModeFa $\in \left[0 : \frac{0.5}{\text{CorrectorISRPeriod}} \right]$

NOTE

This function can be used only with the XPS-Q8 Precision Platform controller.

Refer to *system.ref* file to get CorrectorISRPeriod value.

Description

This functions configures the parameters defined for two post-feedforward compensation notch mode filters.

First notch mode filter parameters:

- NotchModeFr1
- NotchModeFa1
- NotchModeZr1
- NotchModeZa1

Second notch mode filter parameters:

- NotchModeFr2
- NotchModeFa2
- NotchModeZr2
- NotchModeZa2

NOTE

If the "NotchModeFr" value = 0 or the "NotchModeFa" value = 0, then the notch mode filter is not activated.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerCompensationNotchModeFiltersSet $SocketID $FullPositionerName $NotchModeFr1 $NotchModeFa1 $NotchModeZr1 $NotchModeZa1 $NotchModeFr2 $NotchModeFa2 $NotchModeZr2 $NotchModeZa2

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– NotchModeFr1	double	Resonance frequency (Herz) for notch mode filter #1
– NotchModeFa1	double	Anti-resonance frequency (Herz) for notch mode filter #1
– NotchModeZr1	double	Resonance damping factor for notch mode filter #1
– NotchModeZa1	double	Anti-resonance damping factor for notch mode filter #1
– NotchModeFr2	double	Resonance frequency (Herz) for notch mode filter #2
– NotchModeFa2	double	Anti-resonance frequency (Herz) for notch mode filter #2
– NotchModeZr2	double	Resonance damping factor for notch mode filter #2
– NotchModeZa2	double	Anti-resonance damping factor for notch mode filter #2

Output parameter

– None

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCompensationNotchModeFiltersSet** (int SocketID, char FullPositionerName [250], double NotchModeFr1,double NotchModeFa1, double NotchModeZr1, double NotchModeZa1, double NotchModeFr2,double NotchModeFa2, double NotchModeZr2, double NotchModeZa2)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name
– NotchModeFr1	double	Resonance frequency (Herz) for notch mode filter #1
– NotchModeFa1	double	Anti-resonance frequency (Herz) for notch mode filter #1
– NotchModeZr1	double	Resonance damping factor for notch mode filter #1
– NotchModeZa1	double	Anti-resonance damping factor for notch mode filter #1
– NotchModeFr2	double	Resonance frequency (Herz) for notch mode filter #2
– NotchModeFa2	double	Anti-resonance frequency (Herz) for notch mode filter #2
– NotchModeZr2	double	Resonance damping factor for notch mode filter #2
– NotchModeZa2	double	Anti-resonance damping factor for notch mode filter #2

Output parameter

– None

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCompensationNotchModeFiltersSet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal NotchModeFr1 As Double, ByVal NotchModeFa1 As Double, ByVal NotchModeZr1 As Double, ByVal NotchModeZa1 As Double, ByVal NotchModeF2 As Double, ByVal NotchModeFa2 As Double, ByVal NotchModeZr2 As Double, ByVal NotchModeZa2 As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– NotchModeFr1	double	Resonance frequency (Herz) for notch mode filter #1
– NotchModeFa1	double	Anti-resonance frequency (Herz) for notch mode filter #1
– NotchModeZr1	double	Resonance damping factor for notch mode filter #1
– NotchModeZa1	double	Anti-resonance damping factor for notch mode filter #1
– NotchModeFr2	double	Resonance frequency (Herz) for notch mode filter #2
– NotchModeFa2	double	Anti-resonance frequency (Herz) for notch mode filter #2
– NotchModeZr2	double	Resonance damping factor for notch mode filter #2
– NotchModeZa2	double	Anti-resonance damping factor for notch mode filter #2

Output parameter

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **PositionerCompensationNotchModeFiltersSet** (int32 SocketID, cstring FullPositionerName, double NotchFrequency1, double NotchBandwith1, double NotchGain1, double NotchFrequency2, double NotchBandwith2, double NotchGain2)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name
– NotchModeFr1	double	Resonance frequency (Herz) for notch mode filter #1
– NotchModeFa1	double	Anti-resonance frequency (Herz) for notch mode filter #1
– NotchModeZr1	double	Resonance damping factor for notch mode filter #1
– NotchModeZa1	double	Anti-resonance damping factor for notch mode filter #1
– NotchModeFr2	double	Resonance frequency (Herz) for notch mode filter #2
– NotchModeFa2	double	Anti-resonance frequency (Herz) for notch mode filter #2
– NotchModeZr2	double	Resonance damping factor for notch mode filter #2
– NotchModeZa2	double	Anti-resonance damping factor for notch mode filter #2

Return

– Error	int32	Function error code

Python

Prototype

[Error] **PositionerCompensationNotchModeFiltersSet** (integer SocketID, string FullPositionerName, double NotchFrequency1, double NotchBandwith1, double NotchGain1, double NotchFrequency2, double NotchBandwith2, double NotchGain2)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– NotchModeFr1	double	Resonance frequency (Herz) for notch mode filter #1
– NotchModeFa1	double	Anti-resonance frequency (Herz) for notch mode filter #1
– NotchModeZr1	double	Resonance damping factor for notch mode filter #1
– NotchModeZa1	double	Anti-resonance damping factor for notch mode filter #1
– NotchModeFr2	double	Resonance frequency (Herz) for notch mode filter #2
– NotchModeFa2	double	Anti-resonance frequency (Herz) for notch mode filter #2
– NotchModeZr2	double	Resonance damping factor for notch mode filter #2
– NotchModeZa2	double	Anti-resonance damping factor for notch mode filter #2

Return

– Error	int	Function error code

2.2.4.24 PositionerCompensationPhaseCorrectionFiltersGet

Name

PositionerCompensationPhaseCorrectionFiltersGet – Gets the post-feedforward compensation phase correction filter parameters.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)
- Check corrector type: ERR_WRONG_OBJECT_TYPE (-8)

Description

This function returns the system compensation parameters defined for two post-feedforward compensation phase correction filters.

First phase correction filter parameters:

- PhaseCorrectionFn1
- PhaseCorrectionFd1
- PhaseCorrectionGain1

Second phase correction filter parameters:

- PhaseCorrectionFn2
- PhaseCorrectionFd2
- PhaseCorrectionGain2

NOTE

This function can be used only with the XPS-Q8 Precision Platform controller.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerCompensationPhaseCorrectionFiltersGet $SocketID $FullPositionerName PhaseCorrectionFn1 PhaseCorrectionFd1 PhaseCorrectionGain1 PhaseCorrectionFn2 PhaseCorrectionFd2 PhaseCorrectionGain2

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– PhaseCorrectionFn1	double	Numerator frequency (Herz) for phase correction filter #1
– PhaseCorrectionFd1	double	Denominator frequency (Herz) for phase correction filter #1
– PhaseCorrectionGain1	double	Gain for phase correction filter #1
– PhaseCorrectionFn2	double	Numerator frequency (Herz) for phase correction filter #2
– PhaseCorrectionFd2	double	Denominator frequency (Herz) for phase correction filter #2
– PhaseCorrectionGain2	double	Gain for phase correction filter #2

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCompensationPhaseCorrectionFiltersGet** (int SocketID, char FullPositionerName [250], double* PhaseCorrectionFn1,double* PhaseCorrectionFd1, double* PhaseCorrectionGain1, double* PhaseCorrectionFn2,double* PhaseCorrectionFd2, double* PhaseCorrectionGain2)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name

Output parameters

– PhaseCorrectionFn1	double *	Numerator frequency (Herz) for phase correction filter #1
– PhaseCorrectionFd1	double *	Denominator frequency (Herz) for phase correction filter #1
– PhaseCorrectionGain1	double *	Gain for phase correction filter #1
– PhaseCorrectionFn2	double *	Numerator frequency (Herz) for phase correction filter #2
– PhaseCorrectionFd2	double *	Denominator frequency (Herz) for phase correction filter #2
– PhaseCorrectionGain2	double *	Gain for phase correction filter #2

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCompensationPhaseCorrectionFiltersGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, PhaseCorrectionFn1 As Double, PhaseCorrectionFd1 As Double, PhaseCorrectionGain1 As Double, PhaseCorrectionFn2 As Double, PhaseCorrectionFd2 As Double, PhaseCorrectionGain2 As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– PhaseCorrectionFn1	double	Numerator frequency (Herz) for phase correction filter #1
– PhaseCorrectionFd1	double	Denominator frequency (Herz) for phase correction filter #1
– PhaseCorrectionGain1	double	Gain for phase correction filter #1
– PhaseCorrectionFn2	double	Numerator frequency (Herz) for phase correction filter #2
– PhaseCorrectionFd2	double	Denominator frequency (Herz) for phase correction filter #2
– PhaseCorrectionGain2	double	Gain for phase correction filter #2

Return

– Error	long	Function error code

Matlab

Prototype

[Error, PhaseCorrectionFn1, PhaseCorrectionFd1, PhaseCorrectionGain1, PhaseCorrectionFn2, PhaseCorrectionFd2, PhaseCorrectionGain2]
PositionerCompensationPhaseCorrectionFiltersGet (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– PhaseCorrectionFn1	double	Numerator frequency (Herz) for phase correction filter #1
– PhaseCorrectionFd1	double	Denominator frequency (Herz) for phase correction filter #1
– PhaseCorrectionGain1	double	Gain for phase correction filter #1
– PhaseCorrectionFn2	double	Numerator frequency (Herz) for phase correction filter #2
– PhaseCorrectionFd2	double	Denominator frequency (Herz) for phase correction filter #2
– PhaseCorrectionGain2	double	Gain for phase correction filter #2

Python

Prototype

[Error, PhaseCorrectionFn1, PhaseCorrectionFd1, PhaseCorrectionGain1, PhaseCorrectionFn2, PhaseCorrectionFd2, PhaseCorrectionGain2]
PositionerCompensationPhaseCorrectionFiltersGet (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code
– PhaseCorrectionFn1	double	Numerator frequency (Herz) for phase correction filter #1
– PhaseCorrectionFd1	double	Denominator frequency (Herz) for phase correction filter #1
– PhaseCorrectionGain1	double	Gain for phase correction filter #1
– PhaseCorrectionFn2	double	Numerator frequency (Herz) for phase correction filter #2
– PhaseCorrectionFd2	double	Denominator frequency (Herz) for phase correction filter #2
– PhaseCorrectionGain2	double	Gain for phase correction filter #2

2.2.4.25 PositionerCompensationPhaseCorrectionFiltersSet

Name

PositionerCompensationPhaseCorrectionFiltersSet – Sets the post-feedforward compensation phase correction filter parameters.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)
- Check corrector type: ERR_WRONG_OBJECT_TYPE (-8)
- Check parameter values: ERR_PARAMETER_OUT_OF_RANGE (-17)
- PhaseCorrectionFn $\in \left[0 : \frac{0.5}{\text{CorrectorISRPeriod}} \right]$
- PhaseCorrectionFd $\in \left[0 : \frac{0.5}{\text{CorrectorISRPeriod}} \right]$

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

Refer to *system.ref* file to get CorrectorISRPeriod value.

Description

This functions configures the parameters defined for two post-feedforward compensation phase correction filters.

First phase correction filter parameters:

- PhaseCorrectionFn1
- PhaseCorrectionFd1
- PhaseCorrectionGain1

Second phase correction filter parameters:

- PhaseCorrectionFn2
- PhaseCorrectionFd2
- PhaseCorrectionGain2

NOTE

If the "PhaseCorrectionFn" value = 0 or the "PhaseCorrectionFd" value = 0 then the phase correction filter is not activated.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerCompensationPhaseCorrectionFiltersSet $SocketID $FullPositionerName $PhaseCorrectionFn1 $PhaseCorrectionFd1 $PhaseCorrectionGain1 $PhaseCorrectionFn2 $PhaseCorrectionFd2 $PhaseCorrectionGain2

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– PhaseCorrectionFn1	double	Numerator frequency (Herz) for phase correction filter #1
– PhaseCorrectionFd1	double	Denominator frequency (Herz) for phase correction filter #1
– PhaseCorrectionGain1	double	Gain for phase correction filter #1
– PhaseCorrectionFn2	double	Numerator frequency (Herz) for phase correction filter #2
– PhaseCorrectionFd2	double	Denominator frequency (Herz) for phase correction filter #2
– PhaseCorrectionGain2	double	Gain for phase correction filter #2

Output parameter

– None

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCompensationPhaseCorrectionFiltersSet** (int SocketID, char FullPositionerName [250], double PhaseCorrectionFn1,double PhaseCorrectionFd1, double PhaseCorrectionGain1, double PhaseCorrectionFn2,double PhaseCorrectionFd2, double PhaseCorrectionGain2)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	char *	Positioner name
–	PhaseCorrectionFn1	double	Numerator frequency (Herz) for phase correction filter #1
–	PhaseCorrectionFd1	double	Denominator frequency (Herz) for phase correction filter #1
–	PhaseCorrectionGain1	double	Gain for phase correction filter #1
–	PhaseCorrectionFn2	double	Numerator frequency (Herz) for phase correction filter #2
–	PhaseCorrectionFd2	double	Denominator frequency (Herz) for phase correction filter #2
–	PhaseCorrectionGain2	double	Gain for phase correction filter #2

Output parameter

– None

Return

–	Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCompensationPhaseCorrectionFiltersSet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal PhaseCorrectionFn1 As Double, ByVal PhaseCorrectionFd1 As Double, ByVal PhaseCorrectionGain1 As Double, ByVal PhaseCorrectionFn2 As Double, ByVal PhaseCorrectionFd2 As Double, ByVal PhaseCorrectionGain2 As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– PhaseCorrectionFn1	double	Numerator frequency (Herz) for phase correction filter #1
– PhaseCorrectionFd1	double	Denominator frequency (Herz) for phase correction filter #1
– PhaseCorrectionGain1	double	Gain for phase correction filter #1
– PhaseCorrectionFn2	double	Numerator frequency (Herz) for phase correction filter #2
– PhaseCorrectionFd2	double	Denominator frequency (Herz) for phase correction filter #2
– PhaseCorrectionGain2	double	Gain for phase correction filter #2

Output parameter

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **PositionerCompensationPhaseCorrectionFiltersSet** (int32 SocketID, cstring FullPositionerName, double PhaseCorrectionFn1,double PhaseCorrectionFd1, double PhaseCorrectionGain1, double PhaseCorrectionFn2,double PhaseCorrectionFd2, double PhaseCorrectionGain2)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name
– PhaseCorrectionFn1	double	Numerator frequency (Herz) for phase correction filter #1
– PhaseCorrectionFd1	double	Denominator frequency (Herz) for phase correction filter #1
– PhaseCorrectionGain1	double	Gain for phase correction filter #1
– PhaseCorrectionFn2	double	Numerator frequency (Herz) for phase correction filter #2
– PhaseCorrectionFd2	double	Denominator frequency (Herz) for phase correction filter #2
– PhaseCorrectionGain2	double	Gain for phase correction filter #2

Return

– Error	int32	Function error code

Python

Prototype

[Error] **PositionerCompensationPhaseCorrectionFiltersSet** (integer SocketID, string FullPositionerName, double PhaseCorrectionFn1,double PhaseCorrectionFd1, double PhaseCorrectionGain1, double PhaseCorrectionFn2,double PhaseCorrectionFd2, double PhaseCorrectionGain2)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– PhaseCorrectionFn1	double	Numerator frequency (Herz) for phase correction filter #1
– PhaseCorrectionFd1	double	Denominator frequency (Herz) for phase correction filter #1
– PhaseCorrectionGain1	double	Gain for phase correction filter #1
– PhaseCorrectionFn2	double	Numerator frequency (Herz) for phase correction filter #2
– PhaseCorrectionFd2	double	Denominator frequency (Herz) for phase correction filter #2
– PhaseCorrectionGain2	double	Gain for phase correction filter #2

Return

– Error	int	Function error code

2.2.4.26 PositionerCompensationSpatialPeriodicNotchsGet

Name

PositionerCompensationSpatialPeriodicNotchsGet – Gets pre-feedforward compensation spatial periodic filters parameters.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)
- Check corrector type: ERR_WRONG_OBJECT_TYPE (-8)

Description

This functions returns the *CompensationSystemPreFeedForward* spatial periodic filters parameters. These filters reduce the spatial periodic perturbations coming from screw pitch or cogging.

Note that the *CompensationSystemPreFeedForward* feature is available for all corrector types (acceleration, velocity, voltage or position) functioning *in closed loop configuration*.

- SpatialNotchStep1
- SpatialNotchsBandwidth1
- SpatialNotchsGain1
- SpatialNotchStep2
- SpatialNotchsBandwidth2
- SpatialNotchsGain2
- SpatialNotchStep3
- SpatialNotchsBandwidth3
- SpatialNotchsGain3

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerCompensationSpatialPeriodicNotchsGet $SocketID $FullPositionerName SpatialNotchStep1 SpatialNotchBandwidth1 SpatialNotchGain1 SpatialNotchStep2 SpatialNotchBandwidth2 SpatialNotchGain2 SpatialNotchStep3 SpatialNotchBandwidth3 SpatialNotchGain3

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– SpatialNotchStep1	double	Spatial periodic step for filter #1 (units)
– SpatialNotchBandwidth1	double	Spatial periodic bandwidth for filter #1 (Hz)
– SpatialNotchGain1	double	Spatial periodic gain for filter #1
– SpatialNotchStep2	double	Spatial periodic step for filter #2 (units)
– SpatialNotchBandwidth2	double	Spatial periodic bandwidth for filter #2 (Hz)
– SpatialNotchGain2	double	Spatial periodic gain for filter #2
– SpatialNotchStep3	double	Spatial periodic step for filter #3 (units)
– SpatialNotchBandwidth3	double	Spatial periodic bandwidth for filter #3 (Hz)
– SpatialNotchGain3	double	Spatial periodic gain for filter #3

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCompensationSpatialPeriodicNotchsGet** (int SocketID, char FullPositionerName [250], double* SpatialNotchStep1, double* SpatialNotchBandwidth1, double* SpatialNotchGain1, double* SpatialNotchStep2, double* SpatialNotchBandwidth2, double* SpatialNotchGain2, double* SpatialNotchStep3, double* SpatialNotchBandwidth3, double* SpatialNotchGain3)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name

Output parameters

– SpatialNotchStep1	double *	Spatial periodic step for filter #1 (units)
– SpatialNotchBandwidth1	double *	Spatial periodic bandwidth for filter #1 (Hz)
– SpatialNotchGain1	double *	Spatial periodic gain for filter #1
– SpatialNotchStep2	double *	Spatial periodic step for filter #2 (units)
– SpatialNotchBandwidth2	double *	Spatial periodic bandwidth for filter #2 (Hz)
– SpatialNotchGain2	double *	Spatial periodic gain for filter #2
– SpatialNotchStep3	double *	Spatial periodic step for filter #3 (units)
– SpatialNotchBandwidth3	double *	Spatial periodic bandwidth for filter #3 (Hz)
– SpatialNotchGain3	double *	Spatial periodic gain for filter #3

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCompensationSpatialPeriodicNotchsGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, SpatialNotchStep1 As Double, SpatialNotchBandwidth1 As Double, SpatialNotchGain1 As Double, SpatialNotchStep2 As Double, SpatialNotchBandwidth2 As Double, SpatialNotchGain2 As Double, SpatialNotchStep3 As Double, SpatialNotchBandwidth3 As Double, SpatialNotchGain3 As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– SpatialNotchStep1	double	Spatial periodic step for filter #1 (units)
– SpatialNotchBandwidth1	double	Spatial periodic bandwidth for filter #1 (Hz)
– SpatialNotchGain1	double	Spatial periodic gain for filter #1
– SpatialNotchStep2	double	Spatial periodic step for filter #2 (units)
– SpatialNotchBandwidth2	double	Spatial periodic bandwidth for filter #2 (Hz)
– SpatialNotchGain2	double	Spatial periodic gain for filter #2
– SpatialNotchStep3	double	Spatial periodic step for filter #3 (units)
– SpatialNotchBandwidth3	double	Spatial periodic bandwidth for filter #3 (Hz)
– SpatialNotchGain3	double	Spatial periodic gain for filter #3

Return

– Error	long	Function error code

Matlab

Prototype

[Error, SpatialNotchStep1, SpatialNotchBandwidth1, SpatialNotchGain1, SpatialNotchStep2, SpatialNotchBandwidth2, SpatialNotchGain2, SpatialNotchStep3, SpatialNotchBandwidth3, SpatialNotchGain3]
PositionerCompensationSpatialPeriodicNotchsGet (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– SpatialNotchStep1	double	Spatial periodic step for filter #1 (units)
– SpatialNotchBandwidth1	double	Spatial periodic bandwidth for filter #1 (Hz)
– SpatialNotchGain1	double	Spatial periodic gain for filter #1
– SpatialNotchStep2	double	Spatial periodic step for filter #2 (units)
– SpatialNotchBandwidth2	double	Spatial periodic bandwidth for filter #2 (Hz)
– SpatialNotchGain2	double	Spatial periodic gain for filter #2
– SpatialNotchStep3	double	Spatial periodic step for filter #3 (units)
– SpatialNotchBandwidth3	double	Spatial periodic bandwidth for filter #3 (Hz)
– SpatialNotchGain3	double	Spatial periodic gain for filter #3

Python

Prototype

[Error, SpatialNotchStep1, SpatialNotchBandwidth1, SpatialNotchGain1, SpatialNotchStep2, SpatialNotchBandwidth2, SpatialNotchGain2, SpatialNotchStep3, SpatialNotchBandwidth3, SpatialNotchGain3]
PositionerCompensationSpatialPeriodicNotchsGet (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code
– SpatialNotchStep1	double	Spatial periodic step for filter #1 (units)
– SpatialNotchBandwidth1	double	Spatial periodic bandwidth for filter #1 (Hz)
– SpatialNotchGain1	double	Spatial periodic gain for filter #1
– SpatialNotchStep2	double	Spatial periodic step for filter #2 (units)
– SpatialNotchBandwidth2	double	Spatial periodic bandwidth for filter #2 (Hz)
– SpatialNotchGain2	double	Spatial periodic gain for filter #2
– SpatialNotchStep3	double	Spatial periodic step for filter #3 (units)
– SpatialNotchBandwidth3	double	Spatial periodic bandwidth for filter #3 (Hz)
– SpatialNotchGain3	double	Spatial periodic gain for filter #3

2.2.4.27 PositionerCompensationSpatialPeriodicNotchsSet

Name

PositionerCompensationSpatialPeriodicNotchsSet – Sets pre-feedforward compensation spatial periodic filters parameters.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)
- Check correcor type: ERR_WRONG_OBJECT_TYPE (-8)
- Check parameter values: ERR_PARAMETER_OUT_OF_RANGE (-17)
- SpatialNotchStep $\in \left[0 : MaximumVelocity * CorrectorISRPeriod\right]$
- SpatialNotchBandwidth $\in \left[0 : \frac{0.5}{CorrectorISRPeriod}\right]$

NOTE

Refer to system.ref file to get CorrectorISRPeriod and stages.ini for MaximumVelocity values.

Description

This function sets the *CompensationSystemPreFeedForward* spatial periodic filters parameters. These filters reduce the spatial periodic perturbations coming from screw pitch or cogging.

Note that the *CompensationSystemPreFeedForward* feature is available for all corrector types (acceleration, velocity, voltage or position) functioning *in closed loop configuration*.

- SpatialNotchStep1
- SpatialNotchsBandwidth1
- SpatialNotchsGain1
- SpatialNotchStep2
- SpatialNotchsBandwidth2
- SpatialNotchsGain2
- SpatialNotchStep3
- SpatialNotchsBandwidth3
- SpatialNotchsGain3

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerCompensationSpatialPeriodicNotchsSet $SocketID $FullPositionerName $SpatialNotchStep1 $SpatialNotchBandwidth1 $SpatialNotchGain1 $SpatialNotchStep2 $SpatialNotchBandwidth2 $SpatialNotchGain2 $SpatialNotchStep3 $SpatialNotchBandwidth3 $SpatialNotchGain3

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– SpatialNotchStep1	double	Spatial periodic step for filter #1 (units)
– SpatialNotchBandwidth1	double	Spatial periodic bandwidth for filter #1 (Hz)
– SpatialNotchGain1	double	Spatial periodic gain for filter #1
– SpatialNotchStep2	double	Spatial periodic step for filter #2 (units)
– SpatialNotchBandwidth2	double	Spatial periodic bandwidth for filter #2 (Hz)
– SpatialNotchGain2	double	Spatial periodic gain for filter #2
– SpatialNotchStep3	double	Spatial periodic step for filter #3 (units)
– SpatialNotchBandwidth3	double	Spatial periodic bandwidth for filter #3 (Hz)
– SpatialNotchGain3	double	Spatial periodic gain for filter #3

Output parameters

– None

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCompensationSpatialPeriodicNotchsSet** (int SocketID, char FullPositionerName [250], double SpatialNotchStep1, double SpatialNotchBandwidth1, double SpatialNotchGain1, double SpatialNotchStep2, double SpatialNotchBandwidth2, double SpatialNotchGain2, double SpatialNotchStep3, double SpatialNotchBandwidth3, double SpatialNotchGain3)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name
– SpatialNotchStep1	double	Spatial periodic step for filter #1 (units)
– SpatialNotchBandwidth1	double	Spatial periodic bandwidth for filter #1 (Hz)
– SpatialNotchGain1	double	Spatial periodic gain for filter #1
– SpatialNotchStep2	double	Spatial periodic step for filter #2 (units)
– SpatialNotchBandwidth2	double	Spatial periodic bandwidth for filter #2 (Hz)
– SpatialNotchGain2	double	Spatial periodic gain for filter #2
– SpatialNotchStep3	double	Spatial periodic step for filter #3 (units)
– SpatialNotchBandwidth3	double	Spatial periodic bandwidth for filter #3 (Hz)
– SpatialNotchGain3	double	Spatial periodic gain for filter #3

Output parameters

– None

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCompensationSpatialPeriodicNotchsSet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal SpatialNotchStep1 As Double, ByVal SpatialNotchBandwidth1 As Double, ByVal SpatialNotchGain1 As Double, ByVal SpatialNotchStep2 As Double, ByVal SpatialNotchBandwidth2 As Double, ByVal SpatialNotchGain2 As Double, ByVal SpatialNotchStep3 As Double, ByVal SpatialNotchBandwidth3 As Double, ByVal SpatialNotchGain3 As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– SpatialNotchStep1	double	Spatial periodic step for filter #1 (units)
– SpatialNotchBandwidth1	double	Spatial periodic bandwidth for filter #1 (Hz)
– SpatialNotchGain1	double	Spatial periodic gain for filter #1
– SpatialNotchStep2	double	Spatial periodic step for filter #2 (units)
– SpatialNotchBandwidth2	double	Spatial periodic bandwidth for filter #2 (Hz)
– SpatialNotchGain2	double	Spatial periodic gain for filter #2
– SpatialNotchStep3	double	Spatial periodic step for filter #3 (units)
– SpatialNotchBandwidth3	double	Spatial periodic bandwidth for filter #3 (Hz)
– SpatialNotchGain3	double	Spatial periodic gain for filter #3

Output parameters

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **PositionerCompensationSpatialPeriodicNotchsSet** (int32 SocketID, cstring FullPositionerName, double SpatialNotchStep1, double SpatialNotchBandwidth1, double SpatialNotchGain1, double SpatialNotchStep2, double SpatialNotchBandwidth2, double SpatialNotchGain2, double SpatialNotchStep3, double SpatialNotchBandwidth3, double SpatialNotchGain3)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name
– SpatialNotchStep1	double	Spatial periodic step for filter #1 (units)
– SpatialNotchBandwidth1	double	Spatial periodic bandwidth for filter #1 (Hz)
– SpatialNotchGain1	double	Spatial periodic gain for filter #1
– SpatialNotchStep2	double	Spatial periodic step for filter #2 (units)
– SpatialNotchBandwidth2	double	Spatial periodic bandwidth for filter #2 (Hz)
– SpatialNotchGain2	double	Spatial periodic gain for filter #2
– SpatialNotchStep3	double	Spatial periodic step for filter #3 (units)
– SpatialNotchBandwidth3	double	Spatial periodic bandwidth for filter #3 (Hz)
– SpatialNotchGain3	double	Spatial periodic gain for filter #3

Return

– Error	int32	Function error code

Python

Prototype

[Error] **PositionerCompensationSpatialPeriodicNotchsSet** (integer SocketID, string FullPositionerName, double SpatialNotchStep1, double SpatialNotchBandwidth1, double SpatialNotchGain1, double SpatialNotchStep2, double SpatialNotchBandwidth2, double SpatialNotchGain2, double SpatialNotchStep3, double SpatialNotchBandwidth3, double SpatialNotchGain3)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– SpatialNotchStep1	double	Spatial periodic step for filter #1 (units)
– SpatialNotchBandwidth1	double	Spatial periodic bandwidth for filter #1 (Hz)
– SpatialNotchGain1	double	Spatial periodic gain for filter #1
– SpatialNotchStep2	double	Spatial periodic step for filter #2 (units)
– SpatialNotchBandwidth2	double	Spatial periodic bandwidth for filter #2 (Hz)
– SpatialNotchGain2	double	Spatial periodic gain for filter #2
– SpatialNotchStep3	double	Spatial periodic step for filter #3 (units)
– SpatialNotchBandwidth3	double	Spatial periodic bandwidth for filter #3 (Hz)
– SpatialNotchGain3	double	Spatial periodic gain for filter #3

Return

– Error	int	Function error code

2.2.4.28 PositionerCorrectorAutoTuning

Name

PositionerCorrectorAutoTuning – Auto-tuning process for determining position control loop PID values.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check input parameter type: ERR_WRONG_TYPE_INT (-15)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)
- Positioner must not be a "Secondary Positioner": ERR_WRONG_OBJECT_TYPE (-8)
- Check positioner name: ERR_POSITIONER_NAME (-18)
- Check group type: ERR_WRONG_OBJECT_TYPE (-8)
- Control loop type must be "PIDFFVelocity", "PIDDualFFVoltage" or "PIDFFAcceleration": ERR_UNCOMPATIBLE (-24)
- Group status must be "READY": ERR_NOT_ALLOWED_ACTION (-22)

Description

The function executes an auto-tuning process and returns the calculated PID settings (KP, KI and KD values). The selected group must be in "READY" state, else ERR_NOT_ALLOWED_ACTION (-22) is returned.

This function works only if the positioner control loop type is "PIDFFVelocity" (velocity control), "PIDDualFFVoltage" (voltage control) or "PIDFFAcceleration" (acceleration control), else it returns ERR_UNCOMPATIBLE error.

If the function is called when the positioner is not in READY state, ERR_NOT_ALLOWED_ACTION (-22) error will be returned.

The "Mode" input value indicates the control mode of the position loop (**Short Settle** or **High Robustness**).

In the **Short Settle** mode, the PID values are adjusted to have high motion performance (short settling time, less following errors).

The **High Robustness** mode is used for a relatively good performance in motion, but guarantees the robustness (stability) for all stage situations (positions, velocities, accelerations).

If every thing is OK, auto-tuning is executed. If auto-tuning initialization fails ERR_PID_TUNING_INITIALIZATION (-104) is returned, or if the motion becomes disabled then ERR_EMERGENCY_SIGNAL (-26) is returned.

The auto-tuning process is executed in 5 periods. At the end of each period, the auto-tuning process estimates the auto-tuning quality by calculating the noise/signal ratio. If the noise/signal ratio is very close to zero (it means no oscillation), an ERR_RELAY_FEEDBACK_TEST_NO_OSCILLATION (-101) is returned. Else if the noise ratio > MaximumNoiseRatio (normally between 0.1 and 0.2, exact value defined in system.ref) then ERR_RELAY_FEEDBACK_TEST_SIGNAL_NOISY (-102) is returned.

If the number of acquired data points (minimum = 9) or the number of acquired signal periods (minimum = 5) is not enough for a good estimate then ERR_SIGNAL_POINTS_NOT_ENOUGH (-103) is returned.

At end of this function, the new PID setting is returned and the group status becomes "READY" once again.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_POSITIONER_ERROR (-5)
- ERR_POSITIONER_NAME (-18)
- ERR_RELAY_FEEDBACK_TEST_NO_OSCILLATION (-101)
- ERR_RELAY_FEEDBACK_TEST_SIGNAL_NOISY (-102)
- ERR_PID_TUNING_INITIALIZATION (-104)
- ERR_SIGNAL_POINTS_NOT_ENOUGH (-103)
- ERR_UNCOMPATIBLE (-24)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13)
- ERR_WRONG_TYPE_DOUBLE (-14)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

PositionerCorrectorAutoTuning $SocketID $PositionerName $Mode KP KI KD

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– PositionerName	string	Name of a positioner
– Mode	integer.	Loop control mode (0 = short settle, or 1 = robust)

Output parameters

– KP	double *	Calculated KP value
– KI	double *	Calculated KI value
– KD	double *	Calculated KD value

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCorrectorAutoTuning** (int SocketID, char * PositionerName, int Mode, double * KP, double * KI, double * KD)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– PositionerName	char *	Positioner name
– Mode	int	Loop control mode (0 = short settle, or 1 = robust)

Output parameters

– KP	double *	Calculated KP value
– KI	double *	Calculated KI value
– KD	double *	Calculated KD value

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCorrectorAutoTuning** (ByVal SocketID As Long, ByVal PositionerName As String, Mode As Integer, KP As Double, KI As Double, KD As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– PositionerName	string	Positioner name
– Mode	int	Loop control mode (0 = short settle, or 1 = robust)

Output parameters

– KP	double	Calculated KP value
– KI	double	Calculated KI value
– KD	double	Calculated KD value

Return

– Error	long	Function error code

Matlab

Prototype

[Error, KP, KI, KD] **PositionerCorrectorAutoTuning** (int32 SocketID, cstring PositionerName, int32 Mode)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– PositionerName	cstring	Positioner name
– Mode	int32	Loop control mode (0 = short settle, or 1 = robust)

Return

– Error	int32	Function error code
– KP	double	Calculated KP value
– KI	double	Calculated KI value
– KD	double	Calculated KD value

Python

Prototype

[Error, KP, KI, KD] **PositionerCorrectorAutoTuning** (integer SocketID, string PositionerName, integer Mode**,** string Password)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– PositionerName	string	Positioner name
– Mode	int	Loop control mode (0 = short settle, or 1 = robust)

Return

– Error	int	Function error code
– KP	double	Calculated KP value
– KI	double	Calculated KI value
– KD	double	Calculated KD value

2.2.4.29 PositionerCorrectorNotchFiltersGet

Name

PositionerCorrectorNotchFiltersGet – Gets the notch filter parameters.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)
- Check the positioner type: ERR_WRONG_OBJECT_TYPE (-8)

Description

This functions returns the parameters defined for two notch filters.

First notch filter parameters:

- UserNotchFrequency1
- UserNotchBandwidth1
- UserNotchGain1

Second notch filter parameters:

- UserNotchFrequency2
- UserNotchBandwidth2
- UserNotchGain2.

NOTE

If the corrector type is "NoEncoderPositionCorrector" then ERR_UNCOMPATIBLE (-24) is returned.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_UNCOMPATIBLE (-24)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerCorrectorNotchFiltersGet $SocketID $FullPositionerName NotchFrequency1 NotchBandwith1 NotchGain1 NotchFrequency2 NotchBandwith2 NotchGain2

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– NotchFrequency1	double	Frequency (Herz) for notch filter #1
– NotchBandwith1	double	Band width (Herz) for notch filter #1
– NotchGain1	double	Gain for notch filter #1
– NotchFrequency2	double	Frequency (Herz) for notch filter #2
– NotchBandwith2	double	Band width (Herz) for notch filter #2
– NotchGain2	double	Gain for notch filter #2

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCorrectorNotchFiltersGet** (int SocketID, char FullPositionerName [250], double* NotchFrequency1,double* NotchBandwith1, double* NotchGain1, double* NotchFrequency2, double* NotchBandwith2, double* NotchGain2)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name

Output parameters

– NotchFrequency1	double *	Frequency (Herz) for notch filter #1
– NotchBandwith1	double *	Band width (Herz) for notch filter #1
– NotchGain1	double *	Gain for notch filter #1
– NotchFrequency2	double *	Frequency (Herz) for notch filter #2
– NotchBandwith2	double *	Band width (Herz) for notch filter #2
– NotchGain2	double *	Gain for notch filter #2

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCorrectorNotchFiltersGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, NotchFrequency1 As Double, NotchBandwith1 As Double, NotchGain1 As Double, NotchFrequency2 As Double, NotchBandwith2 As Double, NotchGain2 As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– NotchFrequency1	double	Frequency (Herz) for notch filter #1
– NotchBandwith1	double	Band width (Herz) for notch filter #1
– NotchGain1	double	Gain for notch filter #1
– NotchFrequency2	double	Frequency (Herz) for notch filter #2
– NotchBandwith2	double	Band width (Herz) for notch filter #2
– NotchGain2	double	Gain for notch filter #2

Return

– Error	long	Function error code

Matlab

Prototype

[Error, NotchFrequency1, NotchBandwith1, NotchGain1, NotchFrequency2, NotchBandwith2, NotchGain2] **PositionerCorrectorNotchFiltersGet** (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– NotchFrequency1	double	Frequency (Herz) for notch filter #1
– NotchBandwith1	double	Band width (Herz) for notch filter #1
– NotchGain1	double	Gain for notch filter #1
– NotchFrequency2	double	Frequency (Herz) for notch filter #2
– NotchBandwith2	double	Band width (Herz) for notch filter #2
– NotchGain2	double	Gain for notch filter #2

Python

Prototype

[Error, NotchFrequency1, NotchBandwith1, NotchGain1, NotchFrequency2, NotchBandwith2, NotchGain2] **PositionerCorrectorNotchFiltersGet** (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code
– NotchFrequency1	double	Frequency (Herz) for notch filter #1
– NotchBandwith1	double	Band width (Herz) for notch filter #1
– NotchGain1	double	Gain for notch filter #1
– NotchFrequency2	double	Frequency (Herz) for notch filter #2
– NotchBandwith2	double	Band width (Herz) for notch filter #2
– NotchGain2	double	Gain for notch filter #2

2.2.4.30 PositionerCorrectorNotchFiltersSet

Name

PositionerCorrectorNotchFiltersSet – Sets the notch filter parameters.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)
- Check the positioner type: ERR_WRONG_OBJECT_TYPE (-8)
- Check parameter values: ERR_PARAMETER_OUT_OF_RANGE (-17)
- NotchFrequency $\in \left[0 : \frac{0.5}{\text{CorrectorISRPeriod}}\right]$
- NotchBandwidth $\in \left[0 : \frac{0.5}{\text{CorrectorISRPeriod}}\right]$
- NotchGain $\in [0 : 100]$

NOTE

Refer to *system.ref* file to get CorrectorISRPeriod value.

Description

This functions configures the parameters defined for two notch filters. If the "NotchFrequency" value is NULL or the "NotchGain" value is NULL then the notch filter is not activated.

First notch filter parameters:

- NotchFrequency1
- NotchBandwidth1
- NotchGain1
- Second notch filter parameters:
- NotchFrequency2
- NotchBandwidth2
- NotchGain2.

NOTE

If the corrector type is "NoEncoderPositionCorrector" then ERR_UNCOMPATIBLE (-24) is returned.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_UNCOMPATIBLE (-24)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerCorrectorNotchFiltersSet $SocketID $FullPositionerName NotchFrequency1 NotchBandwith1 NotchGain1 NotchFrequency2 NotchBandwith2 NotchGain2

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name
–	NotchFrequency1	double	Frequency (Herz) for notch filter #1
–	NotchBandwith1	double	Band width (Herz) for notch filter #1
–	NotchGain1	double	Gain for notch filter #1
–	NotchFrequency2	double	Frequency (Herz) for notch filter #2
–	NotchBandwith2	double	Band width (Herz) for notch filter #2
–	NotchGain2	double	Gain for notch filter #2

Output parameter

– None

Return

–	Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCorrectorNotchFiltersSet** (int SocketID, char FullPositionerName [250], double* NotchFrequency1,double* NotchBandwith1, double* NotchGain1, double* NotchFrequency2, double* NotchBandwith2, double* NotchGain2)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name
– NotchFrequency1	double	Frequency (Herz) for notch filter #1
– NotchBandwith1	double	Band width (Herz) for notch filter #1
– NotchGain1	double	Gain for notch filter #1
– NotchFrequency2	double	Frequency (Herz) for notch filter #2
– NotchBandwith2	double	Band width (Herz) for notch filter #2
– NotchGain2	double	Gain for notch filter #2

Output parameter

– None

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCorrectorNotchFiltersSet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal NotchFrequency1 As Double, ByVal NotchBandwith1 As Double, ByVal NotchGain1 As Double, ByVal NotchFrequency2 As Double, ByVal NotchBandwith2 As Double, ByVal NotchGain2 As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– NotchFrequency1	double	Frequency (Herz) for notch filter #1
– NotchBandwith1	double	Band width (Herz) for notch filter #1
– NotchGain1	double	Gain for notch filter #1
– NotchFrequency2	double	Frequency (Herz) for notch filter #2
– NotchBandwith2	double	Band width (Herz) for notch filter #2
– NotchGain2	double	Gain for notch filter #2

Output parameter

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **PositionerCorrectorNotchFiltersSet** (int32 SocketID, cstring FullPositionerName, double NotchFrequency1, double NotchBandwith1, double NotchGain1, double NotchFrequency2, double NotchBandwith2, double NotchGain2)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name
– NotchFrequency1	double	Frequency (Herz) for notch filter #1
– NotchBandwith1	double	Band width (Herz) for notch filter #1
– NotchGain1	double	Gain for notch filter #1
– NotchFrequency2	double	Frequency (Herz) for notch filter #2
– NotchBandwith2	double	Band width (Herz) for notch filter #2
– NotchGain2	double	Gain for notch filter #2

Return

– Error	int32	Function error code

Python

Prototype

[Error] **PositionerCorrectorNotchFiltersSet** (integer SocketID, string FullPositionerName, double NotchFrequency1, double NotchBandwith1, double NotchGain1, double NotchFrequency2, double NotchBandwith2, double NotchGain2)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– NotchFrequency1	double	Frequency (Herz) for notch filter #1
– NotchBandwith1	double	Band width (Herz) for notch filter #1
– NotchGain1	double	Gain for notch filter #1
– NotchFrequency2	double	Frequency (Herz) for notch filter #2
– NotchBandwith2	double	Band width (Herz) for notch filter #2
– NotchGain2	double	Gain for notch filter #2

Return

– Error	int	Function error code

2.2.4.31 PositionerCorrectorPIDDualFFVoltageGet

Name

PositionerCorrectorPIDDualFFVoltageGet – Gets the corrector "PIDDualFFVoltage" parameters.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type and the corrector type: ERR_WRONG_OBJECT_TYPE (-8)
- Check output parameter type: ERR_WRONG_TYPE_BOOL (-12), ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function returns the corrector parameter values used by a PID dual feed-forward with a motor voltage output.

NOTE

The "CorrectorType" must be "PIDDualFFVoltage" in the stages.ini file. This servo loop type is used when the position servo loop drives the voltage applied directly to the motor.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_BOOL (-12)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerCorrectorPIDDualFFVoltageGet $SocketID $FullPositionerName ClosedLoopStatus KP KI KD KS IntegrationTime DerivativeFilterCutOffFrequency GKP GKI GKD KForm FeedForwardGainVelocity FeedForwardGainAcceleration Friction

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name

Output parameters

–	ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
–	KP	double	PID servo loop proportional gain
–	KI	double	PID servo loop integral gain
–	KD	double	PID servo loop derivative gain
–	KS	double	PID integral saturation value (0 to 1)
–	IntegrationTime	double	PID integration time (seconds)
–	DerivativeFilterCutOffFrequency	double	PID derivative filter cut off frequency (Hz)
–	GKP	double	Variable PID proportional gain multiplier
–	GKI	double	Variable PID integral gain multiplier
–	GKD	double	Variable PID derivative gain multiplier
–	KForm	double	Variable PID form coefficient
–	FeedForwardGainVelocity	double	Velocity feedforward gain (units)
–	FeedForwardGainAcceleration	double	Acceleration feedforward gain (units)
–	Friction	double	friction compensation

Return

–	Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCorrectorPIDDualFFVoltageGet** (int SocketID, char FullPositionerName[250], bool* ClosedLoopStatus, double* KP, double* KI, double* KD, double* KS, double* IntegrationTime, double* DerivativeFilterCutOffFrequency, double* GKP, double* GKI, double* GKD, double* KForm, double* FeedForwardGainVelocity, double* FeedForwardGainAcceleration, double* Friction)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name

Output parameters

– ClosedLoopStatus	bool *	Position servo loop status (true=closed and false=opened)
– KP	double *	PID servo loop proportional gain
– KI	double *	PID servo loop integral gain
– KD	double *	PID servo loop derivative gain
– KS	double *	PID integral saturation value (0 to 1)
– IntegrationTime	double *	PID integration time (seconds)
– DerivativeFilterCutOffFrequency	double *	PID derivative filter cut off frequency (Hz)
– GKP	double *	Variable PID proportional gain multiplier
– GKI	double *	Variable PID integral gain multiplier
– GKD	double *	Variable PID derivative gain multiplier
– KForm	double *	Variable PID form coefficient
– FeedForwardGainVelocity	double *	Velocity feedforward gain (units)
– FeedForwardGainAcceleration	double *	Acceleration feedforward gain (units)
– Friction	double *	Friction compensation

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCorrectorPIDDualFFVoltageGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ClosedLoopStatus As Boolean, KP As Double, KI As Double, KD As Double, KS As Double, IntegrationTime As Double, DerivativeFilterCutOffFrequency As Double, GKP As Double, GKI As Double, GKD As Double, KForm As Double, FeedForwardGainVelocity As Double, FeedForwardGainAcceleration As Double, Friction As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	PID servo loop proportional gain
– KI	double	PID servo loop integral gain
– KD	double	PID servo loop derivative gain
– KS	double	PID integral saturation value (0 to 1)
– IntegrationTime	double	PID integration time (seconds)
– DerivativeFilterCutOffFrequency	double	PID derivative filter cut off frequency (Hz)
– GKP	double	Variable PID proportional gain multiplier
– GKI	double	Variable PID integral gain multiplier
– GKD	double	Variable PID derivative gain multiplier
– KForm	double	Variable PID form coefficient
– FeedForwardGainVelocity	double	Velocity feedforward gain (units)
– FeedForwardGainAcceleration	double	Acceleration feedforward gain (units)
– Friction	double	Friction compensation

Return

– Error	long	Function error code

Matlab

Prototype

[Error, ClosedLoopStatus, KP, KI, KD, KS, IntegrationTime, DerivativeFilterCutOffFrequency, GKP, GKI, GKD, KForm, FeedForwardGainVelocity, FeedForwardGainAcceleration, Friction] **PositionerCorrectorPIDDualFFVoltageGet** (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	PID servo loop proportional gain
– KI	double	PID servo loop integral gain
– KD	double	PID servo loop derivative gain
– KS	double	PID integral saturation value (0 to 1)
– IntegrationTime	double	PID integration time (seconds)
– DerivativeFilterCutOffFrequency	double	PID derivative filter cut off frequency (Hz)
– GKP	double	Variable PID proportional gain multiplier
– GKI	double	Variable PID integral gain multiplier
– GKD	double	Variable PID derivative gain multiplier
– KForm	double	Variable PID form coefficient
– FeedForwardGainVelocity	double	Velocity feedforward gain (units)
– FeedForwardGainAcceleration	double	Acceleration feedforward gain (units)
– Friction	double	Friction compensation

Python

Prototype

[Error, ClosedLoopStatus, KP, KI, KD, KS, IntegrationTime, DerivativeFilterCutOffFrequency, GKP, GKI, GKD, KForm, FeedForwardGainVelocity, FeedForwardGainAcceleration, Friction]
PositionerCorrectorPIDDualFFVoltageGet (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code
– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	PID servo loop proportional gain
– KI	double	PID servo loop integral gain
– KD	double	PID servo loop derivative gain
– KS	double	PID integral saturation value (0 to 1)
– IntegrationTime	double	PID integration time (seconds)
– DerivativeFilterCutOffFrequency	double	PID derivative filter cut off frequency (Hz)
– GKP	double	Variable PID proportional gain multiplier
– GKI	double	Variable PID integral gain multiplier
– GKD	double	Variable PID derivative gain multiplier
– KForm	double	Variable PID form coefficient
– FeedForwardGainVelocity	double	Velocity feedforward gain (units)
– FeedForwardGainAcceleration	double	Acceleration feedforward gain (units)
– Friction	double	Friction compensation

2.2.4.32 PositionerCorrectorPIDDualFFVoltageSet

Name

PositionerCorrectorPIDDualFFVoltageSet – Configures the corrector "PIDDualFFVoltage" parameters.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type and the corrector type: ERR_WRONG_OBJECT_TYPE (-8)
- Check input parameter type: ERR_WRONG_TYPE_BOOL (-12), ERR_WRONG_TYPE_DOUBLE (-14)
- Check parameter value: ERR_PARAMETER_OUT_OF_RANGE (-17)
- KP ≥ 0
- KI ≥ 0
- KD ≥ 0
- 0 ≤ KS ≤ 1
- IntegrationTime ≥ CorrectorISRPeriod
- GKP > -1
- GKI > -1
- GKD > -1
- KForm ≥ 0
- KFeedForwardVelocity ≥ 0
- KFeedForwardAcceleration ≥ 0
- Friction ≥ 0
- DerivativeFilterCutOffFrequency $\in \left[0 : \frac{0.5}{\text{CorrectorISRPeriod}}\right]$

NOTE

Refer to *system.ref* file to get CorrectorISRPeriod value.

Description

This function configures the "PIDDualFFVoltage" corrector parameters. The "CorrectorType" must be "PIDDualFFVoltage" in the stages.ini file, else ERR_WRONG_OBJECT_TYPE (-8) is returned.

NOTE

This servo loop type is used when the position servo loop drives the voltage applied directly to the motor.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_BOOL (-12)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerCorrectorPIDDualFFVoltageSet $SocketID $FullPositionerName $ClosedLoopStatus $KP $KI $KD $KS $IntegrationTime $DerivativeFilterCutOffFrequency $GKP $GKI $GKD $KForm $FeedForwardGainVelocity $FeedForwardGainAcceleration $Friction

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	PID servo loop proportional gain
– KI	double	PID servo loop integral gain
– KD	double	PID servo loop derivative gain
– KS	double	PID integral saturation value (0 to 1)
– IntegrationTime	double	PID integration time (seconds)
– DerivativeFilterCutOffFrequency	double	PID derivative filter cut off frequency (Hz)
– GKP	double	Variable PID proportional gain multiplier
– GKI	double	Variable PID integral gain multiplier
– GKD	double	Variable PID derivative gain multiplier
– KForm	double	Variable PID form coefficient
– FeedForwardGainVelocity	double	Velocity feedforward gain (units)
– FeedForwardGainAcceleration	double	Acceleration feedforward gain (units)
– Friction	double	Friction compensation

Output parameters

– None

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCorrectorPIDDualFFVoltageSet** (int SocketID, char FullPositionerName[250], bool ClosedLoopStatus, double KP, double KI, double KD, double KS, double IntegrationTime, double DerivativeFilterCutOffFrequency, double GKP, double GKI, double GKD, double KForm, double FeedForwardGainVelocity, double FeedForwardGainAcceleration, double Friction)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name
– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	PID servo loop proportional gain
– KI	double	PID servo loop integral gain
– KD	double	PID servo loop derivative gain
– KS	double	PID integral saturation value (0 to 1)
– IntegrationTime	double	PID integration time (seconds)
– DerivativeFilterCutOffFrequency	double	PID derivative filter cut off frequency (Hz)
– GKP	double	Variable PID proportional gain multiplier
– GKI	double	Variable PID integral gain multiplier
– GKD	double	Variable PID derivative gain multiplier
– KForm	double	Variable PID form coefficient
– FeedForwardGainVelocity	double	Velocity feedforward gain (units)
– FeedForwardGainAcceleration	double	Acceleration feedforward gain (units)
– Friction	double	Friction compensation

Output parameters

– None

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCorrectorPIDDualFFVoltageSet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal ClosedLoopStatus As Boolean, ByVal KP As Double, ByVal KI As Double, ByVal KD As Double, ByVal KS As Double, ByVal IntegrationTime As Double, ByVal DerivativeFilterCutOffFrequency As Double, ByVal GKP As Double, ByVal GKI As Double, ByVal GKD As Double, ByVal KForm As Double, ByVal FeedForwardGainVelocity As Double, ByVal FeedForwardGainAcceleration As Double, ByVal Friction As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	PID servo loop proportional gain
– KI	double	PID servo loop integral gain
– KD	double	PID servo loop derivative gain
– KS	double	PID integral saturation value (0 to 1)
– IntegrationTime	double	PID integration time (seconds)
– DerivativeFilterCutOffFrequency	double	PID derivative filter cut off frequency (Hz)
– GKP	double	Variable PID proportional gain multiplier
– GKI	double	Variable PID integral gain multiplier
– GKD	double	Variable PID derivative gain multiplier
– KForm	double	Variable PID form coefficient
– FeedForwardGainVelocity	double	Velocity feedforward gain (units)
– FeedForwardGainAcceleration	double	Acceleration feedforward gain (units)
– Friction	double	Friction compensation

Output parameters

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **PositionerCorrectorPIDDualFFVoltageSet** (int32 SocketID, cstring FullPositionerName, bool ClosedLoopStatus, double KP, double KI, double KD, double KS, double IntegrationTime, double DerivativeFilterCutOffFrequency, double GKP, double GKI, double GKD, double KForm, double FeedForwardGainVelocity, double FeedForwardGainAcceleration, double Friction)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name
– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	PID servo loop proportional gain
– KI	double	PID servo loop integral gain
– KD	double	PID servo loop derivative gain
– KS	double	PID integral saturation value (0 to 1)
– IntegrationTime	double	PID integration time (seconds)
– DerivativeFilterCutOffFrequency	double	PID derivative filter cut off frequency (Hz)
– GKP	double	Variable PID proportional gain multiplier
– GKI	double	Variable PID integral gain multiplier
– GKD	double	Variable PID derivative gain multiplier
– KForm	double	Variable PID form coefficient
– FeedForwardGainVelocity	double	Velocity feedforward gain (units)
– FeedForwardGainAcceleration	double	Acceleration feedforward gain (units)
– Friction	double	Friction compensation

Return

– Error	int32	Function error code

Python

Prototype

[Error] **PositionerCorrectorPIDDualFFVoltageSet** (integer SocketID, string FullPositionerName, bool ClosedLoopStatus, double KP, double KI, double KD, double KS, double IntegrationTime, double DerivativeFilterCutOffFrequency, double GKP, double GKI, double GKD, double KForm, double FeedForwardGainVelocity, double FeedForwardGainAcceleration, double Friction)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	PID servo loop proportional gain
– KI	double	PID servo loop integral gain
– KD	double	PID servo loop derivative gain
– KS	double	PID integral saturation value (0 to 1)
– IntegrationTime	double	PID integration time (seconds)
– DerivativeFilterCutOffFrequency	double	PID derivative filter cut off frequency (Hz)
– GKP	double	Variable PID proportional gain multiplier
– GKI	double	Variable PID integral gain multiplier
– GKD	double	Variable PID derivative gain multiplier
– KForm	double	Variable PID form coefficient
– FeedForwardGainVelocity	double	Velocity feedforward gain (units)
– FeedForwardGainAcceleration	double	Acceleration feedforward gain (units)
– Friction	double	Friction compensation

Return

– Error	int	Function error code

2.2.4.33 PositionerCorrectorPIDFFAccelerationGet

Name

PositionerCorrectorPIDFFAccelerationGet – Gets the corrector "PIDFFAcceleration" parameters.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type and the corrector type: ERR_WRONG_OBJECT_TYPE (-8)
- Check output parameter type: ERR_WRONG_TYPE_BOOL (-12), ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function returns the corrector parameter values used by a PID feed-forward with an acceleration output.

NOTE

The "CorrectorType" must be "PIDFFAcceleration" in the stages.ini file. This servo loop type is used when a constant value applied to the driver results in a constant acceleration of the stage.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_BOOL (-12)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerCorrectorPIDFFAccelerationGet $SocketID $FullPositionerName ClosedLoopStatus KP KI KD KS IntegrationTime DerivativeFilterCutOffFrequency GKP GKI GKD KForm FeedForwardGainAcceleration FeedForwardGainJerk

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	PID servo loop proportional gain
– KI	double	PID servo loop integral gain
– KD	double	PID servo loop derivative gain
– KS	double	PID integral saturation value (0 to 1)
– IntegrationTime	double	PID integration time (seconds)
– DerivativeFilterCutOffFrequency	double	PID derivative filter cut off frequency (Hz)
– GKP	double	Variable PID proportional gain multiplier
– GKI	double	Variable PID integral gain multiplier
– GKD	double	Variable PID derivative gain multiplier
– KForm	double	Variable PID form coefficient
– FeedForwardGainAcceleration	double	Acceleration feedforward gain
– FeedForwardGainJerk	double	Jerk feedforward gain

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCorrectorPIDFFAccelerationGet** (int SocketID, char FullPositionerName[250], bool* ClosedLoopStatus, double* KP, double* KI, double* KD, double* KS, double* IntegrationTime, double* DerivativeFilterCutOffFrequency, double* GKP, double* GKI, double* GKD, double* KForm, double* FeedForwardGainAcceleration, double* FeedForwardGainJerk)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name

Output parameters

– ClosedLoopStatus	bool *	Position servo loop status (true=closed and false=opened)
– KP	double *	PID servo loop proportional gain
– KI	double *	PID servo loop integral gain
– KD	double *	PID servo loop derivative gain
– KS	double *	PID integral saturation value (0 to 1)
– IntegrationTime	double *	PID integration time (seconds)
– DerivativeFilterCutOffFrequency	double *	PID derivative filter cut off frequency (Hz)
– GKP	double *	Variable PID proportional gain multiplier
– GKI	double *	Variable PID integral gain multiplier
– GKD	double *	Variable PID derivative gain multiplier
– KForm	double *	Variable PID form coefficient
– FeedForwardGainAcceleration	double *	Acceleration feedforward gain
– FeedForwardGainJerk	double *	Jerk feedforward gain

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCorrectorPIDFFAccelerationGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ClosedLoopStatus As Boolean, KP As Double, KI As Double, KD As Double, KS As Double, IntegrationTime As Double, DerivativeFilterCutOffFrequency As Double, GKP As Double, GKI As Double, GKD As Double, KForm As Double, FeedForwardGainAcceleration As Double, FeedForwardGainJerk As Double)

Input parameters

– SocketID	Long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	String	Positioner name

Output parameters

– ClosedLoopStatus	Boolean	Position servo loop status (true=closed and false=opened)
– KP	Double	PID servo loop proportional gain
– KI	Double	PID servo loop integral gain
– KD	Double	PID servo loop derivative gain
– KS	Double	PID integral saturation value (0 to 1)
– IntegrationTime	Double	PID integration time (seconds)
– DerivativeFilterCutOffFrequency	Double	PID derivative filter cut off frequency (Hz)
– GKP	Double	Variable PID proportional gain multiplier
– GKI	Double	Variable PID integral gain multiplier
– GKD	Double	Variable PID derivative gain multiplier
– KForm	Double	Variable PID form coefficient
– FeedForwardGainAcceleration	Double	Acceleration feedforward gain
– FeedForwardGainJerk	Double	Jerk feedforward gain

Return

– Error	Long	Function error code

Matlab

Prototype

[Error, ClosedLoopStatus, KP, KI, KD, KS, IntegrationTime, DerivativeFilterCutOffFrequency, GKP, GKI, GKD, KForm, FeedForwardGainAcceleration, FeedForwardGainJerk]
PositionerCorrectorPIDFFAccelerationGet (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	PID servo loop proportional gain
– KI	double	PID servo loop integral gain
– KD	double	PID servo loop derivative gain
– KS	double	PID integral saturation value (0 to 1)
– IntegrationTime	double	PID integration time (seconds)
– DerivativeFilterCutOffFrequency	double	PID derivative filter cut off frequency (Hz)
– GKP	double	Variable PID proportional gain multiplier
– GKI	double	Variable PID integral gain multiplier
– GKD	double	Variable PID derivative gain multiplier
– KForm	double	Variable PID form coefficient
– FeedForwardGainAcceleration	double	Acceleration feedforward gain
– FeedForwardGainJerk	double	Jerk feedforward gain

Python

Prototype

[Error, ClosedLoopStatus, KP, KI, KD, KS, IntegrationTime, DerivativeFilterCutOffFrequency, GKP, GKI, GKD, KForm, FeedForwardGainAcceleration, FeedForwardGainJerk]
PositionerCorrectorPIDFFAccelerationGet (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code
– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	PID servo loop proportional gain
– KI	double	PID servo loop integral gain
– KD	double	PID servo loop derivative gain
– KS	double	PID integral saturation value (0 to 1)
– IntegrationTime	double	PID integration time (seconds)
– DerivativeFilterCutOffFrequency	double	PID derivative filter cut off frequency (Hz)
– GKP	double	Variable PID proportional gain multiplier
– GKI	double	Variable PID integral gain multiplier
– GKD	double	Variable PID derivative gain multiplier
– KForm	double	Variable PID form coefficient
– FeedForwardGainAcceleration	double	Acceleration feedforward gain
– FeedForwardGainJerk	double	Jerk feedforward gain

2.2.4.34 PositionerCorrectorPIDFFAccelerationSet

Name

PositionerCorrectorPIDFFAccelerationSet – Sets the corrector "PIDFFAcceleration" parameters.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type and the corrector type: ERR_WRONG_OBJECT_TYPE (-8)
- Check input parameter type: ERR_WRONG_TYPE_BOOL (-12), ERR_WRONG_TYPE_DOUBLE (-14)
- Check parameter value: ERR_PARAMETER_OUT_OF_RANGE (-17)
- KP ≥ 0
- KI ≥ 0
- KD ≥ 0
- 0 ≤ KS ≤ 1
- IntegrationTime ≥ CorrectorISRPeriod
- GKP > -1
- GKI > -1
- GKD > -1
- KForm ≥ 0
- KFeedForwardAcceleration ≥ 0
- KFeedForwardJerk ≥ 0
- DerivativeFilterCutOffFrequency $\in \left[0 : \frac{0.5}{\text{CorrectorISRPeriod}} \right]$

NOTE

Refer to *system.ref* file to get CorrectorISRPeriod value.

Description

This function configures the "PIDFFAcceleration" corrector parameters.

NOTE

The "CorrectorType" parameter must be defined as "PIDFFAcceleration" in the "stages.ini" file else ERR_WRONG_OBJECT_TYPE (-8) is returned. This servo loop type is used when a constant value applied to the driver results in a constant acceleration of the stage.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_BOOL (-12)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerCorrectorPIDFFAccelerationSet $SocketID $FullPositionerName $ClosedLoopStatus $KP $KI $KD $KS $IntegrationTime $DerivativeFilterCutOffFrequency $GKP $GKI $GKD $KForm $FeedForwardGainAcceleration $FeedForwardGainJerk

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	PID servo loop proportional gain
– KI	double	PID servo loop integral gain
– KD	double	PID servo loop derivative gain
– KS	double	PID integral saturation value (0 to 1)
– IntegrationTime	double	PID integration time (seconds)
– DerivativeFilterCutOffFrequency	double	PID derivative filter cut off frequency (Hz)
– GKP	double	Variable PID proportional gain multiplier
– GKI	double	Variable PID integral gain multiplier
– GKD	double	Variable PID derivative gain multiplier
– KForm	double	Variable PID form coefficient
– FeedForwardGainAcceleration	double	Acceleration feedforward gain
– FeedForwardGainJerk	double	Jerk feedforward gain

Output parameters

– None

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCorrectorPIDFFAccelerationSet** (int SocketID, char FullPositionerName[250], bool ClosedLoopStatus, double KP, double KI, double KD, double KS, double IntegrationTime, double DerivativeFilterCutOffFrequency, double GKP, double GKI, double GKD, double KForm, double FeedForwardGainAcceleration, double FeedForwardGainJerk)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name
– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	PID servo loop proportional gain
– KI	double	PID servo loop integral gain
– KD	double	PID servo loop derivative gain
– KS	double	PID integral saturation value (0 to 1)
– IntegrationTime	double	PID integration time (seconds)
– DerivativeFilterCutOffFrequency	double	PID derivative filter cut off frequency (Hz)
– GKP	double	Variable PID proportional gain multiplier
– GKI	double	Variable PID integral gain multiplier
– GKD	double	Variable PID derivative gain multiplier
– KForm	double	Variable PID form coefficient
– FeedForwardGainAcceleration	double	Acceleration feedforward gain
– FeedForwardGainJerk	double	Jerk feedforward gain

Output parameters

– None

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCorrectorPIDFFAccelerationSet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal ClosedLoopStatus As Boolean, ByVal KP As Double, ByVal KI As Double, ByVal KD As Double, ByVal KS As Double, ByVal IntegrationTime As Double, ByVal DerivativeFilterCutOffFrequency As Double, ByVal GKP As Double, ByVal GKI As Double, ByVal GKD As Double, ByVal KForm As Double, ByVal FeedForwardGainAcceleration As Double, ByVal FeedForwardGainJerk As Double)

Input parameters

– SocketID	Long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	String	Positioner name
– ClosedLoopStatus	Boolean	Position servo loop status (true=closed and false=opened)
– KP	Double	PID servo loop proportional gain
– KI	Double	PID servo loop integral gain
– KD	Double	PID servo loop derivative gain
– KS	Double	PID integral saturation value (0 to 1)
– IntegrationTime	Double	PID integration time (seconds)
– DerivativeFilterCutOffFrequency	Double	PID derivative filter cut off frequency (Hz)
– GKP	Double	Variable PID proportional gain multiplier
– GKI	Double	Variable PID integral gain multiplier
– GKD	Double	Variable PID derivative gain multiplier
– KForm	Double	Variable PID form coefficient
– FeedForwardGainAcceleration	Double	Acceleration feedforward gain
– FeedForwardGainJerk	Double	Jerk feedforward gain

Output parameters

– None

Return

– Error	Long	Function error code

Matlab

Prototype

[Error] **PositionerCorrectorPIDFFAccelerationSet** (int32 SocketID, cstring FullPositionerName, bool ClosedLoopStatus, double KP, double KI, double KD, double KS, double IntegrationTime, double DerivativeFilterCutOffFrequency, double GKP, double GKI, double GKD, double KForm, double FeedForwardGainAcceleration, double FeedForwardGainJerk)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name
– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	PID servo loop proportional gain
– KI	double	PID servo loop integral gain
– KD	double	PID servo loop derivative gain
– KS	double	PID integral saturation value (0 to 1)
– IntegrationTime	double	PID integration time (seconds)
– DerivativeFilterCutOffFrequency	double	PID derivative filter cut off frequency (Hz)
– GKP	double	Variable PID proportional gain multiplier
– GKI	double	Variable PID integral gain multiplier
– GKD	double	Variable PID derivative gain multiplier
– KForm	double	Variable PID form coefficient
– FeedForwardGainAcceleration	double	Acceleration feedforward gain
– FeedForwardGainJerk	double	Jerk feedforward gain

Return

– Error	int32	Function error code

Python

Prototype

[Error] **PositionerCorrectorPIDFFAccelerationSet** (integer SocketID, string FullPositionerName, bool ClosedLoopStatus, double KP, double KI, double KD, double KS, double IntegrationTime, double DerivativeFilterCutOffFrequency, double GKP, double GKI, double GKD, double KForm, double FeedForwardGainAcceleration, double FeedForwardGainJerk)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	PID servo loop proportional gain
– KI	double	PID servo loop integral gain
– KD	double	PID servo loop derivative gain
– KS	double	PID integral saturation value (0 to 1)
– IntegrationTime	double	PID integration time (seconds)
– DerivativeFilterCutOffFrequency	double	PID derivative filter cut off frequency (Hz)
– GKP	double	Variable PID proportional gain multiplier
– GKI	double	Variable PID integral gain multiplier
– GKD	double	Variable PID derivative gain multiplier
– KForm	double	Variable PID form coefficient
– FeedForwardGainAcceleration	double	Acceleration feedforward gain
– FeedForwardGainJerk	double	Jerk feedforward gain

Return

– Error	int	Function error code

2.2.4.35 PositionerCorrectorSR1AccelerationGet

Name

PositionerCorrectorSR1AccelerationGet – Gets the corrector "SR1Acceleration" parameters.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type and the corrector type: ERR_WRONG_OBJECT_TYPE (-8)
- Check output parameter type: ERR_WRONG_TYPE_BOOL (-12), ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function returns the SR1 corrector parameter current values.

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

The "CorrectorType" must be "SR1Acceleration" in the stages.ini file. This servo loop type is used when a constant value applied to the driver results in a constant acceleration of the stage.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_BOOL (-12)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerCorrectorSR1AccelerationGet $SocketID $FullPositionerName ClosedLoopStatus KP KI KV ObserverFrequency CompensationGainVelocity CompensationGainAcceleration CompensationGainJerk

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	SR1 corrector proportional gain (sec^{-2})
– KI	double	SR1 corrector integral gain (sec^{-3})
– KV	double	SR1 corrector velocity gain (sec^{-1})
– ObserverFrequency	double	SR1 observer frequency (Hz)
– CompensationGainVelocity	double	Velocity compensation gain (sec)
– CompensationGainAcceleration	double	Acceleration compensation gain (sec^2)
– CompensationGainJerk	double	Jerk compensation gain (sec^3)

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCorrectorSR1AccelerationGet** (int SocketID, char FullPositionerName[250], bool* ClosedLoopStatus, double* KP, double* KI, double* KV, double* ObserverFrequency, double* CompensationGainVelocity, double* CompensationGainJerk)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name

Output parameters

– ClosedLoopStatus	bool *	Position servo loop status (true=closed and false=opened)
– KP	double *	SR1 corrector proportional gain (sec^{-2})
– KI	double *	SR1 corrector integral gain (sec^{-3})
– KV	double *	SR1 corrector velocity gain (sec^{-1})
– ObserverFrequency	double *	SR1 observer frequency (Hz)
– CompensationGainVelocity	double *	Velocity compensation gain (sec)
– CompensationGainAcceleration	double *	Acceleration compensation gain (sec^2)
– CompensationGainJerk	double *	Jerk compensation gain (sec^3)

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCorrectorSR1AccelerationGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ClosedLoopStatus As Boolean, KP As Double, KI As Double, KV As Double, ObserverFrequency As Double, CompensationGainVelocity As Double, CompensationGainAcceleration As Double, CompensationGainJerk As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	SR1 corrector proportional gain (sec^{-2})
– KI	double	SR1 corrector integral gain (sec^{-3})
– KV	double	SR1 corrector velocity gain (sec^{-1})
– ObserverFrequency	double	SR1 observer frequency (Hz)
– CompensationGainVelocity	double	Velocity compensation gain (sec)
– CompensationGainAcceleration	double	Acceleration compensation gain (sec^2)
– CompensationGainJerk	double	Jerk compensation gain (sec^3)

Return

– Error	long	Function error code

Matlab

Prototype

[Error, ClosedLoopStatus, KP, KI, KV, ObserverFrequency, CompensationGainVelocity, CompensationGainAcceleration, CompensationGainJerk] **PositionerCorrectorSR1AccelerationGet** (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	SR1 corrector proportional gain (sec^{-2})
– KI	double	SR1 corrector integral gain (sec^{-3})
– KV	double	SR1 corrector velocity gain (sec^{-1})
– ObserverFrequency	double	SR1 observer frequency (Hz)
– CompensationGainVelocity	double	Velocity compensation gain (sec)
– CompensationGainAcceleration	double	Acceleration compensation gain (sec^2)
– CompensationGainJerk	double	Jerk compensation gain (sec^3)

Python

Prototype

[Error, ClosedLoopStatus, KP, KI, KV, ObserverFrequency, CompensationGainVelocity, CompensationGainAcceleration, CompensationGainJerk] **PositionerCorrectorSR1AccelerationGet** (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code
– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	SR1 corrector proportional gain (sec^{-2})
– KI	double	SR1 corrector integral gain (sec^{-3})
– KV	double	SR1 corrector velocity gain (sec^{-1})
– ObserverFrequency	double	SR1 observer frequency (Hz)
– CompensationGainVelocity	double	Velocity compensation gain (sec)
– CompensationGainAcceleration	double	Acceleration compensation gain (sec^2)
– CompensationGainJerk	double	Jerk compensation gain (sec^3)

2.2.4.36 PositionerCorrectorSR1AccelerationSet

Name

PositionerCorrectorSR1AccelerationSet – Sets the corrector "SR1Acceleration" parameters.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type and the corrector type: ERR_WRONG_OBJECT_TYPE (-8)
- Check input parameter type: ERR_WRONG_TYPE_BOOL (-12), ERR_WRONG_TYPE_DOUBLE (-14)
- Check parameter value: ERR_PARAMETER_OUT_OF_RANGE (-17)
- KP > 0
- KI > 0
- KV > 0
- ObserverFrequency $\in \left[0 : \frac{0.5}{\text{CorrectorISRPeriod}} \right]$

Description

This function configures the "SR1Acceleration" corrector parameters.

NOTES

This function can be used only with the XPS-Qn Precision Platform controller.

The "CorrectorType" parameter must be defined as "SR1Acceleration" in the "stages.ini" file, else ERR_WRONG_OBJECT_TYPE (-8) is returned. This servo loop type is used when a constant value applied to the driver results in a constant acceleration of the stage.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_BOOL (-12)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerCorrectorSR1AccelerationSet $SocketID $FullPositionerName $ClosedLoopStatus $KP $KI $KV $ObserverFrequency $CompensationGainVelocity $CompensationGainAcceleration $CompensationGainJerk

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	SR1 corrector proportional gain (sec^{-2})
– KI	double	SR1 corrector integral gain (sec^{-3})
– KV	double	SR1 corrector velocity gain (sec^{-1})
– ObserverFrequency	double	SR1 observer frequency (Hz)
– CompensationGainVelocity	double	Velocity compensation gain (sec)
– CompensationGainAcceleration	double	Acceleration compensation gain (sec^2)
– CompensationGainJerk	double	Jerk compensation gain (sec^3)

Output parameters

– None

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCorrectorSR1AccelerationSet** (int SocketID, char FullPositionerName[250], bool ClosedLoopStatus, double KP, double KI, double KV, double ObserverFrequency, double CompensationGainVelocity, double CompensationGainAcceleration, double CompensationGainJerk)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name
– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	SR1 corrector proportional gain (sec^{-2})
– KI	double	SR1 corrector integral gain (sec^{-3})
– KV	double	SR1 corrector velocity gain (sec^{-1})
– ObserverFrequency	double	SR1 observer frequency (Hz)
– CompensationGainVelocity	double	Velocity compensation gain (sec)
– CompensationGainAcceleration	double	Acceleration compensation gain (sec^2)
– CompensationGainJerk	double	Jerk compensation gain (sec^3)

Output parameters

– None

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCorrectorSR1AccelerationSet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal ClosedLoopStatus As Boolean, ByVal KP As Double, ByVal KI As Double, ByVal KV As Double, ByVal ObserverFrequency As Double, ByVal CompensationGainVelocity As Double, ByVal CompensationGainAcceleration As Double, ByVal CompensationGainJerk As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	SR1 corrector proportional gain (sec^{-2})
– KI	double	SR1 corrector integral gain (sec^{-3})
– KV	double	SR1 corrector velocity gain (sec^{-1})
– ObserverFrequency	double	SR1 observer frequency (Hz)
– CompensationGainVelocity	double	Velocity compensation gain (sec)
– CompensationGainAcceleration	double	Acceleration compensation gain (sec^2)
– CompensationGainJerk	double	Jerk compensation gain (sec^3)

Output parameters

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **PositionerCorrectorSR1AccelerationSet** (int32 SocketID, cstring FullPositionerName, bool ClosedLoopStatus, double KP, double KI, double KV, double ObserverFrequency, double CompensationGainVelocity, double CompensationGainAcceleration, double CompensationGainJerk)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name
– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	SR1 corrector proportional gain (sec^{-2})
– KI	double	SR1 corrector integral gain (sec^{-3})
– KV	double	SR1 corrector velocity gain (sec^{-1})
– ObserverFrequency	double	SR1 observer frequency (Hz)
– CompensationGainVelocity	double	Velocity compensation gain (sec)
– CompensationGainAcceleration	double	Acceleration compensation gain (sec^2)
– CompensationGainJerk	double	Jerk compensation gain (sec^3)

Return

– Error	int32	Function error code

Python

Prototype

[Error] **PositionerCorrectorSR1AccelerationSet** (integer SocketID, string FullPositionerName, bool ClosedLoopStatus, double KP, double KI, double KV, double ObserverFrequency, double CompensationGainVelocity, double CompensationGainAcceleration, double CompensationGainJerk)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name
–	ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
–	KP	double	SR1 corrector proportional gain (sec^{-2})
–	KI	double	SR1 corrector integral gain (sec^{-3})
–	KV	double	SR1 corrector velocity gain (sec^{-1})
–	ObserverFrequency	double	SR1 observer frequency (Hz)
–	CompensationGainVelocity	double	Velocity compensation gain (sec)
–	CompensationGainAcceleration	double	Acceleration compensation gain (sec^2)
–	CompensationGainJerk	double	Jerk compensation gain (sec^3)

Return

–	Error	int	Function error code

2.2.4.37 PositionerCorrectorSR1ObserverAccelerationGet

Name

PositionerCorrectorSR1ObserverAccelerationGet – Gets the corrector "SR1Acceleration" observer parameters.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type and the corrector type: ERR_WRONG_OBJECT_TYPE (-8)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function returns the SR1 observer parameter current values.

NOTES

This function can be used only with the XPS-Qn Precision Platform controller.

The "CorrectorType" must be "SR1Acceleration" in the stages.ini file. This servo loop type is used when a constant value applied to the driver results in a constant acceleration of the stage.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerCorrectorSR1ObserverAccelerationGet $SocketID $FullPositionerName ParameterA ParameterB ParameterC

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– ParameterA	double	SR1 observer parameter A
– ParameterB	double	SR1 observer parameter B
– ParameterC	double	SR1 observer parameter C

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCorrectorSR1ObserverAccelerationGet** (int SocketID, char FullPositionerName[250], double* ParameterA, double* ParameterB, double* ParameterC)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name

Output parameters

– ParameterA	double *	SR1 observer parameter A
– ParameterB	double *	SR1 observer parameter B
– ParameterC	double *	SR1 observer parameter C

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCorrectorSR1ObserverAccelerationGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ParameterA As Double, ParameterB As Double, ParameterC As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– ParameterA	double	SR1 observer parameter A
– ParameterB	double	SR1 observer parameter B
– ParameterC	double	SR1 observer parameter C

Return

– Error	long	Function error code

Matlab

Prototype

[Error, ParameterA, ParameterB, ParameterC]
PositionerCorrectorSR1ObserverAccelerationGet (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– ParameterA	double	SR1 observer parameter A
– ParameterB	double	SR1 observer parameter B
– ParameterC	double	SR1 observer parameter C

Python

Prototype

[Error, ParameterA, ParameterB, ParameterC]
PositionerCorrectorSR1ObserverAccelerationGet (integer SocketID, string FullPositionerName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name

Return

–	Error	int	Function error code
–	ParameterA	double	SR1 observer parameter A
–	ParameterB	double	SR1 observer parameter B
–	ParameterC	double	SR1 observer parameter C

2.2.4.38 PositionerCorrectorSR1ObserverAccelerationSet

Name

PositionerCorrectorSR1ObserverAccelerationSet – Sets the corrector "SR1Acceleration" observer parameters.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type and the corrector type: ERR_WRONG_OBJECT_TYPE (-8)
- Check input parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function sets the SR1 observer parameters.

NOTES

This function can be used only with the XPS-Qn Precision Platform controller.

This function set the SR1 observer parameters, so it erases all of the observer parameter values previously calculated by the PositionerCorrectorSR1AccelerationSet

The "CorrectorType" parameter must be defined as "SR1Acceleration" in the "stages.ini" file, else ERR_WRONG_OBJECT_TYPE (-8) is returned. This servo loop type is used when a constant value applied to the driver results in a constant acceleration of the stage.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerCorrectorSR1ObserverAccelerationSet $SocketID $FullPositionerName $ParameterA $ParameterB $ParameterC

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name
–	ParameterA	double	SR1 observer parameter A
–	ParameterB	double	SR1 observer parameter B
–	ParameterC	double	SR1 observer parameter C

Output parameters

– None

Return

–	Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCorrectorSR1ObserverAccelerationSet** (int SocketID, char FullPositionerName[250], double ParameterA, double ParameterB, double ParameterC)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	char *	Positioner name
–	ParameterA	double	SR1 observer parameter A
–	ParameterB	double	SR1 observer parameter B
–	ParameterC	double	SR1 observer parameter C

Output parameters

– None

Return

–	Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCorrectorSR1ObserverAccelerationSet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal ParameterA As Double, ByVal ParameterB As Double, ByVal ParameterC As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– ParameterA	double	SR1 observer parameter A
– ParameterB	double	SR1 observer parameter B
– ParameterC	double	SR1 observer parameter C

Output parameters

– None

– Return

– Error	long	Function error code

Matlab

Prototype

[Error] **PositionerCorrectorSR1ObserverAccelerationSet** (int32 SocketID, cstring FullPositionerName, double ParameterA, double ParameterB, double ParameterC)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name
– ParameterA	double	SR1 observer parameter A
– ParameterB	double	SR1 observer parameter B
– ParameterC	double	SR1 observer parameter C

Return

– Error	int32	Function error code

Python

Prototype

[Error] **PositionerCorrectorSR1ObserverAccelerationSet** (integer SocketID, string FullPositionerName, double ParameterA, double ParameterB, double ParameterC)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– ParameterA	double	SR1 observer parameter A
– ParameterB	double	SR1 observer parameter B
– ParameterC	double	SR1 observer parameter C

Return

– Error	int	Function error code

2.2.4.39 PositionerCorrectorSR1OffsetAccelerationGet

Name

PositionerCorrectorSR1OffsetAccelerationGet – Gets the corrector "SR1Acceleration" acceleration output offset.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type and the corrector type: ERR_WRONG_OBJECT_TYPE (-8)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function gets the SR1 corrector acceleration output offset current value.

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

The "CorrectorType" must be "SR1Acceleration" in the stages.ini file. This servo loop type is used when a constant value applied to the driver results in a constant acceleration of the stage.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerCorrectorSR1OffsetAccelerationGet $SocketID $FullPositionerName AccelerationOffset

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– AccelerationOffset	double	SR1 corrector acceleration output offset

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCorrectorSR1OffsetAccelerationGet** (int SocketID, char FullPositionerName[250], double* AccelerationOffset)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name

Output parameters

– AccelerationOffset	double *	SR1 corrector acceleration output offset

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCorrectorSR1OffsetAccelerationGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, AccelerationOffset As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– AccelerationOffset	double	SR1 corrector acceleration output offset
– Return		
– Error	long	Function error code

Matlab

Prototype

[Error, AccelerationOffset] **PositionerCorrectorSR1OffsetAccelerationGet** (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– AccelerationOffset	double	SR1 corrector acceleration output offset

Python

Prototype

[Error, AccelerationOffset] **PositionerCorrectorSR1OffsetAccelerationGet** (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code
– AccelerationOffset	double	SR1 corrector acceleration output offset

2.2.4.40 PositionerCorrectorSR1OffsetAccelerationSet

Name

PositionerCorrectorSR1OffsetAccelerationSet – Sets the corrector "SR1Acceleration" acceleration output offset.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type and the corrector type: ERR_WRONG_OBJECT_TYPE (-8)
- Check input parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function sets the SR1 acceleration output offset.

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

The default value of the SR1 acceleration output offset is zero (0) at controller reboot.

The "CorrectorType" parameter must be defined as "SR1Acceleration" in the "stages.ini" file, else ERR_WRONG_OBJECT_TYPE (-8) is returned. This servo loop type is used when a constant value applied to the driver results in a constant acceleration of the stage.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerCorrectorSR1OffsetAccelerationSet $SocketID $FullPositionerName $AccelerationOffset

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– AccelerationOffset	double	SR1 corrector acceleration output offset

Output parameters

– None

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCorrectorSR1OffsetAccelerationSet** (int SocketID, char FullPositionerName[250], double AccelerationOffset)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name
– AccelerationOffset	double	SR1 corrector acceleration output offset

Output parameters

– None

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCorrectorSR1OffsetAccelerationSet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal AccelerationOffset As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– AccelerationOffset	double	SR1 corrector acceleration output offset

Output parameters

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **PositionerCorrectorSR1OffsetAccelerationSet** (int32 SocketID, cstring FullPositionerName, double AccelerationOffset)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name
– AccelerationOffset	double	SR1 corrector acceleration output offset

Return

– Error	int32	Function error code

Python

Prototype

[Error] **PositionerCorrectorSR1OffsetAccelerationSet** (integer SocketID, string FullPositionerName, double AccelerationOffset)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– AccelerationOffset	double	SR1 corrector acceleration output offset

Return

– Error	int	Function error code

2.2.4.41 PositionerCorrectorPIDFFVelocityGet

Name

PositionerCorrectorPIDFFVelocityGet – Gets the corrector "PIDFFVelocity" parameters.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type and the corrector type: ERR_WRONG_OBJECT_TYPE (-8)
- Check output parameter type: ERR_WRONG_TYPE_BOOL (-12), ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function returns the corrector parameter values used by a PID with a velocity output:

ClosedLoopStatus, KP, KI, KD, KS, IntegrationTime, DerivativeFilterCutOffFrequency, GKP, GKI, GKD, Kform and FeedForwardGainVelocity.

NOTE

The "CorrectorType" must be "PIDFFVelocity" in the stages.ini file. This servo loop type is used when a constant value applied to the driver results in a constant velocity of the stage.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_BOOL (-12)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerCorrectorPIDFFVelocityGet $SocketID $FullPositionerName ClosedLoopStatus KP KI KD KS IntegrationTime DerivativeFilterCutOffFrequency GKP GKI GKD KForm FeedForwardGainVelocity

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	PID servo loop proportional gain
– KI	double	PID servo loop integral gain
– KD	double	PID servo loop derivative gain
– KS	double	PID integral saturation value (0 to 1)
– IntegrationTime	double	PID integration time (seconds)
– DerivativeFilterCutOffFrequency	double	PID derivative filter cut off frequency (Hz)
– GKP	double	Variable PID proportional gain multiplier
– GKI	double	Variable PID integral gain multiplier
– GKD	double	Variable PID derivative gain multiplier
– KForm	double	Variable PID form coefficient
– FeedForwardGainVelocity	double	Velocity feedforward gain (units)

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCorrectorPIDFFVelocityGet** (int SocketID, char FullPositionerName[250], bool* ClosedLoopStatus, double* KP, double* KI, double* KD, double* KS, double* IntegrationTime, double* DerivativeFilterCutOffFrequency, double* GKP, double* GKI, double* GKD, double* KForm, double* FeedForwardGainVelocity)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name

Output parameters

– ClosedLoopStatus	bool *	Position servo loop status (true=closed and false=opened)
– KP	double *	PID servo loop proportional gain
– KI	double *	PID servo loop integral gain
– KD	double *	PID servo loop derivative gain
– KS	double *	PID integral saturation value (0 to 1)
– IntegrationTime	double *	PID integration time (seconds)
– DerivativeFilterCutOffFrequency	double *	PID derivative filter cut off frequency (Hz)
– GKP	double *	Variable PID proportional gain multiplier
– GKI	double *	Variable PID integral gain multiplier
– GKD	double *	Variable PID derivative gain multiplier
– KForm	double *	Variable PID form coefficient
– FeedForwardGainVelocity	double *	Velocity feedforward gain (units)

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCorrectorPIDFFVelocityGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ClosedLoopStatus As Boolean, KP As Double, KI As Double, KD As Double, KS As Double, IntegrationTime As Double, DerivativeFilterCutOffFrequency As Double, GKP As Double, GKI As Double, GKD As Double, KForm As Double, FeedForwardGainVelocity As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	PID servo loop proportional gain
– KI	double	PID servo loop integral gain
– KD	double	PID servo loop derivative gain
– KS	double	PID integral saturation value (0 to 1)
– IntegrationTime	double	PID integration time (seconds)
– DerivativeFilterCutOffFrequency	double	PID derivative filter cut off frequency (Hz)
– GKP	double	Variable PID proportional gain multiplier
– GKI	double	Variable PID integral gain multiplier
– GKD	double	Variable PID derivative gain multiplier
– KForm	double	Variable PID form coefficient
– FeedForwardGainVelocity	double	Velocity feedforward gain (units)

Return

– Error	long	Function error code

Matlab

Prototype

[Error, ClosedLoopStatus, KP, KI, KD, KS, IntegrationTime, DerivativeFilterCutOffFrequency, GKP, GKI, GKD, KForm, FeedForwardGainVelocity] **PositionerCorrectorPIDFFVelocityGet** (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	PID servo loop proportional gain
– KI	double	PID servo loop integral gain
– KD	double	PID servo loop derivative gain
– KS	double	PID integral saturation value (0 to 1)
– IntegrationTime	double	PID integration time (seconds)
– DerivativeFilterCutOffFrequency	double	PID derivative filter cut off frequency (Hz)
– GKP	double	Variable PID proportional gain multiplier
– GKI	double	Variable PID integral gain multiplier
– GKD	double	Variable PID derivative gain multiplier
– KForm	double	Variable PID form coefficient
– FeedForwardGainVelocity	double	Velocity feedforward gain (units)

Python

Prototype

[Error, ClosedLoopStatus, KP, KI, KD, KS, IntegrationTime, DerivativeFilterCutOffFrequency, GKP, GKI, GKD, KForm, FeedForwardGainVelocity] **PositionerCorrectorPIDFFVelocityGet** (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code
– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	PID servo loop proportional gain
– KI	double	PID servo loop integral gain
– KD	double	PID servo loop derivative gain
– KS	double	PID integral saturation value (0 to 1)
– IntegrationTime	double	PID integration time (seconds)
– DerivativeFilterCutOffFrequency	double	PID derivative filter cut off frequency (Hz)
– GKP	double	Variable PID proportional gain multiplier
– GKI	double	Variable PID integral gain multiplier
– GKD	double	Variable PID derivative gain multiplier
– KForm	double	Variable PID form coefficient
– FeedForwardGainVelocity	double	Velocity feedforward gain (units)

2.2.4.42 PositionerCorrectorPIDFFVelocitySet

Name

PositionerCorrectorPIDFFVelocitySet – Configures the corrector "PIDFFVelocity" parameters.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type and the corrector type: ERR_WRONG_OBJECT_TYPE (-8)
- Check input parameter type: ERR_WRONG_TYPE_BOOL (-12), ERR_WRONG_TYPE_DOUBLE (-14)
- Check parameter value: ERR_PARAMETER_OUT_OF_RANGE (-17)
- KP ≥ 0
- KI ≥ 0
- KD ≥ 0
- 0 ≤ KS ≤ 1
- IntegrationTime ≥ CorrectorISRPeriod
- GKP > -1
- GKI > -1
- GKD > -1
- KForm ≥ 0
- KFeedForwardVelocity ≥ 0
- DerivativeFilterCutOffFrequency $\in \left[0 : \frac{0.5}{\text{CorrectorISRPeriod}}\right]$

NOTE

Refer to *system.ref* file to get CorrectorISRPeriod value.

Description

This function configures the "PIDFFVelocity" corrector parameters.

NOTE

The "CorrectorType" parameter must be defined as "PIDFFVelocity" in the stages.ini file else ERR_WRONG_OBJECT_TYPE (-8) is returned. This servo loop type is used when a constant value applied to the driver results in a constant velocity of the stage.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_BOOL (-12)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerCorrectorPIDFFVelocitySet $SocketID $FullPositionerName $ClosedLoopStatus $KP $KI $KD $KS $IntegrationTime $DerivativeFilterCutOffFrequency $GKP $GKI $GKD $KForm $FeedForwardGainVelocity

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name
–	ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
–	KP	double	PID servo loop proportional gain
–	KI	double	PID servo loop integral gain
–	KD	double	PID servo loop derivative gain
–	KS	double	PID integral saturation value (0 to 1)
–	IntegrationTime	double	PID integration time (seconds)
–	DerivativeFilterCutOffFrequency	double	PID derivative filter cut off frequency (Hz)
–	GKP	double	Variable PID proportional gain multiplier
–	GKI	double	Variable PID integral gain multiplier
–	GKD	double	Variable PID derivative gain multiplier
–	KForm	double	Variable PID form coefficient
–	FeedForwardGainVelocity	double	Velocity feedforward gain (units)

Output parameters

– None

Return

–	Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCorrectorPIDFFVelocitySet** (int SocketID, char FullPositionerName[250], bool ClosedLoopStatus, double KP, double KI, double KD, double KS, double IntegrationTime, double DerivativeFilterCutOffFrequency, double GKP, double GKI, double GKD, double KForm, double FeedForwardGainVelocity)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name
– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	PID servo loop proportional gain
– KI	double	PID servo loop integral gain
– KD	double	PID servo loop derivative gain
– KS	double	PID integral saturation value (0 to 1)
– IntegrationTime	double	PID integration time (seconds)
– DerivativeFilterCutOffFrequency	double	PID derivative filter cut off frequency (Hz)
– GKP	double	Variable PID proportional gain multiplier
– GKI	double	Variable PID integral gain multiplier
– GKD	double	Variable PID derivative gain multiplier
– KForm	double	Variable PID form coefficient
– FeedForwardGainVelocity	double	Velocity feedforward gain (units)

Output parameters

– None

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCorrectorPIDFFVelocitySet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal ClosedLoopStatus As Boolean, ByVal KP As Double, ByVal KI As Double, ByVal KD As Double, ByVal KS As Double, ByVal IntegrationTime As Double, ByVal DerivativeFilterCutOffFrequency As Double, ByVal GKP As Double, ByVal GKI As Double, ByVal GKD As Double, ByVal KForm As Double, ByVal FeedForwardGainVelocity As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	PID servo loop proportional gain
– KI	double	PID servo loop integral gain
– KD	double	PID servo loop derivative gain
– KS	double	PID integral saturation value (0 to 1)
– IntegrationTime	double	PID integration time (seconds)
– DerivativeFilterCutOffFrequency	double	PID derivative filter cut off frequency (Hz)
– GKP	double	Variable PID proportional gain multiplier
– GKI	double	Variable PID integral gain multiplier
– GKD	double	Variable PID derivative gain multiplier
– KForm	double	Variable PID form coefficient
– FeedForwardGainVelocity	double	Velocity feedforward gain (units)

Output parameters

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **PositionerCorrectorPIDFFVelocitySet** (int32 SocketID, cstring FullPositionerName, bool ClosedLoopStatus, double KP, double KI, double KD, double KS, double IntegrationTime, double DerivativeFilterCutOffFrequency, double GKP, double GKI, double GKD, double KForm, double FeedForwardGainVelocity)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name
– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	PID servo loop proportional gain
– KI	double	PID servo loop integral gain
– KD	double	PID servo loop derivative gain
– KS	double	PID integral saturation value (0 to 1)
– IntegrationTime	double	PID integration time (seconds)
– DerivativeFilterCutOffFrequency	double	PID derivative filter cut off frequency (Hz)
– GKP	double	Variable PID proportional gain multiplier
– GKI	double	Variable PID integral gain multiplier
– GKD	double	Variable PID derivative gain multiplier
– KForm	double	Variable PID form coefficient
– FeedForwardGainVelocity	double	Velocity feedforward gain (units)

Return

– Error	int32	Function error code

Python

Prototype

[Error] **PositionerCorrectorPIDFFVelocitySet** (integer SocketID, string FullPositionerName, bool ClosedLoopStatus, double KP, double KI, double KD, double KS, double IntegrationTime, double DerivativeFilterCutOffFrequency, double GKP, double GKI, double GKD, double KForm, double FeedForwardGainVelocity)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	PID servo loop proportional gain
– KI	double	PID servo loop integral gain
– KD	double	PID servo loop derivative gain
– KS	double	PID integral saturation value (0 to 1)
– IntegrationTime	double	PID integration time (seconds)
– DerivativeFilterCutOffFrequency	double	PID derivative filter cut off frequency (Hz)
– GKP	double	Variable PID proportional gain multiplier
– GKI	double	Variable PID integral gain multiplier
– GKD	double	Variable PID derivative gain multiplier
– KForm	double	Variable PID form coefficient
– FeedForwardGainVelocity	double	Velocity feedforward gain (units)

Return

– Error	int	Function error code

2.2.4.43 PositionerCorrectorPIPositionGet

Name

PositionerCorrectorPIPositionGet – Gets the corrector "PIPosition" parameters.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type and the corrector type: ERR_WRONG_OBJECT_TYPE (-8)
- Check output parameter type: ERR_WRONG_TYPE_BOOL (-12), ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function returns the corrector parameter values used by a PI with a position output: ClosedLoopStatus, KP, KI and IntegrationTime.

NOTE

The "CorrectorType" must be "PIPosition" in the stages.ini file. This servo loop type is used when the position servo loop outputs a position value directly.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_BOOL (-12)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerCorrectorPIPositionGet $SocketID $FullPositionerName ClosedLoopStatus KP KI IntegrationTime

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	PID servo loop proportional gain
– KI	double	PID servo loop integral gain
– IntegrationTime	double	PID integration time (seconds)

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCorrectorPIPositionGet** (int SocketID, char FullPositionerName[250], bool* ClosedLoopStatus, double* KP, double* KI, double* IntegrationTime)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name

Output parameters

– ClosedLoopStatus	bool *	Position servo loop status (true=closed and false=opened)
– KP	double *	PID servo loop proportional gain
– KI	double *	PID servo loop integral gain
– IntegrationTime	double *	PID integration time (seconds)

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCorrectorPIPositionGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ClosedLoopStatus As Boolean, KP As Double, KI As Double, IntegrationTime As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	PID servo loop proportional gain
– KI	double	PID servo loop integral gain
– IntegrationTime	double	PID integration time (seconds)

Return

– Error	long	Function error code

Matlab

Prototype

[Error, ClosedLoopStatus, KP, KI, IntegrationTime]
PositionerCorrectorPIPositionGet (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	PID servo loop proportional gain
– KI	double	PID servo loop integral gain
– IntegrationTime	double	PID integration time (seconds)

Python

Prototype

[Error, ClosedLoopStatus, KP, KI, IntegrationTime]
PositionerCorrectorPIPositionGet (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code
– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	PID servo loop proportional gain
– KI	double	PID servo loop integral gain
– IntegrationTime	double	PID integration time (seconds)

2.2.4.44 PositionerCorrectorPIPositionSet

Name

PositionerCorrectorPIPositionSet – Configures the corrector "PIPosition" parameters.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type and the corrector type: ERR_WRONG_OBJECT_TYPE (-8)
- Check input parameter type: ERR_WRONG_TYPE_BOOL (-12), ERR_WRONG_TYPE_DOUBLE (-14)
- Check parameter value: ERR_PARAMETER_OUT_OF_RANGE (-17)
- KP ≥ 0
- KI ≥ 0
- IntegrationTime ≥ CorrectorISRPeriod

NOTE

Refer to *system.ref* file to get CorrectorISRPeriod value.

Description

This function configures the "PIPosition" corrector parameters.

NOTE

The "CorrectorType" parameter must be defined as "PIPosition" in the stages.ini file else ERR_WRONG_OBJECT_TYPE (-8) is returned. This servo loop type is used when the position servo loop outputs a position value directly.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_BOOL (-12)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerCorrectorPIPositionSet $SocketID $FullPositionerName $ClosedLoopStatus $KP $KI $IntegrationTime

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	PID servo loop proportional gain
– KI	double	PID servo loop integral gain
– IntegrationTime	double	PID integration time (seconds)

Output parameters

– None

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCorrectorPIPositionSet** (int SocketID, char FullPositionerName[250], bool ClosedLoopStatus, double KP, double KI, double IntegrationTime)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name
– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	PID servo loop proportional gain
– KI	double	PID servo loop integral gain
– IntegrationTime	double	PID integration time (seconds)

Output parameters

– None

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCorrectorPIPositionSet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal ClosedLoopStatus As Boolean, ByVal KP As Double, ByVal KI As Double, ByVal IntegrationTime As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	PID servo loop proportional gain
– KI	double	PID servo loop integral gain
– IntegrationTime	double	PID integration time (seconds)

Output parameters

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **PositionerCorrectorPIPositionSet** (int32 SocketID, cstring FullPositionerName, bool ClosedLoopStatus, double KP, double KI, double IntegrationTime)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name
– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	PID servo loop proportional gain
– KI	double	PID servo loop integral gain
– IntegrationTime	double	PID integration time (seconds)

Return

– Error	int32	Function error code

Python

Prototype

[Error] **PositionerCorrectorPIPositionSet** (integer SocketID, string FullPositionerName, bool ClosedLoopStatus, double KP, double KI, double IntegrationTime)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– ClosedLoopStatus	bool	Position servo loop status (true=closed and false=opened)
– KP	double	PID servo loop proportional gain
– KI	double	PID servo loop integral gain
– IntegrationTime	double	PID integration time (seconds)

Return

– Error	int	Function error code

2.2.4.45 PositionerCorrectorTypeGet

Name

PositionerCorrectorTypeGet – Returns the corrector type.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type: ERR_WRONG_OBJECT_TYPE (-8)
- Check output parameter type: ERR_WRONG_TYPE_CHAR (-13)

Description

This function returns the corrector type used by the selected positioner.

The corrector type can be one of this list:

- PositionerCorrectorPIDFFAcceleration
- PositionerCorrectorSR1Acceleration
- PositionerCorrectorPIDFFVelocity
- PositionerCorrectorPIDDualFFVoltage
- PositionerCorrectorPIPosition
- NoCorrector

NOTE

The corrector type is defined in the stages.ini file with the "CorrectorType" parameter.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13)
- SUCCESS (0): no error

TCL

Prototype

PositionerCorrectorTypeGet $SocketID $FullPositionerName CorrectorType

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– CorrectorType	string	Corrector type

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCorrectorTypeGet** (int SocketID, char FullPositionerName[250], char* CorrectorType)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name

Output parameters

– CorrectorType	char *	Corrector type

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCorrectorTypeGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal CorrectorType As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– CorrectorType	string	Corrector type

Return

– Error	long	Function error code

Matlab

Prototype

[Error, CorrectorType] **PositionerCorrectorTypeGet** (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– CorrectorType	cstring	Corrector type

Python

Prototype

[Error, CorrectorType] **PositionerCorrectorTypeGet** (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code
– CorrectorType	string	Corrector type

2.2.4.46 PositionerCurrentVelocityAccelerationFiltersGet

Name

PositionerCurrentVelocityAccelerationFiltersGet – Gets the velocity and acceleration filter cut off frequencies.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type: ERR_WRONG_OBJECT_TYPE (-8)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function returns the current velocity cut off frequency and the current acceleration cut off frequency used by gathering for the selected positioner.

Gathering uses these parameters to filter the current velocity and the current acceleration. These parameters are defined in the stages.ini file.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerCurrentVelocityAccelerationFiltersGet $SocketID $FullPositionerName VelocityCutOffFrequency AccelerationCutOffFrequency

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– VelocityCutOffFrequency	double	Velocity filter cut off frequency (Hz)
– AccelerationCutOffFrequency	double	Acceleration filter cut off frequency (Hz)

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCurrentVelocityAccelerationFiltersGet** (int SocketID, char FullPositionerName[250] , double* VelocityCutOffFrequency, double* AccelerationCutOffFrequency)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name

Output parameters

– VelocityCutOffFrequency	double *	Velocity filter cut off frequency (Hz)
– AccelerationCutOffFrequency	double *	Acceleration filter cut off frequency (Hz)

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCurrentVelocityAccelerationFiltersGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, VelocityCutOffFrequency As Double, AccelerationCutOffFrequency As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– VelocityCutOffFrequency	double	Velocity filter cut off frequency (Hz)
– AccelerationCutOffFrequency	double	Acceleration filter cut off frequency (Hz)

Return

– Error	long	Function error code

Matlab

Prototype

[Error, VelocityCutOffFrequency, AccelerationCutOffFrequency] **PositionerCurrentVelocityAccelerationFiltersGet** (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– VelocityCutOffFrequency	double	Velocity filter cut off frequency (Hz)
– AccelerationCutOffFrequency	double	Acceleration filter cut off frequency (Hz)

Python

Prototype

[Error, VelocityCutOffFrequency, AccelerationCutOffFrequency]
PositionerCurrentVelocityAccelerationFiltersGet (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code
– VelocityCutOffFrequency	double	Velocity filter cut off frequency (Hz)
– AccelerationCutOffFrequency	double	Acceleration filter cut off frequency (Hz)

2.2.4.47 PositionerCurrentVelocityAccelerationFiltersSet

Name

PositionerCurrentVelocityAccelerationFiltersSet – Sets the velocity and acceleration filter cut off frequencies.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type: ERR_WRONG_OBJECT_TYPE (-8)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)
- Check parameter value: ERR_PARAMETER_OUT_OF_RANGE (-17)
- VelocityCutOffFrequency $\in \left[0 : \frac{0.5}{\text{CorrectorISRPeriod}} \right]$
- AccelerationCutOffFrequency $\in \left[0 : \frac{0.5}{\text{CorrectorISRPeriod}} \right]$

NOTE

Refer to *system.ref* file to get CorrectorISRPeriod value.

Description

This function sets a new velocity cut off frequency and a new acceleration cut off frequency for the selected positioner.

Gathering uses these parameters to filter the current velocity and the current acceleration. These parameters are defined in the stages.ini file.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerCurrentVelocityAccelerationFiltersSet $SocketID $FullPositionerName $VelocityCutOffFrequency $AccelerationCutOffFrequency

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– VelocityCutOffFrequency	double	Velocity filter cut off frequency (Hz)
– AccelerationCutOffFrequency	double	Acceleration filter cut off frequency (Hz)

Output parameters

– None

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerCurrentVelocityAccelerationFiltersSet** (int SocketID, char FullPositionerName[250] , double VelocityCutOffFrequency, double AccelerationCutOffFrequency)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name
– VelocityCutOffFrequency	double	Velocity filter cut off frequency (Hz)
– AccelerationCutOffFrequency	double	Acceleration filter cut off frequency (Hz)

Output parameters

– None

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerCurrentVelocityAccelerationFiltersSet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal VelocityCutOffFrequency As Double, ByVal AccelerationCutOffFrequency As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– VelocityCutOffFrequency	double	Velocity filter cut off frequency (Hz)
– AccelerationCutOffFrequency	double	Acceleration filter cut off frequency (Hz)

Output parameters

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **PositionerCurrentVelocityAccelerationFiltersSet** (int32 SocketID, cstring FullPositionerName, double VelocityCutOffFrequency, double AccelerationCutOffFrequency)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name
– VelocityCutOffFrequency	double	Velocity filter cut off frequency (Hz)
– AccelerationCutOffFrequency	double	Acceleration filter cut off frequency (Hz)

Return

– Error	int32	Function error code

Python

Prototype

[Error] **PositionerCurrentVelocityAccelerationFiltersSet** (integer SocketID, string FullPositionerName, double VelocityCutOffFrequency, double AccelerationCutOffFrequency)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name
–	VelocityCutOffFrequency	double	Velocity filter cut off frequency (Hz)
–	AccelerationCutOffFrequency	double	Acceleration filter cut off frequency (Hz)

Return

–	Error	int	Function error code

2.2.4.48 PositionerDriverFiltersGet

Name

PositionerDriverFiltersGet – Gets the piezo driver notch and lowpass filters parameters.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type: ERR_WRONG_OBJECT_TYPE (-8)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)
- Check driver type, if not PIEZO: ERR_UNCOMPATIBLE (-24)
- If piezo driver, check if driver is not initialized: ERR_NOT_ALLOWED_DRIVER_NOT_INITIALIZED (-118)

Description

This function returns current values of the piezo driver filters parameters (KI, notch frequency, notch bandwidth, notch gain, lowpass frequency).

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- ERR_UNCOMPATIBLE (-24)
- ERR_NOT_ALLOWED_DRIVER_NOT_INITIALIZED (-118)
- SUCCESS (0): no error

TCL

Prototype

PositionerDriverFiltersGet $SocketID $FullPositionerName KI NotchFrequency NotchBandwidth NotchGain LowpassFrequency

Input parameters

– SocketID	int	Socket identifier got from "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– KI	double	Driver KI
– NotchFrequency	double	Driver notch frequency (Hz)
– NotchBandwidth	double	Driver notch bandwidth (Hz)
– NotchGain	double	Driver notch gain
– LowpassFrequency	double	Driver lowpass frequency (Hz)

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerDriverFiltersGet** (int SocketID, char FullPositionerName[250] , double * KI, double* NotchFrequency, double* NotchBandwidth, double* NotchGain, double* LowpassFrequency)

Input parameters

– SocketID	int	Socket identifier got from "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name

Output parameters

– KI	double *	Driver KI
– NotchFrequency	double *	Driver notch frequency (Hz)
– NotchBandwidth	double *	Driver notch bandwidth (Hz)
– NotchGain	double *	Driver notch gain
– LowpassFrequency	double *	Driver lowpass frequency (Hz)

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerDriverFiltersGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, KI As Double, NotchFrequency As Double, NotchBandwidth As Double, NotchGain As Double, LowpassFrequency As Double)

Input parameters

– SocketID	long	Socket identifier got from "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– KI	double	Driver KI
– NotchFrequency	double	Driver notch frequency (Hz)
– NotchBandwidth	double	Driver notch bandwidth (Hz)
– NotchGain	double	Driver notch gain
– LowpassFrequency	double	Driver lowpass frequency (Hz)

Return

– Error	long	Function error code

Matlab

Prototype

[Error, KI, NotchFrequency, NotchBandwidth, NotchGain, LowpassFrequency] **PositionerDriverFiltersGet** (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier got from "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– KI	double	Driver KI
– NotchFrequency	double	Driver notch frequency (Hz)
– NotchBandwidth	double	Driver notch bandwidth (Hz)
– NotchGain	double	Driver notch gain
– LowpassFrequency	double	Driver lowpass frequency (Hz)

Python

Prototype

[Error, KI, NotchFrequency, NotchBandwidth,NotchGain, LowpassFrequency]
PositionerDriverFiltersGet (integer SocketID, string FullPositionerName)

Input parameters

–	SocketID	int	Socket identifier got from "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name

Return

–	Error	int	Function error code
–	KI	double	Driver KI
–	NotchFrequency	double	Driver notch frequency (Hz)
–	NotchBandwidth	double	Driver notch bandwidth (Hz)
–	NotchGain	double	Driver notch gain
–	LowpassFrequency	double	Driver lowpass frequency (Hz)

2.2.4.49 PositionerDriverFiltersSet

Name

PositionerDriverFiltersSet – Sets the piezo driver filters parameters.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type: ERR_WRONG_OBJECT_TYPE (-8)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)
- Check parameter value: ERR_PARAMETER_OUT_OF_RANGE (-17)
- Check driver type, if not PIEZO: ERR_UNCOMPATIBLE (-24)
- If piezo driver, check group and driver status:
- If the driver is not initialized:
  ERR_NOT_ALLOWED_DRIVER_NOT_INITIALIZED (-118)
- If the group state is NOTREF or READY:
  ERR_FUNCTION_ONLY_ALLOWED_IN_DISABLED_STATE (-117)
- KI >= 0
- $NotchFrequency \in \left[0 : \frac{0.5}{\text{CorrectorISRPeriod}}\right]$
- $NotchBandwith \in \left[0 : \frac{0.5}{\text{CorrectorISRPeriod}}\right]$
- $NotchGain \in [0 : 100]$
- $LowpassFrequency \in \left[0 : \frac{0.5}{\text{CorrectorISRPeriod}}\right]$

NOTE

Refer to *system.ref* file to get CorrectorISRPeriod value.

Description

This function sets parameters of the driver (KI integral, notch and lowpass filters) for a piezo driver positioner.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- ERR_UNCOMPATIBLE (-24)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_MOTOR_INITIALIZATION_ERROR (-50)
- ERR_FUNCTION_ONLY_ALLOWED_IN_DISABLED_STATE (-117)
- ERR_NOT_ALLOWED_DRIVER_NOT_INITIALIZED (-118)
- SUCCESS (0): no error

TCL

Prototype

PositionerDriverFiltersSet $SocketID $FullPositionerName $KI $NotchFrequency $NotchBandwidth $NotchGain $LowpassFrequency

Input parameters

–	SocketID	int	Socket identifier got from "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name
–	KI	double	Driver KI
–	NotchFrequency	double	Driver notch frequency (Hz)
–	NotchBandwidth	double	Driver notch bandwidth (Hz)
–	NotchGain	double	Driver notch gain
–	LowpassFrequency	double	Driver lowpass frequency (Hz)
–	Output parameters		
–	None		

Return

–	Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerDriverFiltersSet** (int SocketID, char FullPositionerName[250] , double KI, double NotchFrequency, double NotchBandwidth, double NotchGain, double LowpassFrequency)

Input parameters

– SocketID	int	Socket identifier got from "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name
– KI	double	Driver KI
– NotchFrequency	double	Driver notch frequency (Hz)
– NotchBandwidth	double	Driver notch bandwidth (Hz)
– NotchGain	double	Driver notch gain
– LowpassFrequency	double	Driver lowpass frequency (Hz)

Output parameters

– None

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerDriverFiltersSet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal KI As Double, ByVal NotchFrequency As Double, ByVal NotchBandwidth As Double, ByVal NotchGain As Double, ByVal LowpassFrequency As Double)

Input parameters

– SocketID	long	Socket identifier got from "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– KI	double	Driver KI
– NotchFrequency	double	Driver notch frequency (Hz)
– NotchBandwidth	double	Driver notch bandwidth (Hz)
– NotchGain	double	Driver notch gain
– LowpassFrequency	double	Driver lowpass frequency (Hz)

Output parameters

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **PositionerDriverFiltersSet** (int32 SocketID, cstring FullPositionerName, double KI, double NotchFrequency, double NotchBandwidth, double NotchGain, double LowpassFrequency)

Input parameters

– SocketID	int32	Socket identifier got from "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name
– KI	double	Driver KI
– NotchFrequency	double	Driver notch frequency (Hz)
– NotchBandwidth	double	Driver notch bandwidth (Hz)
– NotchGain	double	Driver notch gain
– LowpassFrequency	double	Driver lowpass frequency (Hz)

Return

– Error	int32	Function error code

Python

Prototype

[Error] **PositionerDriverFiltersSet** (integer SocketID, string FullPositionerName, double KI, double NotchFrequency, double NotchBandwidth, double NotchGain, double LowpassFrequency)

Input parameters

– SocketID	int	Socket identifier got from "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– KI	double	Driver KI
– NotchFrequency	double	Driver notch frequency (Hz)
– NotchBandwidth	double	Driver notch bandwidth (Hz)
– NotchGain	double	Driver notch gain
– LowpassFrequency	double	Driver lowpass frequency (Hz)

Return

– Error	int	Function error code

2.2.4.50 PositionerDriverPositionOffsetsGet

Name

PositionerDriverPositionOffsetsGet – Gets the current value of piezo driver stage and gage position offsets.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type: ERR_WRONG_OBJECT_TYPE (-8)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)
- Check driver type, if not PIEZO: ERR_UNCOMPATIBLE (-24)
- If piezo driver, check group and driver status:
- If the driver is not initialized:
  ERR_NOT_ALLOWED_DRIVER_NOT_INITIALIZED (-118)
- If the group state is NOTREF or READY:
  ERR_FUNCTION_ONLY_ALLOWED_IN_DISABLED_STATE (-117)

Description

This function returns current value of the piezo driver position offset parameters (stage position offset, gage position offset).

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- ERR_UNCOMPATIBLE (-24)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_FUNCTION_ONLY_ALLOWED_IN_DISABLED_STATE (-117)
- ERR_NOT_ALLOWED_DRIVER_NOT_INITIALIZED (-118)
- SUCCESS (0): no error

TCL

Prototype

PositionerDriverPositionOffsetsGet $SocketID $FullPositionerName StagePositionOffset GagePositionOffset

Input parameters

– SocketID	int	Socket identifier got from "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– StagePositionOffset	double	Driver stage position offset (units)
– GagePositionOffset	double	Driver gage position offset (units)

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerDriverPositionOffsetsGet** (int SocketID, char FullPositionerName[250] , double* StagePositionOffset, double* GagePositionOffset)

Input parameters

– SocketID	int	Socket identifier got from "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name

Output parameters

– StagePositionOffset	double *	Driver stage position offset (units)
– GagePositionOffset	double *	Driver gage position offset (units)

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerDriverPositionOffsetsGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, StagePositionOffset As Double, GagePositionOffset As Double)

Input parameters

– SocketID	long	Socket identifier got from "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– StagePositionOffset	double	Driver stage position offset (units)
– GagePositionOffset	double	Driver gage position offset (units)

Return

– Error	long	Function error code

Matlab

Prototype

[Error, StagePositionOffset, GagePositionOffset] **PositionerDriverPositionOffsetsGet** (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier got from "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– StagePositionOffset	double	Driver stage position offset (units)
– GagePositionOffset	double	Driver gage position offset (units)

Python

Prototype

[Error, StagePositionOffset, GagePositionOffset] **PositionerDriverPositionOffsetsGet** (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier got from "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code
– StagePositionOffset	double	Driver stage position offset (units)
– GagePositionOffset	double	Driver gage position offset (units)

2.2.4.51 PositionerDriverStatusGet

Name

PositionerDriverStatusGet – Gets the positioner driver status code.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner (must be not a secondary positioner): ERR_WRONG_OBJECT_TYPE (-8), ERR_UNCOMPATIBLE (-24), ERR_POSITIONER_NAME (-18)
- Check output parameter type: ERR_WRONG_TYPE_INT (-15)

Description

This function returns the positioner driver status from the driver board.

Use the "PositionerDriverStatusStringGet" function to get the driver status description.

NOTE

See the positioner driver status list describes in § 2.21

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_UNCOMPATIBLE (-24)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

PositionerDriverStatusGet $SocketID $FullPositionerName PositionerDriverStatus

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name

Output parameters

–	PositionerDriverStatus	int	Driver status code

Return

–	Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerDriverStatusGet** (int SocketID, char FullPositionerName[250] , int * PositionerDriverStatus)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	char *	Positioner name

Output parameters

–	PositionerDriverStatus	int *	Driver status code

Return

–	Error	int	Function error code

Visual Basic

Prototype

Long **PositionerDriverStatusGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, PositionerDriverStatus As Integer)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name

Output parameters

–	PositionerDriverStatus	int	Driver status code

Return

–	Error	long	Function error code

Matlab

Prototype

[Error, PositionerDriverStatus] **PositionerDriverStatusGet** (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– PositionerDriverStatus	int32	Driver status code

Python

Prototype

[Error, PositionerDriverStatus] **PositionerDriverStatusGet** (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code
– PositionerDriverStatus	int	Driver status code

2.2.4.52 PositionerDriverStatusStringGet

Name

PositionerDriverStatusStringGet – Gets the positioner driver status description.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check input parameter type: ERR_WRONG_TYPE_INT (-15)
- Check output parameter type: ERR_WRONG_TYPE_CHAR (-13)

Description

This function returns a driver status description from a positioner driver status code.

NOTE

See the positioner driver status list described in § 2.21

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

PositionerDriverStatusStringGet $SocketID $FullPositionerName $PositionerDriverStatus PositionerDriverStatusString

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– PositionerDriverStatus	int	Driver status code

Output parameters

– PositionerDriverStatusString	int	Driver status description

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerDriverStatusStringGet** (int SocketID, char FullPositionerName[250] , int PositionerDriverStatus, char * PositionerDriverStatusString)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	char *	Positioner name
–	PositionerDriverStatus	int *	Driver status code

Output parameters

–	PositionerDriverStatusString	int *	Driver status description

Return

–	Error	int	Function error code

Visual Basic

Prototype

Long **PositionerDriverStatusStringGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal PositionerDriverStatus As Integer, ByVal PositionerDriverStatusString As String)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name
–	PositionerDriverStatus	int	Driver status code

Output parameters

–	PositionerDriverStatusString	int	Driver status description

Return

–	Error	long	Function error code

Matlab

Prototype

[Error, PositionerDriverStatusString] **PositionerDriverStatusStringGet** (int32 SocketID, cstring FullPositionerName, int32 PositionerDriverStatus)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	cstring	Positioner name
–	PositionerDriverStatus	int32	Driver status code

Return

–	Error	int32	Function error code
–	PositionerDriverStatusString	cstring	Driver status description

Python

Prototype

[Error, PositionerDriverStatusString] **PositionerDriverStatusStringGet** (integer SocketID, string FullPositionerName, integer PositionerDriverStatus)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name
–	PositionerDriverStatus	int	Driver status code

Return

–	Error	int	Function error code
–	PositionerDriverStatusString	string	Driver status description

2.2.4.53 PositionerEncoderAmplitudeValuesGet

Name

PositionerEncoderAmplitudeValuesGet – Gets the encoder amplitude values.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner: ERR_WRONG_OBJECT_TYPE (-8)
- Check the encoder type (must be "AnalogInterpolated"): ERR_WRONG_OBJECT_TYPE (-8)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function returns the maximum and current amplitudes values (in volts) of the analog encoder input.

CAUTION

The encoder type must be "AnalogInterpolated" in the stages.ini file ("EncoderType" parameter).

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerEncoderAmplitudeValuesGet $SocketID $FullPositionerName MaxSinusAmplitude CurrentSinusAmplitude MaxCosinusAmplitude CurrentCosinusAmplitude

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– MaxSinusAmplitude	double	Encoder sinus signal maximum amplitude value (Volts)
– CurrentSinusAmplitude	double	Encoder sinus signal current amplitude value (Volts)
– MaxCosinusAmplitude	double	Encoder cosinus signal maximum amplitude value (Volts)
– CurrentCosinusAmplitude	double	Encoder cosinus signal current amplitude value (Volts)

Return

– Error	int	TCL error code (0=success or 1=syntax error) or function error code

C/C++

Prototype

int **PositionerEncoderAmplitudeValuesGet** (int SocketID, char FullPositionerName[250] , double * MaxSinusAmplitude, double * CurrentSinusAmplitude, double * MaxCosinusAmplitude, double * CurrentCosinusAmplitude)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name

Output parameters

– MaxSinusAmplitude	double *	Encoder sinus signal maximum amplitude value (Volts)
– CurrentSinusAmplitude	double *	Encoder sinus signal current amplitude value (Volts)
– MaxCosinusAmplitude	double *	Encoder cosinus signal maximum amplitude value (Volts)
– CurrentCosinusAmplitude	double *	Encoder cosinus signal current amplitude value (Volts)

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerEncoderAmplitudeValuesGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, MaxSinusAmplitude As Double, CurrentSinusAmplitude As Double, MaxCosinusAmplitude As Double, CurrentCosinusAmplitude As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– MaxSinusAmplitude	double	Encoder sinus signal maximum amplitude value (Volts)
– CurrentSinusAmplitude	double	Encoder sinus signal current amplitude value (Volts)
– MaxCosinusAmplitude	double	Encoder cosinus signal maximum amplitude value (Volts)
– CurrentCosinusAmplitude	double	Encoder cosinus signal current amplitude value (Volts)

Return

– Error	long	Function error code

Matlab

Prototype

[Error, MaxSinusAmplitude, CurrentSinusAmplitude, MaxCosinusAmplitude, CurrentCosinusAmplitude] **PositionerEncoderAmplitudeValuesGet** (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– MaxSinusAmplitude	double	Encoder sinus signal maximum amplitude value (Volts)
– CurrentSinusAmplitude	double	Encoder sinus signal current amplitude value (Volts)
– MaxCosinusAmplitude	double	Encoder cosinus signal maximum amplitude value (Volts)
– CurrentCosinusAmplitude	double	Encoder cosinus signal current amplitude value (Volts)

Python

Prototype

[Error, MaxSinusAmplitude, CurrentSinusAmplitude, MaxCosinusAmplitude, CurrentCosinusAmplitude] **PositionerEncoderAmplitudeValuesGet** (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code
– MaxSinusAmplitude	double	Encoder sinus signal maximum amplitude value (Volts)
– CurrentSinusAmplitude	double	Encoder sinus signal current amplitude value (Volts)
– MaxCosinusAmplitude	double	Encoder cosinus signal maximum amplitude value (Volts)
– CurrentCosinusAmplitude	double	Encoder cosinus signal current amplitude value (Volts)

2.2.4.54 PositionerEncoderCalibrationParametersGet

Name

PositionerEncoderCalibrationParametersGet – Gets the encoder calibration parameters.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner (must be not a secondary positioner): ERR_WRONG_OBJECT_TYPE (-8)
- Check the encoder type (must be "AnalogInterpolated"): ERR_WRONG_OBJECT_TYPE (-8)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

After a calibration of the analog encoder input (by the function "GroupInitializeWithEncoderCalibration"), this function returns the optimum parameters for the analog encoder interface. To take these parameters into account (recommended to achieve best performance), these values must be entered manually in the corresponding section of the stages.ini file. The parameters to set in the stages.ini file are:

EncoderSinusOffset = 0 ; Volts

EncoderCosinusOffset = 0 ; Volts

EncoderDifferentialGain = 0

EncoderPhaseCompensation = 0 ; Deg

CAUTION

The encoder type must be "AnalogInterpolated" in the stages.ini file ("EncoderType" parameter).

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerEncoderCalibrationParametersGet $SocketID $FullPositionerName SinusOffset CosinusOffset DifferentialGain PhaseCompensation

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– SinusOffset	double	Encoder sinus signal offset (Volts)
– CosinusOffset	double	Encoder cosinus signal offset (Volts)
– DifferentialGain	double	Encoder differential gain
– PhaseCompensation	double	Encoder phase compensation (Deg)

Return

– Error	int	TCL error code (0=success or 1=syntax error) or function error code

C/C++

Prototype

int **PositionerEncoderCalibrationParametersGet** (int SocketID, char FullPositionerName[250] , double * SinusOffset, double * CosinusOffset, double * DifferentialGain, double * PhaseCompensation)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name

Output parameters

– SinusOffset	double *	Encoder sinus signal offset (Volts)
– CosinusOffset	double *	Encoder cosinus signal offset (Volts)
– DifferentialGain	double *	Encoder differential gain
– PhaseCompensation	double *	Encoder phase compensation (Deg)

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerEncoderCalibrationParametersGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, SinusOffset As Double, CosinusOffset As Double, DifferentialGain As Double, PhaseCompensation As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– SinusOffset	double	Encoder sinus signal offset (Volts)
– CosinusOffset	double	Encoder cosinus signal offset (Volts)
– DifferentialGain	double	Encoder differential gain
– PhaseCompensation	double	Encoder phase compensation (Deg)

Return

– Error	long	Function error code

Matlab

Prototype

[Error, SinusOffset, CosinusOffset, DifferentialGain, PhaseCompensation] **PositionerEncoderCalibrationParametersGet** (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– SinusOffset	double	Encoder sinus signal offset (Volts)
– CosinusOffset	double	Encoder cosinus signal offset (Volts)
– DifferentialGain	double	Encoder differential gain
– PhaseCompensation	double	Encoder phase compensation (Deg)

Python

Prototype

[Error, SinusOffset, CosinusOffset, DifferentialGain, PhaseCompensation] **PositionerEncoderCalibrationParametersGet** (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code
– SinusOffset	double	Encoder sinus signal offset (Volts)
– CosinusOffset	double	Encoder cosinus signal offset (Volts)
– DifferentialGain	double	Encoder differential gain
– PhaseCompensation	double	Encoder phase compensation (Deg)

2.2.4.55 PositionerErrorGet

Name

PositionerErrorGet – Returns the positioner error code and clears it.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid output parameter type: ERR_WRONG_TYPE_INT (-15)
- Valid positioner name: ERR_POSITIONER _NAME (-18)
- Valid secondary positioner: ERR_UNCOMPATIBLE (-18)

Description

Returns the positioner error code and clears it.

The positioner error codes are listed in the "Positioner error list" § 2.19. The description of the positioner error code can be obtained with the "GroupPositionerErrorStringGet" function.

NOTE

The "PositionerErrorRead" function reads the positioner error without clearing it.

Error codes

ERR_FATAL_INIT (-20)

ERR_IN_INITIALIZATION (-21)

ERR_POSITIONER_NAME (-18)

ERR_UNCOMPATIBLE (-24)

ERR_WRONG_FORMAT (-7)

ERR_WRONG_PARAMETERS_NUMBER (-9)

ERR_WRONG_TYPE_INT (-15)

SUCCESS (0): no error

TCL

Prototype

PositionerErrorGet SocketID PositionerName PositionerError

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– PositionerName	string	Positioner name (maximum size = 250)

Output parameters

– PositionerError	interger	Positioner error code.

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **PositionerErrorGet** (int SocketID, char * PositionerName, int * PositionerError)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– PositionerName	char *	Positioner name

Output parameters

– PositionerError	int *	Positioner error code

Return

– Function error code

Visual Basic

Prototype

Long **PositionerErrorGet** (ByVal SocketID As Long, ByVal PositionerName As String, PositionerError As Long)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– PositionerName	string	Positioner name

Output parameters

– PositionerError	long	Positioner error code

Return

– Function error code

Matlab

Prototype

[Error, PositionerError] **PositionerErrorGet** (int32 SocketID, cstring PositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– PositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– PositionerError	int32	Positioner error code

Python

Prototype

[Error, PositionerError] **PositionerErrorGet** (integer SocketID, string PositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– PositionerName	string	Positioner name

Return

– Error	int	Function error code
– PositionerError	int	Positioner error code

2.2.4.56 PositionerErrorRead

Name

PositionerErrorRead – Returns the positioner error code without clearing it.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid output parameter type: ERR_WRONG_TYPE_INT (-15)
- Valid positioner name: ERR_POSITIONER _NAME (-18)
- Valid secondary positioner: ERR_UNCOMPATIBLE (-18)

Description

Returns the positioner error code without clearing it.

The positioner error codes are listed in the "Positioner error list" § 2.19. The description of the positioner error code can be obtained with the "GroupPositionerErrorStringGet" function.

NOTE

The "PositionerErrorGet" function clears the positioner error.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_UNCOMPATIBLE (-24)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

PositionerErrorRead SocketID PositionerName PositionerError

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	PositionerName	string	Positioner name (maximum size = 250)

Output parameters

–	PositionerError	interger	Positioner error code.

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **PositionerErrorRead** (int SocketID, char * PositionerName, int * PositionerError)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	PositionerName	char *	Positioner name

Output parameters

–	PositionerError	int *	Positioner error code

Return

–		Function error code

Visual Basic

Prototype

Long **PositionerErrorRead** (ByVal SocketID As Long, ByVal PositionerName As String, PositionerError As Long)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	PositionerName	string	Positioner name

Output parameters

–	PositionerError	long	Positioner error code

Return

– Function error code

Matlab

Prototype

[Error, PositionerError] **PositionerErrorRead** (int32 SocketID, cstring PositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– PositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– PositionerError	int32	Positioner error code

Python

Prototype

[Error, PositionerError] **PositionerErrorRead** (integer SocketID, string PositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– PositionerName	string	Positioner name

Return

– Error	int	Function error code
– PositionerError	int	Positioner error code

2.2.4.57 PositionerErrorStringGet

Name

PositionerErrorStringGet – Gets the positioner error description.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check input parameter type: ERR_WRONG_TYPE_INT (-15)
- Check output parameter type: ERR_WRONG_TYPE_CHAR (-13)

Description

This function returns a positioner error description from a positioner error code.

NOTE

See the positioner error list described in § 2.19

Error codes

ERR_FATAL_INIT (-20)

ERR_IN_INITIALIZATION (-21)

ERR_WRONG_FORMAT (-7)

ERR_WRONG_PARAMETERS_NUMBER (-9)

ERR_WRONG_TYPE_CHAR (-13)

ERR_WRONG_TYPE_INT (-15)

SUCCESS (0): no error

TCL

Prototype

PositionerErrorStringGet $SocketID $FullPositionerName $PositionerErrorCode PositionerErrorString

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– PositionerErrorCode	int	Positioner error code

Output parameters

– PositionerErrorString	int	Positioner error description

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerErrorStringGet** (int SocketID, char FullPositionerName[250] , int PositionerErrorCode, char * PositionerErrorString)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name
– PositionerErrorCode	int *	Positioner error code

Output parameters

– PositionerErrorString	int *	Positioner error description

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerErrorStringGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal PositionerErrorCode As Integer, ByVal PositionerErrorString As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– PositionerErrorCode	int	Positioner error code

Output parameters

– PositionerErrorString	int	Positioner error description

Return

– Error	long	Function error code

Matlab

Prototype

[Error, PositionerErrorString] **PositionerErrorStringGet** (int32 SocketID, cstring FullPositionerName, int32 PositionerErrorCode)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name
– PositionerErrorCode	int32	Positioner error code

Return

– Error	int32	Function error code
– PositionerErrorString	cstring	Positioner error description

Python

Prototype

[Error, PositionerErrorString] **PositionerErrorStringGet** (integer SocketID, string FullPositionerName, integer PositionerErrorCode)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– PositionerErrorCode	int	Positioner error code

Return

– Error	int	Function error code
– PositionerErrorString	string	Positioner error description

2.2.4.58 PositionerExcitationSignalGet

Name

PositionerExcitationSignalGet – Returns the currently used parameters of the excitation signal functionality.

Input tests

- Configuration files reading: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid object type: ERR_WRONG_OBJECT_TYPE (-8)
- Valid positioner name: ERR_GROUP_NAME (-19)
- Valid output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)
- Valid output parameter type: ERR_WRONG_TYPE_INT (-15)

Description

This function gets the last configured excitation signal parameters.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0) : no error

TCL

Prototype

PositionerExcitationSignalGet SocketID $PositionerName SignalType Frequency Amplitude Time

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– PositionerName	string	Positioner name (maximum size = 250)

Output parameters

– SignalType	integer	Type of signal
– Frequency	floating point	Frequency (Hz)
– Amplitude	floating point	Amplitude (acceleration, velocity or voltage unit)
– Time	floating point	During time (seconds)

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **PositionerExcitationSignalGet** (int SocketID, char *PositionerName, int* SignalType, double* Frequency, double* Amplitude, double* Time)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– PositionerName	char *	Positioner name

Output parameters

– SignalType	int *	Type of signal
– Frequency	double *	Frequency (Hz)
– Amplitude	double *	Amplitude (acceleration, velocity or voltage unit)
– Time	double *	During time (seconds)

Return

– Function error code

Visual Basic

Prototype

Long **PositionerExcitationSignalGet** (ByVal SocketID As Long, ByVal GroupName As String, SignalType As Long, Frequency As Double, Amplitude As Double, Time As Double)

Input parameters

– SocketID	Long	Socket identifier gets by the "TCP_ConnectToServer" function
– PositionerName	String	Positioner name

Output parameters

– SignalType	Long	Type of signal
– Frequency	Double	Frequency (Hz)
– Amplitude	Double	Amplitude (acceleration, velocity or voltage unit)
– Time	Double	During time (seconds)

Return

– Function error code

Matlab

Prototype

[Error, SignalType, Frequency, Amplitude, Time] **PositionerExcitationSignalGet** (int32 SocketID, cstring PositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– PositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– SignalType	int32Ptr	Type of signal
– Frequency	doublePtr	Frequency (Hz)
– Amplitude	doublePtr	Amplitude (acceleration, velocity or voltage unit)
– Time	doublePtr	During time (seconds)

Python

Prototype

[Error, SignalType, Frequency, Amplitude, Time] **PositionerExcitationSignalGet** (integer SocketID, string PositionerName)

Input parameters

– SocketID	integer	Socket identifier gets by the "TCP_ConnectToServer"function
– PositionerName	string	Positioner name

Return

– Error	integer	Function error code
– Frequency	integerPtr	Type of signal
– Frequency	doublePtr	Frequency (Hz)
– Amplitude	doublePtr	Amplitude (acceleration, velocity or voltage unit)
– Time	doublePtr	During time (seconds)

2.2.4.59 PositionerExcitationSignalSet

Name

PositionerExcitationSignalSet – Configure and activate the signal of excitation.

Input tests

- Configuration files reading: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid object type: ERR_WRONG_OBJECT_TYPE (-8)
- Is secondary positioner or has a secondary positioner: ERR_WRONG_OBJECT_TYPE (-8)
- Valid control loop type: ERR_UNCOMPATIBLE (-24)
- Valid positioner name: ERR_GROUP_NAME (-19)
- Valid output parameter type: ERR_WRONG_TYPE_INT (-15)
- Valid output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)
- Check group status (*must be READY*): ERR_NOT_ALLOWED_ACTION (-22)
- Check frequency (*must ≥ 0.1 and ≤ 0.5/CorrectorISRPeriod*): ERR_PARAMETER_OUT_OF_RANGE (-17)
- Check excitation time (*must ≥ 4*CorrectorISRPeriod*): ERR_PARAMETER_OUT_OF_RANGE (-17)
- Check signal amplitude [-*Acceleration (Velocity or Voltage) limit to Acceleration (Velocity or Voltage) limit*]: ERR_PARAMETER_OUT_OF_RANGE (-17)
- Check type of signal (0, 1, 2 or 3): ERR_PARAMETER_OUT_OF_RANGE (-17)

Description

The excitation signal functionality generates a typical signal (a sine, a blank noise or an echelon signal) that the controller sends to motors to excite the system. In measuring the output signal of the excited system, we can determine some system characteristics, like the system transfer function.

The excitation signal functionality is only available with the stages controlled in acceleration (acceleration control, ex: brushless / linear motors), velocity (velocity control) or in voltage (voltage control). It does not exist with the stages controlled in position (ex: stepper motors).

The excitation-signal function **PositionerExcitationSignalSet** can be executed only when the positioner is in "READY" state. When the excitation-signal process is in progress, the positioner is in the "ExcitationSignal" state. At the end of the process, the positioner returns to "READY" state (see group state diagram).

The **PositionerExcitationSignalSet** function sends an excitation signal to the motor for a brief time. This function is allowed for "PIDFFAcceleration", "PIDFFVelocity" or "PIDDualFFVoltage" control loops. The parameters to configure are *signal type* (0:sine, 1:echelon,2:random-amplitude,3:random-pulse-width binary-amplitude, integer), *frequency* (Hz, double), *amplitude* (acceleration, velocity or voltage unit, double) and *during time* (seconds, double).

The function effective parameters for each mode are : (here : Limit means AccelerationLimit, VelocityLimit or VoltageLimit)

- Sine signal mode : Frequency (≥ 0.1 and ≤ 0.5/CorrectorISRPeriod), Amplitude (> 0 and ≤ Limit), Time (≥ 4 * CorrectorISRPeriod)
- Echelon signal mode : Amplitude (> 0 and ≤ Limit, or < 0 and ≥ -Limit), Time(≥ 4 * CorrectorISRPeriod).

  + During *Time* : Signal = *Amplitude*

  + End of *Time* : Signal = 0
- Random-amplitude signal mode : Amplitude (>0 and ≤ Limit), Time(>0), Frequency (≥ 0.1 and <= 0.5/CorrectorISRPeriod).

  Signal is generated with a random value at with a period defined by the controller base time (*CorrectorISRPeriod , default value 0.125 ms*), then is filtered with a second order low-pass filter at the cut-off *Frequency* value.
- Random-pulse-width binary-amplitude signal mode :

  Amplitude (>0 and ≤ Limit), *Time* (≥ *4 * CorrectorISRPeriod*), *Frequency* (≥ *0.1 and ≤ 0.5/CorrectorISRPeriod*).

  Signal is a sequence of pulses (*Signal = Amplitude or = 0*) with pulse randomly varied width (multiple of Tbase).

  Frequency is the controlled system band-width (*cut-off frequency*), necessary for the PRBS (*Pseudo Random Binary Sequence*) function configuration.

The function non-effective parameters can accept any value, the value 0 is recommended for simplicity.

NOTE

If during the excitation signal generation the stage position exceeds the user minimum or maximum target positions, the motor excitation command is stopped and an error is returned.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_IN_INITIALIZATION (-21)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_UNCOMPATIBLE (-24)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- ERR_WRONG_TYPE_INT (-15)
- ERR_EXCITATION_SIGNAL_INITIALIZATION (-112)
- SUCCESS (0) : no error

TCL

Prototype

PositionerExcitationSignalSet SocketID $PositionerName $SignalType $Frequency $Amplitude $Time

Input parameters

– SocketID	integer	Socket identifier gets by the "TCP_ConnectToServer" function
– PositionerName	string	Positioner name (maximum size = 250)
– SignalType	integer	Type of signal
– Frequency	floating point	Frequency (Hz)
– Amplitude	floating point	Amplitude (acceleration, velocity or voltage unit)
– Time	floating point	During time (seconds)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **PositionerExcitationSignalSet** (int SocketID, char *PositionerName, int SignalType, double Frequency, double Amplitude, double Time)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– PositionerName	char *	Positioner name
– SignalType	int	Type of signal
– Frequency	double	Frequency (Hz)
– Amplitude	double	Amplitude (acceleration, velocity or voltage unit)
– Time	double	During time (seconds)

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **PositionerExcitationSignalSet** (ByVal SocketID As Long, ByVal GroupName As String, ByVal SignalType As Long, ByVal Frequency As Double, ByVal Amplitude As Double, ByVal Time As Double)

Input parameters

– SocketID	Long	Socket identifier gets by the "TCP_ConnectToServer" function
– PositionerName	String	Positioner name
– SignalType	Long	Type of signal
– Frequency	Double	Frequency (Hz)
– Amplitude	double	Amplitude (acceleration, velocity or voltage unit)
– Time	Double	During time (seconds)

Output parameters

– None

Return

– Function error code

Matlab

Prototype

[Error] **PositionerExcitationSignalSet** (int32 SocketID, cstring PositionerName, SignalType, Frequency, Amplitude, Time)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– PositionerName	cstring	Positioner name
– SignalType	int32	Type of signal
– Frequency	double	Frequency (Hz)
– Amplitude	double	Amplitude (acceleration, velocity or voltage unit)
– Time	double	During time (seconds)

Return

– Error	int32	Function error code

Python

Prototype

[Error] **PositionerExcitationSignalSet** (integer SocketID, string PositionerName, SignalType, Frequency, Amplitude, Time)

Input parameters

– SocketID	integer	Socket identifier gets by the "TCP_ConnectToServer"function
– PositionerName	string	Positioner name
– SignalType	integer	Type of signal
– Frequency	double	Frequency (Hz)
– Amplitude	double	Amplitude (acceleration, velocity or voltage unit)
– Time	double	During time (seconds)

Return

– Error	integer	Function error code

2.2.4.60 PositionerHardInterpolatorFactorGet

Name

PositionerHardInterpolatorFactorGet – Gets the interpolation factor for position compare mode.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner (must be not a secondary positioner): ERR_WRONG_OBJECT_TYPE (-8)
- Check the encoder type (must be "AnalogInterpolated"): ERR_WRONG_OBJECT_TYPE (-8)
- Check output parameter type: ERR_WRONG_TYPE_INT (-15)

Description

This function returns the interpolation factor of the hardware interpolator used in the "Position Compare" mode. The interpolation factor value is defined as:

InterpolationFactor = round (EncoderScalePitch/HardInterpolatorResolution)

NOTE

The encoder type must be "AnalogInterpolated" in the stages.ini file ("EncoderType" parameter).

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

PositionerHardInterpolatorFactorGet $SocketID $FullPositionerName InterpolationFactor

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– InterpolationFactor	interger	Interpolation factor

Return

– Error	int	TCL error code (0=success or 1=syntax error) or function error code

C/C++

Prototype

int **PositionerHardInterpolatorFactorGet** (int SocketID, char FullPositionerName[250] , int * InterpolationFactor)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name

Output parameters

– InterpolationFactor	int *	Interpolation factor

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerHardInterpolatorFactorGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, InterpolationFactor As Integer)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– InterpolationFactor	int	Interpolation factor

Return

– Error	long	Function error code

Matlab

Prototype

[Error, InterpolationFactor] **PositionerHardInterpolatorFactorGet** (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– InterpolationFactor	int32	Interpolation factor

Python

Prototype

[Error, InterpolationFactor] **PositionerHardInterpolatorFactorGet** (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code
– InterpolationFactor	int	Interpolation factor

2.2.4.61 PositionerHardInterpolatorFactorSet

Name

PositionerHardInterpolatorFactorSet – Sets the interpolation factor for position compare mode.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type (must be not a secondary positioner): ERR_WRONG_OBJECT_TYPE (-8)
- Check the positioner name: ERR_POSITIONER_NAME (-18)
- Check the group state (must be NOTINIT): ERR_NOT_ALLOWED_ACTION (-22)
- Check the encoder type (must be "AnalogInterpolated"): ERR_WRONG_OBJECT_TYPE (-8)
- Check input parameter type: ERR_WRONG_TYPE_INT (-15)
- Check input parameter value:ERR_PARAMETER_OUT_OF_RANGE (-17)

Description

This function sets the interpolation factor of the hardware interpolator used in the "PositionCompare" mode. The IP200 is updated and the position compare resolution is set as follows:

PositionCompareResolution = EncoderScalePitch/InterpolationFactor

The "InterpolationFactor" value must be define with one of these values:

20	25	40	50	80	100	160	200

If the input interpolator factor value is different from these values then ERR_PARAMETER_OUT_OF_RANGE is returned.

NOTE

The group must be NOTINIT to use this function else ERR_NOT_ALLOWED_ACTION is returned.

The encoder type must be "AnalogInterpolated" in the stages.ini file ("EncoderType" parameter) else the error is returned

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

PositionerHardInterpolatorFactorSet $SocketID $FullPositionerName $InterpolationFactor

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– InterpolationFactor	int	Interpolation factor

Return

– Error	int	TCL error code (0=success or 1=syntax error) or function error code

C/C++

Prototype

int **PositionerHardInterpolatorFactorSet** (int SocketID, char FullPositionerName[250] , int InterpolationFactor)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name
– InterpolationFactor	int	Interpolation factor

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerHardInterpolatorFactorSet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal InterpolationFactor As Integer)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– InterpolationFactor	int	Interpolation factor

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **PositionerHardInterpolatorFactorSet** (int32 SocketID, cstring FullPositionerName, int32 InterpolationFactor)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name
– InterpolationFactor	int32	Interpolation factor

Return

– Error	int32	Function error code

Python

Prototype

[Error] **PositionerHardInterpolatorFactorSet** (integer SocketID, string FullPositionerName, integer InterpolationFactor)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– InterpolationFactor	int	Interpolation factor

Return

– Error	int	Function error code

2.2.4.62 PositionerHardInterpolatorPositionGet

Name

PositionerHardInterpolatorPositionGet – Gets interpolated position from the encoder hard interpolator.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner (must not be a secondary positioner): ERR_WRONG_OBJECT_TYPE (-8)
- Check the encoder type (must be "AnalogInterpolated"): ERR_WRONG_OBJECT_TYPE (-8)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function returns the position interpolated by the encoder hard interpolator.

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

The encoder type must be "*AnalogInterpolated*" or "*AnalogInterpolatedTheta*" in the stages.ini file ("EncoderType" parameter).

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerHardInterpolatorPositionGet $SocketID $FullPositionerName Position

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name

Output parameters

–	Position	double	Interpolated position

Return

–	Error	int	TCL error code (0=success or 1=syntax error) or function error code

C/C++

Prototype

int **PositionerHardInterpolatorPositionGet** (int SocketID, char FullPositionerName[250] , double * Position)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	char *	Positioner name

Output parameters

–	Position	double *	Interpolated position

Return

–	Error	int	Function error code

Visual Basic

Prototype

Long **PositionerHardInterpolatorPositionGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, Position As Double)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name

Output parameters

–	Position	double	Interpolated position

Return

–	Error	long	Function error code

Matlab

Prototype

[Error, Position] **PositionerHardInterpolatorPositionGet** (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– Position	double	Interpolated position

Python

Prototype

[Error, Position] **PositionerHardInterpolatorPositionGet** (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code
– Position	double	Interpolated position

2.2.4.63 PositionerHardwareStatusGet

Name

PositionerHardwareStatusGet – Gets the positioner hardware status code.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner (must be not a secondary positioner): ERR_WRONG_OBJECT_TYPE (-8), ERR_UNCOMPATIBLE (-24), ERR_POSITIONER_NAME (-18)
- Check output parameter type: ERR_WRONG_TYPE_INT (-15)

Description

This function returns the hardware status of the selected positioner. The positioner hardware status is composed of the "corrector" hardware status and the "servitudes" hardware status:

The "Corrector" returns the motor interface and the position encoder hardware status.

The "Servitudes" returns the general inhibit and the end of runs hardware status.

NOTE

See the positioner hardware status list describes in § 2.20

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

PositionerHardwareStatusGet $SocketID $FullPositionerName PositionerHardwareStatus

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– PositionerHardwareStatus	int	Hardware status code

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error

C/C++

Prototype

int **PositionerHardwareStatusGet** (int SocketID, char FullPositionerName[250] , int * PositionerHardwareStatus)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name

Output parameters

– PositionerHardwareStatus	int *	Hardware status code

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerHardwareStatusGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, PositionerHardwareStatus As Integer)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– PositionerHardwareStatus	int	Hardware status code

Return

– Error	long	Function error code

Matlab

Prototype

[Error, PositionerHardwareStatus] **PositionerHardwareStatusGet** (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– PositionerHardwareStatus	int32	Hardware status code

Python

Prototype

[Error, PositionerHardwareStatus] **PositionerHardwareStatusGet** (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code
– PositionerHardwareStatus	int	Hardware status code

2.2.4.64 PositionerHardwareStatusStringGet

Name

PositionerHardwareStatusStringGet – Gets the positioner error description.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check input parameter type: ERR_WRONG_TYPE_INT (-15)
- Check output parameter type: ERR_WRONG_TYPE_CHAR (-13)

Description

This function returns the hardware status description from a positioner hardware status code.

NOTE

See the positioner hardware status list describes in § 2.20.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

PositionerHardwareStatusStringGet $SocketID $FullPositionerName $PositionerHardwareStatusCode PositionerHardwareStatusString

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– PositionerHardwareStatusCode	int	Positioner hardware status code

Output parameters

– PositionerHardwareStatusString	int	Positioner hardware status description

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerHarwareStatusStringGet** (int SocketID, char FullPositionerName[250] , int PositionerHardwareStatusCode, char * PositionerHardwareStatusString)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name
– PositionerHardwareStatusCode	int *	Positioner hardware status code

Output parameters

– PositionerHardwareStatusString	int *	Positioner hardware status description

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerHardwareStatusStringGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal PositionerHardwareStatusCode As Integer, ByVal PositionerHardwareStatusString As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– PositionerHardwareStatusCode	int	Positioner hardware status code

Output parameters

– PositionerHardwareStatusString	int	Positioner hardware status description

Return

– Error	long	Function error code

Matlab

Prototype

[Error, PositionerHardwareStatusString] **PositionerHardwareStatusStringGet** (int32 SocketID, cstring FullPositionerName, int32 PositionerHardwareStatusCode)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name
– PositionerHardwareStatusCode	int32	Positioner hardware status code

Return

– Error	int32	Function error code
– PositionerHardwareStatusString	cstring	Positioner hardware status description

Python

Prototype

[Error, PositionerHardwareStatusString] **PositionerHardwareStatusStringGet** (integer SocketID, string FullPositionerName, integer PositionerHardwareStatusCode)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– PositionerHardwareStatusCode	int	Positioner hardware status code

Return

– Error	int	Function error code
– PositionerHardwareStatusString	string	Positioner hardware status description

2.2.4.65 PositionerMaximumVelocityAndAccelerationGet

Name

PositionerMaximumVelocityAndAccelerationGet – Gets the maximum velocity and acceleration from the profiler generators.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner (must not be a secondary positioner): ERR_WRONG_OBJECT_TYPE (-8)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function returns the maximum velocity and acceleration of the profile generators. These parameters represent the limits for the profiler and are defined in the stages.ini file:

MaximumVelocity = ; unit/second

MaximumAcceleration = ; unit/second^2

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerMaximumVelocityAndAccelerationGet $SocketID $FullPositionerName MaximumVelocity MaximumAcceleration

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name

Output parameters

–	MaximumVelocity	double	Maximum velocity (units/seconds)
–	MaximumAcceleration	double	Maximum acceleration (units/seconds^2)

Return

–	Error	int	TCL error code (0=success or 1=syntax error) or function error code

C/C++

Prototype

int **PositionerMaximumVelocityAndAccelerationGet** (int SocketID, char FullPositionerName[250] , double * MaximumVelocity, double * MaximumAcceleration)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	char *	Positioner name

Output parameters

–	MaximumVelocity	double *	Maximum velocity (units/seconds)
–	MaximumAcceleration	double *	Maximum acceleration (units/seconds²)

Return

–	Error	int	Function error code

Visual Basic

Prototype

Long **PositionerMaximumVelocityAndAccelerationGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, MaximumVelocity As Double, MaximumAcceleration As Double)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name

Output parameters

–	MaximumVelocity	double	Maximum velocity (units/seconds)
–	MaximumAcceleration	double	Maximum acceleration (units/seconds²)

Return

–	Error	long	Function error code

Matlab

Prototype

[Error, MaximumVelocity, MaximumAcceleration]
PositionerMaximumVelocityAndAccelerationGet (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– MaximumVelocity	double	Maximum velocity (units/seconds)
– MaximumAcceleration	double	Maximum acceleration (units/seconds²)

Python

Prototype

[Error, MaximumVelocity, MaximumAcceleration]
PositionerMaximumVelocityAndAccelerationGet (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code
– MaximumVelocity	double	Maximum velocity (units/seconds)
– MaximumAcceleration	double	Maximum acceleration (units/seconds²)

2.2.4.66 PositionerMotionDoneGet

Name

PositionerMotionDoneGet – Gets the motion done parameters.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner (must not be a secondary positioner): ERR_WRONG_OBJECT_TYPE (-8)
- Check the motion done mode (must be "VelocityAndPositionWindow"): ERR_WRONG_OBJECT_TYPE (-8)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function returns the motion done parameters only for the "VelocityAndPositionWindow" MotionDone mode. If the MotionDone mode is defined as "Theoretical" then ERR_WRONG_OBJECT_TYPE (-8) is returned.

The "MotionDoneMode" parameter from the stages.ini file defines the motion done mode. The motion done can be defined as "Theoretical" (the motion done mode is not used) or "VelocityAndPositionWindow". For a more thorough description of the motion done mode, please refer to the XPS Motion Tutorial section Motion/Motion Done.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerMotionDoneGet $SocketID $FullPositionerName PositionWindow VelocityWindow CheckingTime MeanPeriod Timeout

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– PositionWindow	double	Position window (units)
– VelocityWindow	double	Velocity window (units/seconds)
– CheckingTime	double	Checking time (seconds)
– MeanPeriod	double	Mean period (seconds)
– Timeout	double	Motion done time out (seconds)

Return

– Error	int	TCL error code (0=success or 1=syntax error) or function error code

C/C++

Prototype

int **PositionerMotionDoneGet** (int SocketID, char FullPositionerName[250] , double * PositionWindow, double * VelocityWindow, double * CheckingTime, double * MeanPeriod, double * Timeout)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name

Output parameters

– PositionWindow	double *	Position window (units)
– VelocityWindow	double *	Velocity window (units/seconds)
– CheckingTime	double *	Checking time (seconds)
– MeanPeriod	double *	Mean period (seconds)
– Timeout	double *	Motion done time out (seconds)

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerMotionDoneGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, PositionWindow As Double, VelocityWindow As Double, CheckingTime As Double, MeanPeriod As Double, Timeout As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– PositionWindow	double	Position window (units)
– VelocityWindow	double	Velocity window (units/seconds)
– CheckingTime	double	Checking time (seconds)
– MeanPeriod	double	Mean period (seconds)
– Timeout	double	Motion done time out (seconds)

Return

– Error	long	Function error code

Matlab

Prototype

[Error, PositionWindow, VelocityWindow, CheckingTime, MeanPeriod, Timeout] **PositionerMotionDoneGet** (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– PositionWindow	double	Position window (units)
– VelocityWindow	double	Velocity window (units/seconds)
– CheckingTime	double	Checking time (seconds)
– MeanPeriod	double	Mean period (seconds)
– Timeout	double	Motion done time out (seconds)

Python

Prototype

[Error, PositionWindow, VelocityWindow, CheckingTime, MeanPeriod, Timeout]
PositionerMotionDoneGet (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code
– PositionWindow	double	Position window (units)
– VelocityWindow	double	Velocity window (units/seconds)
– CheckingTime	double	Checking time (seconds)
– MeanPeriod	double	Mean period (seconds)
– Timeout	double	Motion done time out (seconds)

2.2.4.67 PositionerMotionDoneSet

Name

PositionerMotionDoneSet – Sets the motion done parameters.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type: ERR_WRONG_OBJECT_TYPE (-8)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function updates the motion done parameters only for the "VelocityAndPositionWindow" MotionDone mode. The "MotionDoneMode" parameter from the stages.ini file must be defined as "VelocityAndPositionWindow" else ERR_WRONG_OBJECT_TYPE (-8) is returned.

For a more thorough description of the Motion Done mode, please refer to the XPS Motion Tutorial section Motion/Motion Done.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerMotionDoneSet $SocketID $FullPositionerName $PositionWindow $VelocityWindow $CheckingTime $MeanPeriod $Timeout

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– PositionWindow	double	Position window (units)
– VelocityWindow	double	Velocity window (units/seconds)
– CheckingTime	double	Checking time (seconds)
– MeanPeriod	double	Mean period (seconds)
– Timeout	double	Motion done time out (seconds)

Output parameters

– None

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerMotionDoneSet** (int SocketID, char FullPositionerName[250] , double * PositionWindow, double * VelocityWindow, double * CheckingTime, double * MeanPeriod, double * Timeout)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name
– PositionWindow	double	Position window (units)
– VelocityWindow	double	Velocity window (units/seconds)
– CheckingTime	double	Checking time (seconds)
– MeanPeriod	double	Mean period (seconds)
– Timeout	double	Motion done time out (seconds)

Output parameters

– None

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerMotionDoneSet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal PositionWindow As Double, ByVal VelocityWindow As Double, ByVal CheckingTime As Double, ByVal MeanPeriod As Double, ByVal Timeout As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– PositionWindow	double	Position window (units)
– VelocityWindow	double	Velocity window (units/seconds)
– CheckingTime	double	Checking time (seconds)
– MeanPeriod	double	Mean period (seconds)
– Timeout	double	Motion done time out (seconds)

Output parameters

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **PositionerMotionDoneSet** (int32 SocketID, cstring FullPositionerName, double PositionWindow, double VelocityWindow, double CheckingTime, double MeanPeriod, double Timeout)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name
– PositionWindow	double	Position window (units)
– VelocityWindow	double	Velocity window (units/seconds)
– CheckingTime	double	Checking time (seconds)
– MeanPeriod	double	Mean period (seconds)
– Timeout	double	Motion done time out (seconds)

Return

– Error	int32	Function error code

Python

Prototype

[Error] **PositionerMotionDoneSet** (integer SocketID, string FullPositionerName, double PositionWindow, double VelocityWindow, double CheckingTime, double MeanPeriod, double Timeout)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– PositionWindow	double	Position window (units)
– VelocityWindow	double	Velocity window (units/seconds)
– CheckingTime	double	Checking time (seconds)
– MeanPeriod	double	Mean period (seconds)
– Timeout	double	Motion done time out (seconds)

Return

– Error	int	Function error code

2.2.4.68 PositionerPositionCompareDisable

Name

PositionerPositionCompareDisable – Disables the position compare mode.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner (must not be a secondary positioner): ERR_WRONG_OBJECT_TYPE (-8)
- Check the encoder ("AquadB" or "AnalogInterpolated"): ERR_WRONG_OBJECT_TYPE (-8)

Description

This function disables the position compare mode.

For a more thorough description of the position compare output, please refer to the XPS Motion Tutorial section Triggers/Position Compare Output.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

PositionerPositionCompareDisable $SocketID $FullPositionerName

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– None

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerPositionCompareDisable** (int SocketID, char FullPositionerName[250])

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	char *	Positioner name

Output parameters

– None

Return

–	Error	int	Function error code

Visual Basic

Prototype

Long **PositionerPositionCompareDisable** (ByVal SocketID As Long, ByVal FullPositionerName As String)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name

Output parameters

– None

Return

–	Error	long	Function error code

Matlab

Prototype

[Error] **PositionerPositionCompareDisable** (int32 SocketID, cstring FullPositionerName)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	cstring	Positioner name

Return

–	Error	int32	Function error code

Python

Prototype

[Error] **PositionerPositionCompareDisable** (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code

2.2.4.69 PositionerPositionCompareEnable

Name

PositionerPositionCompareEnable – Enables the position compare mode.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner (must not be a secondary positioner): ERR_WRONG_OBJECT_TYPE (-8)
- Check the positioner name: ERR_POSITIONER_NAME (-18)
- Check the encoder ("AquadB" or "AnalogInterpolated"): ERR_WRONG_OBJECT_TYPE (-8)
- Check the group state (must be READY): ERR_NOT_ALLOWED_ACTION (-22)
- Check the position compare parameters (must be configured): ERR_NOT_ALLOWED_ACTION (-22)

Description

This function enables the position compare mode. The group must be in READY state to use this function else ERR_NOT_ALLOWED_ACTION (-22) is returned.

If the position compare parameters are not configured (by the "PositionerPositionCompareSet" function) then ERR_NOT_ALLOWED_ACTION (-22) is returned.

For a more thorough description of the position compare output, please refer to the XPS Motion Tutorial section Triggers/Position Compare Output.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

PositionerPositionCompareEnable $SocketID $FullPositionerName

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name

Output parameters

– None

Return

–	Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerPositionCompareEnable** (int SocketID, char FullPositionerName[250])

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	char *	Positioner name

Output parameters

– None

Return

–	Error	int	Function error code

Visual Basic

Prototype

Long **PositionerPositionCompareEnable** (ByVal SocketID As Long, ByVal FullPositionerName As String)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name

Output parameters

– None

Return

–	Error	long	Function error code

Matlab

Prototype

[Error] **PositionerPositionCompareEnable** (int32 SocketID, cstring FullPositionerName)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	cstring	Positioner name

Return

–	Error	int32	Function error code

Python

Prototype

[Error] **PositionerPositionCompareEnable** (integer SocketID, string FullPositionerName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name

Return

–	Error	int	Function error code

2.2.4.70 PositionerPositionCompareGet

Name

PositionerPositionCompareGet – Gets the position compare parameters.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Check the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner (must not be a secondary positioner): ERR_WRONG_OBJECT_TYPE (-8)
- Check the position compare parameters (must be configured): ERR_POSITION_COMPARE_NOT_SET (-23)
- Check the configured mode type (must be PositionCompare): ERR_UNCOMPATIBLE (-24)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14), ERR_WRONG_TYPE_BOOL (-12)

Description

This function returns the real value (without correction) of parameters of the position compare output trigger and returns current state (enabled or disabled).

For a more thorough description of the position compare output, please refer to the XPS Motion Tutorial section Triggers/Position Compare Output.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITION_COMPARE_NOT_SET (-23)
- ERR_UNCOMPATIBLE (-24)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_BOOL (-12)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerPositionCompareGet $SocketID $FullPositionerName MinimumPosition MaximumPosition PositionStep EnableState

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– PositionStep	double	Position step (units)
– EnableState	bool	Position compare state (true=enabled or false=disabled)

Return

– Error	int	TCL error code (0=success or 1=syntax error) or function error code

C/C++

Prototype

int **PositionerPositionCompareGet** (int SocketID, char FullPositionerName[250] , double* MinimumPosition, double* MaximumPosition, double* PositionStep, bool * EnableState)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name

Output parameters

– MinimumPosition	double *	Minimum position (units)
– MaximumPosition	double *	Maximum position (units)
– PositionStep	double *	Position step (units)
– EnableState	bool *	Position compare state (true=enabled or false=disabled)

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerPositionCompareGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, MinimumPosition As Double, MaximumPosition As Double, PositionStep As Double, EnableState As Boolean)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– PositionStep	double	Position step (units)
– EnableState	bool	Position compare state (true=enabled or false=disabled)

Return

– Error	long	Function error code

Matlab

Prototype

[Error, MinimumPosition, MaximumPosition, PositionStep, EnableState] **PositionerPositionCompareGet** (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– PositionStep	double	Position step (units)
– EnableState	bool	Position compare state (true=enabled or false=disabled)

Python

Prototype

[Error, MinimumPosition, MaximumPosition, PositionStep, EnableState]
PositionerPositionCompareGet (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code
– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– PositionStep	double	Position step (units)
– EnableState	bool	Position compare state (true=enabled or false=disabled)

2.2.4.71 PositionerPositionCompareSet

Name

PositionerPositionCompareSet – Sets the position compare parameters.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Check the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type: ERR_WRONG_OBJECT_TYPE (-8)
- Check the position encoder ("AquadB" or "AnalogInterpolated"): ERR_UNCOMPATIBLE (-24)
- Check input parameter values: ERR_PARAMETER_OUT_OF_RANGE (-17)
- MinimumPosition < MaximumPosition
- MinimumPosition >= MinimumTargetPosition
- MaximumPosition <= MaximumTargetPosition
- 0 <= PositionStep <= (MaximumPosition – MinimumPosition)
- Check position compare state (must be disabled): ERR_NOT_ALLOWED_ACTION (-22)

Description

This function sets the parameters for the position compare output trigger of the PCO connector on the XPS controller cards.

These parameters are used only when the position compare mode is enabled. For a more thorough description of the position compare output, please refer to the XPS Motion Tutorial section Triggers/Position Compare Output.

NOTE

This function can be used only with a position encoder. If no position encoder then ERR_UNCOMPATIBLE (-24) is returned.

In the PositionCompare mode (activated with PositionerPositionCompareEnable() function), during the move (relative or absolute) and inside the zone set by PositionerPositionCompareSet(), if the current following error exceeds the Warning Following Error value, the PositionCompareWarningFollowingErrorFlag is activated and the move returns an error (-120: Warning following error during move with position compare enabled). To reset the PositionCompareWarningFollowingErrorFlag, send the PositionerPositionCompareDisable() function. The WarningFollowingError is set to FatalFollowingError (defined in stages.ini file) by default, but it can be modified by PositionerWarningErrorSet().

In the PositionCompare mode (activated with PositionerPositionCompareEnable() function), during the move (relative or absolute) and inside the zone set by PositionerPositionCompareSet(), the CorrectorOutput is limited to MaximumAcceleration (*defined in stages.ini*).

Error codes

- ERR_FATAL_INIT (-20)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_IN_INITIALIZATION (-21)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_UNCOMPATIBLE (-24)
- SUCCESS (0): no error

TCL

Prototype

PositionerPositionCompareSet $SocketID $FullPositionerName $MinimumPosition $MaximumPosition $PositionStep

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– PositionStep	double	Position step (units)

Output parameters

– None

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerPositionCompareSet** (int SocketID, char FullPositionerName[250] , double MinimumPosition, double MinimumPosition, double PositionStep)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name
– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– PositionStep	double	Position step (units)

Output parameters

– None

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerPositionCompareSet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal MinimumPosition As Double, ByVal MaximumPosition As Double, ByVal PositionStep As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– PositionStep	double	Position step (units)

Output parameters

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **PositionerPositionCompareSet** (int32 SocketID, cstring FullPositionerName, double MinimumPosition, double MaximumPosition, double PositionStep)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name
– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– PositionStep	double	Position step (units)

Return

– Error	int32	Function error code

Python

Prototype

[Error] **PositionerPositionCompareSet** (integer SocketID, string FullPositionerName, double MinimumPosition, double MaximumPosition, double PositionStep)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– PositionStep	double	Position step (units)

Return

– Error	int	Function error code

2.2.4.72 PositionerPositionComparePulseParametersGet

Name

PositionerPositionComparePulseParametersGet – Gets the position compare PCO pulse parameters.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Check the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)
- Check the positioner (must not be a secondary positioner): ERR_WRONG_OBJECT_TYPE (-8)

Description

This function returns the configured parameters of the position compare PCO pulse parameters.

For a more thorough description of the position compare output, please refer to the XPS Motion Tutorial section Triggers/Position Compare Output.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerPositionComparePulseParametersGet $SocketID $FullPositionerName PCOPulseWidth EncoderSettlingTime

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name

Output parameters

–	PCOPulseWidth	double	Width of PCO pulses (µs)
–	EncoderSettlingTime	double	Encoder signal settling time (µs)

Return

–	Error	int	TCL error code (0=success or 1=syntax error) or function error code

C/C++

Prototype

int **PositionerPositionComparePulseParametersGet** (int SocketID, char FullPositionerName[250] , double* PCOPulseWidth, double* EncoderSettlingTime)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	char *	Positioner name

Output parameters

–	PCOPulseWidth	double *	Width of PCO pulses (µs)
–	EncoderSettlingTime	double *	Encoder signal settling time (µs)

Return

–	Error	int	Function error code

Visual Basic

Prototype

Long **PositionerPositionComparePulseParametersGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, PCOPulseWidth As Double, EncoderSettlingTime As Double)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name

Output parameters

–	PCOPulseWidth	double	Width of PCO pulses (µs)
–	EncoderSettlingTime	double	Encoder signal settling time (µs)

Return

–	Error	long	Function error code

Matlab

Prototype

[Error, PCOPulseWidth, EncoderSettlingTime]
PositionerPositionComparePulseParametersGet (int32 SocketID, cstring FullPositionerName)

Input parameter

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	cstring	Positioner name

Return

–	Error	int32	Function error code
–	PCOPulseWidth	double	Width of PCO pulses (µs)
–	EncoderSettlingTime	double	Encoder signal settling time (µs)

Python

Prototype

[Error, PCOPulseWidth, EncoderSettlingTime]
PositionerPositionComparePulseParametersGet (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code
– PCOPulseWidth	double	Width of PCO pulses (µs)
– EncoderSettlingTime	double	Encoder signal settling time (µs)

2.2.4.73 PositionerPositionComparePulseParametersSet

Name

PositionerPositionComparePulseParametersSet – Sets the position compare PCO pulse parameters.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Check the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type (must not be a secondary positioner): ERR_WRONG_OBJECT_TYPE (-8)
- Check the position encoder ("AquadB" or "AnalogInterpolated"): ERR_UNCOMPATIBLE (-24)
- Check input parameter values: ERR_PARAMETER_OUT_OF_RANGE (-17)
- PCOPulseWidth value must equal to 0.2 (default), 1, 2.5 or 10 (µs)
- EncoderSettlingTime value must equal to 0.075 (default), 1, 4 or 12 (µs)
- Check position compare state (must be disabled): ERR_NOT_ALLOWED_ACTION (-22)
- Check if the CIE board supports this function: ERR_HARDWARE_FUNCTION_NOT_SUPPORTED (-115)

Description

This function sets two additional parameters for the position compare output trigger of the PCO connector on the XPS controller cards. The first additional parameter is the pulse width. The second parameter is the encoder settling time value, which is the time the encoder inputs have to stabilize after a change of state is detected.

These parameters are used only when using the position compare mode. For a more thorough description of the position compare output, please refer to the XPS Motion Tutorial section Triggers/Position Compare Output.

NOTE

When changing the PCO Pulse settle time you must limit the maximum velocity of the stage accordingly, otherwise you will loose the PCO position and generate the wrong number of pulses at wrong positions. For example, if you set the EncoderSettlingTime to 4 µs, the maximum PCO encoder frequency need to be limited to less than 0.25 / $4e^{-6}$ = 62.5 kHz. So, if EncoderScalePitch = 0.004 mm and HardInterpolatorFactor = 200 then the stage maximum velocity must ≤ $62.5e^{3}$ * 0.004 / 200 = 1.25 mm/s, otherwise the PCO will not work properly.

How to determine PCO encoder frequency:

1. ● For AquadB encoder:
   PCO encoder frequency = Velocity / EncoderResolution
2. ● For analog interpolated encoder:
   *PCO encoder frequency = Velocity * HardInterpolatorFactor / EncoderScalePitch*

Example: XML310 stage (EncoderScalePitch=0.004 mm, HardInterpolatorFactor=200).
If Velocity=10mm/s => PCO encoder frequency = 10*200/0.004 = 500 kHz

NOTE

This function can be used only with a position encoder. If no position encoder then ERR_UNCOMPATIBLE (-24) is returned.

This function is called automatically at controller reboot and at GroupInitialize() execution to set the position compare pulse parameters to default values (PCOPulseWidth to 0.2 µs, EncoderSettlingTime to 0.075 µs).

Error codes

- ERR_FATAL_INIT (-20)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_IN_INITIALIZATION (-21)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_UNCOMPATIBLE (-24)
- ERR_HARDWARE_FUNCTION_NOT_SUPPORTED (-115)
- SUCCESS (0): no error

TCL

Prototype

PositionerPositionComparePulseParametersSet $SocketID $FullPositionerName $PCOPulseWidth $EncoderSettlingTime

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– PCOPulseWidth	double	Width of PCO pulses (µs)
– EncoderSettlingTime	double	Encoder signal settling time (µs)

Output parameters

– None

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerPositionComparePulseParametersSet** (int SocketID, char FullPositionerName[250] , double PCOPulseWidth, double EncoderSettlingTime)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	char *	Positioner name
–	PCOPulseWidth	double	Width of PCO pulses (µs)
–	EncoderSettlingTime	double	Encoder signal settling time (µs)

Output parameters

– None

Return

–	Error	int	Function error code

Visual Basic

Prototype

Long **PositionerPositionComparePulseParametersSet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal PCOPulseWidth As Double, ByVal EncoderSettlingTime As Double)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name
–	PCOPulseWidth	double	Width of PCO pulses (µs)
–	EncoderSettlingTime	double	Encoder signal settling time (µs)

Output parameters

– None

Return

–	Error	long	Function error code

Matlab

Prototype

[Error] **PositionerPositionComparePulseParametersSet** (int32 SocketID, cstring FullPositionerName, double PCOPulseWidth, EncoderSettlingTime)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name
– PCOPulseWidth	double	Width of PCO pulses (µs)
– EncoderSettlingTime	double	Encoder signal settling time (µs)

Return

– Error	int32	Function error code

Python

Prototype

[Error] **PositionerPositionComparePulseParametersSet** (integer SocketID, string FullPositionerName, double PCOPulseWidth, double EncoderSettlingTime)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– PCOPulseWidth	double	Width of PCO pulses (µs)
– EncoderSettlingTime	double	Encoder signal settling time (µs)

Return

– Error	int	Function error code

2.2.4.74 PositionerPositionCompareScanAccelerationLimitGet

Name

PositionerPositionCompareScanAccelerationLimitGet – Get the position compare scan acceleration limit.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Check the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)
- Check the corrector type (must be *PIDFFAcceleration* corrector): ERR_UNCOMPATIBLE (-24)

Description

This function returns the position compare scan acceleration limit.

During scan of position compare, the motor output will be limited to this value instead of the *AccelerationLimit*.

The position compare scan acceleration limit takes effect only with *PIDFFAcceleration* corrector type.

This function can be used only with a *PIDFFAcceleration* corrector else ERR_UNCOMPATIBLE is returned.

For a more thorough description of the position compare output, please refer to the XPS Motion Tutorial section Triggers/Position Compare Output.

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_UNCOMPATIBLE (-24)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

Positioner**PositionCompareScanAccelerationLimitGet** $SocketID $FullPositionerName ScanAccelerationLimit

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– ScanAccelerationLimit	double	Limit of position compare scan acceleration (units/s^2)

Return

– Error	int	TCL error code (0=success or 1=syntax error) or function error code

C/C++

Prototype

int **PositionerPositionCompareScanAccelerationLimitGet** (int SocketID, char FullPositionerName[250] , double* ScanAccelerationLimit)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name

Output parameters

– ScanAccelerationLimit	double *	Limit of position compare scan acceleration (units/s^2)

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerPositionCompareScanAccelerationLimitGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ScanAccelerationLimit As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– ScanAccelerationLimit	double	Limit of position compare scan acceleration (units/s^2)

Return

– Error	long	Function error code

Matlab

Prototype

[Error, ScanAccelerationLimit]
PositionerPositionCompareScanAccelerationLimitGet (int32 SocketID, cstring FullPositionerName)

Input parameter

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– ScanAccelerationLimit	double	Limit of position compare scan acceleration (units/s^2)

Python

Prototype

[Error, ScanAccelerationLimit]
PositionerPositionCompareScanAccelerationLimitGet (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code
– ScanAccelerationLimit	double	Limit of position compare scan acceleration (units/s^2)

2.2.4.75 PositionerPositionCompareScanAccelerationLimitSet

Name

PositionerPositionCompareScanAccelerationLimitSet – Sets the position compare acceleration limit.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Check the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the corrector type (must be *PIDFFAcceleration* corrector): ERR_UNCOMPATIBLE (-24)
- Check input parameter values: ERR_PARAMETER_OUT_OF_RANGE (-17)
- PositionCompareScanAccelerationLimit value must > 0 and <= MaximumAcceleration (value in stages.ini)

Description

This function sets the position compare scan acceleration limit.

During position compare, the motor output will be limited to this value instead of *AccelerationLimit*.

The position compare scan acceleration limit takes effect only with *PIDFFAcceleration* corrector type.

This function can be used only with a *PIDFFAcceleration* corrector otherwise ERR_UNCOMPATIBLE is returned.

For a more thorough description of the position compare output, please refer to the XPS Motion Tutorial section Triggers/Position Compare Output.

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_UNCOMPATIBLE (-24)
- SUCCESS (0): no error

TCL

Prototype

PositionerPositionCompareScanAccelerationLimitSet $SocketID $FullPositionerName $ScanAccelerationLimit

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name
–	ScanAccelerationLimit	double	Limit of position compare scan acceleration (units/s^2)

Output parameters

– None

Return

–	Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerPositionCompareScanAccelerationLimitSet** (int SocketID, char FullPositionerName[250] , double ScanAccelerationLimit)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	char *	Positioner name
–	ScanAccelerationLimit	double	Limit of position compare scan acceleration (units/s^2)

Output parameters

– None

Return

–	Error	int	Function error code

Visual Basic

Prototype

Long **PositionerPositionCompareScanAccelerationLimitSet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal ScanAccelerationLimit As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– ScanAccelerationLimit	double	Limit of position compare scan acceleration (units/s^2)

Output parameters

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **PositionerPositionCompareScanAccelerationLimitSet** (int32 SocketID, cstring FullPositionerName, double ScanAccelerationLimit)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name
– ScanAccelerationLimit	double	Limit of position compare scan acceleration (units/s^2)

Return

– Error	int32	Function error code

Python

Prototype

[Error] **PositionerPositionCompareScanAccelerationLimitSet** (integer SocketID, string FullPositionerName, double ScanAccelerationLimit)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– ScanAccelerationLimit	double	Limit of position compare scan acceleration (units/s^2)

Return

– Error	int	Function error code

2.2.4.76 PositionerPositionCompareAquadBAlwaysEnable

Name

PositionerPositionCompareAquadBAlwaysEnable – Enables the AquadB signal in the always mode.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Check the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type: ERR_WRONG_OBJECT_TYPE (-8)
- Check the position encoder ("AquadB" or "AnalogInterpolated"): ERR_UNCOMPATIBLE (-24)
- Check if the CIE board supports this function: ERR_HARDWARE_FUNCTION_NOT_SUPPORTED (-115)

Description

This function enables the generation of AquadB output signals on the PCO connector (the 2&3 or 4&5 pins) of the XPS controller cards. The "always" mode means that the AquadB signal is generated all the time (not position windowed).

NOTE

This function can be used only with a position encoder. If there is no position encoder then ERR_UNCOMPATIBLE (-24) is returned.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_UNCOMPATIBLE (-24)
- ERR_HARDWARE_FUNCTION_NOT_SUPPORTED (-115)
- SUCCESS (0): no error

TCL

Prototype

PositionerPositionCompareAquadBAlwaysEnable $SocketID $FullPositionerName

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– None

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerPositionCompareAquadBAlwaysEnable** (int SocketID, char FullPositionerName[250])

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name

Output parameters

– None

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerPositionCompareAquadBAlwaysEnable** (ByVal SocketID As Long, ByVal FullPositionerName As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **PositionerPositionCompareAquadBAlwaysEnable** (int32 SocketID, cstring FullPositionerName)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	cstring	Positioner name

Return

–	Error	int32	Function error code

Python

Prototype

[Error] **PositionerPositionCompareAquadBAlwaysEnable** (integer SocketID, string FullPositionerName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name

Return

–	Error	int	Function error code

2.2.4.77 PositionerPositionCompareAquadBWindowedGet

Name

PositionerPositionCompareAquadBWindowedGet – Gets the windowed AquadB mode parameters and state..

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Check the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner (must not be a secondary positioner): ERR_WRONG_OBJECT_TYPE (-8)
- Check the position compare parameters (must be configured): ERR_POSITION_COMPARE_NOT_SET (-23)
- Check the configured mode type (must be WindowedAquadB): ERR_UNCOMPATIBLE (-24)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14), ERR_WRONG_TYPE_BOOL (-12)

Description

This function returns the configured parameters of the position windowed AquadB output signal and gives its state (enabled or disabled).

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITION_COMPARE_NOT_SET (-23)
- ERR_UNCOMPATIBLE (-24)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_BOOL (-12)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerPositionCompareAquadBWindowedGet $SocketID $FullPositionerName MinimumPosition MaximumPosition EnableState

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– EnableState	bool	Windowed AquadB state (true=enabled or false=disabled)

Return

– Error	int	TCL error code (0=success or 1=syntax error) or function error code

C/C++

Prototype

int **PositionerPositionCompareAquadBWindowedGet** (int SocketID, char FullPositionerName[250] , double* MinimumPosition, double* MaximumPosition, bool * EnableState)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name

Output parameters

– MinimumPosition	double *	Minimum position (units)
– MaximumPosition	double *	Maximum position (units)
– EnableState	bool *	Windowed AquadB state (true=enabled or false=disabled)

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerPositionCompareAquadBWindowedGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, MinimumPosition As Double, MaximumPosition As Double, EnableState As Boolean)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– EnableState	bool	Windowed AquadB state (true=enabled or false=disabled)

Return

– Error	long	Function error code

Matlab

Prototype

[Error, MinimumPosition, MaximumPosition, EnableState]
PositionerPositionCompareAquadBWindowedGet (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– EnableState	bool	Windowed AquadB state (true=enabled or false=disabled)

Python

Prototype

[Error, MinimumPosition, MaximumPosition, EnableState]
PositionerPositionCompareAquadBWindowedGet (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code
– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– EnableState	bool	Windowed AquadB state (true=enabled or false=disabled)

2.2.4.78 PositionerPositionCompareAquadBWindowedSet

Name

PositionerPositionCompareAquadBWindowedSet – Sets the windowed AquadB signal parameters.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Check the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type: ERR_WRONG_OBJECT_TYPE (-8)
- Check the position encoder ("AquadB" or "AnalogInterpolated"): ERR_UNCOMPATIBLE (-24)
- Check input parameter values: ERR_PARAMETER_OUT_OF_RANGE (-17)
- MinimumPosition < MaximumPosition
- MinimumPosition >= MinimumTargetPosition
- MaximumPosition <= MaximumTargetPosition
- Check if the CIE board supports this function: ERR_HARDWARE_FUNCTION_NOT_SUPPORTED (-115)
- Check position compare state (must be disabled): ERR_NOT_ALLOWED_ACTION (-22)

Description

This function sets the parameters for the position windowed AquadB output signal on the PCO connector (the 2&3 or 4&5 pins) of the XPS controller cards.

These parameters are in effect only when the position compare mode is enabled by the *PositionerPositionCompareEnable()* function.

NOTE

This function can be used only with a position encoder ("AquadB" or "AnalogInterpolated"). If there is no position encoder then ERR_UNCOMPATIBLE (-24) is returned.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_IN_INITIALIZATION (-21)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_UNCOMPATIBLE (-24)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_HARDWARE_FUNCTION_NOT_SUPPORTED (-115)
- SUCCESS (0): no error

TCL

Prototype

PositionerPositionCompareAquadBWindowedSet $SocketID $FullPositionerName $MinimumPosition $MaximumPosition

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name
–	MinimumPosition	double	Minimum position (units)
–	MaximumPosition	double	Maximum position (units)

Output parameters

– None

Return

–	Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **PositionerPositionCompareAquadBWindowedSet** (int SocketID, char FullPositionerName[250] , double * MinimumPosition, double * MaximumPosition)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	char *	Positioner name
–	MinimumPosition	double	Minimum position (units)
–	MaximumPosition	double	Maximum position (units)

Output parameters

– None

Return

–	Error	int	Function error code

Visual Basic

Prototype

Long **PositionerPositionCompareAquadBWindowedSet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal MinimumPosition As Double, ByVal MaximumPosition As Double)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name
–	MinimumPosition	double	Minimum position (units)
–	MaximumPosition	double	Maximum position (units)

Output parameters

– None

Return

–	Error	long	Function error code

Matlab

Prototype

[Error] **PositionerPositionCompareAquadBWindowedSet** (int32 SocketID, cstring FullPositionerName, double MinimumPosition, double MaximumPosition)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name
– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)

Return

– Error	int32	Function error code

Python

Prototype

[Error] **PositionerPositionCompareAquadBWindowedSet** (integer SocketID, string FullPositionerName, double MinimumPosition, double MaximumPosition)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)

Return

– Error	int	Function error code

2.2.4.79 PositionerPreCorrectorExcitationSignalGet

Name

PositionerPreCorrectorExcitationSignalGet – Returns the currently used parameters of the pre-corrector excitation signal functionality.

Input tests

- Configuration files reading: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid object type: ERR_WRONG_OBJECT_TYPE (-8)
- Valid positioner name: ERR_GROUP_NAME (-19)
- Valid output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function gets the last configured pre-corrector excitation signal parameters.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0) : no error

TCL

Prototype

PositionerPreCorrectorExcitationSignalGet SocketID $PositionerName Frequency Amplitude Time

Input parameters

– SocketID	integer	Socket identifier gets by the "TCP_ConnectToServer" function
– PositionerName	string	Positioner name (maximum size = 250)

Output parameters

– Frequency	floating point	Frequency (Hz)
– Amplitude	floating point	Amplitude (position unit)
– Time	floating point	During time (seconds)

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **PositionerPreCorrectorExcitationSignalGet** (int SocketID, char *PositionerName, double* Frequency, double* Amplitude, double* Time)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– PositionerName	char *	Positioner name

Output parameters

– Frequency	double *	Frequency (Hz)
– Amplitude	double *	Amplitude (position units)
– Time	double *	During time (seconds)

Return

– Function error code

Visual Basic

Prototype

Long **PositionerPreCorrectorExcitationSignalGet** (ByVal SocketID As Long, ByVal GroupName As String, Frequency As Double, Amplitude As Double, Time As Double)

Input parameters

– SocketID	Long	Socket identifier gets by the "TCP_ConnectToServer" function
– PositionerName	String	Positioner name

Output parameters

– Frequency	Double	Frequency (Hz)
– Amplitude	Double	Amplitude (position unit)
– Time	Double	During time (seconds)

Return

– Function error code

Matlab

Prototype

[Error, Frequency, Amplitude, Time] **PositionerPreCorrectorExcitationSignalGet** (int32 SocketID, cstring PositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– PositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– Frequency	doublePtr	Frequency (Hz)
– Amplitude	doublePtr	Amplitude (position unit)
– Time	doublePtr	During time (seconds)

Python

Prototype

[Error, Frequency, Amplitude, Time] **PositionerPreCorrectorExcitationSignalGet** (integer SocketID, string PositionerName)

Input parameters

– SocketID	integer	Socket identifier gets by the "TCP_ConnectToServer"function
– PositionerName	string	Positioner name

Return

– Error	integer	Function error code
– Frequency	doublePtr	Frequency (Hz)
– Amplitude	doublePtr	Amplitude (position unit)
– Time	doublePtr	During time (seconds)

2.2.4.80 PositionerPreCorrectorExcitationSignalSet

Name

PositionerPreCorrectorExcitationSignalSet – Configure and activate the signal of pre-corrector excitation.

Input tests

- Configuration files reading: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid object type: ERR_WRONG_OBJECT_TYPE (-8)
- Is secondary positioner or has a secondary positioner: ERR_WRONG_OBJECT_TYPE (-8)
- Valid control loop type: ERR_UNCOMPATIBLE (-24)
- Valid positioner name: ERR_GROUP_NAME (-19)
- Valid output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)
- Check group status (*must be READY*): ERR_NOT_ALLOWED_ACTION (-22)
- Check frequency (*must ≥ 0.1 and ≤0.5/CorrectorISRPeriod*): ERR_PARAMETER_OUT_OF_RANGE (-17)
- Check amplitude (*must > 0*): ERR_PARAMETER_OUT_OF_RANGE (-17)
- Check excitation time (*must ≥ 4*CorrectorISRPeriod*): ERR_PARAMETER_OUT_OF_RANGE (-17)
- Check position (*CurrSetpointPos ≤ UserMaximumTargetPosition and CurrSetpointPos-2Amplitude ≥ UserMinimumTargetPosition)* : ERR_TRAVEL_LIMITS (-35)
- Check maximum velocity (*Amplitude * ω < MaximumVelocity*): ERR_TRAJ_VEL_LIMIT (-68)
- Check maximum acceleration (*Amplitude * ω² < MaximumAcceleration*): ERR_TRAJ_ACC_LIMIT (-69)

Description

The pre-corrector excitation signal functionality generates sine-wave signals (*ExcitationPosition, ExcitationVelocity, ExcitationAcceleration, ExcitationJerk*) inserted in the position control loop (*in closed or open loop configuration*) to excite the system. In measuring the output signal (*eg. encoder position, velocity or acceleration*) of the excited system, we can determine some system characteristics, like the system transfer function.

The exact forms of generated pre-corrector excitation signals are as follows:

$\omega = 2\pi F$ *(F: excitation frequency)*

$ExcitationPosition = A*\cos(\omega t) - A$ *(A: excitation amplitude, t: current time)*

$ExcitationVelocity = (-A\omega)*\sin(\omega t)$

$ExcitationAcceleration = (-A\omega^2)*\cos(\omega t)$

$ExcitationJerk = (A\omega^3)*\sin(\omega t)$

The pre-corrector excitation signal functionality is only available with the stages controlled in acceleration (acceleration control, ex: brushless / linear motors), velocity

(velocity control) or in voltage (voltage control). It does not exist with stages controlled in position (ex: stepper motors).

The excitation signal function **PositionerPreCorrectorExcitationSignalSet** can be executed only when the positioner is in "READY" state. When the excitation-signal process is in progression, the positioner is in the "ExcitationSignal" state. At the end of the process, the positioner returns to the "READY" state (see group state diagram).

The **PositionerPreCorrectorExcitationSignalSet** function sends a sine form excitation signal to entry of position control loop during a time. This function is allowed for "PIDFFAcceleration", "PIDFFVelocity" or "PIDDualFFVoltage" control loops. The parameters to configure include: *frequency* (Hz, double), *amplitude* (position unit, double) and *during time* (seconds, double).

The function effective parameters are : *Frequency* ($\geq$ 0.1 and $\leq$ *0.5/CorrectorISRPeriod*), *Amplitude* ($>$ 0), *Time* ($\geq$ *4*CorrectorISRPeriod*).

NOTE

If during the excitation signal generation the stage position exceeds the user minimum or maximum target positions, the motor excitation command is stopped and an error is returned.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_IN_INITIALIZATION (-21)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_UNCOMPATIBLE (-24)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- ERR_TRAVEL_LIMITS (-35)
- ERR_TRAJ_VEL_LIMIT (-68)
- ERR_TRAJ_ACC_LIMIT (-69)
- ERR_EXCITATION_SIGNAL_INITIALIZATION (-112)
- SUCCESS (0) : no error

TCL

Prototype

PositionerPreCorrectorExcitationSignalSet SocketID $PositionerName $Frequency $Amplitude $Time

Input parameters

– SocketID	integer	Socket identifier gets by the "TCP_ConnectToServer" function
– PositionerName	string	Positioner name (maximum size = 250)
– Frequency	floating point	Frequency (Hz)
– Amplitude	floating point	Amplitude (position unit)
– Time	floating point	During time (seconds)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **PositionerPreCorrectorExcitationSignalSet** (int SocketID, char *PositionerName, double Frequency, double Amplitude, double Time)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– PositionerName	char *	Positioner name
– Frequency	double	Frequency (Hz)
– Amplitude	double	Amplitude (position unit)
– Time	double	During time (seconds)

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **PositionerPreCorrectorExcitationSignalSet** (ByVal SocketID As Long, ByVal GroupName As String, ByVal Frequency As Double, ByVal Amplitude As Double, ByVal Time As Double)

Input parameters

– SocketID	Long	Socket identifier gets by the "TCP_ConnectToServer" function
– PositionerName	String	Positioner name
– Frequency	Double	Frequency (Hz)
– Amplitude	double	Amplitude (position unit)
– Time	Double	During time (seconds)

Output parameters

– None

Return

– Function error code

Matlab

Prototype

[Error] **PositionerPreCorrectorExcitationSignalSet** (int32 SocketID, cstring PositionerName, Frequency, Amplitude, Time)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– PositionerName	cstring	Positioner name
– Frequency	double	Frequency (Hz)
– Amplitude	double	Amplitude (position unit)
– Time	double	During time (seconds)

Return

– Error	int32	Function error code

Python

Prototype

[Error] **PositionerPreCorrectorExcitationSignalSet** (integer SocketID, string PositionerName, Frequency, Amplitude, Time)

Input parameters

– SocketID	integer	Socket identifier gets by the "TCP_ConnectToServer"function
– PositionerName	string	Positioner name
– Frequency	double	Frequency (Hz)
– Amplitude	double	Amplitude (position unit)
– Time	double	During time (seconds)

Return

– Error	integer	Function error code

2.2.4.81 PositionerRawEncoderPositionGet

Name

PositionerRawEncoderPositionGet – Returns the raw encoder position for a positioner.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid object type: ERR_WRONG_OBJECT_TYPE (-8)
- Valid positioner name: ERR_GROUP_NAME (-19)
- Valid output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function returns the raw encoder position from a corrected position for a positioner.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerRawEncoderPositionGet SocketID PositionerName UserEncoderPosition RawEncoderPosition

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	PositionerName	string	Positioner name (maximum size = 250)
–	UserEncoderPosition	floating point	User corrected encoder position

Output parameters

–	RawEncoderPosition	floating point	Raw encoder position

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **PositionerRawEncoderPositionGet** (int SocketID, char * PositionerName, double UserEncoderPosition, double * RawEncoderPosition)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	PositionerName	char *	Positioner name
–	UserEncoderPosition	double	User corrected encoder position

Output parameters

–	RawEncoderPosition	double *	Raw encoder position

Return

– Function error code

Visual Basic

Prototype

Long **PositionerRawEncoderPositionGet** (ByVal SocketID As Long, ByVal PositionerName As String, ByVal UserEncoderPosition As Double, RawEncoderPosition As Double)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	PositionerName	string	Positioner name
–	UserEncoderPosition	double	User corrected encoder position

Output parameters

–	RawEncoderPosition	double	Raw encoder position

Return

– Function error code

Matlab

Prototype

[Error, RawEncoderPosition] **PositionerRawEncoderPositionGet** (int32 SocketID, cstring PositionerName, double UserEncoderPosition)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
–	PositionerName	cstring	Positioner name
–	UserEncoderPosition	double	User corrected encoder position

Return

–	Error	int32	Function error code
–	RawEncoderPosition	doubletPtr	Raw encoder position

Python

Prototype

[Error, RawEncoderPosition] **PositionerRawEncoderPositionGet** (integer SocketID, string PositionerName, double UserEncoderPosition)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	PositionerName	string	Positioner name
–	UserEncoderPosition	double	User corrected encoder position

Return

–	Error	int	Function error code
–	RawEncoderPosition	doubletPtr	Raw encoder position

2.2.4.82 PositionersEncoderIndexDifferenceGet

Name

PositionersEncoderIndexDifferenceGet – Gets the distance between the two index encoders (gantry).

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Check the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the group type (must be a SingleAxis or an XY): ERR_WRONG_OBJECT_TYPE (-8)
- Check the positioner name: ERR_POSITIONER_NAME (-18)
- Check the positioner type (must not be a secondary positioner): ERR_WRONG_OBJECT_TYPE (-8)
- Check the positioner type (must be "gantry"): ERR_UNCOMPATIBLE (-24)
- Check the positioner was at least once homed: ERR_NEED_TO_BE_HOMED_AT_LEAST_ONCE (-109)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function returns the distance between the two encoders indexes of a "primary positioner – secondary positioner" couple. To use this function, the positioner must be configured in "gantry" mode else ERR_UNCOMPATIBLE (-24) is returned.

For further information about gantry mode, refer to the "SYSTEM – Manual Configuration – Gantries (Secondary Positioners)" section in the XPS user's manual.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_NEED_TO_BE_HOMED_AT_LEAST_ONCE (-109)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionersEncoderIndexDifferenceGet $SocketID $FullPositionerName Distance

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– Distance	double	Distance between the two index encoders (units)

Return

– Error	int	TCL error code (0=success or 1=syntax error) or function error code

C/C++

Prototype

int **PositionersEncoderIndexDifferenceGet** (int SocketID, char FullPositionerName[250] , double* Distance)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name

Output parameters

– Distance	double *	Distance between the two index encoders (units)

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionersEncoderIndexDifferenceGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, Distance As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– Distance	double	Distance between the two index encoders (units)

Return

– Error	long	Function error code

Matlab

Prototype

[Error, Distance] **PositionersEncoderIndexDifferenceGet** (int32 SocketID, cstring FullPositionerName)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	cstring	Positioner name

Return

–	Error	int32	Function error code
–	Distance	double	Distance between the two index encoders (units)

Python

Prototype

[Error, Distance] **PositionersEncoderIndexDifferenceGet** (integer SocketID, string FullPositionerName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name

Return

–	Error	int	Function error code
–	Distance	double	Distance between the two index encoders (units)

2.2.4.83 PositionerSGammaExactVelocityAjustedDisplacementGet

Name

PositionerSGammaExactVelocityAjustedDisplacementGet – Gets the adjusted displacement to get exact velocity.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Check command format: ERR_WRONG_FORMAT (-7)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the profiler type (must be "SGamma"): ERR_WRONG_OBJECT_TYPE (-8)
- Check the positioner type (must not be a secondary positioner): ERR_WRONG_OBJECT_TYPE (-8)
- Check output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function returns the closest optimum displacement to obtain the most precise velocity during the displacement.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerSGammaExactVelocityAjustedDisplacementGet $SocketID $FullPositionerName $DesiredDisplacement AdjustedDisplacement

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name
–	DesiredDisplacement	double	Desired displacement (units)

Output parameters

–	AdjustedDisplacement	double	Ajusted displacement (units)

Return

–	Error	int	TCL error code (0=success or 1=syntax error) or function error code

C/C++

Prototype

int **PositionerSGammaExactVelocityAjustedDisplacementGet** (int SocketID, char FullPositionerName[250] , double DesiredDisplacement, double * AdjustedDisplacement)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name
– DesiredDisplacement	double	Desired displacement (units)

Output parameters

– AdjustedDisplacement	double *	Ajusted displacement (units)

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerSGammaExactVelocityAjustedDisplacementGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal DesiredDisplacement As Double, AdjustedDisplacement As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– DesiredDisplacement	double	Desired displacement (units)

Output parameters

– AdjustedDisplacement	double	Ajusted displacement (units)

Return

– Error	long	Function error code

Matlab

Prototype

[Error, AdjustedDisplacement]
PositionerSGammaExactVelocityAjustedDisplacementGet (int32 SocketID, cstring FullPositionerName, double DesiredDisplacement)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name
– DesiredDisplacement	double	Desired displacement (units)

Return

– Error	int32	Function error code
– AdjustedDisplacement	double	Ajusted displacement (units)

Python

Prototype

[Error, AdjustedDisplacement]
PositionerSGammaExactVelocityAjustedDisplacementGet (integer SocketID, string FullPositionerName, double DesiredDisplacement)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– DesiredDisplacement	double	Desired displacement (units)

Return

– Error	int	Function error code
– AdjustedDisplacement	double	Ajusted displacement (units)

2.2.4.84 PositionerSGammaParametersSet

Name

PositionerSGammaParametersSet – Sets new motion values for the SGamma profiler.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Check the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the profiler type (must be "SGamma"): ERR_WRONG_OBJECT_TYPE (-8)
- Check the positioner type (must not be a secondary positioner): ERR_WRONG_OBJECT_TYPE (-8)
- Check input parameter types: ERR_WRONG_TYPE_DOUBLE (-14)
- Check input parameter values: ERR_PARAMETER_OUT_OF_RANGE (-17)
- 0 < **NewVelocity** ≤ MaximumVelocity
- 0 < **NewAcceleration** ≤ MaximumAcceleration
- 2 * ISRProfileGeneratorPeriod ≤ NewMinimumJerkTime ≤ NewMaximumJerkTime
- (with ISRProfileGeneratorPeriod = 0.0004 ms)

Description

This function defines the new SGamma profiler values that will be used in future displacements.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerSGammaParametersSet $SocketID $FullPositionerName $Velocity $Acceleration $MinimumJerkTime $MaximumJerkTime

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– Velocity	double	motion velocity (units/seconds)
– Acceleration	double	motion acceleration (units/seconds²)
– MinimumJerkTime	double	Minimum jerk time (seconds)
– MaximumJerkTime	double	Maximum jerk time (seconds)

Output parameters

– None

Return

– Error	int	TCL error code (0=success or 1=syntax error) or function error code

C/C++

Prototype

int **PositionerSGammaParametersSet** (int SocketID, char FullPositionerName[250] , double Velocity, double Acceleration, double MinimumJerkTime, double MaximumJerkTime)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name
– Velocity	double	motion velocity (units/seconds)
– Acceleration	double	motion acceleration (units/seconds²)
– MinimumJerkTime	double	Minimum jerk time (seconds)
– MaximumJerkTime	double	Maximum jerk time (seconds)

Output parameters

– None

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerSGammaParametersSet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal Velocity As Double, ByVal Acceleration As Double, ByVal MinimumJerkTime As Double, ByVal MaximumJerkTime As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– Velocity	double	motion velocity (units/seconds)
– Acceleration	double	motion acceleration (units/seconds²)
– MinimumJerkTime	double	Minimum jerk time (seconds)
– MaximumJerkTime	double	Maximum jerk time (seconds)

Output parameters

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **PositionerSGammaParametersSet** (int32 SocketID, cstring FullPositionerName, double Velocity, double Acceleration, double MinimumJerkTime, double MaximumJerkTime)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name
– Velocity	double	motion velocity (units/seconds)
– Acceleration	double	motion acceleration (units/seconds²)
– MinimumJerkTime	double	Minimum jerk time (seconds)
– MaximumJerkTime	double	Maximum jerk time (seconds)

Return

– Error	int32	Function error code

Python

Prototype

[Error] **PositionerSGammaParametersSet** (integer SocketID, string FullPositionerName, double Velocity, double Acceleration, double MinimumJerkTime, double MaximumJerkTime)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name
–	Velocity	double	motion velocity (units/seconds)
–	Acceleration	double	motion acceleration (units/seconds²)
–	MinimumJerkTime	double	Minimum jerk time (seconds)
–	MaximumJerkTime	double	Maximum jerk time (seconds)

Return

–	Error	int	Function error code

2.2.4.85 PositionerSGammaParametersGet

Name

PositionerSGammaParametersGet – Gets the current motion values from the SGamma profiler.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Check the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the profiler type (must be "SGamma"): ERR_WRONG_OBJECT_TYPE (-8)
- Check the positioner type (must not be a secondary positioner): ERR_WRONG_OBJECT_TYPE (-8)
- Check input parameter types: ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function gets the current SGamma profiler values that are used in displacements.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerSGammaParametersGet $SocketID $FullPositionerName Velocity Acceleration MinimumJerkTime MaximumJerkTime

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– Velocity	double	motion velocity (units/seconds)
– Acceleration	double	motion acceleration (units/seconds²)
– MinimumJerkTime	double	Minimum jerk time (seconds)
– MaximumJerkTime	double	Maximum jerk time (seconds)

Return

– Error	int	TCL error code (0=success or 1=syntax error) or function error code

C/C++

Prototype

int **PositionerSGammaParametersGet** (int SocketID, char FullPositionerName[250] , double * Velocity, double * Acceleration, double * MinimumJerkTime, double * MaximumJerkTime)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name

Output parameters

– Velocity	double *	motion velocity (units/seconds)
– Acceleration	double *	motion acceleration (units/seconds²)
– MinimumJerkTime	double *	Minimum jerk time (seconds)
– MaximumJerkTime	double *	Maximum jerk time (seconds)

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerSGammaParametersGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, Velocity As Double, Acceleration As Double, MinimumJerkTime As Double, MaximumJerkTime As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– Velocity	double	motion velocity (units/seconds)
– Acceleration	double	motion acceleration (units/seconds²)
– MinimumJerkTime	double	Minimum jerk time (seconds)
– MaximumJerkTime	double	Maximum jerk time (seconds)

Return

– Error	long	Function error code

Matlab

Prototype

[Error, Velocity, Acceleration, MinimumJerkTime, MaximumJerkTime] **PositionerSGammaParametersGet** (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– Velocity	double	motion velocity (units/seconds)
– Acceleration	double	motion acceleration (units/seconds²)
– MinimumJerkTime	double	Minimum jerk time (seconds)
– MaximumJerkTime	double	Maximum jerk time (seconds)

Python

Prototype

[Error, Velocity, Acceleration, MinimumJerkTime, MaximumJerkTime]
PositionerSGammaParametersGet (integer SocketID, string FullPositionerName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name

Return

–	Error	int	Function error code
–	Velocity	double	motion velocity (units/seconds)
–	Acceleration	double	motion acceleration (units/seconds²)
–	MinimumJerkTime	double	Minimum jerk time (seconds)
–	MaximumJerkTime	double	Maximum jerk time (seconds)

2.2.4.86 PositionerSGammaPreviousMotionTimesGet

Name

PositionerSGammaPreviousMotionTimesGet – Gets the motion and the settling time.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the profiler type (must be "SGamma"): ERR_WRONG_OBJECT_TYPE (-8)
- Check the positioner type (must not be a secondary positioner): ERR_WRONG_OBJECT_TYPE (-8)
- Check input parameter types: ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function returns the motion (setting) and settling times from the previous motion.

The motion time represents the defined time to complete the previous displacement.

The settling time represents the effective settling time for a motion done.

Error codes

- ERR_FATAL_INIT (-20
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerSGammaPreviousMotionTimesGet $SocketID $FullPositionerName SettingTime SettlingTime

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– SettingTime	double	Setting time (seconds)
– SettlingTime	double	settling time (seconds)

Return

– Error	int	TCL error code (0=success or 1=syntax error) or function error code

C/C++

Prototype

int **PositionerSGammaPreviousMotionTimesGet** (int SocketID, char FullPositionerName[250] , double* SettingTime, double* SettlingTime)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name

Output parameters

– SettingTime	double *	Setting time (seconds)
– SettlingTime	double *	settling time (seconds)

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerSGammaPreviousMotionTimesGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, SettingTime As Double, SettlingTime As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– SettingTime	double	Setting time (seconds)
– SettlingTime	double	Settling time (seconds)

Return

– Error	long	Function error code

Matlab

Prototype

[Error, SettingTime, SettlingTime] **PositionerSGammaPreviousMotionTimesGet** (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– SettingTime	double	Setting time (seconds)
– SettlingTime	double	Settling time (seconds)

Python

Prototype

[Error, SettingTime, SettlingTime] **PositionerSGammaPreviousMotionTimesGet** (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code
– SettingTime	double	Setting time (seconds)
– SettlingTime	double	Settling time (seconds)

2.2.4.87 PositionerStageParameterGet

Name

PositionerStageParameterGet – Gets a stage parameter value from the stages.ini file.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check input/output parameter types: ERR_WRONG_TYPE_CHAR (-13)
- Check the positioner name and the parameter name: ERR_UNCOMPATIBLE (-24)

Description

This function returns stage parameter values from the stages.ini file of a selected positioner.

The positioner name is the stage name. And the parameter name is read in the section under this stage name.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_UNCOMPATIBLE (-24)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13)
- SUCCESS (0): no error

TCL

Prototype

PositionerStageParameterGet $SocketID $FullPositionerName $ParameterName ParameterValue

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– ParameterName	string	Parameter name

Output parameters

– ParameterValue	string	Parameter value

Return

– Error	int	TCL error code (0=success or 1=syntax error) or function error code

C/C++

Prototype

int **PositionerStageParameterGet** (int SocketID, char FullPositionerName[250] , char * ParameterName, char * ParameterValue)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name
– ParameterName	char *	Parameter name

Output parameters

– ParameterValue	char *	Parameter value

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerStageParameterGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal ParameterName As String, ByVal ParameterValue As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– ParameterName	string	Parameter name

Output parameters

– ParameterValue	string	Parameter value

Return

– Error	long	Function error code

Matlab

Prototype

[Error, ParameterValue] **PositionerStageParameterGet** (int32 SocketID, cstring FullPositionerName, cstring ParameterName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name
– ParameterName	cstring	Parameter name

Return

– Error	int32	Function error code
– ParameterValue	cstring	Parameter value

Python

Prototype

[Error, ParameterValue] **PositionerStageParameterGet** (integer SocketID, string FullPositionerName, string ParameterName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– ParameterName	string	Parameter name

Return

– Error	int	Function error code
– ParameterValue	string	Parameter value

2.2.4.88 PositionerStageParameterSet

Name

PositionerStageParameterSet – Saves a new stage parameter value into the stages.ini file.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check input/output parameter types: ERR_WRONG_TYPE_CHAR (-13)
- Check the positioner name and the parameter name: ERR_UNCOMPATIBLE (-24)
- Check the user rights (must be identified as administrator): ERR_NEED_ADMINISTRATOR_RIGHTS (-107)

Description

This function saves a new stage parameter value in the "stages.ini" file.

The positioner name sets the stage name and the parameter name is searched in the section of this stage name. Once the parameter is found, the parameter value is modified to the new value.

If file reading fails then ERR_READ_FILE (-61) is returned

If file writing fails then ERR_WRITE_FILE (-60) is returned

NOTE

To use this function, the user must login with administrator rights ("Login" function).

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_NEED_ADMINISTRATOR_RIGHTS (-107)
- ERR_READ_FILE (-61)
- ERR_UNCOMPATIBLE (-24)
- ERR_WRITE_FILE (-60)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13)
- SUCCESS (0): no error

TCL

Prototype

PositionerStageParameterSet $SocketID $FullPositionerName $ParameterName ParameterValue

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– ParameterName	string	Parameter name

Output parameters

– ParameterValue	string	Parameter value

Return

– Error	int	TCL error code (0=success or 1=syntax error) or function error code

C/C++

Prototype

int **PositionerStageParameterSet** (int SocketID, char FullPositionerName[250] , char * ParameterName, char * ParameterValue)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name
– ParameterName	char *	Parameter name

Output parameters

– ParameterValue	char *	Parameter value

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerStageParameterSet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal ParameterName As String, ByVal ParameterValue As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– ParameterName	string	Parameter name

Output parameters

– ParameterValue	string	Parameter value

Return

– Error	long	Function error code

Matlab

Prototype

[Error, ParameterValue] **PositionerStageParameterSet** (int32 SocketID, cstring FullPositionerName, cstring ParameterName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name
– ParameterName	cstring	Parameter name

Return

– Error	int32	Function error code
– ParameterValue	cstring	Parameter value

Python

Prototype

[Error, ParameterValue] **PositionerStageParameterSet** (integer SocketID, string FullPositionerName, string ParameterName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– ParameterName	string	Parameter name

Return

– Error	int	Function error code
– ParameterValue	string	Parameter value

2.2.4.89 PositionerTimeFlasherDisable

Name

PositionerTimeFlasherDisable – Disables the time flasher mode.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type (must not be a secondary positioner): ERR_WRONG_OBJECT_TYPE (-8)

Description

This function disables the time flasher mode. The time flasher mode is a trigger output per axis that can be either configured to output distance spaced pulses or time spaced pulses. The output pulses are accessible from the PCO connector at the back of the XPS controller.

For a more thorough description of the position compare output, please refer to the XPS User's manual, section named Triggers/Position Compare Output.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_OBJECT_TYPE (-8)
- SUCCESS (0): no error

TCL

Prototype

PositionerTimeFlasherDisable $SocketID $FullPositionerName

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name

Output parameters

– None

Return

–	Error	int	TCL error code (0=success or 1=syntax error) or function error code

C/C++

Prototype

int **PositionerTimeFlasherDisable** (int SocketID, char FullPositionerName[250])

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	char *	Positioner name

Output parameters

– None

Return

–	Error	int	Function error code

Visual Basic

Prototype

Long **PositionerTimeFlasherDisable** (ByVal SocketID As Long, ByVal FullPositionerName As String)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name

Output parameters

– None

Return

–	Error	long	Function error code

Matlab

Prototype

[Error] **PositionerTimeFlasherDisable** (int32 SocketID, cstring FullPositionerName)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	cstring	Positioner name

Return

–	Error	int32	Function error code

Python

Prototype

[Error] **PositionerTimeFlasherDisable** (integer SocketID, string FullPositionerName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name

Return

–	Error	int	Function error code

2.2.4.90 PositionerTimeFlasherEnable

Name

PositionerTimeFlasherEnable – Enables the time flasher mode.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type (must not be a secondary positioner): ERR_WRONG_OBJECT_TYPE (-8)
- Check the time flasher parameters (must be configured): ERR_NOT_ALLOWED_ACTION (-22)

Description

This function enables the time flasher mode. The time flasher mode is a trigger output per axis that can be either configured to output distance spaced pulses or time spaced pulses. The output pulses are accessible from the PCO connector at the back of the XPS controller.

To use this function, the group must be in READY state else ERR_NOT_ALLOWED_ACTION (-22) is returned.

For a more thorough description of the position compare output, please refer to the XPS User's manual, section named Triggers/Position Compare Output.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_OBJECT_TYPE (-8)
- SUCCESS (0): no error

TCL

Prototype

PositionerTimeFlasherEnable $SocketID $FullPositionerName

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name

Output parameters

– None

Return

–	Error	int	TCL error code (0=success or 1=syntax error) or function error code

C/C++

Prototype

int **PositionerTimeFlasherEnable** (int SocketID, char FullPositionerName[250])

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	char *	Positioner name

Output parameters

– None

Return

–	Error	int	Function error code

Visual Basic

Prototype

Long **PositionerTimeFlasherEnable** (ByVal SocketID As Long, ByVal FullPositionerName As String)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name

Output parameters

– None

Return

–	Error	long	Function error code

Matlab

Prototype

[Error] **PositionerTimeFlasherEnable** (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code

Python

Prototype

[Error] **PositionerTimeFlasherEnable** (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code

2.2.4.91 PositionerTimeFlasherGet

Name

PositionerTimeFlasherGet – Gets the time flasher parameters.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type (must not be a secondary positioner): ERR_WRONG_OBJECT_TYPE (-8)
- Check output parameter types: ERR_WRONG_TYPE_DOUBLE (-14), ERR_WRONG_TYPE_BOOL (-12)
- Check the time flasher parameters (must be configured): ERR_POSITION_COMPARE_NOT_SET (-23)
- Check the configured mode type (must be TimeFlasher): ERR_UNCOMPATIBLE (-24)

Description

This function returns the parameters of the time flasher trigger. The time flasher mode is defined by:

a position window defined by a minimum position and a maximum position

a time period to set the time spaced pulses.

For a more thorough description of the position compare output, please refer to the XPS Motion Tutorial section Triggers/Position Compare Output.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITION_COMPARE_NOT_SET (-23)
- ERR_UNCOMPATIBLE (-24)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_BOOL (-12)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerTimeFlasherGet $SocketID $FullPositionerName MinimumPosition MaximumPosition TimePeriod EnableState

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– TimePeriod	double	Time period (seconds)
– EnableState	bool	Enable time flasher state (true=enabled and false=disabled)

Return

– Error	int	TCL error code (0=success or 1=syntax error) or function error code

C/C++

Prototype

int **PositionerTimeFlasherGet** (int SocketID, char FullPositionerName[250] , double * MinimumPosition, double * MaximumPosition, double * TimePeriod, bool * EnableState)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name

Output parameters

– MinimumPosition	double *	Minimum position (units)
– MaximumPosition	double *	Maximum position (units)
– TimePeriod	double *	Time period (seconds)
– EnableState	bool *	Enable time flasher state (true=enabled and false=disabled)

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerTimeFlasherGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, MinimumPosition As Double, MaximumPosition As Double, TimePeriod As Double, EnableState As Boolean)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– TimePeriod	double	Time period (seconds)
– EnableState	bool	Enable time flasher state (true=enabled and false=disabled)

Return

– Error	long	Function error code

Matlab

Prototype

[Error, MinimumPosition, MaximumPosition, TimePeriod, EnableState] **PositionerTimeFlasherGet** (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– TimePeriod	double	Time period (seconds)
– EnableState	bool	Enable time flasher state (true=enabled and false=disabled)

Python

Prototype

[Error, MinimumPosition, MaximumPosition, TimePeriod, EnableState]
PositionerTimeFlasherGet (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code
– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– TimePeriod	double	Time period (seconds)
– EnableState	bool	Enable time flasher state (true=enabled and false=disabled)

2.2.4.92 PositionerTimeFlasherSet

Name

PositionerTimeFlasherSet – Sets the time flasher parameters.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Check the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type (must not be a secondary positioner): ERR_WRONG_OBJECT_TYPE (-8)
- Check the position encoder (must be used): ERR_UNCOMPATIBLE (-24)
- Check input parameter types: ERR_WRONG_TYPE_DOUBLE (-14)
- Check if the CIE board supports this function: ERR_HARDWARE_FUNCTION_NOT_SUPPORTED (-115)
- Check the time flasher state (must be disabled): ERR_NOT_ALLOWED_ACTION (-22)
- Check input parameter values: ERR_PARAMETER_OUT_OF_RANGE (-17)
- MinimumPosition < MaximumPosition
- MinimumPosition >= MinimumTravelLimit
- MaximumPosition <= MaximumTravelLimit
- 0.0000004 <= TimePeriod <= 50.0 (Max 2.5 MHz and Min 0.02 Hz)

Description

This function configures the time flasher parameters. The time flasher output trigger uses the PCO connector on the XPS controller cards. The time flasher mode is defined by:

a position window defined by a minimum position and a maximum position

a time period to set the time spaced pulses.

NOTES

This function is not available without a position encoder.

These parameters are used only when the time flasher mode is enabled. To enable the time flasher mode, use the "PositionerPositionCompareEnable" function.

For a more thorough description of the position compare output, please refer to the XPS Motion Tutorial section Triggers/Position Compare Output.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_UNCOMPATIBLE (-24)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- ERR_HARDWARE_FUNCTION_NOT_SUPPORTED (-115)
- SUCCESS (0): no error

TCL

Prototype

PositionerTimeFlasherSet $SocketID $FullPositionerName $MinimumPosition $MaximumPosition $TimePeriod

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– TimePeriod	double	Time period (seconds)

Output parameters

– None

Return

– Error	int	TCL error code (0=success or 1=syntax error) or function error code

C/C++

Prototype

int **PositionerTimeFlasherSet** (int SocketID, char FullPositionerName[250] , double * MinimumPosition, double * MaximumPosition, double * TimePeriod, bool * EnableState)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name
– MinimumPosition	double *	Minimum position (units)
– MaximumPosition	double *	Maximum position (units)
– TimePeriod	double *	Time period (seconds)

Output parameters

– None

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerTimeFlasherSet** (ByVal SocketID As Long, ByVal FullPositionerName As String, MinimumPosition As Double, MaximumPosition As Double, TimePeriod As Double, EnableState As Boolean)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– TimePeriod	double	Time period (seconds)

Output parameters

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **PositionerTimeFlasherSet** (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name
– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– TimePeriod	double	Time period (seconds)

Return

– Error	int32	Function error code

Python

Prototype

[Error] **PositionerTimeFlasherSet** (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– TimePeriod	double	Time period (seconds)

Return

– Error	int	Function error code

2.2.4.93 PositionerUserTravelLimitsGet

Name

PositionerUserTravelLimitsGet – Gets the user travel limits.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type (must not be a secondary positioner): ERR_WRONG_OBJECT_TYPE (-8)
- Check output parameter types: ERR_WRONG_TYPE_DOUBLE (-14)
- If piezo driver, check if driver is not initialized: ERR_NOT_ALLOWED_DRIVER_NOT_INITIALIZED (-118)

Description

This function returns the user-defined travel limits for the selected positioner.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- ERR_NOT_ALLOWED_DRIVER_NOT_INITIALIZED (-118)
- SUCCESS (0): no error

TCL

Prototype

PositionerUserTravelLimitsGet $SocketID $FullPositionerName UserMinimumTarget UserMaximumTarget

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

–	UserMinimumTarget	double	User minimum travel limit (units)
–	UserMaximumTarget	double	User maximum travel limit (units)

Return

–	Error	int	TCL error code (0=success or 1=syntax error) or function error code

C/C++

Prototype

int **PositionerUserTravelLimitsGet** (int SocketID, char FullPositionerName[250] , double * UserMinimumTarget, double * UserMaximumTarget)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name

Output parameters

– UserMinimumTarget	double *	User minimum travel limit (units)
– UserMaximumTarget	double *	User maximum travel limit (units)

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerUserTravelLimitsGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, UserMinimumTarget As Double, UserMaximumTarget As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– UserMinimumTarget	double	User minimum travel limit (units)
– UserMaximumTarget	double	User maximum travel limit (units)

Return

– Error	long	Function error code

Matlab

Prototype

[Error, UserMinimumTarget, UserMaximumTarget] **PositionerUserTravelLimitsGet** (int32 SocketID, cstring FullPositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name

Return

– Error	int32	Function error code
– UserMinimumTarget	double	User minimum travel limit (units)
– UserMaximumTarget	double	User maximum travel limit (units)

Python

Prototype

[Error, UserMinimumTarget, UserMaximumTarget] **PositionerUserTravelLimitsGet** (integer SocketID, string FullPositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Return

– Error	int	Function error code
– UserMinimumTarget	double	User minimum travel limit (units)
– UserMaximumTarget	double	User maximum travel limit (units)

2.2.4.94 PositionerUserTravelLimitsSet

Name

PositionerUserTravelLimitsSet – Sets the user travel limits.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Check the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner type (must not be a secondary positioner): ERR_WRONG_OBJECT_TYPE (-8)
- Check output parameter types: ERR_WRONG_TYPE_DOUBLE (-14)
- Check input parameter values: ERR_PARAMETER_OUT_OF_RANGE (-17)
- UserMinimumTargetPosition < UserMaximumTargetPosition
- MinimumTargetPosition <= UserMinimumTargetPosition <= MaximumTargetPosition
- MinimumTargetPosition <= UserMaximumTargetPosition <= MaximumTargetPosition
- UserMinimumTargetPosition <= ProfilerPosition
- UserMaximumTargetPosition >= ProfilerPosition
- If piezo driver, check if driver is not initialized: ERR_NOT_ALLOWED_DRIVER_NOT_INITIALIZED (-118)

Description

This function sets the new user travel limits of the selected positioner.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- ERR_NOT_ALLOWED_DRIVER_NOT_INITIALIZED (-118)
- SUCCESS (0): no error

TCL

Prototype

PositionerUserTravelLimitsSet $SocketID $FullPositionerName $UserMinimumTarget $UserMaximumTarget

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– UserMinimumTarget	double	User minimum travel limit (units)
– UserMaximumTarget	double	User maximum travel limit (units)

Output parameters

– None

Return

– Error	int	TCL error code (0=success or 1=syntax error) or function error code

C/C++

Prototype

int **PositionerUserTravelLimitsSet** (int SocketID, char FullPositionerName[250] , double UserMinimumTarget, double UserMaximumTarget)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name
– UserMinimumTarget	double	User minimum travel limit (units)
– UserMaximumTarget	double	User maximum travel limit (units)

Output parameters

– None

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerUserTravelLimitsSet** (ByVal SocketID As Long, ByVal FullPositionerName As String, ByVal UserMinimumTarget As Double, ByVal UserMaximumTarget As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– UserMinimumTarget	double	User minimum travel limit (units)
– UserMaximumTarget	double	User maximum travel limit (units)

Output parameters

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **PositionerUserTravelLimitsSet** (int32 SocketID, cstring FullPositionerName, double UserMinimumTarget, double UserMaximumTarget)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name
– UserMinimumTarget	double	User minimum travel limit (units)
– UserMaximumTarget	double	User maximum travel limit (units)

Return

– Error	int32	Function error code

Python

Prototype

[Error] **PositionerUserTravelLimitsSet** (integer SocketID, string FullPositionerName, double UserMinimumTarget, double UserMaximumTarget)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– UserMinimumTarget	double	User minimum travel limit (units)
– UserMaximumTarget	double	User maximum travel limit (units)

Return

– Error	int	Function error code

2.2.4.95 PositionerWarningFollowingErrorGet

Name

PositionerWarningFollowingErrorGet – Returns the warning following error for a positioner.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid object type: ERR_WRONG_OBJECT_TYPE (-8)
- Valid positioner name: ERR_POSITIONER_NAME (-18)
- Valid output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function gets the current value of the warning following error for a positioner.

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

PositionerWarningFollowingErrorGet $SocketID $FullPositionerName WarningFollowingError

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– WarningFollowingError	double	Warning following error (units)

Return

– Error	int	TCL error code (0=success or 1=syntax error) or function error code

C/C++

Prototype

int **PositionerWarningFollowingErrorGet** (int SocketID, char FullPositionerName[250] , double * WarningFollowingError)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name

Output parameters

– WarningFollowingError	double *	Warning following error (units)

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerWarningFollowingErrorGet** (ByVal SocketID As Long, ByVal FullPositionerName As String, WarningFollowingError As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name

Output parameters

– WarningFollowingError	double	Warning following error (units)

Return

– Error	long	Function error code

Matlab

Prototype

[Error, WarningFollowingError] **PositionerWarningFollowingErrorGet** (int32 SocketID, cstring FullPositionerName)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	cstring	Positioner name

Return

–	Error	int32	Function error code
–	WarningFollowingError	double	Warning following error limit (units)

Python

Prototype

[Error, WarningFollowingError] **PositionerWarningFollowingErrorGet** (integer SocketID, string FullPositionerName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	FullPositionerName	string	Positioner name

Return

–	Error	int	Function error code
–	WarningFollowingError	double	Warning following error (units)

2.2.4.96 PositionerWarningFollowingErrorSet

Name

PositionerWarningFollowingErrorSet – Sets value of the warning following error for a positioner.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid object type: ERR_WRONG_OBJECT_TYPE (-8)
- Valid positioner name: ERR_POSITIONER_NAME (-18)
- Valid output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)
- Check input parameter values: ERR_PARAMETER_OUT_OF_RANGE (-17)
- 0 < WarningFollowingError <= FatalFollowingError

Description

This function sets a new value of the warning following error for a positioner.

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- SUCCESS (0): no error

TCL

Prototype

PositionerWarningFollowingErrorSet $SocketID $FullPositionerName $WarningFollowingError

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– WarningFollowingError	double	Warning following error (units)

Output parameters

– None

Return

– Error	int	TCL error code (0=success or 1=syntax error) or function error code

C/C++

Prototype

int **PositionerWarningFollowingErrorSet** (int SocketID, char FullPositionerName[250] , double WarningFollowingError)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	char *	Positioner name
– WarningFollowingError	double	Warning following error (units)

Output parameters

– None

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **PositionerWarningFollowingErrorSet** (ByVal SocketID As Long, ByVal FullPositionerName As String, WarningFollowingError As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– WarningFollowingError	double	Warning following error (units)

Output parameters

– None

Return

– Error	long	Function error code

Matlab

Prototype

[Error] **PositionerWarningFollowingErrorSet** (int32 SocketID, cstring FullPositionerName, double WarningFollowingError)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	cstring	Positioner name
– WarningFollowingError	double	Warning following error (units)

Return

– Error	int32	Function error code

Python

Prototype

[Error] **PositionerWarningFollowingErrorSet** (integer SocketID, string FullPositionerName, double WarningFollowingError)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– FullPositionerName	string	Positioner name
– WarningFollowingError	double	Warning following error (units)

Return

– Error	int	Function error code

2.2.5 Configuration Files

Two configuration files are used by the controller: "System.ini" and "Stages.ini". These configuration files are read during boot of the controller.

1. The **system.ref** file specifies the hardware configuration and defines general parameters.
2. The **system.ini** file specifies the system configuration and defines the motion groups used.
3. The **stages.ini** file defines the stage parameters for all positioners (see "ConfigurationWizard" document to learn about the stage parameters). The stages.ini file must include at least those positioners referenced in the system.ini file, but it might also include positioners that are currently not used by the system.

- **System.ref file:**

```
[GENERAL]
HardwareType = XPS                          ; XPS, SPX or SPS
FirmwareName = MainController
ExternalModuleNames =  XPSRemoteControl
ISRBaseFrequency = 12e6                     ; Hertz
CorrectorISRPeriod = 100e-6                 ; seconds
IRQDelay = 6e-6                             ; seconds
DACUpdateDelay = 90e-6                      ; seconds
ProfileGeneratorISRRatio = 4
ServitudesISRRatio = 10
GatheringBufferSize = 1000000
DelayBeforeStartup = 0                      ; seconds
ProfilerPriorityOverrunPatch = Disabled     ; Disabled or Enabled
[AUTOTUNING]
VelocityAutoTuningParameters = 20, 10, 15, 0.5, 0.5
VoltageAutoTuningParameters = 15, 10, 15, 0.5, 0.5
AccelerationAutoTuningParameters = 10, 10, 15, 0.5, 0.5, 50, 1000, 750
AccelerationAutoScalingParameters = 10, 15, 5, 5
[FEATURES]
CompensationSystemPreFeedForwardMode = Disabled     ; Disabled or Enabled
CompensationSystemPostFeedForwardMode = Disabled    ; Disabled or Enabled
```

- **System.ini file:**

NOTE

PIDBaseFilter parameter in system.ini file is to be configured only when using the XPS-Qn Precision Platform controller.

```
[GENERAL]
BootScriptFileName =
BootScriptArguments =              ; the separator is the comma
[GROUPS]
InterlockedGroups =                ; list of groups involved in GroupInterlocked mode
SingleAxisInUse = SINGLE           ; the separator is the comma, eg:  S1, S2, ...
SingleAxisWithClampingInUse = SCLAMP ; Separator is comma, eg:  SCL1, SCL2, ...
SingleAxisThetaInUse = STHETA      ; the separator is the comma, eg:  ST1, ST2, ...
SpindleInUse = SPIN                ; the separator is the comma, eg:  SP1, SP2, ...
XYInUse = XY                       ; the separator is the comma, eg:  XY1, XY2, ...
XYZInUse = XYZ                     ; the separator is the comma, eg:  XYZ1, XYZ2, ...
TZInUse = TZ                       ; the separator is the comma, eg:  TZ1, TZ2, ...
MultipleAxesInUse = MULTI          ; the separator is the comma, eg:  M1,M2, ...
; ------------------------------------------------------------------------------
[SINGLE]
PositionerInUse =  Positioner
[SINGLE. Positioner]
PlugNumber =
InterferometerPlugNumber =         ;   If Agilent interferometer is used
StageName = MySTAGE                ;   see " stages.ini" file
; --- Time flasher
TimeFlasherBaseFrequency =         ; default value 40e6, must be between 39.5e6 and
                                   40.5e6 Hz
;--- PIDBase filter (take into account the base movements, effective only with
PIDFFAcceleration corrector) To be added only when using XPS-Q8 Precision
Platform controller
PIDBaseFilter =                    ; Enabled or Disabled
MovingMass =                       ; If PIDBaseFilter = Enabled
StaticMass =                       ; If PIDBaseFilter = Enabled
Viscosity =                        ; If PIDBaseFilter = Enabled
Stiffness =                        ; If PIDBaseFilter = Enabled
; CIE08CompensatedPCO mode
CIE08CompensatedPCOMode = Disabled                    ; Enabled or Disabled
;CIE08CompensatedPCOMaximumDataNumber = 1000000       ; Value <= 1000000
;--- If Gantry (secondary positioner)
SecondaryPositionerGantry =        ; Enabled or Disabled
SecondaryPlugNumber =              ; If SecondaryPositionerGantry = Enabled
SecondaryStageName =               ; If SecondaryPositionerGantry = Enabled
```

```
SecondaryPositionerGantryEndReferencingPosition =    ; If
                                    SecondaryPositionerGantry = Enabled
SecondaryPositionerGantryEndReferencingTolerance = ; If
                                    SecondaryPositionerGantry = Enabled
SecondaryPositionerGantryOffsetAfterInitialization =    ; If
                                    SecondaryPositionerGantry = Enabled
SecondaryPositionerGantryMotorEnableDelay = ; seconds,  if
                                    SecondaryPositionerGantry = Enabled
; ----------------------------------------------------------------------------------
[SPIN]
PositionerInUse =  Positioner
[SPIN.Positioner]
….                                  ; The same lines as in the SINGLE.Positioner
                                    section, unless gantry lines
….
; ----------------------------------------------------------------------------------
[SCLAMP]
PositionerInUse =  Positioner
;--- Clamping
ClampInputBit =             ; GPIOx.DIy (x=1,y=1..8 or x=2,y=1..6 or x=3,y=1..6 or
                            ; x=4,y=1..16) or NoFeedback (sensor simulation)
ClampInputMode =            ; NonInverted (clamped if input=1, unclamped if input=0)
                            ;  or Inverted (clamped if input=0, unclamped if input=1),
                            ;  parameter not needed if ClampInputBit=NoFeedback
ClampOutputBit =            ; GPIOx.DOy (x=1,y=1..8 or x=3,y=1..6 or x=4,y=1..16)
ClampOutputMode =           ; NonInverted (output 1 to clamp, output 0 to unclamp) or
                            ; Inverted (output 0 to clamp, output 1 to unclamp)
ClampingActivatingTime =    ; seconds
ClampingActivatingTimeout   ; seconds
ClampingReleaseTime =       ; seconds
ClampingReleaseTimeout =    ; seconds
ClampingPositionOffset =    ; units
 [SCLAMP.Positioner]
….                          ; The same lines as in the SINGLE.Positioner
                            section

….
; ----------------------------------------------------------------------------------
[STHETA]
PositionerInUse =  Positioner
; Theta correction on XY
ThetaCorrectionXYGroupName =
ThetaCorrectionLowPassCutOffFrequency =     ; Hz
; Yaw mapping
YawMappingXYGroupName =
YawMappingToThetaFileName =
```

```
YawMappingToThetaLineNumber =
YawMappingToThetaColumnNumber =
YawMappingToThetaMaxPositionError =
YawMappingToXFileName =
YawMappingToXLineNumber =
YawMappingToXColumnNumber =
YawMappingToXMaxPositionError =
YawMappingToYFileName =
YawMappingToYMaxPositionError =
YawMappingToYLineNumber =
YawMappingToYColumnNumber =
; Clamping
ClampRestType =                 ; Clamped or Unclamped
ClampActivatingTime =           ; seconds
ClampActivatingTimeOut =        ; seconds
ClampReleaseTime =              ; seconds
ClampReleaseTimeOut =           ; seconds
[STHETA.Positioner]
….                              ; The same lines as in the SINGLE.Positioner
                                section
….
; ------------------------------------------------------------------------
[XY]
PositionerInUse =  X,Y          ; the separator is the comma
InitializationAndHomeSearchSequence =  ; Together, XThenY or YThenX
; XY gantry motor force
XMotorForceBalance =            ; Enabled or Disabled
YOffsetForForceRatio =          ; If XMotorForceBalance = Enabled
PrimaryYForceRatio =            ; If XMotorForceBalance = Enabled
SecondaryYForceRatio =          ; If XMotorForceBalance = Enabled
; Mapping X
XMappingFileName =              ; XYMappingX.map
XMappingLineNumber =
XMappingColumnNumber =
XMappingMaxPositionError =
; Mapping Y
YMappingFileName =              ; XYMappingY.map
YMappingLineNumber =
YMappingColumnNumber =
YMappingMaxPositionError =
[XY. X]
….                              ; The same lines as in the SINGLE.Positioner
                                section
….
```

```
[XY. Y]
….                          ; The same lines as in the SINGLE.Positioner
                            section
….
; ------------------------------------------------------------------------------
[XYZ]
PositionerInUse =  X,Y,Z    ; The separator is the comma
InitializationAndHomeSearchSequence =  ; Together or XThenYThenZ
; Mapping X
XMappingFileName =          ; XYZMappingX.map
XMappingLineNumber =
XMappingColumnNumber =
XMappingMaxPositionError =
; Mapping Y
YMappingFileName =          ; XYZMappingY.map
YMappingLineNumber =
YMappingColumnNumber =
YMappingMaxPositionError =
; Mapping Z
ZMappingFileName =          ; XYZMappingZ.map
ZMappingLineNumber =
ZMappingColumnNumber =
ZMappingMaxPositionError =
[XYZ. X]
….                          ; The same lines as in the SINGLE.Positioner
                            section, unless gantry lines
….
[XYZ. Y]
….                          ; The same lines as in the SINGLE.Positioner
                            section, unless gantry lines
….
[XYZ. Z]
….                          ; The same lines as in the SINGLE.Positioner
                            section, unless gantry lines
….
; ------------------------------------------------------------------------------
[TZ]
PositionerInUse =  Z1,Z2,Z3 ; the separator is the comma
InitializationAndHomeSearchSequence = ; Together or OneAfterAnother
TZDecouplingGainMatrixFileName =  ; TZ_decoupling_matrix_filename.txt
MaximumZZZTargetDifference = ; Maximum difference between Z target positions
                            (units)
[TZ. Z1]
….                          ; The same lines as in the SINGLE.Positioner
                            section, unless gantry lines
```

```
….
[TZ. Z2]
….                              ; The same lines as in the SINGLE.Positioner
                                section, unless gantry lines
….
[TZ. Z3]
….                              ; The same lines as in the SINGLE.Positioner
                                section, unless gantry lines
….
; ------------------------------------------------------------------------------
[MULTI]
PositionerInUse =  M1,M2,M3,M4  ; From 1 to 8
InitializationAndHomeSearchSequence =  ; Together, OneAfterAnother or
                                OneAfterAnotherInReverseOrder
[MULTI.M1]
….                              ; The same lines as in the SINGLE.Positioner
                                section
….
[MULTI.M2]
….                              ; The same lines as in the SINGLE.Positioner
                                section
….
[MULTI.M3]
….                              ; The same lines as in the SINGLE.Positioner
                                section
….
[MULTI.M4]
….                              ; The same lines as in the SINGLE.Positioner
                                section
….
```

- **Stages.ini file:**

```
[MySTAGE]
;--- Stage
SmartStageName =            ; Smart stage reference
;--- DRIVER
DriverName =          ; XPS-DRV00 (pass through board for external driver)
; XPS-DRV00P (pass through board for external driver with PulseDir output)
; XPS-DRV01 (driver for DC servo and stepper motors)
; XPS-DRV02 (driver for 3-phase brushless or linear motors)
; XPS-DRV02P (driver for 3-phase brushless or linear motors)
; XPS-DRVMx (driver for DC motors )
; XPS-DRV03 (driver for DC motors)
; XPS-DRV03L (driver for DC motors)
; XPS-D3PD6U (high power external driver, using XPS-DRV00P pass-through board).
; XPS-DRVPx (x = 1, 2, … , driver for piezo actuator).
```

```
; If DriverName = XPS-DRV01 driver
;  If  MotorDriverInterface = AnalogVelocity
DriverPWMFrequency =
DriverErrorAmplifierGain =
DriverTachometerGain=
;  If  MotorDriverInterface = AnalogVoltage
PWMFrequency =
;  If  MotorDriverInterface = AnalogStepper
PWMFrequency =
DriverStepperWinding=
; If DriverName = XPS-DRVMx driver   (x = 1 to 5)
DriverBridgeFreeWheel =
DriverBrake =
; If DriverName = XPS-DRV02/XPS-DRV02P  driver
DriverMotorResistance =
DriverMotorInductance =
DriverCutOffFrequency =
DriverMaximumPeakCurrent =
DriverMaximumRMSCurrent =
DriverRMSIntegrationTime =
DriverThermistanceThreshold =
; If DriverName = XPS-DRV03 driver
;  If  MotorDriverInterface = AnalogAcceleration
DriverMotorResistance =
DriverMotorInductance =
DriverCurrentCutOffFrequency =
DriverMaximumPeakCurrent =
DriverMaximumRMSCurrent =
DriverRMSIntegrationTime =
DriverMaximumMotorVoltage =
;  If  MotorDriverInterface = AnalogVoltage
DriverMaximumRMSCurrent =
DriverRMSIntegrationTime =
;  If  MotorDriverInterface = AnalogVelocity
DriverMotorResistance =
DriverMotorInductance =
DriverCurrentCutOffFrequency =
DriverMaximumRMSCurrent =
DriverRMSIntegrationTime =
DriverMaximumMotorVoltage =
DriverVelocityCutOffFrequency =
DriverMotorVoltageConstant =
DriverTachoGeneratorVoltage =
DriverStageInertia =
```

```
DriverGearRatio =
; If DriverName = XPS-DRVPx (x = 1,2, …) piezo driver
; Motor driver interface type
MotorDriverInterface = AnalogPositionPiezo
; Limit sensors input plug
ServitudesType = Piezo
; Piezo driver parameters
DriverNotchFrequency =          ; Hz
DriverNotchBandwidth =          ; Hz
DriverNotchGain =               ;
DriverLowpassFrequency =        ; Hz
DriverKI =                      ;
DriverFatalFollowingError =     ; units
DriverStagePositionOffset =     ; units
DriverTravelCorrection =        ; ppm
;--- MOTOR DRIVER INTERFACE
MotorDriverInterface =          ; AnalogStepperPosition
                                ; AnalogVelocity
                                ; AnalogVoltage
                                ; AnalogAcceleration
                                ; AnalogAccelerationTZ   ;// Specific initialization for TZ group
                                ; AnalogSin60Acceleration
                                ; AnalogSin90Acceleration
                                ; AnalogSin120Acceleration
                                ; AnalogDualSin60Acceleration
                                ; AnalogDualSin90Acceleration
                                ; AnalogDualSin120Acceleration
                                ; AnalogPosition
                                ; PulseDir
                                ; PulsePulse
                                ; AnalogSin60AccelerationLMI
                                ; AnalogSin90AccelerationLMI
                                ; AnalogSin120AccelerationLMI
                                ; AnalogPositionPiezo
; If MotorDriverInterface = AnalogVelocity
ScalingVelocity =               ; unit/s
VelocityLimit =                 ; unit/s
; If MotorDriverInterface = AnalogAcceleration
ScalingAcceleration =           ; unit/s²
AccelerationLimit =             ; unit/s²
; If MotorDriverInterface = AnalogVoltage
MaximumCurrent =                ; amps
VoltageLimit =                  ; volts
```

```
; If MotorDriverInterface = AnalogPosition
MinimumTargetPositionVoltage = ; volts
MaximumTargetPositionVoltage =          ; volts
; If MotorDriverInterface = AnalogStepperPosition
DisplacementPerFullStep =          ; units
ScalingCurrent =                   ; amps for 10 volts
PeakCurrentPerPhase =              ; amps
StandbyPeakCurrentPerPhase =       ; amps
BaseVelocity =                     ; units/s
; If MotorDriverInterface = PulseDir or PulsePulse
DigitalStepperPulseLogic =         ; Positive or Negative
DigitalStepperDirectionLogic =     ; Positive or Negative
DisplacementPerFullStep =          ; units
BaseVelocity =                     ; units/s
; If MotorDriverInterface = AnalogSin …
ScalingAcceleration =              ; unit/s²
AccelerationLimit =                ; unit/s²
MagneticTrackPeriod =              ; units
InitializationAccelerationLevel =  ; percent (LMI)
InitializationCycleDuration =      ; seconds (LMI)
; If MotorDriverInterface = AnalogDualSin …
ScalingAcceleration =              ; unit/s²
AccelerationLimit =                ; unit/s²
MagneticTrackPeriod =              ; units
InitializationAccelerationLevel =  ; percent
InitializationCycleDuration =      ; seconds
FirstMotorForceBalance =
SecondMotorForceBalance =
; If MotorDriverInterface = AnalogPositionPiezo
                                   ; nothing
;--- Encoder
EncoderType =                      ; AquadB
                                   ; AnalogInterpolated
                                   ; Interferometer      ; Agilent interferometer
LinearEncoderCorrection =          ; ppm
; If EncoderType = AquadB
EncoderIndexOffset =                ; default value 0
EncoderResolution =                ; units
; If EncoderType = AnalogInterpolated
EncoderIndexOffset =                ; default value 0
EncoderHardInterpolatorErrorCheck =      ; Enabled (default value) or Disabled
EncoderZMPlug =                    ; Driver
                                   ; Encoder
EncoderResolution =                ; units
```

```
EncoderInterpolationFactor =
EncoderScalePitch =                ; units
EncoderADC1Offset =                ; volts
EncoderADC2Offset =                ; volts
EncoderPhaseCompensation =         ; deg
EncoderDifferentialGain =
; If EncoderType = Interferometer
InterferometerCountDirection =     ; Normal or Reverse
InterferometerResolution =         ; units
EncoderResolution =                ; units
;--- Backlash
Backlash =                         ; unit  (0 = not activated)
;--- Positioner Mapping
PositionerMappingFileName =
; If PositionerMappingFileName is defined then the mapping is enabled and must be
                                   configured:
PositionerMappingLineNumber =
PositionerMappingMaxPositionError =
;--- Travels
MinimumTargetPosition =            ; units
HomePreset =                       ; units
MaximumTargetPosition =            ; units
;--- Profiler
MaximumVelocity =                  ; units/second
MaximumAcceleration =              ; units/second²
EmergencyDecelerationMultiplier =
MinimumJerkTime =                  ; seconds
MaximumJerkTime =                  ; seconds
TrackingCutOffFrequency =          ; Hz
;--- HOME
HomeSearchSequenceType =           ; MechanicalZeroAndIndexHomeSearch
                                   ; MechanicalZeroHomeSearch
                                   ; MinusEndOfRunAndIndexHomeSearch
                                   ; MinusEndOfRunHomeSearch
                                   ; PlusEndOfRunHomeSearch
                                   ; IndexHomeSearch
                                   ; CurrentPositionAsHome
HomeSearchMaximumVelocity =        ; units/second
HomeSearchMaximumAcceleration =    ; units/second²
HomeSearchTimeout =                ; seconds
HomingSensorOffset =               ; units
;--- CORRECTOR
CorrectorType =                    ; PIDFFAcceleration      =>
                                   MotorDriverInterface « Acceleration »
```

```
                                ; SR1Acceleration   => MotorDriverInterface
                                « Acceleration »
                                ; PIDFFVelocity     => MotorDriverInterface
                                « Velocity »
                                ; PIDDualFFVoltage        =>
                                MotorDriverInterface « Voltage »
                                ; PIPosition   => MotorDriverInterface « Position »
                                ; NoEncoderPosition       =>
                                MotorDriverInterface « Position »
; If   CorrectorType is PIDFFAcceleration
KP =                            ; 1/seconds²
KI =                            ; 1/seconds²
KD =                            ; 1/seconds²
KS =
IntegrationTime =               ; seconds
DerivativeFilterCutOffFrequency =       ; Hertz
GKP =
GKD =
GKI =
KForm =                         ; units
KFeedForwardAcceleration =      ; units/second²
KFeedForwardJerk =              ; units/second³
ClosedLoopStatus =              ; Opened or Closed
FatalFollowingError =           ; units
DeadBandThreshold =             ; units
; If   CorrectorType is SR1Acceleration
KP =                            ; 1/seconds²
KI =                            ; 1/seconds³
KV =                            ; 1/seconds
ObserverFrequency =             ; Hz
CompensationGainVelocity =      ; sec
CompensationGainAcceleration =  ; sec2
CompensationGainJerk =          ;  sec3
ClosedLoopStatus =              ; Opened or Closed
FatalFollowingError =           ; units
; If CorrectorType is PIDFFVelocity
KP =                            ; 1/seconds
KI =                            ; 1/seconds²
KD =
KS =
IntegrationTime =               ; seconds
DerivativeFilterCutOffFrequency =       ; Hertz
GKP =
GKD =
GKI =
```

```
KForm =                                 ; units
KFeedForwardVelocity =
ClosedLoopStatus =                      ; Opened or Closed
FatalFollowingError =                   ; units
DeadBandThreshold =                     ; units
; If CorrectorType is PIDDualFFVoltage
KP =                                    ; volts/units
KI =                                    ; volts/units/seconds
KD =                                    ; volts * seconds/units
KS =
IntegrationTime =                       ; seconds
DerivativeFilterCutOffFrequency =       ; Hertz
GKP =
GKD =
GKI =
KForm =                                 ; units
KFeedForwardAcceleration =              ; volts/(units/seconds²)
KFeedForwardVelocity =                  ; volts/(units/seconds)
KFeedForwardVelocityOpenLoop =          ;
Friction =                              ; volts
ClosedLoopStatus =                      ; Opened or Closed
FatalFollowingError =                   ; units
DeadBandThreshold =                     ; units
; If CorrectorType is PIPosition
KP =
KI =                                    ; 1/seconds
KD =                                    ; seconds
KS =
IntegrationTime =                       ; seconds
DerivativeFilterCutOffFrequency =       ; Hertz
GKP =
GKD =
GKI =
KForm =                                 ; units
ClosedLoopStatus =                      ; Opened or Closed
FatalFollowingError =                   ; units
DeadBandThreshold =                     ; units
;--- NOTCH FILTER
NotchFrequency1 =                       ; Hertz  (0 = not activated)
NotchBandwith1 =                        ; Hertz
NotchGain1 =
NotchFrequency2 =                       ; Hertz (0 = not activated)
NotchBandwith2 =                        ; Hertz
NotchGain2 =
```

```
;--- GATHERING FILTERS
CurrentVelocityCutOffFrequency =        ; Hertz
CurrentAccelerationCutOffFrequency =    ; Hertz
;--- MOTION DONE
MotionDoneMode =                        ; Theoretical
                                        ; VelocityAndPositionWindow
; If MotionDoneMode = VelocityAndPositionWindow
MotionDonePositionThreshold =           ; units
MotionDoneVelocityThreshold =           ; units/second
MotionDoneCheckingTime =                ; seconds
MotionDoneMeanPeriod =                  ; seconds
MotionDoneTimeout =                     ; seconds
;--- SERVITUDES
ServitudesType =                        ; StandardEORDriverPlug
                                        ; StandardLimitAndHomeEncoderPlug ;
                                        OldName: StandardEOREncoderPlug
                                        ; StandardLimitAndLimitEncoderPlug
                                        ; Spindle
                                        ; Piezo
;--- System compensation pre-feedforward filters
;--- (if CompensationSystemPreFeedForwardMode = Enabled)
CompensationSpatialPeriodicNotchsStep1 = 42                 ;   units
CompensationSpatialPeriodicNotchsBandwidth1 = 1             ;   Hz
CompensationSpatialPeriodicNotchsGain1 = 0.5                ;
CompensationSpatialPeriodicNotchsStep2 = 63                 ;   units
CompensationSpatialPeriodicNotchsBandwidth2 = 2             ;   Hz
CompensationSpatialPeriodicNotchsGain2 = 0.4                ;
CompensationSpatialPeriodicNotchsStep3 = 84                 ;   units
CompensationSpatialPeriodicNotchsBandwidth3 = 3             ;   Hz
CompensationSpatialPeriodicNotchsGain3 = 0.3                ;
CompensationFrequencyNotchsFrequency1 = 1001                ;   Hz
CompensationFrequencyNotchsBandwidth1 = 11                  ;   Hz
CompensationFrequencyNotchsGain1 = 0.9                      ;
CompensationFrequencyNotchsFrequency2 = 1101                ;   Hz
CompensationFrequencyNotchsBandwidth2 = 12                  ;   Hz
CompensationFrequencyNotchsGain2 = 0.8                      ;
CompensationFrequencyNotchsFrequency3 = 1201                ;   Hz
CompensationFrequencyNotchsBandwidth3 = 13                  ;   Hz
CompensationFrequencyNotchsGain3 = 0.7                      ;
;--- System Compensation post-feedforward filters
;--- (if CompensationSystemPostFeedForwardMode = Enabled)
;--- CompensationNotch mode filter #1
CompensationNotchModeFr1 = 101                              ;   Hz
CompensationNotchModeFa1 = 102                              ;   Hz
```

```
CompensationNotchModeZr1 = 0.101                    ;
CompensationNotchModeZa1 = 0.102                    ;
;--- CompensationNotch mode filter #2
CompensationNotchModeFr2 =  201                     ;   Hz
CompensationNotchModeFa2 = 202                      ;   Hz
CompensationNotchModeZr2 = 0.201                    ;
CompensationNotchModeZa2 = 0.202                    ;
;--- CompensationPhase correction filter #1
CompensationPhaseCorrectionFn1 = 301                ;   Hz
CompensationPhaseCorrectionFd1 = 302                ;   Hz
CompensationPhaseCorrectionGain1 = 0.302            ;
;--- CompensationPhase correction filter #2
CompensationPhaseCorrectionFn2 =  401               ;   Hz
CompensationPhaseCorrectionFd2 = 402                ;   Hz
CompensationPhaseCorrectionGain2 = 0.402            ;
;--- CompensationLowPassFilterCutOffFrequency filter
CompensationLowPassFilterCutOffFrequency = 501      ;   Hz
```

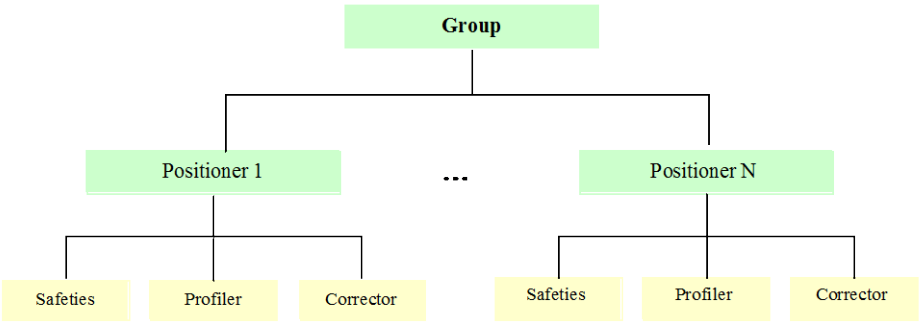
2.3 Group

2.3.1 Description

The "Group" objects are used to define one or several "positioners" in the same motion group. The available motion groups are defined in the section [GROUPS] in the system.ini file and the group types are:

- **SingleAxis** (1 positioner)/"**Gantry" SingleAxis** (2 positioners)
- **SingleAxisWithClamping** (1 positioner)
- **SingleAxisTheta** (1 positioner)
- **Spindle** (1 positioner)
- **XY** (2 positioners)/"**Gantry" XY** (3 or 4 positioners)
- **XYZ** (3 positioners)
- **MultipleAxes/"Gantry" MultipleAxes** (1 to 8 positioners)
- **TZ** (3 positioners)

2.3.2 Object Structure



A motion "**Group**" is built in relation to a **group type** (SingleAxis, SingleAxisWithClamping, SingleAxisTheta, Spindle, XY, XYZ, TZ or MultipleAxes).

A group is defined by a **group name**.

To define a new group see §2.2.4.95 (configuration file).

NOTE

The maximum number of positioners in the same group is limited to 8.

2.3.3 Function Description

2.3.3.1 GroupAccelerationSetpointGet

Name

GroupAccelerationSetpointGet – Returns the setpoint acceleration for one or all positioners of the selected group.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid object type (group or positioner): ERR_WRONG_OBJECT_TYPE (-8)
- Valid group name: ERR_GROUP_NAME (-19)
- Valid positioner name: ERR_POSITIONER_NAME (-18)
- Valid output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

Returns the setpoint acceleration for one or all positioners of the selected group.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

GroupAccelerationSetpointGet SocketID GroupName SetpointAcceleration …

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Group name (maximum size = 250)

Output parameters

–	SetpointAcceleration	floating point	Setpoint acceleration (must be repeated for each positioner of group)

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupAccelerationSetpointGet** (int SocketID, char *GroupName, int NbPositioners, double * SetpointAcceleration)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	char *	Group name
–	NbPositioners	int	Number of positioners in the selected group

Output parameters

–	SetpointAcceleration	double *	Setpoint Acceleration array

Return

– Function error code

Visual Basic

Prototype

Long **GroupAccelerationSetpointGet** (ByVal SocketID As Long, ByVal GroupName As String, ByVal NbPositioners As Long, SetpointAcceleration As Double)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Group name
–	NbPositioners	long	Number of positioners in the selected group

Output parameters

–	SetpointAcceleration	double	Setpoint Acceleration array

Return

– Function error code

Matlab

Prototype

[Error, SetpointAcceleration] **GroupAccelerationSetpointGet** (int32 SocketID, cstring GroupName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	Group name

Return

– Error	int32	Function error code
– SetpointAcceleration	doubletPtr	Setpoint Acceleration array

Python

Prototype

[Error, SetpointAcceleration] **GroupAccelerationSetpointGet** (integer SocketID, string GroupName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	Group name

Return

– Error	int	Function error code
– SetpointAcceleration	doubletPtr	Setpoint Acceleration array

2.3.3.2 GroupAnalogTrackingModeDisable

Name

GroupAnalogTrackingModeDisable - Exits the analog tracking mode.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid group name: ERR_GROUP_NAME (-19)
- Group status must be "ANALOG TRACKING": ERR_NOT_ALLOWED_ACTION (-22)

Description

Disables the analog tracking mode. The group exits the "ANALOG TRACKING" state returning to the "READY" state. If the group state is not "ANALOG TRACKING", ERR_NOT_ALLOWED_ACTION (-22) is returned.

NOTE

The tracking mode interprets ADC value as a position command or as a velocity command.

To enable the analog tracking mode use the "GroupAnalogTrackingModeEnable" function.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

GroupAnalogTrackingModeDisable SocketID GroupName

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	SingleAxis group name (maximum size = 250)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupAnalogTrackingModeDisable** (int SocketID, char *GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	char *	SingleAxis group name

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **GroupAnalogTrackingModeDisable** (ByVal SocketID As Long, ByVal GroupName As String)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	SingleAxis group name

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **GroupAnalogTrackingModeDisable** (int32 SocketID, cstring GroupName)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	cstring	SingleAxis group name

Return

– Function error code

Python

Prototype

integer **GroupAnalogTrackingModeDisable** (integer SocketID, string GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	string	SingleAxis group name

Return

– Function error code

2.3.3.3 GroupAnalogTrackingModeEnable

Name

GroupAnalogTrackingModeEnable - Enables the analog tracking mode.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Group status must be "READY": ERR_NOT_ALLOWED_ACTION (-22)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid group name: ERR_GROUP_NAME (-19)
- Valid tracking type ("Position" or "Velocity"): ERR_WRONG_OBJECT_TYPE (-8)
- Configured tracking: ERR_NOT_ALLOWED_ACTION (-22)

Description

Enables the analog tracking mode. To use this function, the group must be in READY state and tracking must be configured before, else an error ERR_NOT_ALLOWED_ACTION (-22) is returned.

Once the tracking mode is enabled, the group status must be "ANALOG TRACKING" (48 is the code for Analog tracking state due to a TrackingEnable command).

"Position" analog tracking

In case of "Position" tracking type, the analog input is interpreted as a position command. The parameters must be set by the "AnalogTrackingPositionParametersSet" function and can be read by the "AnalogTrackingPositionParametersGet" function.

"Velocity" analog tracking

In case of "Velocity" tracking type, the analog input is interpreted as a velocity command. The parameters must be set by the "AnalogTrackingVelocityParametersSet" function and can be read by the "AnalogTrackingVelocityParametersGet" function.

NOTE

To disable the analog tracking mode use the "GroupAnalogTrackingModeDisable" function.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_OBJECT_TYPE (-8)
- SUCCESS (0): no error

TCL

Prototype

GroupAnalogTrackingModeEnable SocketID GroupName Type

Input parameters

- SocketID int Socket identifier gets by the "TCP_ConnectToServer" function
- GroupName string SingleAxis group name (maximum size = 250)
- TypeString Tracking type ("Position" or "Velocity")

Output parameters

- None

Return

- TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupAnalogTrackingModeEnable** (int SocketID, char *GroupName, char *Type)

Input parameters

SocketID int Socket identifier gets by the "TCP_ConnectToServer"function

GroupName char * SingleAxis group name

- Type char * Tracking type ("Position" or "Velocity")

Output parameters

- None

Return

- Function error code

Visual Basic

Prototype

Long **GroupAnalogTrackingModeEnable** (ByVal SocketID As Long, ByVal GroupName As String, ByVal Type As String)

Input parameters

- SocketID long Socket identifier gets by the "TCP_ConnectToServer" function
- GroupName string SingleAxis group name
- TypeString Tracking type ("Position" or "Velocity")

Output parameters

- None

Return

- Function error code

Matlab

Prototype

int32 **GroupAnalogTrackingModeEnable** (int32 SocketID, cstring GroupName, cstring Type)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	SingleAxis group name
– Type	cstring	Tracking type ("Position" or "Velocity")

Return

– Function error code

Python

Prototype

integer **GroupAnalogTrackingModeEnable** (integer SocketID, string GroupName, string Type)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	SingleAxis group name
– TypeString	Tracking type ("Position" or "Velocity")	

Return

– Function error code

2.3.3.4 GroupCorrectorOutputGet

Name

GroupCorrectorOutputGet – Returns corrector output for one or all positioners of the selected group.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Valid output parameter: ERR_WRONG_TYPE_DOUBLE (-14)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)

Description

Returns corrector output for one or all positioners of the selected group.

The input parameter "group name" can be a positioner name.

For a group, this function returns the corrector output for each positioner from the selected group.

For a positioner, this function returns only the corrector output associated with the selected positioner.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

GroupCorrectorOutputGet SocketID GroupName CorrectorOutput

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Group name or Positioner name (maximum size = 250)

Output parameters

–	CorrectorOutput	floating point	Corrector output

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupCorrectorOutputGet** (int SocketID, char *GroupName, int NbPositioners, double * CorrectorOutput)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	Group name or Positioner name
– NbPositioners	int	Number of positioners in the selected group (1 if positioner)

Output parameters

– CorrectorOutput	double *	Corrector output array

Return

– Function error code

Visual Basic

Prototype

Long **GroupCorrectorOutputGet** (ByVal SocketID As Long, ByVal GroupName As String, ByVal NbPositioners As Long, CorrectorOutput As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	Group name or Positioner name
– NbPositioners	long	Number of positioners in the selected group (1 if positioner)

Output parameters

– CorrectorOutput	doubletPtr	Corrector output array

Return

– Function error code

Matlab

Prototype

[Error, CorrectorOutput] **GroupCorrectorOutputGet** (int32 SocketID, cstring GroupName, int32 NbPositioners)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	Group name or Positioner name
– NbPositioners	int32	Number of positioners in the selected group (1 if positioner)

Return

– Error	int32	Function error code
– CorrectorOutput	double	Corrector output array

Python

Prototype

[Error, CorrectorOutput] **GroupCorrectorOutputGet** (integer SocketID, string GroupName, integer NbPositioners)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	Group name or Positioner name
– NbPositioners	int	Number of positioners in the selected group (1 if positioner)

Return

– Error	int	Function error code
– CorrectorOutput	double	Corrector output array

2.3.3.5 GroupCurrentFollowingErrorGet

Name

GroupCurrentFollowingErrorGet – Returns the current following error for one or all positioners of the selected group.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid object type (group or positioner): ERR_WRONG_OBJECT_TYPE (-8)
- Valid group name: ERR_GROUP_NAME (-19)
- Valid positioner name: ERR_POSITIONER_NAME (-18)
- Valid output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

Returns the current following error for one or all positioners of the selected group.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

GroupCurrentFollowingErrorGet SocketID GroupName CurrentFollowing error …

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Group name (maximum size = 250)

Output parameters

–	CurrentFollowingError	floating point	Current following error (must be repeated for each positioner of group)

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupCurrentFollowingErrorGet** (int SocketID, char *GroupName, int NbPositioners, double * CurrentFollowingError)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	Group name
– NbPositioners	int	Number of positioners in the selected group

Output parameters

– CurrentFollowingError	double *	Current following error array

Return

– Function error code

Visual Basic

Prototype

Long **GroupCurrentFollowingErrorGet** (ByVal SocketID As Long, ByVal GroupName As String, ByVal NbPositioners As Long, CurrentFollowingError As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	Group name
– NbPositioners	long	Number of positioners in the selected group

Output parameters

– CurrentFollowingError	double	Current following error array

Return

– Function error code

Matlab

Prototype

[Error, CurrentFollowingError] **GroupCurrentFollowingErrorGet** (int32 SocketID, cstring GroupName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	Group name

Return

– Error	int32	Function error code
– CurrentFollowingError	doubletPtr	Current following error array

Python

Prototype

[Error, CurrentFollowingError] **GroupCurrentFollowingErrorGet** (integer SocketID, string GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	string	Group name

Return

–	Error	int	Function error code
–	CurrentFollowingError	doubletPtr	Current following error array

2.3.3.6 GroupInitialize

Name

GroupInitialize - Initializes the motor and activates the servo loop of the selected group.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Group status must be "NOTINIT": ERR_NOT_ALLOWED_ACTION (-22)
- Actor must be a group: ERR_WRONG_OBJECT_TYPE (-8), ERR_POSITIONER_NAME (-18)
- Valid group name: ERR_GROUP_NAME (-19)
- Check state of physical ends of run: ERR_BOTH_ENDS_OF_RUN_ACTIVATED (-113).

Description

The selected group must be in not initialized "NOTINIT" state, else ERR_NOT_ALLOWED_ACTION (-22) is returned.

This function begins to check the positioner error. If an error is detected, the hardware status register is reset (motor on) and the positioner error is cleared before checking it again. If a positioner error is always present, the motor is turned off, ERR_POSITIONER_ERROR (-5) is returned and the group becomes "NOTINIT".

If no positioner error then the group status becomes "MOTOR_INIT". The master-slave error is cleared, the encoder is reset (update encoder position) and the user travel limits are checked. If a travel limit error is detected then the motor is turned off, the error ERR_TRAVEL_LIMITS (-35) is returned and the group becomes "NOTINIT".

Moreover, the function checks the state of the physical ends of run. If both physical ends of run are activated, then the motor is turned off, the error ERR_BOTH_ENDS_OF_RUN_ACTIVATED (-113) is returned and the group becomes "NOTINIT".

If no error, the motor is initialized in case of "AnalogSinAcc" or "AnalogDualSinAcc". The error ERR_MOTOR_INITIALIZATION_ERROR (-50) is returned if the initialization failed and the group becomes "NOTINIT".

If successful, the positions are reset, the servo loop is activated and the motor is on. The group is now "NOT REFERENCED".

NOTES

In Master-Slave mode, after an emergency stop, the master group and the slave group are in "NOTINIT" state. To restart a master-slave relation the slave group(s) must be reinitialized before the master group.

For the XPS-Qn Precision Platform the controller checks and sets its external drivers not at boot-up, but at the beginning of the GroupInitialize() call.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_ERROR (-5)
- ERR_POSITIONER_NAME (-18)
- ERR_MOTOR_INITIALIZATION_ERROR (-50)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_TRAVEL_LIMITS (-35)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_BOTH_ENDS_OF_RUN_ACTIVATED (-113)
- SUCCESS (0): no error

TCL

Prototype

GroupInitialize SocketID GroupName

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	Group name (maximum size = 250)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupInitialize** (int SocketID, char *GroupName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	Goup name

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **GroupInitialize** (ByVal SocketID As Long, ByVal GroupName As String)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Group name

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **GroupInitialize** (int32 SocketID, cstring GroupName)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	cstring	Group name

Return

– Function error code

Python

Prototype

integer **GroupInitialize** (integer SocketID, string GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	string	Group name

Return

– Function error code

2.3.3.7 GroupInitializeNoEncoderReset

Name

GroupInitializeNoEncoderReset - Initializes the motor without reset encoder and activates the servo loop of the selected group.

Input tests

- Configuration files reading: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Group status must be "NOTINIT": ERR_NOT_ALLOWED_ACTION (-22)
- Actor must be a group: ERR_WRONG_OBJECT_TYPE (-8), ERR_POSITIONER_NAME (-18)
- Valid group name: ERR_GROUP_NAME (-19)
- Check state of physical ends of run: ERR_BOTH_ENDS_OF_RUN_ACTIVATED (-113).

Description

The selected group must be in "NOTINIT" state, else ERR_NOT_ALLOWED_ACTION (-22) is returned.

This function begins to check the positioner error. If an error is detected, the hardware status register is reset (motor on) and the positioner error is cleared before checking it again. If a positioner error is always present, the motor is turned off, ERR_POSITIONER_ERROR (-5) is returned and the group becomes "NOTINIT".

If no positioner error then the group status becomes "MOTOR_INIT". The master-slave error is cleared, the encoder is reset (update encoder position) and the user travel limits are checked. If a travel limit error is detected then the motor is turned off, the error ERR_TRAVEL_LIMITS (-35) is returned and the group becomes "NOTINIT".

Moreover, the function checks the state of the physical ends of run. If both physical ends of run are activated, then the motor is turned off, the error ERR_BOTH_ENDS_OF_RUN_ACTIVATED (-113) is returned and the group becomes "NOTINIT".

If no error, the motor is initialized in case of "AnalogSinAcc" or "AnalogDualSinAcc". The error ERR_MOTOR_INITIALIZATION_ERROR (-50) is returned if the initialization failed and the group becomes "NOTINIT".

If successful, the positions are no reset, the servo loop is activated and the motor is on. The group is now "NOT REFERENCED".

NOTES

In Master-Slave mode, after an emergency stop, the master group and the slave group are in "NOTINIT" state. To restart a master-slave relation the slave group(s) must be reinitialized before the master group.

For the XPS-Qn Precision Platform the controller checks and sets its external drivers not at boot-up, but at the beginning of the GroupInitialize() call.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_ERROR (-5)
- ERR_POSITIONER_NAME (-18)
- ERR_MOTOR_INITIALIZATION_ERROR (-50)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_TRAVEL_LIMITS (-35)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_BOTH_ENDS_OF_RUN_ACTIVATED (-113)
- SUCCESS (0): no error

TCL

Prototype

GroupInitializeNoEncoderReset SocketID GroupName

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Group name (maximum size = 250)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupInitializeNoEncoderReset GroupInitialize** (int SocketID, char *GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	char *	Goup name

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **GroupInitializeNoEncoderReset GroupInitialize** (ByVal SocketID As Long, ByVal GroupName As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	Group name

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **GroupInitializeNoEncoderReset GroupInitialize** (int32 SocketID, cstring GroupName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	Group name

Return

– Function error code

Python

Prototype

integer **GroupInitializeNoEncoderReset GroupInitialize** (integer SocketID, string GroupName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	Group name

Return

– Function error code

2.3.3.8 GroupInitializeWithEncoderCalibration

Name

GroupInitializeWithEncoderCalibration - Initializes motor, calibrates encoder and activates servo loop.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Group status must be "NOTINIT": ERR_NOT_ALLOWED_ACTION (-22)
- Actor must be a group: ERR_WRONG_OBJECT_TYPE (-8), ERR_POSITIONER_NAME (-18)
- Valid group name: ERR_GROUP_NAME (-19)
- Check state of physical ends of run: ERR_BOTH_ENDS_OF_RUN_ACTIVATED (-113).

Description

If the selected group is not in "NOTINIT" state, then the "ERR_NOT_ALLOWED_ACTION (-22)" is returned by this function.

Initializes the motor, calibrates the encoder and activates the servo loop of each positioner of the selected group. To get the calibration results for each positioner, use the "PositionerEncoderCalibrationParametersGet" function.

This function checks the positioner error. If an error is detected, the hardware status register is reset (motor on) and the positioner error is cleared before checking it again. If a positioner error is always present, the motor is turned off, ERR_POSITIONER_ERROR (-5) is returned and the group becomes "NOTINIT".

If no positioner error then the group status becomes "MOTOR_INIT". The master-slave error is cleared, the encoder is reset (update encoder position) and the user travel limits are checked. If a travel limit error is detected then the motor is turned off, the error ERR_TRAVEL_LIMITS (-35) is returned and the group becomes "NOTINIT".

Moreover, the function checks the state of the physical ends of run. If both physical ends of run are activated, then the motor is turned off, the error ERR_BOTH_ENDS_OF_RUN_ACTIVATED (-113) is returned and the group becomes "NOTINIT".

If no error, the motor is initialized in case of "AnalogSinAcc" or "AnalogDualSinAcc". The ERR_MOTOR_INITIALIZATION_ERROR (-50) is returned if the initialization failed and the group becomes "NOTINIT".

After the group initialization the group status is "MOTOR_INIT", next the encoder undergoes calibration and the group status becomes "ENCODER_CALIBRATING". If a following error occurs during calibration, ERR_FOLLOWING_ERROR (-25) is returned and the group becomes "NOTINIT".

If successful, the motor is initialized, the encoder is calibrated and the servo loop is activated. The group is now "NOT REFERENCED".

NOTE

In Master-Slave mode, after an emergency stop, the master group and the slave group are in "NOTINIT" status. To restart a master-slave relation the slave group(s) must be reinitialized before the master group.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_FOLLOWING_ERROR (-25)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_MOTOR_INITIALIZATION_ERROR (-50)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_POSITIONER_ERROR (-5)
- ERR_POSITIONER_NAME (-18)
- ERR_TRAVEL_LIMITS (-35)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_BOTH_ENDS_OF_RUN_ACTIVATED (-113)
- SUCCESS (0): no error

TCL

Prototype

GroupInitializeWithEncoderCalibration SocketID GroupName

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Group name (maximum size = 250)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupInitializeWithEncoderCalibration** (int SocketID, char *GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	char *	Goup name

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **GroupInitializeWithEncoderCalibration** (ByVal SocketID As Long, ByVal GroupName As String)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Group name

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **GroupInitializeWithEncoderCalibration** (int32 SocketID, cstring GroupName)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	cstring	Group name

Return

– Function error code

Python

Prototype

integer **GroupInitializeWithEncoderCalibration** (integer SocketID, string GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	string	Group name

Return

– Function error code

2.3.3.9 GroupHomeSearch

Name

GroupHomeSearch - Initiates a home search.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Group status must be "Not referenced": ERR_NOT_ALLOWED_ACTION (-22)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- The actor must be a valid group name: ERR_GROUP_NAME (-19)

Description

This function initiates a home search for each positioner of the selected group.

The group must be initialized and the group must be in "NOT REFERENCED" state else this function returns the ERR_NOT_ALLOWED_ACTION (-22) error. If no error then the group status becomes "HOMING".

The home search can fail due to:

a following error: ERR_FOLLOWING_ERROR (-25)

a ZM detection error: ERR_GROUP_HOME_SEARCH_ZM_ERROR (-49)

a motion done time out when a dynamic error of the positioner is detected during one of the moves during the home search process: ERR_GROUP_MOTION_DONE_TIMEOUT (-33)

a home search timeout when the complete (and complex) home search procedure was not executed in the allowed time: ERR_HOME_SEARCH_TIMEOUT (-28)

For all these errors, the group returns to the "NOTINIT" state.

After the home search sequence, each positioner error is checked. If an error is detected, the hardware status register is reset (motor on) and the positioner error is cleared before checking it again. If a positioner error is always present, ERR_TRAVEL_LIMITS (-35) is returned and the group becomes "NOTINIT".

Once the home search is successful, the group is in "READY" state.

NOTE

The home search routine for each positioner is defined in the *"stages.ini"* file by the "HomeSearchSequenceType" key.

The home search time out is defined in the *"stages.ini"* file by the "HomeSearchTimeOut" key.

The home search sequence is defined in the *"system.ini"* file by the "InitializationAndHomeSearchSequence" key for each group with several positioners.

XY group

The home search sequence can be "Together", "XthenY" or "YthenX" in a standard XY configuration.

If the XY group is "Gantry" (dual positioner on X or on Y axis) only "XthenY" or "YthenX" are allowed.

XYZ group

The home search sequence can be "Together" or "XthenYthenZ".

MultipleAxes group

The home search sequence can be "Together", "OneAfterAnother" or "OneAfterAnotherInReverseOrder".

If the MultipleAxes group has at least one "Gantry" positioner (dual positioner on one axis or some axes) only "OneAfterAnother" or "OneAfterAnotherInReverseOrder" is allowed.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_FOLLOWING_ERROR (-25)
- ERR_GROUP_HOME_SEARCH_TIMEOUT (-28)
- ERR_GROUP_HOME_SEARCH_ZM_ERROR (-49)
- ERR_GROUP_MOTION_DONE_TIMEOUT (-33)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_TRAVEL_LIMITS (-35)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

GroupHomeSearch SocketID GroupName

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Group name (maximum size = 250)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupHomeSearch** (int SocketID, char *GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	char *	Group name

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **GroupHomeSearch** (ByVal SocketID As Long, ByVal GroupName As String)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Group name

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **GroupHomeSearch** (int32 SocketID, cstring GroupName)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	cstring	Group name

Return

– Function error code

Python

Prototype

integer **GroupHomeSearch** (integer SocketID, string GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	string	Group name

Return

– Function error code

2.3.3.10 GroupHomeSearchAndRelativeMove

Name

GroupHomeSearchAndRelativeMove - Initiates a home search followed by a relative move.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Group status must be "Not referenced": ERR_NOT_ALLOWED_ACTION (-22)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid group name: ERR_GROUP_NAME (-19)
- Valid "Displacement" parameter: ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function initiates a home search followed by a relative move at the end of the home search.

The group must be initialized and the group must be in "NOT REFERENCED" state else this function returns the ERR_NOT_ALLOWED_ACTION (-22) error. If no error then the group status becomes "HOMING".

The home search sequence can fail due to:

a following error: ERR_FOLLOWING_ERROR (-25)

a ZM detection error: ERR_GROUP_HOME_SEARCH_ZM_ERROR (-49)

a home search time out: ERR_GROUP_MOTION_DONE_TIMEOUT (-33)

For all these errors, the group returns to the "NOTINIT" state.

Once the home search is completed, a relative move is executed. After this sequence without error, each positioner is checked for error. If an error is detected, the hardware status register is reset (motor on) and the positioner error is cleared before checking it again. If a positioner error is always present, ERR_TRAVEL_LIMITS (-35) is returned and the group becomes "NOTINIT".

If the home search is successful, the group will be in "READY" state.

NOTE

The home search routine for each positioner is defined in the *stages.ini* file by the "HomeSearchSequenceType" key.

The home search time out is defined in the *stages.ini* file by the "HomeSearchTimeOut" key.

The home search sequence is defined in the *system.ini* file by the "InitializationAndHomeSearchSequence" key for each group with several positioners:

XY group

The home search sequence can be "Together", "XthenY" or "YthenX" if the XY group is standard configuration. If the XY group is Gantry (dual positioner on X or on Y axis) only the "XthenY" or "YthenX" are allowed.

XYZ group

The home search sequence can be "Together" or "XthenYthenZ".

MultipleAxes group

The home search sequence can be "Together", "OneAfterAnother" or "OneAfterAnotherInReverseOrder".

If the MultipleAxes group has at least one "Gantry" positioner (dual positioner on one axis or some axes), only "OneAfterAnother" or "OneAfterAnotherInReverseOrder" are allowed.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_FOLLOWING_ERROR (-25)
- ERR_GROUP_NAME (-19)
- ERR_GROUP_HOME_SEARCH_TIMEOUT (-28)
- ERR_GROUP_HOME_SEARCH_ZM_ERROR (-49)
- ERR_GROUP_MOTION_DONE_TIMEOUT (-33)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_TRAVEL_LIMITS (-35)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

GroupHomeSearchAndRelativeMove SocketID GroupName Displacement

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Group name (maximum size = 250)
–	Displacement	floating point	Relative displacement (must be repeated for each positioner of the selected group)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupHomeSearchAndRelativeMove** (int SocketID, char *GroupName, int NbPositioners, double * Displacement)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	char *	Group name
–	NbPositioners	int	Number of positioners in the selected group
–	Displacement	double *	Relative displacement array

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **GroupHomeSearchAndRelativeMove** (ByVal SocketID As Long, ByVal GroupName As String, ByVal NbPositioners As Long, Displacement As Double)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	SingleAxis group name
–	NbPositioners	long	Number of positioners in the selected group
–	Displacement	double	Relative displacement array

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **GroupHomeSearchAndRelativeMove** (int32 SocketID, cstring GroupName, doublePtr Displacement)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	SingleAxis group name
– Displacement	doubletPtr	Relative displacement array

Return

– Function error code

Python

Prototype

integer **GroupHomeSearchAndRelativeMove** (integer SocketID, string GroupName, doublePtr Displacement)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	SingleAxis group name
– Displacement	doubletPtr	Relative displacement array

Return

– Function error code

2.3.3.11 GroupInterlockDisable

Name

GroupInterlockDisable – Disables group interlock mode.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Actor must be a group: ERR_WRONG_OBJECT_TYPE (-8), ERR_POSITIONER_NAME (-18)
- Valid group name: ERR_GROUP_NAME (-19)

Description

This function removes the dependency between this group and the groups that are included in GroupInterlock mode. So, if the function executes the group can initialize, home or move independent of all errors coming from other interlocked groups.

GroupInterlock mode: Actions that a group takes based on the activities of other groups: execute actions (like stop axis, power-off, change state…) immediately after an error (or an user command Disable/KillGroup) detected from one of its interlocked groups.

Example: The list of interlocked groups is G1, G2, G3, this means:

1. G1 depends on G2 and G3 (G1 in action if an error occurs on G2 or G3)
2. G2 depends on G1 and G3 (G2 in action if an error occurs on G1 or G3)
3. G3 depends on G1 and G2 (G3 in action if an error occurs on G1 or G2)

The interlocked groups are listed in the [GROUPS] section of *system.ini* file:

InterlockedGroups = …; Names of groups involved in the GroupInterlock mode.

The GroupInterlock mode is enabled by default at boot of the XPS controller.

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

GroupInterlockDisable SocketID GroupName

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Group name (maximum size = 250)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupInterlockDisable** (int SocketID, char *GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	char *	Goup name

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **GroupInterlockDisable** (ByVal SocketID As Long, ByVal GroupName As String)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Group name

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **GroupInterlockDisable** (int32 SocketID, cstring GroupName)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	cstring	Group name

Return

– Function error code

Python

Prototype

integer **GroupInterlockDisable** (integer SocketID, string GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	string	Group name

Return

– Function error code

2.3.3.12 GroupInterlockEnable

Name

GroupInterlockEnable – Enables group interlock mode.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Actor must be a group: ERR_WRONG_OBJECT_TYPE (-8), ERR_POSITIONER_NAME (-18)
- Valid group name: ERR_GROUP_NAME (-19)

Description

This function enables the dependency between this group and the groups that are involved in GroupInterlock mode. So, if the function executes the group cannot initialize, home or move without correcting the errors coming from its interlocked groups.

GroupInterlock mode: Activities that a group takes based on the activities of other groups: execute actions (like stop axis, power-off, change state…) immediately after an error (or an user command Disable/KillGroup) detected from one of its interlocked groups.

Example: The list of interlocked groups is G1, G2, G3 , this means:

4. G1 depends on G2 and G3 (G1 in action if an error occurs on G2 or G3)
5. G2 depends on G1 and G3 (G2 in action if an error occurs on G1 or G3)
6. G3 depends on G1 and G2 (G3 in action if an error occurs on G1 or G2)

The interlocked groups are listed in the [GROUPS] section of *system.ini* file:

InterlockedGroups = …; Names of groups involved in the GroupInterlock mode.

The GroupInterlock mode is enabled by default at boot of the XPS controller.

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

GroupInterlockEnable SocketID GroupName

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	Group name (maximum size = 250)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupInterlockEnable** (int SocketID, char *GroupName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	Goup name

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **GroupInterlockEnable** (ByVal SocketID As Long, ByVal GroupName As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	Group name

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **GroupInterlockEnable** (int32 SocketID, cstring GroupName)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	cstring	Group name

Return

– Function error code

Python

Prototype

integer **GroupInterlockEnable** (integer SocketID, string GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	string	Group name

Return

– Function error code

2.3.3.13 GroupJogModeDisable

Name

GroupJogModeDisable – Disables the jog mode *
1. Not allowed for a spindle group –

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters [1]: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid object type (group): ERR_WRONG_OBJECT_TYPE (-8)
- Valid group name: ERR_GROUP_NAME (-19)
- Group status must be "JOGGING": ERR_NOT_ALLOWED_ACTION (-22)

Description

Disables the Jog mode. To use this function, the group must be in the "JOGGING" state and all positioners must be idle (meaning velocity must be 0).

This function exits the "JOGGING" state and to returns to the "READY" state. If the group state is not in the "JOGGING" state or if the profiler velocity is not null then the error ERR_NOT_ALLOWED_ACTION (-22) is returned.

NOTE

To enable the jog mode use the "GroupJogModeEnable" function.

CAUTION

The jog mode cannot be used with a spindle group.

Error codes

ERR_FATAL_INIT (-20)
ERR_GROUP_NAME (-19)
ERR_IN_INITIALIZATION (-21)
ERR_NOT_ALLOWED_ACTION (-22)
ERR_WRONG_FORMAT (-7)
ERR_WRONG_OBJECT_TYPE (-8)
ERR_WRONG_PARAMETERS_NUMBER (-9)
SUCCESS (0): no error

TCL

Prototype

GroupJogModeDisable SocketID GroupName

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Group name (maximum size = 250)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupJogModeDisable** (int SocketID, char *GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	char *	Group name

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **GroupJogModeDisable** (ByVal SocketID As Long, ByVal GroupName As String)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Group name

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **GroupJogModeDisable** (int32 SocketID, cstring GroupName)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	cstring	Group name

Return

– Function error code

Python

Prototype

integer **GroupJogModeDisable** (integer SocketID, string GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	string	Group name

Return

– Function error code

2.3.3.14 GroupJogModeEnable

Name

GroupJogModeEnable – Enables the jog mode *- Not allowed for a spindle group -*

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Backlash must not be activated: ERR_NOT_ALLOWED_BACKLASH (-46)
- Number of command parameters [1]: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid object type (group): ERR_WRONG_OBJECT_TYPE (-8), ERR_POSITIONER_NAME (-18)
- Valid group name: ERR_GROUP_NAME (-19)
- Group status must be "READY": ERR_NOT_ALLOWED_ACTION (-22)

Description

Enables the Jog mode. To use this function, the group must be in the "READY" state and all positioners must be idle (meaning velocity must be 0).

This function goes to the "JOGGING" state. If the group state is not "READY", ERR_NOT_ALLOWED_ACTION (-22) is returned.

NOTE

To disable the jog mode use the "GroupJogModeDisable" function.

CAUTION

The jog mode cannot be used with a spindle group.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_NOT_ALLOWED_BACKLASH (-46)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

GroupJogModeEnable SocketID GroupName

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Group name (maximum size = 250)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupJogModeEnable** (int SocketID, char *GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	char *	Group name

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **GroupJogModeEnable** (ByVal SocketID As Long, ByVal GroupName As String)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Group name

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **GroupJogModeEnable** (int32 SocketID, cstring GroupName)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	cstring	Group name

Return

– Function error code

Python

Prototype

integer **GroupJogModeEnable** (integer SocketID, string GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	string	Group name

Return

– Function error code

2.3.3.15 GroupJogCurrentGet

Name

GroupJogCurrentGet – Returns the current velocity and acceleration from the jog profiler.

- Not allowed for a spindle group -

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Valid object type (group or positioner): ERR_WRONG_OBJECT_TYPE (-8)
- Valid group name: ERR_GROUP_NAME (-19)
- Valid positioner name: ERR_POSITIONER_NAME (-18)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function returns the current velocity and acceleration from the jog profiler for one positioner or for all positioners of the selected group.

So, this function must be called when the group is in "JOGGING" mode else the current velocity and the current acceleration will be null.

CAUTION

The jog mode cannot be used with a spindle group.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

GroupJogCurrentGet SocketID GroupName Velocity Acceleration …

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	Group name or Positioner name (maximum size = 250)

Output parameters

– Velocity	floating point	Current velocity
– Acceleration	floating point	Current Acceleration

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupJogCurrentGet** (int SocketID, char *GroupName, int NbPositioners, double * Velocity, double * Acceleration)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	Group name or Positioner name
– NbPositioners	int	Number of positioners in the selected group (1 if a positioner)

Output parameters

–	Velocity	double *	Current velocity array
– Acceleration	double *	Current Acceleration array	

Return

– Function error code

Visual Basic

Prototype

Long **GroupJogCurrentGet** (ByVal SocketID As Long, ByVal GroupName As String, ByVal NbPositioners As Long, Velocity As Double, Acceleration As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	Group name or Positioner name
– NbPositioners	long	Number of positioners in the selected group (1 if a positioner)

Output parameters

–	Velocity	double	Current velocity array
– Acceleration	double	Current Acceleration array	

Return

– Function error code

Matlab

Prototype

[Error, Velocity, Acceleration] **GroupJogCurrentGet** (int32 SocketID, cstring GroupName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	Group name or Positioner name

Return

– Error	int32	Function error code	
–	Velocity	doubletPtr	Current velocity array
– Acceleration	doubletPtr	Current Acceleration array	

Python

Prototype

[Error, Velocity, Acceleration] **GroupJogCurrentGet** (integer SocketID, string GroupName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	Group name or Positioner name

Return

– Error	int	Function error code	
–	Velocity	doubletPtr	Current velocity array
– Acceleration	doubletPtr	Current Acceleration array	

2.3.3.16 GroupJogParametersGet

Name

GroupJogParametersGet – Returns the velocity and acceleration set by "GroupJogParametersSet".

- Not allowed for a spindle group -

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid object type (group or positioner): ERR_WRONG_OBJECT_TYPE (-8)
- Valid group name: ERR_GROUP_NAME (-19)
- Valid positioner name: ERR_POSITIONER_NAME (-18)
- Output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function returns the velocity and the acceleration set by the user to use the jog mode for one positioner or for all positioners of the selected group.

So, this function must be called when the group is in "JOGGING" mode else the velocity and the acceleration will be null.

To change the velocity and the acceleration on the fly, in the jog mode, call the "GroupJogParametersSet" function.

CAUTION

The jog mode cannot be used with a spindle group.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

GroupJogParametersGet SocketID GroupName Velocity Acceleration …

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Group name (maximum size = 250)

Output parameters

–	Velocity	floating point	User jog velocity
–	Acceleration	floating point	User jog Acceleration

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupJogParametersGet** (int SocketID, char *GroupName, int NbPositioners, double * Velocity, double * Acceleration)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	char *	Group name
–	NbPositioners	int	Number of positioners in the selected group

Output parameters

–	Velocity	double *	User jog velocity array
–	Acceleration	double *	User jog Acceleration array

Return

– Function error code

Visual Basic

Prototype

Long **GroupJogParametersGet** (ByVal SocketID As Long, ByVal GroupName As String, ByVal NbPositioners As Long, Velocity As Double, Acceleration As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	Group name
– NbPositioners	long	Number of positioners in the selected group

Output parameters

– Velocity	double	User jog velocity array
– Acceleration	double	User jog Acceleration array

Return

– Function error code

Matlab

Prototype

[Error, Velocity, Acceleration] **GroupJogParametersGet** (int32 SocketID, cstring GroupName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	Group name

Return

– Error	int32	Function error code
– Velocity	doubletPtr	User jog velocity array
– Acceleration	doubletPtr	User jog Acceleration array

Python

Prototype

[Error, Velocity, Acceleration] **GroupJogParametersGet** (integer SocketID, string GroupName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	Group name

Return

– Error	int	Function error code
– Velocity	doubletPtr	User jog velocity array
– Acceleration	doubletPtr	User jog Acceleration array

2.3.3.17 GroupJogParametersSet

Name

GroupJogParametersSet – Changes the velocity and the acceleration on the fly, in the jog mode.

- Not allowed for a spindle group -

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Group status must be "JOGGING": ERR_NOT_ALLOWED_ACTION (-22)
- Valid object type (group or positioner): ERR_WRONG_OBJECT_TYPE (-8)
- Valid group name: ERR_GROUP_NAME (-19)
- Valid positioner name: ERR_POSITIONER_NAME (-18)
- Input parameters for each positioner:
  1. Velocity > MaximumVelocity => Velocity = MaximumVelocity
  2. Velocity < -MaximumVelocity => Velocity = -MaximumVelocity
  3. Acceleration ≤ 0 => ERR_JOG_OUT_OF_RANGE (-42)
  4. Acceleration > MaximumAcceleration => Acceleration = MaximumAcceleration

Description

This function changes the velocity and the acceleration on the fly, used in the jog mode. If an error occurs, each positioner stops and the velocity value is set to zero.

To use this function, the jog mode must be enabled (requires call of the "GroupJogModeEnable" function). If the group status is not "JOGGING" then an "ERR_NOT_ALLOWED_ACTION (-22)" is returned.

If a slave or following error is detected during the jog setting then an "ERR_FOLLOWING_ERROR (-25)" or "ERR_SLAVE (-44)" is returned. In this case, the motion is stopped, the jog mode is disabled and the group status becomes "DISABLE".

CAUTION

The jog mode cannot be used with a spindle group.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_FOLLOWING_ERROR (-25)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_JOG_OUT_OF_RANGE (-42)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_POSITIONER_NAME (-18)
- ERR_SLAVE (-44)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_OBJECT_TYPE (-8)
- SUCCESS (0): no error

TCL

Prototype

GroupJogParametersSet SocketID GroupName Velocity Acceleration

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	Group name (maximum size = 250)
– Velocity	floating point	User jog velocity
– Acceleration	floating point	User jog Acceleration

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupJogParametersSet** (int SocketID, char *GroupName, int NbPositioners, double * Velocity, double * Acceleration)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	Group name
– NbPositioners	int	Number of positioners in the selected group
– Velocity	double *	User jog velocity array
– Acceleration	double *	User jog Acceleration array

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **GroupJogParametersSet** (ByVal SocketID As Long, ByVal GroupName As String, ByVal NbPositioners As Long, Velocity As Double, Acceleration As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	Group name
– NbPositioners	long	Number of positioners in the selected group
– Velocity	double	User jog velocity array
– Acceleration	double	User jog Acceleration array

Output parameters

– None

Return

– Function error code

Matlab

Prototype

[Error, Velocity, Acceleration] **GroupJogParametersSet** (int32 SocketID, cstring GroupName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	Group name
– Velocity	doubletPtr	User jog velocity array
– Acceleration	doubletPtr	User jog Acceleration array

Return

– Error	int32	Function error code

Python

Prototype

[Error, Velocity, Acceleration] **GroupJogParametersSet** (integer SocketID, string GroupName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	Group name
– Velocity	doubletPtr	User jog velocity array
– Acceleration	doubletPtr	User jog Acceleration array

Return

– Error	int	Function error code

2.3.3.18 GroupKill

Name

GroupKill - Kills the selected group to go to "NOTINIT" status.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid object type (group): ERR_WRONG_OBJECT_TYPE (-8), ERR_POSITIONER_NAME (-18)
- Valid group name: ERR_GROUP_NAME (-19)

Description

Kills the selected group to stop its action. The group returns to the "NOTINIT" state. If the group is already in this state then it stays in the "NOT INIT" state.

The GroupKill is a high priority command that is executed in any condition.

NOTE

If an initialization, encoder calibration homing, referencing, motion or a trajectory is in progress, an "emergency stop" will be done. So, for each of these functions, an " ERR_EMERGENCY_SIGNAL (-26)" error will be generated.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

GroupKill SocketID GroupName

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Group name (maximum size = 250)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupKill** (int SocketID, char *GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	char *	Goup name

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **GroupKill** (ByVal SocketID As Long, ByVal GroupName As String)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Group name

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **GroupKill** (int32 SocketID, cstring GroupName)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	cstring	Group name

Return

– Function error code

Python

Prototype

integer **GroupKill** (integer SocketID, string GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	string	Group name

Return

– Function error code

2.3.3.19 GroupMotionDisable

Name

GroupMotionDisable – Disables a "READY" group.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Group status must be "READY": ERR_NOT_ALLOWED_ACTION (-22)
- Actor must be a group: ERR_WRONG_OBJECT_TYPE (-8), ERR_POSITIONER_NAME (-18)
- Valid group name: ERR_GROUP_NAME (-19)

Description

Turns OFF the motors, stops the corrector servo loop and disables the position compare mode if active. The group status becomes "DISABLE".

If the group is not in the "READY" state then an ERR_NOT_ALLOWED_ACTION (-22) is returned.

NOTE

In the "DISABLED" state the encoder is still read.

To return to the "READY" state, call the "GroupMotionEnable" function.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

GroupMotionDisable SocketID GroupName

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Group name (maximum size = 250)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupMotionDisable** (int SocketID, char *GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	char *	Goup name

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **GroupMotionDisable** (ByVal SocketID As Long, ByVal GroupName As String)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Group name

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **GroupMotionDisable** (int32 SocketID, cstring GroupName)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	cstring	Group name

Return

– Function error code

Python

Prototype

integer **GroupMotionDisable** (integer SocketID, string GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	string	Group name

Return

– Function error code

2.3.3.20 GroupMotionEnable

Name

GroupMotionEnable – Enables a group in a DISABLE state to turn the motors on and to restart corrector loops.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Group status must be "DISABLE": ERR_NOT_ALLOWED_ACTION (-22)
- Actor must be a group: ERR_WRONG_OBJECT_TYPE (-8), ERR_POSITIONER_NAME (-18)
- Valid group name: ERR_GROUP_NAME (-19)

Description

Turns ON the motors and restarts the corrector servo loops. The group state becomes "READY".

If the group is not in the "DISABLE" state then the "ERR_NOT_ALLOWED_ACTION (-22)" is returned.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

GroupMotionEnable SocketID GroupName

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Group name (maximum size = 250)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupMotionEnable** (int SocketID, char *GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	char *	Goup name

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **GroupMotionEnable** (ByVal SocketID As Long, ByVal GroupName As String)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Group name

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **GroupMotionEnable** (int32 SocketID, cstring GroupName)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	cstring	Group name

Return

– Function error code

Python

Prototype

integer **GroupMotionEnable** (integer SocketID, string GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	string	Group name

Return

– Function error code

2.3.3.21 GroupMotionStatusGet

Name

GroupMotionStatusGet – Returns the motion status for one or all positioners of the selected group.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid object type (group or positioner): ERR_WRONG_OBJECT_TYPE (-8)
- Valid group name: ERR_GROUP_NAME (-19)
- Valid positioner name: ERR_POSITIONER_NAME (-18)
- Valid output parameter type: ERR_WRONG_TYPE_INT (-15)

Description

Returns the motion status for one or all positioners of the selected group.

The motion status possible values are :

0 : Not moving state (group status in NOT_INIT, NOT_REF or READY).

1 : Busy state (positioner in moving, homing, referencing, spinning, analog tracking, trajectory, encoder calibrating, slave mode).

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0) : no error

TCL

Prototype

GroupMotionStatusGet SocketID GroupName Status1 Status2 …

or

GroupMotionStatusGet SocketID GroupName.PositionerName Status

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Group name (maximum size = 250)

Output parameters

–	Status	int	Positioner status (to be repeated for each positioner of group)

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupMotionStatusGet** (int SocketID, char *GroupName, int NbPositioners, int * Status)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	char *	Group name
–	NbPositioners	int	Number of positioners in the selected group

Output parameters

–	Status	int *	Positioner status

Return

– Function error code

Visual Basic

Prototype

Long **GroupMotionStatusGet** (ByVal SocketID As Long, ByVal GroupName As String, ByVal NbPositioners As Long, Status As Long)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	Group name
– NbPositioners	long	Number of positioners in the selected group

Output parameters

– Status	long	Positioner status

Return

– Function error code

Matlab

Prototype

[Error, Status] **GroupMotionStatusGet** (int32 SocketID, cstring GroupName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	Group name

Return

– Error	int32	Function error code
– Status	int32	PtrPositioner status

Python

Prototype

[Error, Status] **GroupMotionStatusGet** (integer SocketID, string GroupName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	StringGroup name	

Return

– Error	int	Function error code
– Status	int	PtrPositioner status

2.3.3.22 GroupMoveAbort

Name

GroupMoveAbort – aborts the motion or the jog in progress for a group or a positioner.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters [1]: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Group status must be "NOTINITMOVING" or "JOGGING": ERR_NOT_ALLOWED_ACTION (-22)
- Valid object type (group or positioner): ERR_WRONG_OBJECT_TYPE (-8)
- Valid group name: ERR_GROUP_NAME (-19)
- Valid positioner name: ERR_POSITIONER_NAME (-18)

Description

This function aborts a motion or a jog in progress. The group state must be "MOVING" or "JOGGING" else the "ERR_NOT_ALLOWED_ACTION (-22)" is returned.

For a group:

If the group status is "MOVING", this function stops all motion in progress.

If the group status is "JOGGING", this function stops all "jog" motions in progress and disables the jog mode. After this "group move abort" action, the group status becomes "READY".

For a positioner:

If the group status is "MOVING", this function stops the motion of the selected positioner.

If the group status is "JOGGING", this function stops the "jog" motion of the selected positioner.

If the positioner is idle, an ERR_NOT_ALLOWED_ACTION (-22) is returned.

After this "positioner move abort" action, if all positioners are idle then the group status becomes "READY", else the group stays in the same state.

NOTE

If the "move abort" action fails, an ERR_GROUP_ABORT_MOTION (-27) is returned.

This error is generated when GroupMoveAbort is used to abort a motion of a positioner in a group and the name of the positioner is incorrect.

This error will also be return by the "GroupMove" function.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_ABORT_MOTION (-27)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

GroupMoveAbort SocketID GroupName

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	Group name (maximum size = 250)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupMoveAbort** (int SocketID, char *GroupName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	Goup name

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **GroupMoveAbort** (ByVal SocketID As Long, ByVal GroupName As String)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Group name

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **GroupMoveAbort** (int32 SocketID, cstring GroupName)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	cstring	Group name

Return

– Function error code

Python

Prototype

integer **GroupMoveAbort** (integer SocketID, string GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	string	Group name

Return

– Function error code

2.3.3.23 GroupMoveAbortFast

Name

GroupMoveAbortFast – aborts with user-defined deceleration a motion or a jog in progress for a group or positioner.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters [1]: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Group status must be "NOTINITMOVING" or "JOGGING": ERR_NOT_ALLOWED_ACTION (-22)
- Valid object type (group or positioner): ERR_WRONG_OBJECT_TYPE (-8)
- Valid group name: ERR_GROUP_NAME (-19)
- Valid positioner name: ERR_POSITIONER_NAME (-18)
- Valid UserDecelerationMultiplier value (**>= 1** and **<= 100**): ERR_PARAMETER_OUT_OF_RANGE (-17)

Description

This function aborts a motion or a jog in progress with a deceleration value defined by user (*UserDeceleration*):

UserDeceleration = DecelerationMultiplier * MaximumAcceleration.

Here: *DecelerationMultiplier*: GroupMoveAbortFast function parameter

MaximumAcceleration: Stage parameter, defined in the stages.ini file.

The group state must be "MOVING" or "JOGGING" else the "ERR_NOT_ALLOWED_ACTION (-22)" is returned.

For a group:

If the group state is "MOVING", this function stops all motion.

If the group state is "JOGGING", this function stops all "jog" motion and disables the jog mode. After this "group move abort" action, the group state becomes "READY".

For a positioner:

If the group state is "MOVING", this function stops the motion of the selected positioner.

If the group state is "JOGGING", this function stops the "jog" motion of the selected positioner.

If the positioner is idle, an ERR_NOT_ALLOWED_ACTION (-22) is returned.

After this "positioner move abort" action, if all positioners are idle then the group state becomes "READY", else the group stays in the same state.

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

If the "move abort" action fails, an ERR_GROUP_ABORT_MOTION (-27) is returned.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_ABORT_MOTION (-27)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- SUCCESS (0): no error

TCL

Prototype

GroupMoveAbortFast SocketID GroupName DecelerationMultiplier

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Group name (maximum size = 250)
–	DecelerationMultiplier	int	Braking deceleration multiplier

Output parameters

– None

Return

TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupMoveAbortFast** (int SocketID, char *GroupName, int DecelerationMultiplier)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	char *	Group name
–	DecelerationMultiplier	int	Braking deceleration multiplier

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **GroupMoveAbortFast** (ByVal SocketID As Long, ByVal GroupName As String, ByVal DecelerationMultiplier As Long)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	Group name
– DecelerationMultiplier	long	Braking deceleration multiplier

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **GroupMoveAbortFast** (int32 SocketID, cstring GroupName, int32 DecelerationMultiplier)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	Group name
– DecelerationMultiplier	int32	Braking deceleration multiplier

Return

– Function error code

Python

Prototype

integer **GroupMoveAbortFast** (integer SocketID, string GroupName, integer DecelerationMultiplier)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	Group name
– DecelerationMultiplier	int	Braking deceleration multiplier

Return

– Function error code

2.3.3.24 GroupMoveAbsolute

Name

GroupMoveAbsolute - Initiates an absolute move for a positioner or a group.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Group status must be "READY" or "MOVING": ERR_NOT_ALLOWED_ACTION (-22)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid object type (group or positioner): ERR_WRONG_OBJECT_TYPE (-8)
- Valid group name: ERR_GROUP_NAME (-19)
- Valid positioner name: ERR_POSITIONER_NAME (-18)
- Valid input parameter type: ERR_WRONG_TYPE_DOUBLE (-14)
- Verify target position in relation with the travel limits: ERR_PARAMETER_OUT_OF_RANGE (-17)
  - TargetPosition $\geq$ MinimumTargetPosition
  - TargetPosition $\leq$ MaximumTargetPosition

Description

This function initiates an absolute move to one or all positioners of the selected group. The group state must be "READY" or "MOVING" else ERR_NOT_ALLOWED_ACTION (-22) is returned. If the group is "READY" then the group state becomes "MOVING".

An absolute motion is defined by the distance between to the zero position and the target position. If the current position is the same as the target position then no move will be done.

Each "positioner" move refers to the acceleration, velocity, minimum jerkTime and maximum jerkTime as defined in the "Stages.ini" file or as redefined by the "PositionerSGammaParametersSet" function.

If a slave or following error is detected during the move then ERR_FOLLOWING_ERROR (-25) or ERR_SLAVE (-44) is returned. In this case, the motion in progress is stopped and the group state becomes "DISABLE".

If a "MotionDoneMode" is defined as "VelocityAndPositionWindowMotionDone" then an ERR_GROUP_MOTION_DONE_TIMEOUT (-33) error will be returned if the time out (defined by "MotionDoneTimeout" in the stages.ini file) is reached before the motion is done. The group state becomes "DISABLE".

If a "GroupMoveAbort" is done, an ERR_GROUP_ABORT_MOTION (-27) is returned. In this case, the motion in progress is stopped and the group state becomes "READY".

During a move with PositionCompare (or TimeFlasher) scan enabled, if the current following error exceeds *WarningFollowingError* value inside the PositionCompare (or TimeFlasher) scan zone, a *WarningFollowingErrorFlag* is latched. In this case the motion continues then finishes normally (the group status becomes "READY"), but the *GroupMoveAbsolute* function returns ERR_WARNING_FOLLOWING_ERROR (-120) error instead of SUCCESS (0). To reset *WarningFollowingErrorFlag* for next

moves, execute *PositionerPositionCompareDisable* (or *PositionerTimeFlasherDisable*) function.

If in "GroupKill" command an emergency brake or an emergency stop has occurred, an "ERR_EMERGENCY_SIGNAL (-26)" is returned. In this case, the motion in progress is stopped and the group state becomes "NOTINIT".

NOTE

Asynchronous moves for positioners of the same group are possible through the use of different sockets to send functions.

Error codes

- ERR_EMERGENCY_SIGNAL (-26)
- ERR_FATAL_INIT (-20)
- ERR_FOLLOWING_ERROR (-25)
- ERR_GROUP_ABORT_MOTION (-27)
- ERR_GROUP_MOTION_DONE_TIMEOUT (-33)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_POSITIONER_NAME (-18)
- ERR_SLAVE (-44)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- ERR_WARNING_FOLLOWING_ERROR (-120)
- SUCCESS (0): no error

TCL

Prototype

GroupMoveAbsolute SocketID GroupName TargetPosition

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Group name (maximum size = 250)
–	TargetPosition	floating point	Target position (must be repeated for each positioner of the selected group)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupMoveAbsolute** (int SocketID, char *GroupName, int NbPositioners, double * TargetPosition)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	char *	Group name
–	NbPositioners	int	Number of positioners in the selected group
–	TargetPosition	double	*Target position array

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **GroupMoveAbsolute** (ByVal SocketID As Long, ByVal GroupName As String, ByVal NbPositioners As Long, TargetPosition As Double)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	SingleAxis group name
–	NbPositioners	long	Number of positioners in the selected group
–	TargetPosition	double	Target position array

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **GroupMoveAbsolute** (int32 SocketID, cstring GroupName, DoublePtr TargetPosition)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	cstring	SingleAxis group name
–	TargetPosition	doubletPtr	Target position array

Return

– Function error code

Python

Prototype

integer **GroupMoveAbsolute** (integer SocketID, string GroupName, doublePtr TargetPosition)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	string	SingleAxis group name
–	TargetPosition	doubletPtr	Target position array

Return

– Function error code

2.3.3.25 GroupMoveRelative

Name

GroupMoveRelative - Initiates a relative move for a positioner or a group.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Group status must be "READY" or "MOVING": ERR_NOT_ALLOWED_ACTION (-22)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid object type (group or positioner): ERR_WRONG_OBJECT_TYPE (-8)
- Valid group name: ERR_GROUP_NAME (-19)
- Valid positioner name: ERR_POSITIONER_NAME (-18)
- Valid input parameter type: ERR_WRONG_TYPE_DOUBLE (-14)
- Verify target displacment in relation with the travel limits: ERR_PARAMETER_OUT_OF_RANGE (-17)
  - TargetPosition $\geq$ MinimumTargetPosition
  - TargetPosition $\leq$ MaximumTargetPosition

Description

This function initiates a relative move defined by the target displacement to one or all positioners of the selected group. The group state must be "READY" or "MOVING" else ERR_NOT_ALLOWED_ACTION (-22) is returned. If the group is "READY" then the group state becomes "MOVING".

The target displacement and the current position defines the new target position to reach:

NewTargetPosition = CurrentTargetPosition + TargetDisplacement

Each "positioner" move refers to the acceleration, velocity, minimum jerkTime and maximum jerkTime as defined in the "Stages.ini" file or as redefined by the "PositionerSGammaParametersSet" function.

If a slave or following error is detected during the move then an "ERR_FOLLOWING_ERROR (-25)" or "ERR_SLAVE (-44)" is returned. In this case, the motion in progress is stopped and the group status becomes "DISABLE".

If a "MotionDoneMode" is defined as "VelocityAndPositionWindowMotionDone" then an "ERR_GROUP_MOTION_DONE_TIMEOUT (-33)" error will be returned if the time out (defined by "MotionDoneTimeout" in the stages.ini file) is reached before the motion is done. The group state becomes "DISABLE".

If a "GroupMoveAbort" is done, an "ERR_GROUP_ABORT_MOTION (-27)" is returned. In this case, the motion in progress is stopped and the group state becomes "READY".

During move with PositionCompare (or TimeFlasher) scan enabled, if the current following error exceeds *WarningFollowingError* value inside the PositionCompare (or TimeFlasher) scan zone, a *WarningFollowingErrorFlag* is latched. In this case the motion continues then finishes normally (the group status becomes "READY"), but the *GroupMoveRelative* function returns ERR_WARNING_FOLLOWING_ERROR (-120) error instead of SUCCESS (0). To reset *WarningFollowingErrorFlag* for the next

moves, execute *PositionerPositionCompareDisable* (or *PositionerTimeFlasherDisable*) function.

If in "GroupKill" command an emergency brake or an emergency stop has occurred, an "ERR_EMERGENCY_SIGNAL (-26)" is returned. In this case, the motion in progress is stopped and the group state becomes "NOTINIT".

NOTE

Asynchronous moves for positioners of a same group are possible through the use of different sockets to send the functions.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_EMERGENCY_SIGNAL (-26)
- ERR_FOLLOWING_ERROR (-25)
- ERR_GROUP_ABORT_MOTION (-27)
- ERR_GROUP_MOTION_DONE_TIMEOUT (-33)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_POSITIONER_NAME (-18)
- ERR_SLAVE (-44)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- ERR_WARNING_FOLLOWING_ERROR (-120)
- SUCCESS (0): no error

TCL

Prototype

GroupMoveRelative SocketID GroupName Displacement

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Group name (maximum size = 250)
–	Displacement	floating point	Relative displacement (must be repeated for each positioner of the selected group)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupMoveRelative** (int SocketID, char *GroupName, int NbPositioners, double * Displacement)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	char *	Group name
–	NbPositioners	int	Number of positioners in the selected group
–	Displacement	double *	Relative displacement array

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **GroupMoveRelative** (ByVal SocketID As Long, ByVal GroupName As String, ByVal NbPositioners As Long, Displacement As Double)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	SingleAxis group name
–	NbPositioners	long	Number of positioners in the selected group
–	Displacement	double	Relative displacement array

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **GroupMoveRelative** (int32 SocketID, cstring GroupName, DoublePtr Displacement)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	cstring	SingleAxis group name
–	Displacement	doubletPtr	Relative displacement array

Return

– Function error code

Python

Prototype

integer **GroupMoveRelative** (integer SocketID, string GroupName, doublePtr Displacement)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	string	SingleAxis group name
–	Displacement	doubletPtr	Relative displacement array

Return

– Function error code

2.3.3.26 GroupPositionCorrectedProfilerGet

Name

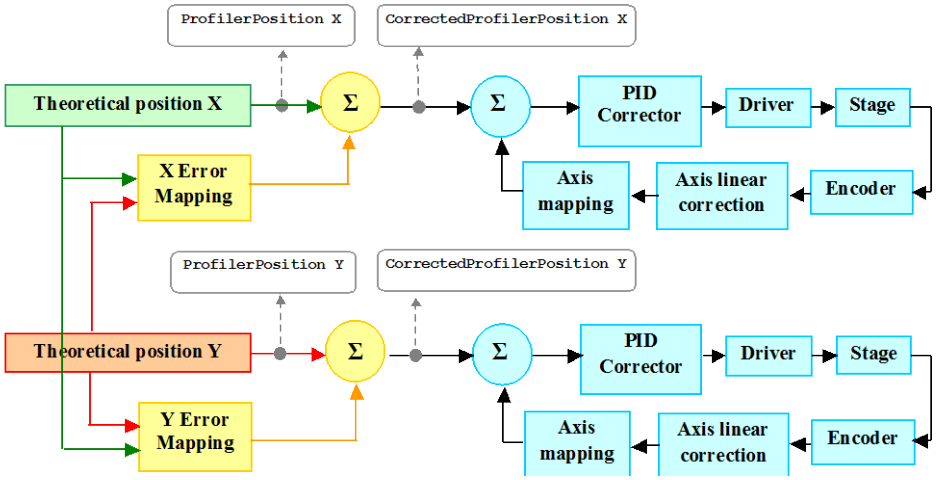
GroupPositionCorrectedProfilerGet – Returns the corrected profiler position for all positioners of an XY group.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid object type: ERR_WRONG_OBJECT_TYPE (-8)
- Valid group type (must be an XY group): ERR_POSITIONER_NAME (-18)
- Valid group name: ERR_GROUP_NAME (-19)
- Valid output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

The **GroupPositionCorrectedProfilerGet** function gets the corrected position which is the theoretical position recalculated with the XY mapping correction.



This function applies the XY mapping on the theoretical user-defined positions and returns the corrected positions.

NOTE

This function is only allowed with an XY group.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

GroupPositionCorrectedProfilerGet SocketID GroupName PositionX PositionY CorrectedPositionX CorrectedPositionY

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	XY group name (maximum size = 250)
– PositionX	floating point	Theoretical position X
– PositionY	floating point	Theoretical position Y

Output parameters

– CorrectedPositionX	floating point	Corrected theoretical position X
– CorrectedPositionY	floating point	Corrected theoretical position Y

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupPositionCorrectedProfilerGet** (int SocketID, char *GroupName, double PositionX, double PositionY, double * CorrectedPositionX, double * CorrectedPositionY)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	char *	XY group name
–	PositionX	double	Theoretical position X
–	PositionY	double	Theoretical position Y

Output parameters

–	CorrectedPositionX	double *	Corrected theoretical position X
–	CorrectedPositionY	double *	Corrected theoretical position Y

Return

– Function error code

Visual Basic

Prototype

Long **GroupPositionCorrectedProfilerGet** (ByVal SocketID As Long, ByVal GroupName As String, ByVal PositionX As Double, ByVal PositionY As Double, CorrectedPositionX As Double, CorrectedPositionY As Double)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	XY group name
–	PositionX	double	Theoretical position X
–	PositionY	double	Theoretical position Y

Output parameters

–	CorrectedPositionX	double	Corrected theoretical position X
–	CorrectedPositionY	double	Corrected theoretical position Y

Return

– Function error code

Matlab

Prototype

[Error, CorrectedPositionX, CorrectedPositionY] **GroupPositionCorrectedProfilerGet** (int32 SocketID, cstring GroupName, double PositionX, double PositionY)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	XY group name
– PositionX	double	Theoretical position X
– PositionY	double	Theoretical position Y

Return

– Error	int32	Function error code
– CorrectedPositionX	doubletPtr	Corrected theoretical position X
– CorrectedPositionY	doubletPtr	Corrected theoretical position Y

Python

Prototype

[Error, CorrectedPositionX, CorrectedPositionY] **GroupPositionCorrectedProfilerGet** (integer SocketID, string GroupName, double PositionX, double PositionY)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	XY group name
– PositionX	double	Theoretical position X
– PositionY	double	Theoretical position Y

Return

– Error	int	Function error code
– CorrectedPositionX	doubletPtr	Corrected theoretical position X
– CorrectedPositionY	doubletPtr	Corrected theoretical position Y

2.3.3.27 GroupPositionCurrentGet

Name

GroupPositionCurrentGet – Returns the current position for one or all positioners of the selected group.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid object type (group or positioner): ERR_WRONG_OBJECT_TYPE (-8)
- Valid group name: ERR_GROUP_NAME (-19)
- Valid positioner name: ERR_POSITIONER_NAME (-18)
- Valid output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

Returns the current position for one or all positioners of the selected group.

The current position is defined as:

CurrentPosition = SetpointPosition - FollowingError

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

GroupPositionCurrentGet SocketID GroupName CurrentPosition …

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	Group name (maximum size = 250)

Output parameters

– CurrentPosition	floating point	Current Position (must be repeated for each positioner of group)

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupPositionCurrentGet** (int SocketID, char *GroupName, int NbPositioners, double * CurrentPosition)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	Group name
– NbPositioners	int	Number of positioners in the selected group

Output parameters

– CurrentPosition	double *	Current position array

Return

– Function error code

Visual Basic

Prototype

Long **GroupPositionCurrentGet** (ByVal SocketID As Long, ByVal GroupName As String, ByVal NbPositioners As Long, CurrentPosition As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	Group name
– NbPositioners	long	Number of positioners in the selected group

Output parameters

– CurrentPosition	double	Current position array

Return

– Function error code

Matlab

Prototype

[Error, CurrentPosition] **GroupPositionCurrentGet** (int32 SocketID, cstring GroupName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	Group name

Return

– Error	int32	Function error code
– CurrentPosition	doubletPtr	Current position array

Python

Prototype

[Error, CurrentPosition] **GroupPositionCurrentGet** (integer SocketID, string GroupName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	Group name

Return

– Error	int	Function error code
– CurrentPosition	doubletPtr	Current position array

2.3.3.28 GroupPositionPCORawEncoderGet

Name

GroupPositionPCORawEncoderGet – Returns the PCO raw encoder positions for an XY group.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid object type: ERR_WRONG_OBJECT_TYPE (-8)
- Valid group type (must be an XY group): ERR_POSITIONER_NAME (-18)
- Valid group name: ERR_GROUP_NAME (-19)
- Valid output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function returns the PCO raw encoder positions X and Y from the user specified positions X and Y.

NOTE

This function is only allowed with an XY group.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

Group**PositionPCORawEncoderGet** SocketID GroupName PositionX PositionY PCORawPositionX CorrectedPositionY

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	XY group name (maximum size = 250)
– PositionX	floating point	User corrected position X
– PositionY	floating point	User corrected position Y

Output parameters

– PCORawPositionX	floating point	PCO Raw position X
– PCORawPositionY	floating point	PCO Raw position Y

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupPositionPCORawEncoderGet** (int SocketID, char *GroupName, double PositionX, double PositionY, double * PCORawPositionX, double * PCORawPositionY)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	XY group name
– PositionX	double	User corrected position X
– PositionY	double	User corrected position Y

Output parameters

– PCORawPositionX	double *	PCO Raw position X
– PCORawPositionY	double *	PCO Raw position Y

Return

– Function error code

Visual Basic

Prototype

Long **GroupPositionPCORawEncoderGet** (ByVal SocketID As Long, ByVal GroupName As String, ByVal PositionX As Double, ByVal PositionY As Double, PCORawPositionX As Double, PCORawPositionY As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	XY group name
– PositionX	double	User corrected position X
– PositionY	double	User corrected position Y

Output parameters

– PCORawPositionX	double	PCO Raw position X
– PCORawPositionY	double	PCO Raw position Y

Return

– Function error code

Matlab

Prototype

[Error, PCORawPositionX, PCORawPositionY] **GroupPositionPCORawEncoderGet** (int32 SocketID, cstring GroupName, double PositionX, double PositionY)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	XY group name
– PositionX	double	User corrected position X
– PositionY	double	User corrected position Y

Return

– Error	int32	Function error code
– PCORawPositionX	doubletPtr	PCO Raw position X
– PCORawPositionY	doubletPtr	PCO Raw position Y

Python

Prototype

[Error, PCORawPositionX, PCORawPositionY] **GroupPositionPCORawEncoderGet** (integer SocketID, string GroupName, double PositionX, double PositionY)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	XY group name
– PositionX	double	User corrected position X
– PositionY	double	User corrected position Y

Return

– Error	int	Function error code
– PCORawPositionX	doubletPtr	PCO Raw position X
– PCORawPositionY	doubletPtr	PCO Raw position Y

2.3.3.29 GroupPositionSetpointGet

Name

GroupPositionSetpointGet – Returns the setpoint position for one or all positioners of the selected group.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid object type (group or positioner): ERR_WRONG_OBJECT_TYPE (-8)
- Valid group name: ERR_GROUP_NAME (-19)
- Valid positioner name: ERR_POSITIONER_NAME (-18)
- Valid output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

Returns the setpoint position for one or all positioners of the selected group.

The "setpoint" position is calculated by the motion profiler and represents the "theoretical" position to reach.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

GroupPositionSetpointGet SocketID GroupName SetpointPosition …

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Group name (maximum size = 250)

Output parameters

–	SetpointPosition	floating point	Setpoint position (must be repeated for each positioner of group)

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupPositionSetpointGet** (int SocketID, char *GroupName, int NbPositioners, double * SetpointPosition)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	char *	Group name
–	NbPositioners	int	Number of positioners in the selected group

Output parameters

–	SetpointPosition	double *	Setpoint position array

Return

Function error code

Visual Basic

Prototype

Long **GroupPositionSetpointGet** (ByVal SocketID As Long, ByVal GroupName As String, ByVal NbPositioners As Long, SetpointPosition As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	Group name
– NbPositioners	long	Number of positioners in the selected group

Output parameters

– SetpointPosition	double	Setpoint position array

Return

– Function error code

Matlab

Prototype

[Error, SetpointPosition] **GroupPositionSetpointGet** (int32 SocketID, cstring GroupName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	Group name

Return

– Error	int32	Function error code
– SetpointPosition	doubletPtr	Setpoint position array

Python

Prototype

[Error, SetpointPosition] **GroupPositionSetpointGet** (integer SocketID, string GroupName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	Group name

Return

– Error	int	Function error code
– SetpointPosition	doubletPtr	Setpoint position array

2.3.3.30 GroupPositionTargetGet

Name

GroupPositionTargetGet – Returns the target position for one or all positioners of the selected group.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid object type (group or positioner): ERR_WRONG_OBJECT_TYPE (-8)
- Valid group name: ERR_GROUP_NAME (-19)
- Valid positioner name: ERR_POSITIONER_NAME (-18)
- Valid output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

Returns the target position for one or all positioners of the selected group.

The target position represents the "end" position after the move.

For instance, during a move from 0 to 10 units, the position values are:

GroupPositionTargetGet => **10.0000**

GroupPositionCurrentGet => **5.0005**

GroupPositionSetpointGet => **5.0055**

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- ERR_WRONG_OBJECT_TYPE (-8)
- SUCCESS (0): no error

TCL

Prototype

GroupPositionTargetGet SocketID GroupName TargetPosition …

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	Group name (maximum size = 250)

Output parameters

– TargetPosition	floating point	Target position (must be repeated for each positioner of group)

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupPositionTargetGet** (int SocketID, char *GroupName, int NbPositioners, double * TargetPosition)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	Group name
– NbPositioners	int	Number of positioners in the selected group

Output parameters

– TargetPosition	double *	Target position array

Return

– Function error code

Visual Basic

Prototype

Long **GroupPositionTargetGet** (ByVal SocketID As Long, ByVal GroupName As String, ByVal NbPositioners As Long, TargetPosition As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	Group name
– NbPositioners	long	Number of positioners in the selected group

Output parameters

– TargetPosition	double	Target position array

Return

Function error code

Matlab

Prototype

[Error, TargetPosition] **GroupPositionTargetGet** (int32 SocketID, cstring GroupName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	Group name

Return

– Error	int32	Function error code
– TargetPosition	doubletPtr	Target position array

Python

Prototype

[Error, TargetPosition] **GroupPositionTargetGet** (integer SocketID, string GroupName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	Group name

Return

– Error	int	Function error code
– TargetPosition	doubletPtr	Target position array

2.3.3.31 GroupStatusGet

Name

GroupStatusGet – Returns the group status code.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid output parameter type: ERR_WRONG_TYPE_INT (-15)
- Valid group name: ERR_GROUP_NAME (-19)

Description

Returns the group status code. The group status codes are listed in the "Group status list" § 0.

The description of the group status code can be retrieved from the "GroupStatusStringGet" function.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

GroupStatusGet SocketID GroupName GroupStatus

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Group name (maximum size = 250)

Output parameters

–	GroupStatus	interger	State of the group.

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupStatusGet** (int SocketID, char *GroupName, int * GroupStatus)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	Group name

Output parameters

– GroupStatus	int *	Status of the group

Return

– Function error code

Visual Basic

Prototype

Long **GroupStatusGet** (ByVal SocketID As Long, ByVal GroupName As String, GroupStatus As Long)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	Group name

Output parameters

– GroupStatus	LongStatus of the group

Return

– Function error code

Matlab

Prototype

[Error, GroupStatus] **GroupStatusGet** (int32 SocketID, cstring GroupName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	Group name

Return

– Error	int32	Function error code
– GroupStatus	int32Status of the group	

Python

Prototype

[Error, GroupStatus] **GroupStatusGet** (integer SocketID, string GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	string	Group name

Return

–	Error	int	Function error code
–	GroupStatus	integer	Status of the group

2.3.3.32 GroupReferencingActionExecute

Name

GroupReferencingActionExecute – Initiates the given action, with the given sensor and parameters.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid positioner name: ERR_POSITIONER_NAME (-18)
- Group status must be "NOT REFERENCED": ERR_NOT_ALLOWED_ACTION (-22)
- Valid action name and sensor name: ERR_WRONG_OBJECT_TYPE (-8)
- Valid input parameter type: ERR_WRONG_TYPE_DOUBLE (-14)
- Input parameter coherence: ERR_PARAMETER_OUT_OF_RANGE (-17)
- For a "LatchOnHighToLowTransition" or "LatchOnLowToHighTransition" or "LatchOnIndex" or "LatchOnIndexAfterSensorHighToLowTransition" or "MoveToPreviouslyLatchedPosition" action.
- Parameter ≤ MaximumVelocity
- Parameter ≠ 0
- Referencing state: ERR_NOT_ALLOWED_ACTION (-22)
- Latch must be done since referencing start for a "MoveToPreviouslyLatchedPosition" action.

Description

Initiates a referencing action for a positioner. A referencing action is defined by a given action name (see action list), with a given sensor name (see sensor list) and parameters. For more detail, see the XPS User's manual referencing section.

Action list	Sensor **to be defined**
LatchOnHighToLowTransition	Yes
LatchOnIndex	None
LatchOnIndexAfterSensorHighToLowTransition	Yes
LatchOnLowToHighTransition	Yes
MoveRelative	None
MoveToPreviouslyLatchedPosition	None
SetPosition	None
SetPositionToHomePreset	None

Sensor list

MechanicalZero

MinusEndOfRun

PlusEndOfRun

None

If a following error occurs during the referencing and motion is in progress, an emergency brake is applied and ERR_FOLLOWING_ERROR (-25) is returned. The group state becomes "NOTINIT".

If the home search time out is reached, ERR_GROUP_HOME_SEARCH_TIMEOUT (-28) is returned. The group state becomes "NOTINIT".

When referencing is done, you can exit the "REFERENCING" state to go in "READY" state with the "GroupReferencingStop" function.

CAUTION

This function must be used with a positioner.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_FOLLOWING_ERROR (-25)
- ERR_GROUP_HOME_SEARCH_TIMEOUT (-28)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

GroupReferencingActionExecute SocketID PositionerName Action Sensor Parameter

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	PositionerName	string	Positioner name (maximum size = 250)
–	ActionString	Referencing action name	
–	SensorString	Referencing sensor name	
–	Parameter	floating point	Referencing parameter (related to the referencing action)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupReferencingActionExecute** (int SocketID, char * PositionerName, char * Action, char * Sensor, double Parameter)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	PositionerName	char *	Positioner name
–	Action	char *	Referencing action name
–	Sensor	char *	Referencing sensor name
–	Parameter	double	Referencing parameter (related to the referencing action)

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **GroupReferencingActionExecute** (ByVal SocketID As Long, String PositionerName, ByVal Action As String, ByVal Sensor As String, ByVal Parameter As Double)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	PositionerName	string	Positioner name
–	ActionString	Referencing action name	
–	SensorString	Referencing sensor name	
–	Parameter	double	Referencing parameter (related to the referencing action)

Output parameters

– None

Return

– Function error code

Matlab

Prototype

[Error] **GroupReferencingActionExecute** (int32 SocketID, cstring PositionerName, cstring Action, cstring Sensor, double Parameter)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– PositionerName	cstring	Positioner name
– Action	cstring	Referencing action name
– Sensor	cstring	Referencing sensor name
– Parameter	double	Referencing parameter (related to the referencing action)

Return

– Error	int32	Function error code

Python

Prototype

[Error] **GroupReferencingActionExecute** (integer SocketID, string PositionerName, string Action, string Sensor, double Parameter)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– PositionerName	string	Positioner name
– ActionString	Referencing action name	
– SensorString	Referencing sensor name	
– Parameter	double	Referencing parameter (related to the referencing action)

Return

– Error	int	Function error code

2.3.3.33 GroupReferencingStart

Name

GroupReferencingStart – Starts the referencing mode.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid group name: ERR_GROUP_NAME (-19)
- Group status must be "NOT REFERENCED": ERR_NOT_ALLOWED_ACTION (-22)

Description

Starts the referencing mode and sets the group status as "REFERENCING".

To use this function, the selected group must be in "NOT REFERENCED" state, else ERR_NOT_ALLOWED_ACTION (-22) is returned.

To stop the referencing mode and to go in the "READY" state, call the "GroupReferencingStop" function.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

GroupReferencingStart SocketID GroupName

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Group name

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupReferencingStart** (int SocketID, char * GroupName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char * Group name	

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **GroupReferencingStart** (ByVal SocketID As Long, ByVal GroupName As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	Group name

Output parameters

– None

Return

Function error code

Matlab

Prototype

[Error] **GroupReferencingStart** (int32 SocketID, cstring GroupName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	Group name

Return

– Error	int32	Function error code

Python

Prototype

[Error] **GroupReferencingStart** (integer SocketID, string GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	string	Group name

Return

–	Error	int	Function error code

2.3.3.34 GroupReferencingStop

Name

GroupReferencingStop – Stops the referencing mode.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid group name: ERR_GROUP_NAME (-19)
- Group status must be "REFERENCING": ERR_NOT_ALLOWED_ACTION (-22)

Description

Stops the referencing mode and sets the group state to "READY".

To use this function, the selected group must be in "REFERENCING" state, else ERR_NOT_ALLOWED_ACTION (-22) is returned.

The travel limits are checked before stopping referencing mode. The ERR_TRAVEL_LIMITS (-35) is returned if the profiler position is out of range of the software travel limits and the group stays in the "REFERENCING" state.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_TRAVEL_LIMITS (-35)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

GroupReferencingStop SocketID GroupName

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Group name

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupReferencingStop** (int SocketID, char * GroupName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char * Group name	

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **GroupReferencingStop** (ByVal SocketID As Long, ByVal GroupName As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	Group name

Output parameters

– None

Return

– Function error code

Matlab

Prototype

[Error] **GroupReferencingStop** (int32 SocketID, cstring GroupName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	Group name

Return

– Error	int32	Function error code

Python

Prototype

[Error] **GroupReferencingStop** (integer SocketID, string GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	string	Group name

Return

–	Error	int	Function error code

2.3.3.35 GroupVelocityCurrentGet

Name

GroupVelocityCurrentGet – Returns the current velocity for one or all positioners of the selected group.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid object type (group or positioner): ERR_WRONG_OBJECT_TYPE (-8)
- Valid group name: ERR_GROUP_NAME (-19)
- Valid positioner name: ERR_POSITIONER_NAME (-18)
- Valid output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

Returns the current velocity for one or all positioners of the selected group.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

GroupVelocityCurrentGet SocketID GroupName CurrentVelocity …

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Group name (maximum size = 250)

Output parameters

–	CurrentPosition	floating point	Current Velocity (must be repeated for each positioner of group)

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupVelocityCurrentGet** (int SocketID, char *GroupName, int NbPositioners, double * CurrentVelocity)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	Group name
– NbPositioners	int	Number of positioners in the selected group

Output parameters

– CurrentVelocity	double *	Current Velocity array

Return

– Function error code

Visual Basic

Prototype

Long **GroupVelocityCurrentGet** (ByVal SocketID As Long, ByVal GroupName As String, ByVal NbPositioners As Long, CurrentVelocity As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	Group name
– NbPositioners	long	Number of positioners in the selected group

Output parameters

– CurrentVelocity	double	Current Velocity array

Return

– Function error code

Matlab

Prototype

[Error, CurrentVelocity] **GroupVelocityCurrentGet** (int32 SocketID, cstring GroupName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	Group name

Return

– Error	int32	Function error code
– CurrentVelocity	doubletPtr	Current Velocity array

Python

Prototype

[Error, CurrentVelocity] **GroupVelocityCurrentGet** (integer SocketID, string GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	string	Group name

Return

–	Error	int	Function error code
–	CurrentVelocity	doubletPtr	Current Velocity array

2.4 SingleAxis Group

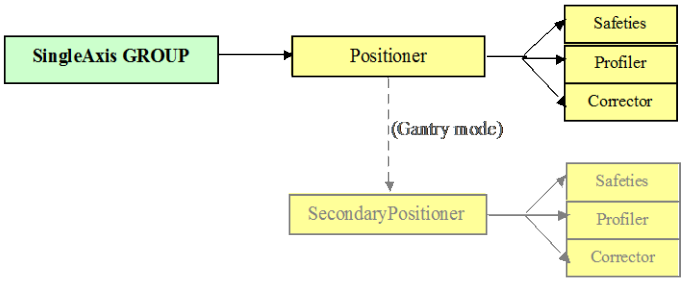
2.4.1 Description

The SingleAxis is composed of one single positioner for the execution of motion commands.

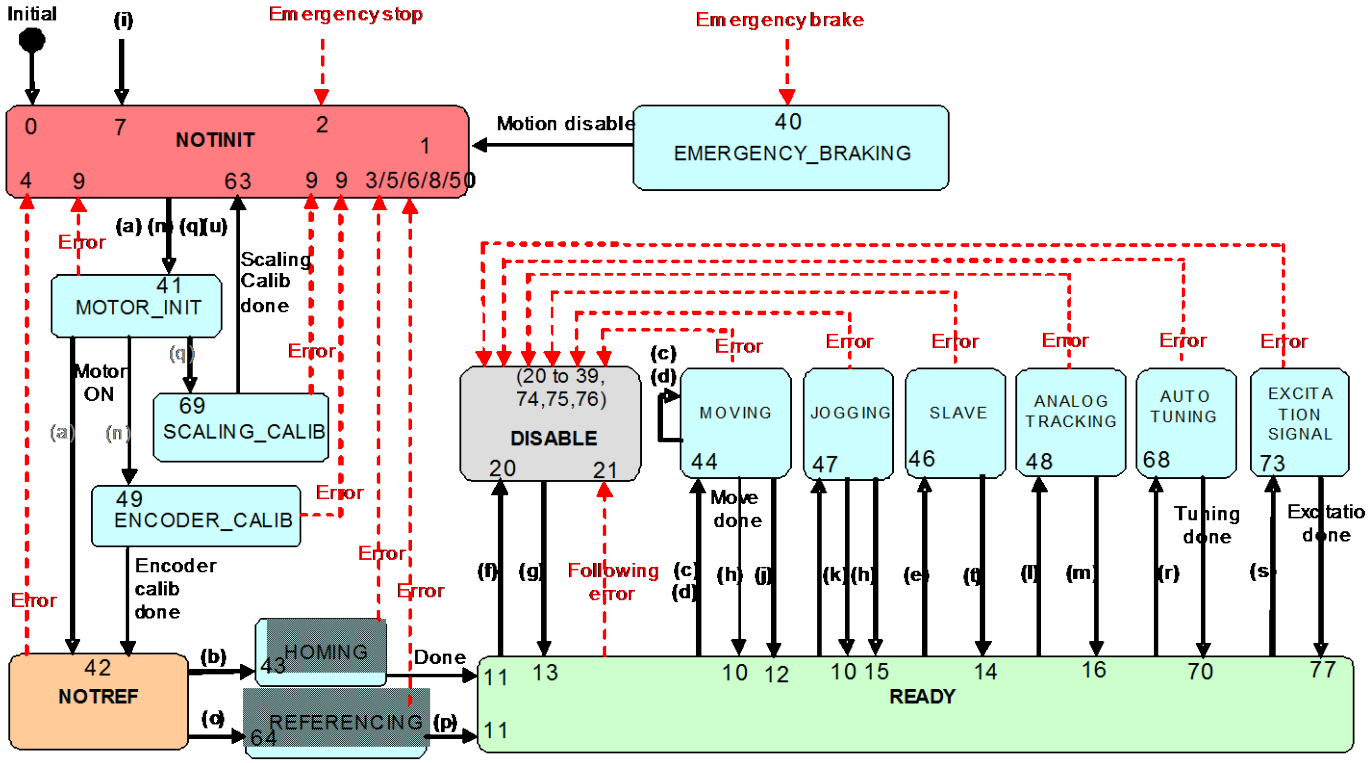
A SingleAxis group can be use in GANTRY mode (dual positioner).

The XPS controller can handle several SingleAxis objects (1 to 8).

There is no relation between SingleAxis objects and other objects handled by the controller.



2.4.2 State Diagram



Called functions:

(a) GroupInitialize

(b) GroupHomeSearch

(c) GroupMoveAbsolute

(d) GroupMoveRelative

(e) GroupSlaveModeEnable

(f) GroupMotionDisable

(g) GroupMotionEnable

(h) GroupMoveAbort

(i) GroupKill or KillAll

(j) GroupJogModeEnable

(k) GroupJogModeDisable

(l) GroupAnalogTrackingModeEnable

(m) GroupAnalogTrackingModeDisable

(n) GroupInitializeWithEncoderCalibration

(o) GroupReferencingStart

(p) GroupReferencingStop

(q) PositionerAccelerationAutoScaling

(r) PositionerCorrectorAutoTuning

(s) PositionerExcitationSignalSet / PositionerPreCorrectorExcitationSignalSet

(t) GroupSlaveModeDisable

(u) GroupInitializeNoEncoderReset

2.4.3 Specific Function Description

2.4.3.1 SingleAxisSlaveModeDisable

Name

SingleAxisSlaveModeDisable – Disables the slave-master mode.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Group state must be "SLAVE": ERR_NOT_ALLOWED_ACTION (-22)
- Check the group name: ERR_GROUP_NAME (-19)
- Check the positioner name: ERR_POSITIONER_NAME (-18)
- Check the group type (must be a SingleAxis group): ERR_WRONG_OBJECT_TYPE (-8)

Description

This function disables the master-slave mode. If a motion is in progress then it is aborted.

To use this function, the group state must be SLAVE (46). If it's not the case then ERR_NOT_ALLOWED_ACTION (-22) is returned.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

SingleAxisSlaveModeDisable $SocketID $GroupName

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	SingleAxis group name (maximum size = 250)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **SingleAxisSlaveModeDisable** (int SocketID, char *GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	char *	SingleAxis group name

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **SingleAxisSlaveModeDisable** (ByVal SocketID As Long, ByVal GroupName As String)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	SingleAxis group name

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **SingleAxisSlaveModeDisable** (int32 SocketID, cstring GroupName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	SingleAxis group name

Return

– Function error code

Python

Prototype

integer **SingleAxisSlaveModeDisable** (integer SocketID, string GroupName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	SingleAxis group name

Return

– Function error code

2.4.3.2 SingleAxisSlaveModeEnable

Name

SingleAxisSlaveModeEnable – Enables the slave-master mode.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Group state must be "READY": ERR_NOT_ALLOWED_ACTION (-22)
- Check the group name: ERR_GROUP_NAME (-19)
- Check the positioner name: ERR_POSITIONER_NAME (-18)
- Check the group type (must be a SingleAxis group): ERR_WRONG_OBJECT_TYPE (-8)
- Check the slave parameters (must be configured): ERR_SLAVE_CONFIGURATION (-41)

Description

This function enables the master-slave mode only if the slave group is in "READY" state. In this mode the slave must be defined as a SingleAxis group and the master can be a positioner from any group.

To use this function, the SingleAxis group must be in the READY state. If it's not the case then ERR_NOT_ALLOWED_ACTION (-22) is returned.

To use this function, the master positioner and the slave ratio must be configured using the "SingleAxisSlaveParametersSet" function. If it's not the case then ERR_SLAVE_CONFIGURATION (-41) is returned.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_POSITIONER_NAME (-18)
- ERR_SLAVE_CONFIGURATION (-41)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

SingleAxisSlaveModeEnable $SocketID $GroupName

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	SingleAxis group name (maximum size = 250)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **SingleAxisSlaveModeEnable** (int SocketID, char *GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	char *	SingleAxis group name

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **SingleAxisSlaveModeEnable** (ByVal SocketID As Long, ByVal GroupName As String)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	SingleAxis group name

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **SingleAxisSlaveModeEnable** (int32 SocketID, cstring GroupName)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	cstring	SingleAxis group name

Return

– Function error code

Python

Prototype

integer **SingleAxisSlaveModeEnable** (integer SocketID, string GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	string	SingleAxis group name

Return

– Function error code

2.4.3.3 SingleAxisSlaveParametersGet

Name

SingleAxisSlaveParametersGet – Returns the slave parameters.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner name: ERR_POSITIONER_NAME (-18)
- Check the group type (must be a SingleAxis group): ERR_WRONG_OBJECT_TYPE (-8)
- Valid output parameter type: ERR_WRONG_TYPE_DOUBLE (-14), ERR_WRONG_TYPE_CHAR (-13)
- Check the slave parameters (must be configured): ERR_NOT_ALLOWED_ACTION (-22)

Description

This function returns the slave parameters: the master positioner name and the master-slave ratio.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

SingleAxisSlaveParametersGet $SocketID $GroupName MasterPositionerName Ratio

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	Group name (maximum size = 250)

Output parameters

– MasterPositionerName	string	Master positioner name from any group
– Ratio	double	Gear ratio between the master and the slave

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **SingleAxisSlaveParametersGet** (int SocketID, char *GroupName, int NbPositioners, char * MasterPositionerName , double * Ratio)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	Group name

Output parameters

– MasterPositionerName	char *	Master positioner name from any group
– Ratio	double *	Gear ratio between the master and the slave

Return

– Function error code

Visual Basic

Prototype

Long **SingleAxisSlaveParametersGet** (ByVal SocketID As Long, ByVal GroupName As String, ByVal MasterPositionerName As String, Ratio As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	Group name

Output parameters

– MasterPositionerName	string	Master positioner name from any group
– Ratio	double	Gear ratio between the master and the slave

Return

– Function error code

Matlab

Prototype

[Error, MasterPositionerName, Ratio] **SingleAxisSlaveParametersGet** (int32 SocketID, cstring GroupName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	Group name

Return

– Error	int32	Function error code
– MasterPositionerName	cstring	Master positioner name from any group
– Ratio	double	Gear ratio between the master and the slave

Python

Prototype

[Error, MasterPositionerName, Ratio] **SingleAxisSlaveParametersGet** (integer SocketID, string GroupName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	Group name

Return

– Error	int	Function error code
– MasterPositionerName	string	Master positioner name from any group
– Ratio	double	Gear ratio between the master and the slave

2.4.3.4 SingleAxisSlaveParametersSet

Name

SingleAxisSlaveParametersSet – Sets the slave parameters.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the master positioner name: ERR_POSITIONER_NAME (-18)
- Check the master group type: ERR_WRONG_OBJECT_TYPE (-8)
- Check input parameter type: ERR_WRONG_TYPE_DOUBLE (-14), ERR_WRONG_TYPE_CHAR (-13)
- Check the slave parameters (must be configured): ERR_NOT_ALLOWED_ACTION (-22)
- Check the base velocity value (must be null): ERR_BASE_VELOCITY (-48)
- Check the ratio value (Ratio > 0): ERR_PARAMETER_OUT_OF_RANGE (-17)

Description

This function configures the slave parameters only for a SingleAxis group.

The slave is a copy of the master and a ratio can be applied: Slave = Ratio * Master.

The slave-master mode is activated only after the call of "SingleAxisSlaveModeEnable" function.

The master can be a positioner from any group, except from a spindle group. If the master group is a spindle then ERR_NOT_ALLOWED_ACTION (-22) is returned. The master positioner must be different from the slave positioner else ERR_WRONG_OBJECT_TYPE (-8) is returned.

NOTE

After an emergency stop, the master group and the slave group are in "NOTINIT" status. To restart a master-slave relation, the slave group(s) must be reinitialized before the master group.

Error codes

- ERR_BASE_VELOCITY (-48)
- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

SingleAxisSlaveParametersSet $SocketID $GroupName $MasterPositionerName $Ratio

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Group name (maximum size = 250)
–	MasterPositionerName	string	Master positioner name from any group
–	Ratio	double	Gear ratio between the master and the slave

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **SingleAxisSlaveParametersSet** (int SocketID, char *GroupName, int NbPositioners, char * MasterPositionerName , double Ratio)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	Group name
– MasterPositionerName	char *	Master positioner name from any group
– Ratio	double	Gear ratio between the master and the slave

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **SingleAxisSlaveParametersSet** (ByVal SocketID As Long, ByVal GroupName As String, ByVal MasterPositionerName As String, ByVal Ratio As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	Group name
– MasterPositionerName	string	Master positioner name from any group
– Ratio	double	Gear ratio between the master and the slave

Output parameters

– None

Return

– Function error code

Matlab

Prototype

[Error] **SingleAxisSlaveParametersSet** (int32 SocketID, cstring GroupName, cstring MasterPositionerName, double Ratio)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	Group name
– MasterPositionerName	cstring	Master positioner name from any group
– Ratio	double	Gear ratio between the master and the slave

Return

– Error	int32	Function error code

Python

Prototype

[Error] **SingleAxisSlaveParametersSet** (integer SocketID, string GroupName, string MasterPositionerName, double Ratio)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	Group name
– MasterPositionerName	string	Master positioner name from any group
– Ratio	double	Gear ratio between the master and the slave

Return

– Error	int	Function error code

2.4.4 Configuration Files

An example of a SingleAxis group (named "MySingleAxis") is in the system.ini file below. The "MySingleAxis" group is comprised of a positioner named "MyPositioner".

The positioner "MyPositioner" uses the parameters of "MYSTAGE" from the stages.ini file and is connected to the plug 1 of the XPS controller. The Positioner "MyPositioner" has a secondary positioner in gantry mode. This secondary positioner uses the parameters of "MYSTAGE" from the stages.ini file and is connected to the plug 2 of the XPS controller.

NOTE

PIDBaseFilter parameter in system.ini file is to be configured only when using the XPS-Qn Precision Platform controller.

System.ini file:

```
[GROUPS]
SingleAxisInUse = MySingleAxis

[MySingleAxis]      ; Configuration of "MySingleAxis" SingleAxis group
PositionerInUse = MyPositioner

[MySingleAxis. MyPositioner]
PlugNumber = 1
StageName = MYSTAGE
; --- Time flasher
TimeFlasherBaseFrequency =      ; default value 40e6, must be between 39.5e6 and
                                  40.5e6 Hz
;--- PIDBase filter (take into account the base movements, effective only with
PIDFFAcceleration corrector) To be added only when using XPS-Q8 Precision
Platform controller
PIDBaseFilter =                 ; Enabled or Disabled.
MovingMass =                    ; If PIDBaseFilter = Enabled
StaticMass =                    ; If PIDBaseFilter = Enabled
Viscosity =                     ; If PIDBaseFilter = Enabled
Stiffness =                     ; If PIDBaseFilter = Enabled
;--- If Gantry (secondary positioner)
SecondaryPositionerGantry =             ; Enabled or Disabled
SecondaryPlugNumber = 2                 ; If SecondaryPositionerGantry = Enabled
SecondaryStageName = MYSTAGE            ;If SecondaryPositionerGantry = Enabled
SecondaryPositionerGantryEndReferencingPosition = 0         ;If
                                          SecondaryPositionerGantry = Enabled
SecondaryPositionerGantryEndReferencingTolerance =1         ;If
                                          SecondaryPositionerGantry = Enabled
SecondaryPositionerGantryOffsetAfterInitialization =0       ;If
                                          SecondaryPositionerGantry = Enabled
SecondaryPositionerGantryMotorEnableDelay =         ;seconds,if
                                          SecondaryPositionerGantry = Enabled
```

Stages.ini file:

[MYSTAGE]

MYSTAGE configuration => See § "Positioner: Configuration filess"

2.5 SingleAxisWithClamping Group

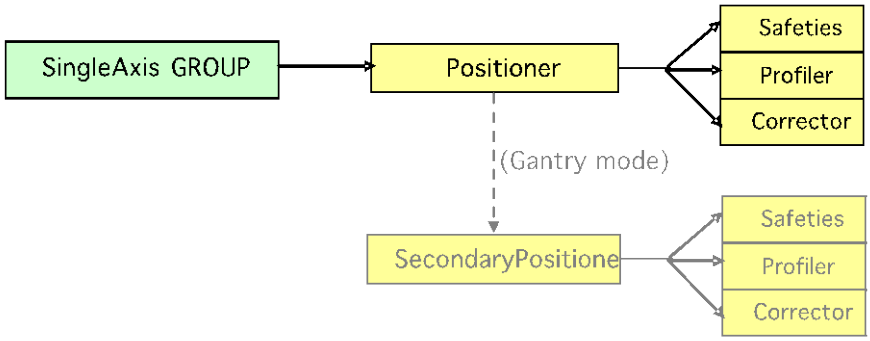
2.5.1 Description

The SingleAxisWithClamping is composed of one single positioner that allows execution of motion commands.

In general it is similar to SingleAxis group, the only difference is the support of clamping motions: axis is clamped before moves, unclamped during moves and reclamped at stop.

A SingleAxisWithClamping group can NOT be used in GANTRY mode (*secondary positioner is impossible*).

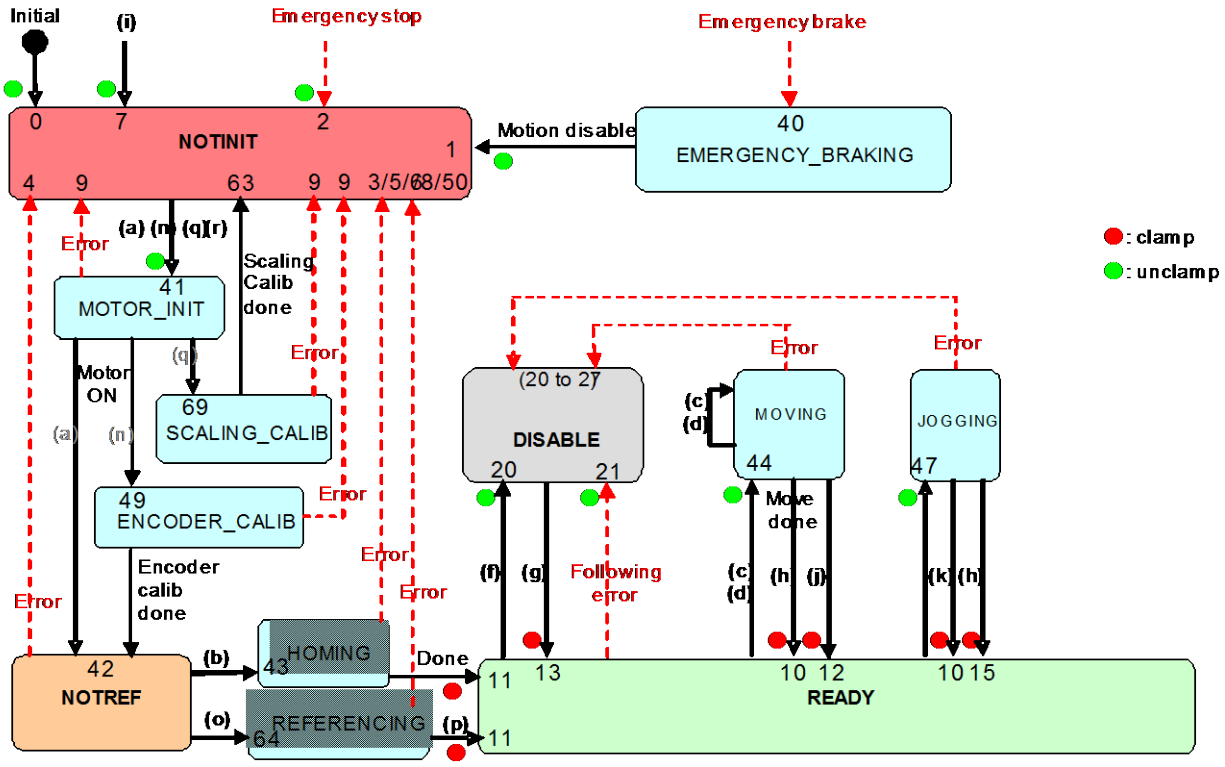
The XPS controller can handle several SingleAxisWithClamping objects (1 to 8).



NOTE

This group can be used only with the XPS-Qn Precision Platform controller.

2.5.2 State Diagram



Called functions:

(a) GroupInitialize
(b) GroupHomeSearch
(c) GroupMoveAbsolute
(d) GroupMoveRelative
(f) GroupMotionDisable
(g) GroupMotionEnable
(h) GroupMoveAbort
(i) GroupKill or KillAll
(j) GroupJogModeEnable
(k) GroupJogModeDisable
(l) GroupAnalogTrackingModeEnable
(m) GroupAnalogTrackingModeDisable
(n) GroupInitializeWithEncoderCalibration
(o) GroupReferencingStart
(p) GroupReferencingStop
(q) PositionerAccelerationAutoScaling
(r) GroupInitializeNoEncoderReset

2.5.3 Configuration Files

An example of a SingleAxisWithClamping group (named "MySingleAxisWithClamping") is shown in the system.ini file below. The "MySingleAxisWithClamping" group is composed of a positioner named "MyPositioner".

The positioner "MyPositioner" uses the parameters of "MYSTAGE" from the stages.ini file and is connected to plug 1 of the XPS controller. The Positioner "MyPositioner" has a secondary positioner (option to build a "gantry" position). This secondary positioner uses the parameters of "MYSTAGE" from the stages.ini file and is connected to plug 2 of the XPS controller.

NOTE

Configuration below is applicable only to the XPS-Qn Precision Platform controller.

System.ini file:

[GROUPS]

SingleAxisWithClampingInUse = **MySingleAxisWithClamping**

[MySingleAxisWithClamping] *; Configuration of "MySingleAxisWithClamping" SingleAxisWithClamping group*

PositionerInUse = **MyPositioner**

ClampInputBit = NoFeedback *; GPIOx.DIy (x=1,y=1..8 or x=2,y=1..6 or x=3,y=1..6 or x=4,y=1..16) or NoFeedback (sensor simulation)*

ClampInputMode = NonInverted *; NonInverted (clamped if input=1, unclamped if input=0) or Inverted (clamped if input=0, unclamped if input=1), parameter not needed if ClampInputBit=NoFeedback*

ClampOutputBit = GPIO4.DO2 *; GPIOx.DOy (x=1,y=1..8 or x=3,y=1..6 or x=4,y=1..16)*

ClampOutputMode = Inverted *; NonInverted (output 1 to clamp, output 0 to unclamp) or Inverted (output 0 to clamp, output 1 to unclamp)*

ClampingActivatingTime = 0.03 *; seconds*

ClampingActivatingTimeout = 0.5 *; seconds*

ClampingReleaseTime = 0.12 *; seconds*

ClampingReleaseTimeout = 0.5 *; seconds*

ClampingPositionOffset = 0 *; units*

[MySingleAxisWithClamping. MyPositioner]

PlugNumber = 1

StageName = MYSTAGE

; --- Time flasher

TimeFlasherBaseFrequency = *; default value 40e6, must be between 39.5e6 and 40.5e6 Hz*

;--- PIDBase filter (take into account the base movements, effective only with PIDFFAcceleration corrector)

PIDBaseFilter = *; Enabled or Disabled*

MovingMass = *; If PIDBaseFilter = Enabled*

```
StaticMass =                    ; If PIDBaseFilter = Enabled
Viscosity =                     ; If PIDBaseFilter = Enabled
Stiffness =                     ; If PIDBaseFilter = Enabled
```

Stages.ini file:

```
[MYSTAGE]
MYSTAGE configuration => See § "Positioner: Configuration filess"
```

–

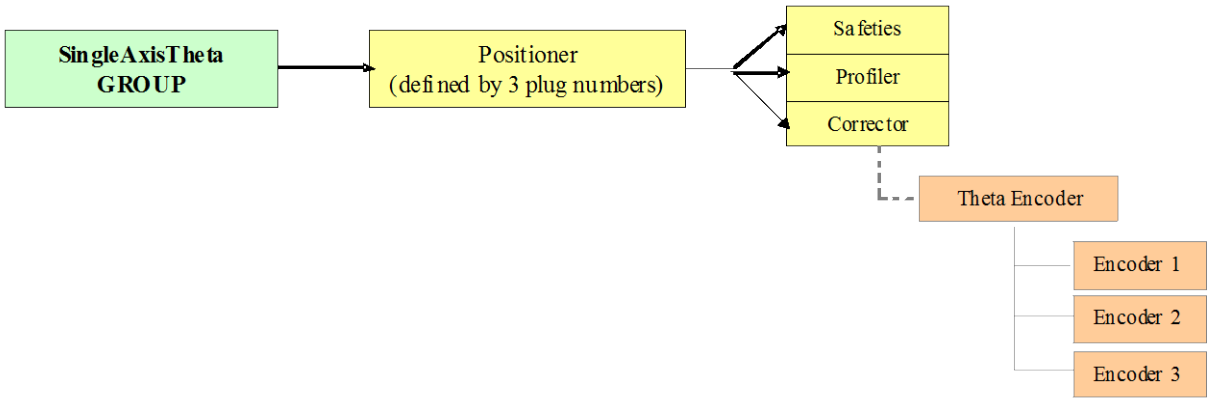
2.6 SingleAxisTheta Group

2.6.1 Description

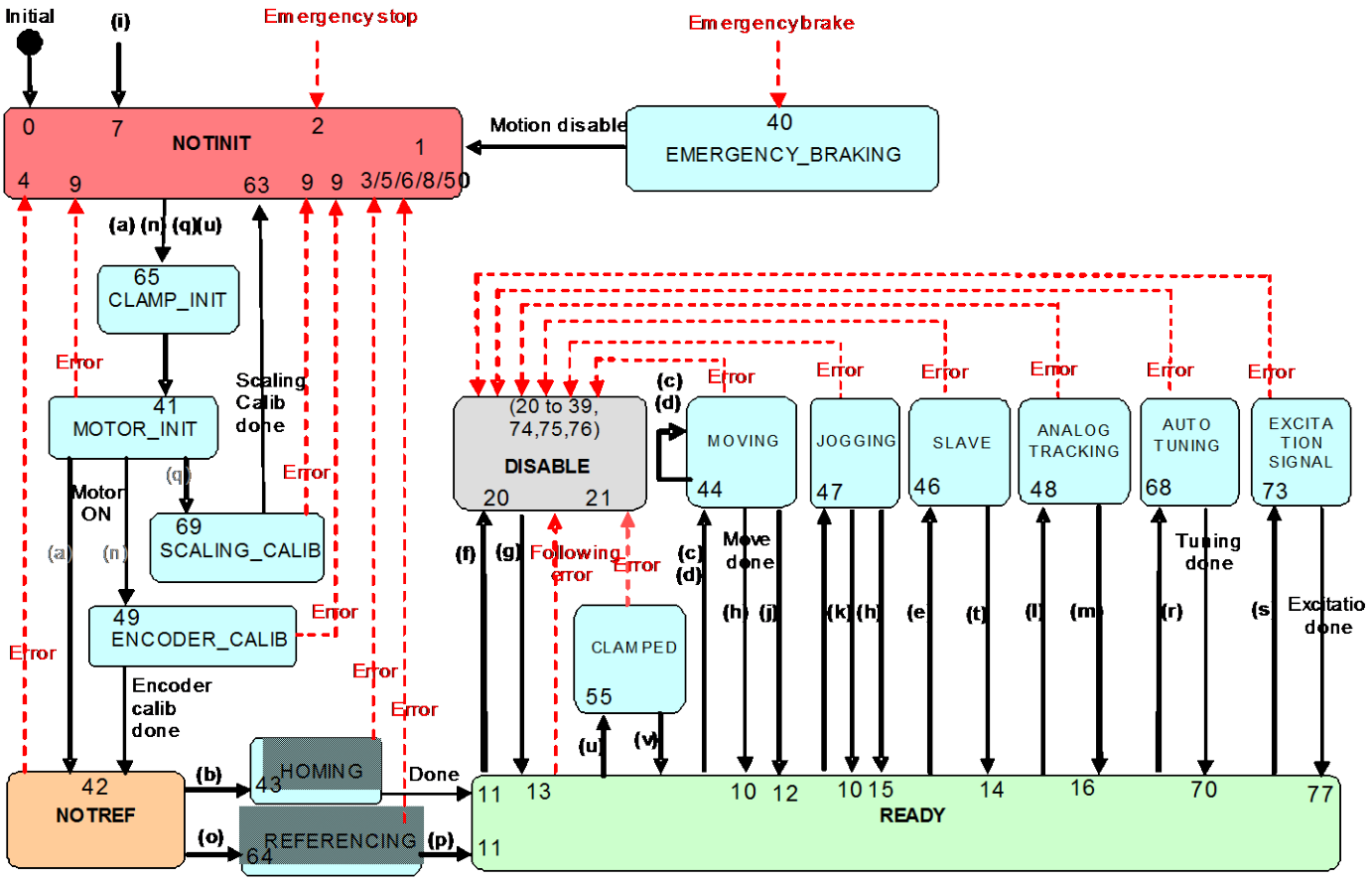
The SingleAxisTheta is composed of one single positioner object with three encoders (Theta encoder) for the execution of motion commands. It includes a "Yaw" mapping and a "Theta" correction on an XY group.

NOTE

This group can be used only with the XPS-Qn Precision Platform controller.



2.6.2 State Diagram



Called functions:

(a) GroupInitialize

(b) GroupHomeSearch

(c) GroupMoveAbsolute

(d) GroupMoveRelative

(e) GroupSlaveModeEnable

(f) GroupMotionDisable

(g) GroupMotionEnable

(h) GroupMoveAbort

(i) GroupKill or KillAll

(j) GroupJogModeEnable

(k) GroupJogModeDisable

(l) GroupAnalogTrackingModeEnable

(m) GroupAnalogTrackingModeDisable

(n) GroupInitializeWithEncoderCalibration

(o) GroupReferencingStart

(p) GroupReferencingStop

(q) PositionerAccelerationAutoScaling

(r) PositionerCorrectorAutoTuning

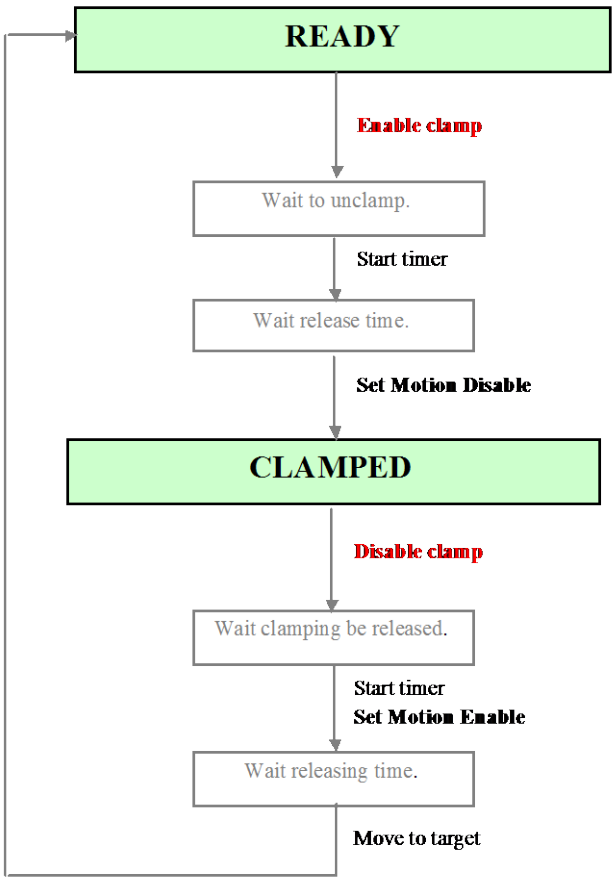
(s) PositionerExcitationSignalSet / PositionerPreCorrectorExcitationSignalSet

(t) GroupSlaveModeDisable

(u) GroupInitializeNoEncoderReset

2.6.3 Clamping Sequence

2.6.3.1 Clamping state diagram



NOTES

In case of recoverable errors (go to DISABLE state or go to READY state), the clamp is unclamped.

In case of fatal errors (go to NOT INIT state), the clamp stays in the current state (clamped).

2.6.3.2 Clamping configuration

```
; Clamping
ClampRestType = Unclamped       ; Clamped or Unclamped
ClampActivatingTime = 0.03; seconds
ClampActivatingTimeOut = 0.5    ; seconds
ClampReleaseTime = 0.12  ; seconds
ClampReleaseTimeOut = 0.5       ; seconds
```

2.6.4 Specific Functions Description

2.6.4.1 SingleAxisThetaClampDisable

Name

SingleAxisThetaClampDisable – unclamp a SingleAxisTheta group.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid object type (group): ERR_WRONG_OBJECT_TYPE (-8)
- Valid group name: ERR_GROUP_NAME (-19)
- Valid positioner name: ERR_POSITIONER_NAME (-18)
- Valid output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function unclamps the selected SingleAxisTheta group

The group must be in the CLAMPED state. If unclamping is successful, the group is unclamped and the group state becomes "READY".

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

SingleAxisThetaClampDisable SocketID GroupName

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	SingleAxisTheta group name (maximum size = 250)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **SingleAxisThetaClampDisable** (int SocketID, char *GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	char *	SingleAxisTheta group name

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **SingleAxisThetaClampDisable** (ByVal SocketID As Long, ByVal GroupName As String)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	SingleAxisTheta group name

Output parameters

– None

Return

Function error code

Matlab

Prototype

[Error] **SingleAxisThetaClampDisable** (int32 SocketID, cstring GroupName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	SingleAxisTheta group name

Return

– Error	int32	Function error code

Python

Prototype

[Error] **SingleAxisThetaClampDisable** (integer SocketID, string GroupName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	SingleAxisTheta group name

Return

– Error	int	Function error code

2.6.4.2 SingleAxisThetaClampEnable

Name

SingleAxisThetaClampEnable – clamps a SingleAxisTheta group.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid object type (group): ERR_WRONG_OBJECT_TYPE (-8)
- Valid group name: ERR_GROUP_NAME (-19)
- Valid positioner name: ERR_POSITIONER_NAME (-18)
- Valid output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function clamps the selected SingleAxisTheta group.

The group must be in the READY state. If clamping is successful, the group is clamped and the group state becomes "CLAMPED".

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

SingleAxisThetaClampEnable SocketID GroupName

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	SingleAxisTheta group name (maximum size = 250)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **SingleAxisThetaClampEnable** (int SocketID, char *GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	char *	SingleAxisTheta group name

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **SingleAxisThetaClampEnable** (ByVal SocketID As Long, ByVal GroupName As String)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	SingleAxisTheta group name

Output parameters

– None

Return

– Function error code

Matlab

Prototype

[Error] **SingleAxisThetaClampEnable** (int32 SocketID, cstring GroupName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	SingleAxisTheta group name

Return

– Error	int32	Function error code

Python

Prototype

[Error] **SingleAxisThetaClampEnable** (integer SocketID, string GroupName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	SingleAxisTheta group name

Return

– Error	int	Function error code

2.6.5 Configuration Files

Example of configuration file for a "SingleAxisTheta" group named "THETA" composed of one only positioner named "MyPOSITIONER".

NOTE

Configuration below relates only XPS-Q8 Precision Platform controller.

System.ini file:

```
[GROUPS]
SingleAxisThetaInUse = THETA

[THETA]     ; THETA SingleAxisTheta group configuration
PositionerInUse = MyPOSITIONER

; Theta correction on XY
ThetaCorrectionXYGroupName =
ThetaCorrectionLowPassCutOffFrequency = 20  ; Hz

; Yaw mapping
YawMappingXYGroupName =
YawMappingToThetaFileName =
YawMappingToThetaLineNumber =
YawMappingToThetaColumnNumber =
YawMappingToThetaMaxPositionError =
YawMappingToXFileName =
YawMappingToXLineNumber =
YawMappingToXColumnNumber =
YawMappingToXMaxPositionError =
YawMappingToYFileName =
YawMappingToYMaxPositionError =
YawMappingToYLineNumber =
YawMappingToYColumnNumber =


; Clamping
ClampRestType = Unclamped          ; Clamped or Unclamped
ClampActivatingTime = 0.03         ; seconds
ClampActivatingTimeOut = 0.5       ; seconds
ClampReleaseTime = 0.12            ; seconds
ClampReleaseTimeOut = 0.5          ; seconds

[THETA.MyPOSITIONER]
PlugNumber = 1,2,3                 ; Theta with 3 encoders
```

```
StageName = MySTAGE
; --- Time flasher
TimeFlasherBaseFrequency =      ; default value 40e6, must be between 39.5e6 and
                                  40.5e6 Hz
;--- PIDBase filter (take into account the base movements, effective only with
PIDFFAcceleration corrector)
PIDBaseFilter =                 ;  Enabled or Disabled
MovingMass =                    ;  If PIDBaseFilter = Enabled
StaticMass =                    ;  If PIDBaseFilter = Enabled
Viscosity =                     ;  If PIDBaseFilter = Enabled
Stiffness =                     ;  If PIDBaseFilter = Enabled
```

Stiffness = Stages.ini file:

```
[MySTAGE]
;--- Position encoder interface parameters
EncoderType = AquadBTheta       ; AquadBTheta or AnalogInterpolatedTheta

;--- THETA encoder features
EncoderRadius = 150e-6          ; units XY * 10-6
MaximumEncoderCorrectionX = 10000
MaximumEncoderCorrectionY = 10000

[…]
```

For more information about MYSTAGE configuration => See § "Positioner: Configuration filess"

NOTE

EncoderType must be "AquadBTheta" or "AnalogInterpolatedTheta".

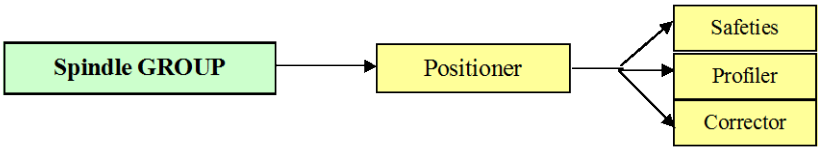
2.7 Spindle Group

2.7.1 Description

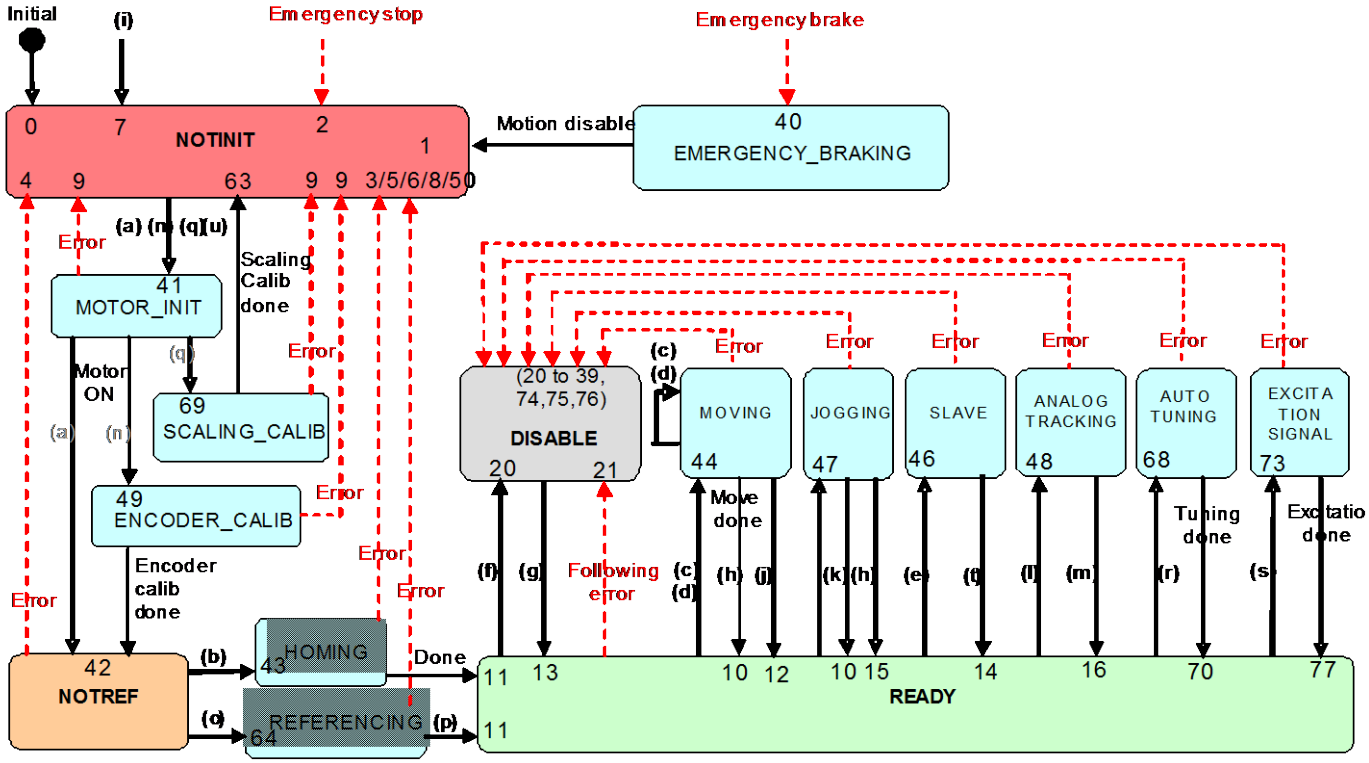
A Spindle group is very similar to the SingleAxis group. It's composed of only one positioner with one main difference, it does not handle software or hardware end of runs. Therefore, it is allowed to spin indefinitely in any direction. SingleAxis group motion commands are still allowed (except jog, which is replaced by spin).

The controller can handle several Spindle groups.

There is no relation between Spindle groups and other objects handled by the controller.



2.7.2 State Diagram



Called functions:

(a) GroupInitialize

(b) GroupHomeSearch

(c) GroupMoveAbsolute

(d) GroupMoveRelative

(e) GroupSlaveModeEnable

(f) GroupMotionDisable

(g) GroupMotionEnable

(h) GroupMoveAbort

(i) GroupKill or KillAll

(j) GroupJogModeEnable

(k) GroupJogModeDisable

(l) GroupAnalogTrackingModeEnable

(m) GroupAnalogTrackingModeDisable

(n) GroupInitializeWithEncoderCalibration

(o) GroupReferencingStart

(p) GroupReferencingStop

(q) PositionerAccelerationAutoScaling

(r) PositionerCorrectorAutoTuning

(s) PositionerExcitationSignalSet / PositionerPreCorrectorExcitationSignalSet

(t) GroupSlaveModeDisable

(u) GroupInitializeNoEncoderReset

2.7.3 Specific Function Description

2.7.3.1 GroupSpinCurrentGet

Name

GroupSpinCurrentGet – Returns the spin mode parameters.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner name: ERR_POSITIONER_NAME (-18)
- Check the group type (must be a Spindle group): ERR_WRONG_OBJECT_TYPE (-8)
- Valid output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function returns the current (or actual) velocity and acceleration used by the SPIN mode.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

GroupSpinCurrentGet $SocketID $GroupName Velocity Acceleration

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	Spindle group name (maximum size = 250)

Output parameters

– Velocity	double	Velocity (units/s)
– Acceleration	double	Acceleration (units/s²)

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupSpinCurrentGet** (int SocketID, char *GroupName, double * Velocity, double * Acceleration)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	Spindle group name

Output parameters

– Velocity	double *	Velocity (units/s)
– Acceleration	double *	Acceleration (units/s²)

Return

– Function error code

Visual Basic

Prototype

Long **GroupSpinCurrentGet** (ByVal SocketID As Long, ByVal GroupName As String, Velocity As Double, Acceleration As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	Spindle group name

Output parameters

– Velocity	double	Velocity (units/s)
– Acceleration	double	Acceleration (units/s²)

Return

– Function error code

Matlab

Prototype

[Error, Velocity, Acceleration] **GroupSpinCurrentGet** (int32 SocketID, cstring GroupName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	Spindle group name

Return

– Error	int32	Function error code
– Velocity	double	Velocity (units/s)
– Acceleration	double	Acceleration (units/s²)

Python

Prototype

[Error, Velocity, Acceleration] **GroupSpinCurrentGet** (integer SocketID, string GroupName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	Spindle group name

Return

– Error	int	Function error code
– Velocity	double	Velocity (units/s)
– Acceleration	double	Acceleration (units/s²)

2.7.3.2 GroupSpinModeStop

Name

GroupSpinModeStop – Stops the motion of the spindle group.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner name (must be a group name): ERR_POSITIONER_NAME (-18)
- Check the group type (must be a Spindle group): ERR_WRONG_OBJECT_TYPE (-8)

Description

This function stops the motion of a spindle group and sets the group state to READY.

To use this function, the group must be in SPINNING state else ERR_NOT_ALLOWED_ACTION (-22) is returned.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

GroupSpinModeStop $SocketID $GroupName

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Spindle group name (maximum size = 250)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupSpinModeStop** (int SocketID, char *GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	char *	Spindle group name

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **GroupSpinModeStop** (ByVal SocketID As Long, ByVal GroupName As String)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Spindle group name

Output parameters

– None

Return

– Function error code

Matlab

Prototype

[Error] **GroupSpinModeStop** (int32 SocketID, cstring GroupName)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	cstring	Spindle group name

Return

–	Error	int32	Function error code

Python

Prototype

[Error] **GroupSpinModeStop** (integer SocketID, string GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	string	Spindle group name

Return

–	Error	int	Function error code

2.7.3.3 GroupSpinParametersGet

Name

GroupSpinParametersGet – Returns the spin profiler parameters.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the group type (must be a spindle group): ERR_WRONG_OBJECT_TYPE (-8)
- Check the function (must be a spindle function): ERR_POSITIONER_NAME (-18)
- Valid output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function returns the "Setpoint" (theoretical) velocity and acceleration used in the SPIN mode.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

GroupSpinParametersGet $SocketID $GroupName Velocity Acceleration

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Spindle group name (maximum size = 250)

Output parameters

–	Velocity	double	Setpoint Velocity (units/s)
–	Acceleration	double	Setpoint Acceleration (units/s²)

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupSpinParametersGet** (int SocketID, char *GroupName, double * Velocity, double * Acceleration)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	Spindle group name

Output parameters

– Velocity	double *	Setpoint Velocity (units/s)
– Acceleration	double *	Setpoint Acceleration (units/s²)

Return

– Function error code

Visual Basic

Prototype

Long **GroupSpinParametersGet** (ByVal SocketID As Long, ByVal GroupName As String, Velocity As Double, Acceleration As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	Spindle group name

Output parameters

– Velocity	double	Setpoint Velocity (units/s)
– Acceleration	double	Setpoint Acceleration (units/s²)

Return

– Function error code

Matlab

Prototype

[Error, Velocity, Acceleration] **GroupSpinParametersGet** (int32 SocketID, cstring GroupName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	Spindle group name

Return

– Error	int32	Function error code
– Velocity	double	Setpoint Velocity (units/s)
– Acceleration	double	Setpoint Acceleration (units/s²)

Python

Prototype

[Error, Velocity, Acceleration] **GroupSpinParametersGet** (integer SocketID, string GroupName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	Spindle group name

Return

– Error	int	Function error code
– Velocity	double	Setpoint Velocity (units/s)
– Acceleration	double	Setpoint Acceleration (units/s²)

2.7.3.4 GroupSpinParametersSet

Name

GroupSpinParametersSet – Sets the spin profiler parameters.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the group type (must be a spindle group): ERR_WRONG_OBJECT_TYPE (-8)
- Check the function (must be a spindle function): ERR_POSITIONER_NAME (-18)
- Valid output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)
- Check input parameter value: ERR_PARAMETER_OUT_OF_RANGE (-17)
  - Velocity <= MaximumVelocity
  - Velocity >= - MaximumVelocity
  - Acceleration > 0
  - Acceleration ≤ MaximumAcceleration

Description

This function starts the SPIN mode and also allows on-the-fly changes to the velocity and acceleration used by the SPIN mode. If an error occurs, the positioner stops and the velocity value is set to zero.

After the tests on input values:

If Velocity > MaximumVelocity => Velocity = MaximumVelocity

If Velocity < - MaximumVelocity => Velocity = - MaximumVelocity

If Acceleration ≤ 0 => ERROR and stop motion

If Acceleration > MaximumAcceleration => Acceleration = MaximumAcc.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

GroupSpinParametersSet $SocketID $GroupName $Velocity $Acceleration

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	Spindle group name (maximum size = 250)
– Velocity	double	Setpoint Velocity (units/s)
– Acceleration	double	Setpoint Acceleration (units/s²)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GroupSpinParametersSet** (int SocketID, char *GroupName, double Velocity, double Acceleration)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	Spindle group name
– Velocity	double	Setpoint Velocity (units/s)
– Acceleration	double	Setpoint Acceleration (units/s²)

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **GroupSpinParametersSet** (ByVal SocketID As Long, ByVal GroupName As String, ByVal Velocity As Double, ByVal Acceleration As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	Spindle group name
– Velocity	double	Setpoint Velocity (units/s)
– Acceleration	double	Setpoint Acceleration (units/s²)

Output parameters

– None

Return

– Function error code

Matlab

Prototype

[Error] **GroupSpinParametersSet** (int32 SocketID, cstring GroupName, double Velocity, double Acceleration)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	Spindle group name
– Velocity	double	Setpoint Velocity (units/s)
– Acceleration	double	Setpoint Acceleration (units/s²)

Return

– Error	int32	Function error code

Python

Prototype

[Error, Velocity, Acceleration] **GroupSpinParametersSet** (integer SocketID, string GroupName, double Velocity, double Acceleration)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	Spindle group name
– Velocity	double	Setpoint Velocity (units/s)
– Acceleration	double	Setpoint Acceleration (units/s²)

Return

– Error	int	Function error code

2.7.3.5 SpindleSlaveModeDisable

Name

SpindleSlaveModeDisable – Disables the slave-master mode.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Group state must be "SLAVE": ERR_NOT_ALLOWED_ACTION (-22)
- Check the group name: ERR_GROUP_NAME (-19)
- Check the positioner name: ERR_POSITIONER_NAME (-18)
- Check the group type (must be a Spindle group): ERR_WRONG_OBJECT_TYPE (-8)

Description

This function disables the master-slave mode for a spindle group. If a motion is in progress then it is aborted.

To use this function, the group state must be SLAVE (46). If it's not the case then ERR_NOT_ALLOWED_ACTION (-22) is returned.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

SpindleSlaveModeDisable $SocketID $GroupName

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Spindle group name (maximum size = 250)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **SpindleSlaveModeDisable** (int SocketID, char *GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	char *	Spindle group name

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **SpindleSlaveModeDisable** (ByVal SocketID As Long, ByVal GroupName As String)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Spindle group name

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **SpindleSlaveModeDisable** (int32 SocketID, cstring GroupName)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	cstring	Spindle group name

Return

– Function error code

Python

Prototype

integer **SpindleSlaveModeDisable** (integer SocketID, string GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	string	Spindle group name

Return

– Function error code

2.7.3.6 SpindleSlaveModeEnable

Name

SpindleSlaveModeEnable – Enables the slave-master mode.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Group state must be "READY": ERR_NOT_ALLOWED_ACTION (-22)
- Check the group name: ERR_GROUP_NAME (-19)
- Check the positioner name: ERR_POSITIONER_NAME (-18)
- Check the group type (must be a Spindle group): ERR_WRONG_OBJECT_TYPE (-8)
- Check the slave parameters (must be configured): ERR_SLAVE_CONFIGURATION (-41)

Description

This function enables the master-slave mode only if the slave group is in ready mode. In this mode the slave must be defined as a Spindle group and the master can be a positioner from any group.

To use this function, the Spindle group must be in the READY state. If it's not the case then ERR_NOT_ALLOWED_ACTION (-22) is returned.

To use this function, the master positioner and the slave ratio must be configured with the "SpindleSlaveParametersSet" function. If it's not the case then ERR_SLAVE_CONFIGURATION (-41) is returned.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_POSITIONER_NAME (-18)
- ERR_SLAVE_CONFIGURATION (-41)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

SpindleSlaveModeEnable $SocketID $GroupName

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Spindle group name (maximum size = 250)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **SpindleSlaveModeEnable** (int SocketID, char *GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	char *	Spindle group name

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **SpindleSlaveModeEnable** (ByVal SocketID As Long, ByVal GroupName As String)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Spindle group name

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **SpindleSlaveModeEnable** (int32 SocketID, cstring GroupName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	Spindle group name

Return

– Function error code

Python

Prototype

integer **SpindleSlaveModeEnable** (integer SocketID, string GroupName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	Spindle group name

Return

– Function error code

2.7.3.7 SpindleSlaveParametersGet

Name

SpindleSlaveParametersGet – Returns the spindle slave parameters.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the positioner name: ERR_POSITIONER_NAME (-18)
- Check the group type (must be a Spindle group): ERR_WRONG_OBJECT_TYPE (-8)
- Valid output parameter type: ERR_WRONG_TYPE_DOUBLE (-14), ERR_WRONG_TYPE_CHAR (-13)
- Check the slave parameters (must be configured): ERR_NOT_ALLOWED_ACTION (-22)

Description

This function returns the slave parameters: the master positioner name and the master-slave ratio.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

SpindleSlaveParametersGet $SocketID $GroupName MasterPositionerName Ratio

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	Spindle Group name (maximum size = 250)

Output parameters

– MasterPositionerName	string	Master positioner name from any group
– Ratio	double	Gear ratio between the master and the slave

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **SpindleSlaveParametersGet** (int SocketID, char *GroupName, int NbPositioners, char * MasterPositionerName , double * Ratio)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	Spindle Group name

Output parameters

– MasterPositionerName	char *	Master positioner name from any group
– Ratio	double *	Gear ratio between the master and the slave

Return

– Function error code

Visual Basic

Prototype

Long **SpindleSlaveParametersGet** (ByVal SocketID As Long, ByVal GroupName As String, ByVal MasterPositionerName As String, Ratio As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	Spindle Group name

Output parameters

– MasterPositionerName	string	Master positioner name from any group
– Ratio	double	Gear ratio between the master and the slave

Return

– Function error code

Matlab

Prototype

[Error, MasterPositionerName, Ratio] **SpindleSlaveParametersGet** (int32 SocketID, cstring GroupName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	Spindle Group name

Return

– Error	int32	Function error code
– MasterPositionerName	cstring	Master positioner name from any group
– Ratio	double	Gear ratio between the master and the slave

Python

Prototype

[Error, MasterPositionerName, Ratio] **SpindleSlaveParametersGet** (integer SocketID, string GroupName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	Spindle Group name

Return

– Error	int	Function error code
– MasterPositionerName	string	Master positioner name from any group
– Ratio	double	Gear ratio between the master and the slave

2.7.3.8 SpindleSlaveParametersSet

Name

SpindleSlaveParametersSet – Sets the spindle slave parameters.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the master positioner name: ERR_POSITIONER_NAME (-18)
- Check the master group type: ERR_WRONG_OBJECT_TYPE (-8)
- Check input parameter type: ERR_WRONG_TYPE_DOUBLE (-14), ERR_WRONG_TYPE_CHAR (-13)
- Check the slave parameters (must be configured): ERR_NOT_ALLOWED_ACTION (-22)
- Check the base velocity value (must be null): ERR_BASE_VELOCITY (-48)
- Check the ratio value (Ratio > 0): ERR_PARAMETER_OUT_OF_RANGE (-17)

Description

This function configures the slave parameters only for a Spindle group.

The slave is a master copy and a ratio can be applied: Slave = Ratio * Master.

The slave-master mode is activated only after calling of "SingleAxisSlaveModeEnable" function.

The master can be a positioner from a spindle group only. If the master group is another group type then ERR_NOT_ALLOWED_ACTION (-22) is returned. The master positioner must be different from the slave positioner else ERR_WRONG_OBJECT_TYPE (-8) is returned.

NOTE

After an emergency stop, the master group and the slave group are in "NOTINIT" state. To restart a master-slave relation, the slave group(s) must be reinitialized before the master group.

Error codes

- ERR_BASE_VELOCITY (-48)
- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

SpindleSlaveParametersSet $SocketID $GroupName $MasterPositionerName $Ratio

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Spindle Group name (maximum size = 250)
–	MasterPositionerName	string	Master positioner name from any group
–	Ratio	double	Gear ratio between the master and the slave

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **SpindleSlaveParametersSet** (int SocketID, char *GroupName, int NbPositioners, char * MasterPositionerName , double Ratio)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	char *	Spindle Group name
–	MasterPositionerName	char *	Master positioner name from any group
–	Ratio	double	Gear ratio between the master and the slave

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **SpindleSlaveParametersSet** (ByVal SocketID As Long, ByVal GroupName As String, ByVal MasterPositionerName As String, ByVal Ratio As Double)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	Spindle Group name
–	MasterPositionerName	string	Master positioner name from any group
–	Ratio	double	Gear ratio between the master and the slave

Output parameters

– None

Return

– Function error code

Matlab

Prototype

[Error] **SpindleSlaveParametersSet** (int32 SocketID, cstring GroupName, cstring MasterPositionerName, double Ratio)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	Spindle Group name
– MasterPositionerName	cstring	Master positioner name from any group
– Ratio	double	Gear ratio between the master and the slave

Return

– Error	int32	Function error code

Python

Prototype

[Error] **SpindleSlaveParametersSet** (integer SocketID, string GroupName, string MasterPositionerName, double Ratio)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	Spindle Group name
– MasterPositionerName	string	Master positioner name from any group
– Ratio	double	Gear ratio between the master and the slave

Return

– Error	int	Function error code

2.7.4 Configuration Files

Below is an example of a Spindle group (named "AXIS") in the system.ini file. The group POSITIONER is built with a positioner named "STAGE". The positioner "STAGE" uses the parameters of "MYSTAGE" from the stages.ini file and is connected to plug 1 of the XPS controller.

NOTE

PIDBaseFilter parameter in system.ini file is to be configured only when using the XPS-Qn Precision Platform controller.

System.ini file:

```
[GROUPS]
SpindleInUse = POSITIONER
[POSITIONER]    ; AXIS Spindle group configuration
PositionerInUse = STAGE
[POSITIONER.STAGE]
PlugNumber = 1
StageName = MYSTAGE
; --- Time flasher
TimeFlasherBaseFrequency =      ; default value 40e6, must be between 39.5e6 and 40.5e6 Hz
;--- PIDBase filter (take into account the base movements, effective only with PIDFFAcceleration corrector) To be added only when using XPS-Q8 Precision Platform controller
PIDBaseFilter =                 ; Enabled or Disabled
MovingMass =                    ; If PIDBaseFilter = Enabled
StaticMass =                    ; If PIDBaseFilter = Enabled
Viscosity =                     ; If PIDBaseFilter = Enabled
Stiffness =                     ; If PIDBaseFilter = Enabled
```

Stages.ini file:

```
[MYSTAGE]
MYSTAGE configuration => See § "Positioner: Configuration filess"
```

2.8 XY Group

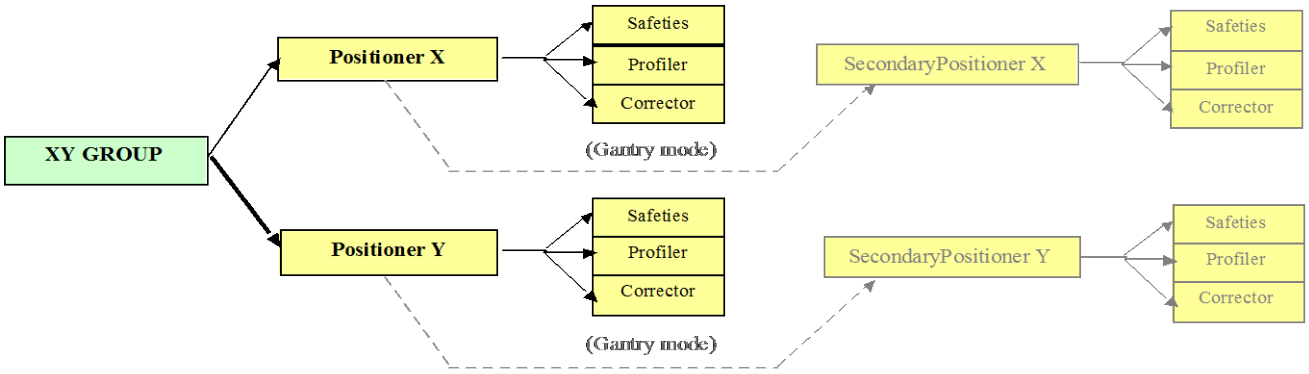
2.8.1 Description

An XY group is composed of two positioners, typically in an orthogonal XY configuration.

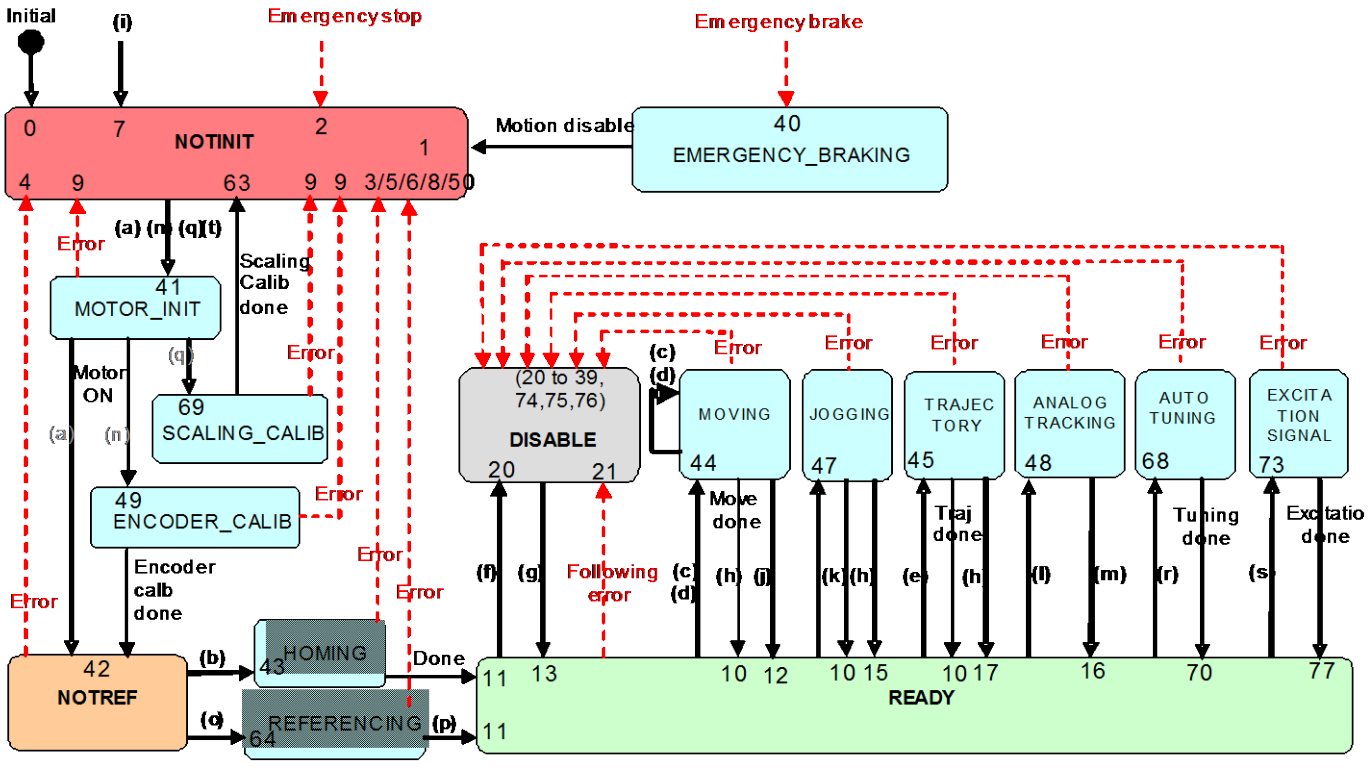
An XY group can be used in GANTRY mode (dual positioner for X or for Y).

It includes an XY mapping feature: XY = f (XY)

It includes an XY line-arc trajectory (2D).



2.8.2 State Diagram



Called function:

(a) GroupInitialize
(b) GroupHomeSearch
(c) GroupMoveAbsolute
(d) GroupMoveRelative
(e) XYLineArcExecution
(f) GroupMotionDisable
(g) GroupMotionEnable
(h) GroupMoveAbort
(i) GroupKill or KillAll
(j) GroupJogModeEnable
(k) GroupJogModeDisable
(l) GroupAnalogTrackingModeEnable
(m) GroupAnalogTrackingModeDisable
(n) GroupInitializeWithEncoderCalibration
(o) GroupReferencingStart
(p) GroupReferencingStop
(q) PositionerAccelerationAutoScaling
(r) PositionerCorrectorAutoTuning
(s) PositionerExcitationSignalSet / PositionerPreCorrectorExcitationSignalSet
(t) GroupInitializeNoEncoderReset

2.8.3 Specific Function Description

2.8.3.1 XYLineArcExecution

Name

XYLineArcExecution – Executes an XY LineArc trajectory.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Group state must be "READY": ERR_NOT_ALLOWED_ACTION (-22)
- Check the group (must not be a gantry): ERR_POSITIONER_NAME (-18)
- Check the group type (must be an XY group): ERR_WRONG_OBJECT_TYPE (-8)
- Check backlash (must not be enabled): ERR_NOT_ALLOWED_BACKLASH (-46)
- Check the trajectory file name length: ERR_STRING_TOO_LONG (-3)
- Check the trajectory file existence: ERR_READ_FILE (-61)
- Check the input value (> 0): ERR_PARAMETER_OUT_OF_RANGE (-17)
- Check the velocity (0 < Velocity <= TrajectoryMaximumVelocity): ERR_TRAJ_VEL_LIMIT (-68)
- Check the acceleration (0 < Acceleration <= TrajectoryMaximumAcceleration): ERR_TRAJ_ACC_LIMIT (-69)
- Check input parameter types: ERR_WRONG_TYPE_DOUBLE (-14), ERR_WRONG_TYPE_INT (-15)

Description

This function executes an XY LineArc trajectory. The trajectory file must be stored in the folder "\ADMIN\Public\Trajectory" of the XPS controller. If the trajectory cannot be initialized (message queue or task error) then ERR_TRAJ_INITIALIZATION (-72) is returned.

Before a trajectory execution, it is recommended to verify whether the trajectory is within stage parameter capability and is able execute by using the "XYLineArcVerification" and "XYLineArcVerificationResultGet" functions.

For a more thorough description of the line-arc trajectory capability, please refer to the XPS Motion Tutorial, section Trajectories/Line-Arc Trajectories.

NOTE

In case of an ERR_GROUP_MOTION_DONE_TIMEOUT (-33) error, an ERR_FOLLOWING_ERROR (-25) error or ERR_SLAVE (-44) error, the group state becomes DISABLE. To help determine the error source, check the positioner errors, the hardware status and the driver status.

Error codes

- ERR_BASE_VELOCITY (-48)
- ERR_FATAL_INIT (-20)
- ERR_FOLLOWING_ERROR (-25)
- ERR_GROUP_MOTION_DONE_TIMEOUT (-33)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_NOT_ALLOWED_BACKLASH (-46)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_POSITIONER_NAME (-18)
- ERR_READ_FILE (-61)
- ERR_SLAVE (-44)
- ERR_STRING_TOO_LONG (-3)
- ERR_TRAJ_ACC_LIMIT (-69)
- ERR_TRAJ_VEL_LIMIT (-68)
- ERR_TRAJ_INITIALIZATION (-72)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14),
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

XYLineArcExecution $SocketID $GroupName $FileName $Velocity $Acceleration $ExecutionNumber

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	XY group name (maximum size = 250)
– FileName	string	Trajectory file name (maximum size = 250)
– Velocity	double	Trajectory velocity (units/seconds)
– Acceleration	double	Trajectory acceleration (units/seconds²)
– ExecutionNumber	int	Number of trajectory executions

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **XYLineArcExecution** (int SocketID, char *GroupName, char *FileName , double Velocity, double Acceleration, int ExecutionNumber)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	XY group name
– FileName	char *	Trajectory file name (maximum size = 250)
– Velocity	double	Trajectory velocity (units/seconds)
– Acceleration	double	Trajectory acceleration (units/seconds²)
– ExecutionNumber	int	Number of trajectory executions

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **XYLineArcExecution** (ByVal SocketID As Long, ByVal GroupName As String, ByVal FileName As String, ByVal Velocity As Double, ByVal Acceleration As Double, ByVal ExecutionNumber As Integer)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	XY group name
– FileName	string	Trajectory file name (maximum size = 250)
– Velocity	double	Trajectory velocity (units/seconds)
– Acceleration	double	Trajectory acceleration (units/seconds²)
– ExecutionNumber	int	Number of trajectory executions

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **XYLineArcExecution** (int32 SocketID, cstring GroupName, cstring FileName, double Velocity, double Acceleration, int ExecutionNumber)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	XY group name
– FileName	cstring	Trajectory file name (maximum size = 250)
– Velocity	double	Trajectory velocity (units/seconds)
– Acceleration	double	Trajectory acceleration (units/seconds²)
– ExecutionNumber	int32	Number of trajectory executions

Return

– Function error code

Python

Prototype

integer **XYLineArcExecution** (integer SocketID, string GroupName, string FileName, double Velocity, double Acceleration, int ExecutionNumber)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	XY group name
– FileName	string	Trajectory file name (maximum size = 250)
– Velocity	double	Trajectory velocity (units/seconds)
– Acceleration	double	Trajectory acceleration (units/seconds²)
– ExecutionNumber	int	Number of trajectory executions

Return

– Function error code

2.8.3.2 XYLineArcParametersGet

Name

XYLineArcParametersGet – Returns the LineArc trajectory parameters.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the group (must not be a gantry): ERR_POSITIONER_NAME (-18)
- Check the group type (must be a XY group): ERR_WRONG_OBJECT_TYPE (-8)
- Check the LineArc trajectory type: ERR_NOT_ALLOWED_ACTION (-22)
- Valid output parameter type: ERR_WRONG_TYPE_CHAR (-13), ERR_WRONG_TYPE_DOUBLE (-14), ERR_WRONG_TYPE_INT (-15)

Description

This function returns the XY LineArc trajectory parameters (trajectory name, trajectory velocity, trajectory acceleration, current executing element number) of the currently executed LineArc trajectory.

For a more thorough description of the line-arc trajectory capability, please refer to the XPS Motion Tutorial, section Trajectories/Line-Arc Trajectories.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13)
- ERR_WRONG_TYPE_DOUBLE (-14)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

XYLineArcParametersGet $SocketID $GroupName FileName Velocity Acceleration ElementNumber

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	XY Group name (maximum size = 250)

Output parameters

– FileName	string	Executing trajectory file name (maximum size = 250)
– Velocity	double	Trajectory velocity (units/seconds)
– Acceleration	double	Trajectory acceleration (units/seconds²)
– ElementNumber	int	Current executing element number

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **XYLineArcParametersGet** (int SocketID, char *GroupName, char * FileName, double * Velocity, double * Acceleration, int * ElementNumber)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	XY Group name

Output parameters

– FileName	char *	Executing trajectory file name (maximum size = 250)
– Velocity	double *	Trajectory velocity (units/seconds)
– Acceleration	double *	Trajectory acceleration (units/seconds²)
– ElementNumber	int *	Current executing element number

Return

– Function error code

Visual Basic

Prototype

Long **XYLineArcParametersGet** (ByVal SocketID As Long, ByVal GroupName As String, ByVal FileName As String, Velocity As Double, Acceleration As Double, ElementNumber As Integer)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	XY Group name

Output parameters

– FileName	string	Executing trajectory file name (maximum size = 250)
– Velocity	double	Trajectory velocity (units/seconds)
– Acceleration	double	Trajectory acceleration (units/seconds²)
– ElementNumber	int	Current executing element number

Return

– Function error code

Matlab

Prototype

[Error, FileName, Velocity, Acceleration, ElementNumber]
XYLineArcParametersGet (int32 SocketID, cstring GroupName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	XY Group name

Return

– Error	int32	Function error code
– FileName	cstring	Executing trajectory file name (maximum size = 250)
– Velocity	double	Trajectory velocity (units/seconds)
– Acceleration	double	Trajectory acceleration (units/seconds²)
– ElementNumber	int32	Current executing element number

Python

Prototype

[Error, FileName, Velocity, Acceleration, ElementNumber]
XYLineArcParametersGet (integer SocketID, string GroupName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	XY Group name

Return

– Error	int	Function error code
– FileName	string	Executing trajectory file name (maximum size = 250)
– Velocity	double	Trajectory velocity (units/seconds)
– Acceleration	double	Trajectory acceleration (units/seconds²)
– ElementNumber	int	Current executing element number

2.8.3.3 XYLineArcPulseOutputGet

Name

XYLineArcPulseOutputGet – Returns the configuration of pulse generation in a LineArc trajectory.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the group (must not be a positioner): ERR_POSITIONER_NAME (-18)
- Check the group type (must be a XY group): ERR_WRONG_OBJECT_TYPE (-8)
- Valid output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function returns the last configuration of pulse generation of an XY LineArc trajectory.

The pulse output configuration is defined by a start length, an end length, and a path length interval.

Example:

One pulse is generated every 10 µm on the Line-Arc trajectory between 10 mm and 30 mm.

Start length = 10 mm

End length = 30 mm

Path length interval = 0.01 mm

For a more thorough description of the line-arc trajectory capability, please refer to the XPS Motion Tutorial, section Trajectories/Line-Arc Trajectories.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

XYLineArcPulseOutputGet $SocketID $GroupName StartLength EndLength PathLengthInterval

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	XY Group name (maximum size = 250)

Output parameters

– StartLength	double	Start length (units)
– EndLength	double	End length (units)
– PathLengthInterval	double	Path length interval (units)

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **XYLineArcPulseOutputGet** (int SocketID, char *GroupName, double * StartLength, double * EndLength, double * PathLengthInterval)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	XY Group name

Output parameters

– StartLength	double *	Start length (units)
– EndLength	double *	End length (units)
– PathLengthInterval	double *	Path length interval (units)

Return

– Function error code

Visual Basic

Prototype

Long **XYLineArcPulseOutputGet** (ByVal SocketID As Long, ByVal GroupName As String, StartLength As Double, EndLength As Double, PathLengthInterval As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	XY Group name

Output parameters

– StartLength	double	Start length (units)
– EndLength	double	End length (units)
– PathLengthInterval	double	Path length interval (units)

Return

– Function error code

Matlab

Prototype

[Error, StartLength, EndLength, PathLengthInterval] **XYLineArcPulseOutputGet** (int32 SocketID, cstring GroupName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	XY Group name

Return

– Error	int32	Function error code
– StartLength	double	Start length (units)
– EndLength	double	End length (units)
– PathLengthInterval	double	Path length interval (units)

Python

Prototype

[Error, StartLength, EndLength, PathLengthInterval] **XYLineArcPulseOutputGet** (integer SocketID, string GroupName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	XY Group name

Return

– Error	int	Function error code
– StartLength	double	Start length (units)
– EndLength	double	End length (units)
– PathLengthInterval	double	Path length interval (units)

2.8.3.4 XYLineArcPulseOutputSet

Name

XYLineArcPulseOutputSet – Sets the configuration of pulse generation on LineArc trajectory.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the group (must not be a positioner): ERR_POSITIONER_NAME (-18)
- Check the group type (must be a XY group): ERR_WRONG_OBJECT_TYPE (-8)
- Check the pulse generation must not be in progress: ERR_NOT_ALLOWED_ACTION (-22)
- Valid input parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function configures and activates the pulse generation on XY LineArc trajectory. Pulse generation is defined by a start length, an end length, and a path length interval. If a pulse is already activated on the selected XY Line-Arc trajectory then this function returns ERR_NOT_ALLOWED_ACTION error.

Please note that the pulse output settings are automatically removed when the trajectory is over. Hence, with the execution of every new trajectory, it is required to define the pulse output settings again.

This capability allows output of pulses at constant length intervals on a Line-Arc trajectories. The pulses are generated between a start length and an end length. All lengths are calculated in an orthogonal XY plane. The StartLength, EndLength, and PathLengthInterval refer to the Setpoint positions.

Example:

XYLineArcPulseOutputSet(XY, 10, 30, 0.01)

One pulse will be generated every 10 µm on the next Line-Arc trajectory between 10 mm and 30 mm

For a more thorough description of the line-arc trajectory capability, please refer to the XPS Motion Tutorial section Trajectories/Line-Arc Trajectories.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

XYLineArcPulseOutputSet $SocketID $GroupName StartLength EndLength PathLengthInterval

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	XY Group name (maximum size = 250)
– StartLength	double	Start length (units)
– EndLength	double	End length (units)
– PathLengthInterval	double	Path length interval (units)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **XYLineArcPulseOutputSet** (int SocketID, char *GroupName, double StartLength, double EndLength, double PathLengthInterval)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	XY Group name
– StartLength	double	Start length (units)
– EndLength	double	End length (units)
– PathLengthInterval	double	Path length interval (units)

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **XYLineArcPulseOutputSet** (ByVal SocketID As Long, ByVal GroupName As String, ByVal StartLength As Double, ByVal EndLength As Double, ByVal PathLengthInterval As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	XY Group name
– StartLength	double	Start length (units)
– EndLength	double	End length (units)
– PathLengthInterval	double	Path length interval (units)

Output parameters

– None

Return

– Function error code

Matlab

Prototype

[Error] **XYLineArcPulseOutputSet** (int32 SocketID, cstring GroupName, double StartLength, double EndLength, double PathLengthInterval)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	XY Group name
– StartLength	double	Start length (units)
– EndLength	double	End length (units)
– PathLengthInterval	double	Path length interval (units)

Return

– Error	int32	Function error code

Python

Prototype

[Error] **XYLineArcPulseOutputSet** (integer SocketID, string GroupName, double StartLength, double EndLength, double PathLengthInterval)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	XY Group name
– StartLength	double	Start length (units)
– EndLength	double	End length (units)
– PathLengthInterval	double	Path length interval (units)

Return

– Error	int	Function error code

2.8.3.5 XYLineArcVerification

Name

XYLineArcVerification – Verifies a line-arc trajectory data file.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the group (must not be a gantry): ERR_POSITIONER_NAME (-18)
- Check the group type (must be an XY group): ERR_WRONG_OBJECT_TYPE (-8)
- Check the trajectory file name length: ERR_STRING_TOO_LONG (-3)
- Check the trajectory file existence: ERR_READ_FILE (-61)
- Check the trajectory (number of elements > 0): ERR_TRAJ_EMPTY (-66)
- Check trajectory element types in file (FirstTangent, …): ERR_WRONG_TYPE_DOUBLE (-14)
- Check keys in file ("FirstTangent" and "DiscontinuityAngle"): ERR_READ_FILE_PARAMETER_KEY (-74)
- Check trajectory element (distance and tangent): ERR_TRAJ_ELEM_LINE (-65)
  - $|\ XElementDistance\ | \geq 1e\text{-}14$
  - $|\ YElementDistance\ | \geq 1e\text{-}14$
  - $TangentOut \neq 1.797e308$
- Check trajectory element (radius): ERR_TRAJ_ELEM_RADIUS (-63)
  - $Radius \geq 1e\text{-}14$
- Check trajectory element (sweep angle): ERR_TRAJ_ELEM_ SWEEP (-64)
  - $SweepAngle \geq 1e\text{-}14$

Description

This function verifies the execution of an XY LineArc trajectory. The results of the verification can be gathered with the "XYLineArcVerificationResultGet" function. The trajectory file must be stored in the folder "\ADMIN\Public\Trajectory" of the XPS controller. If the trajectory cannot be initialized (message queue or task error) then ERR_TRAJ_INITIALIZATION (-72) is returned.

This function can be executed at any time and is independent of trajectory execution. This function performs the following:

- Checks the trajectory file for data coherence.
- Calculates the trajectory limits, which are: the required travel per positioner, the maximum possible trajectory velocity and the maximum possible trajectory acceleration. This function helps define the parameters for trajectory execution.
- If all is OK, it returns "SUCCESS" (0). Otherwise, it returns a corresponding error.

NOTE

The "XYLineArcVerification" function is independent of the "XYLineArcExecution" function. It is recommended, but not necessary, to execute this function before executing a LineArc trajectory.

For a more thorough description of the line-arc trajectory capability, please refer to the XPS Motion Tutorial, section Trajectories/Line-Arc Trajectories.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_READ_FILE (-61)
- ERR_READ_FILE_PARAMETER_KEY (-74)
- ERR_STRING_TOO_LONG (-3)
- ERR_TRAJ_ELEM_LINE (-65)
- ERR_TRAJ_ELEM_RADIUS (-63)
- ERR_TRAJ_ELEM_SWEEP (-64)
- ERR_TRAJ_EMPTY (-66)
- ERR_TRAJ_INITIALIZATION (-72)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

XYLineArcVerification $SocketID $GroupName $FileName

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	XY group name (maximum size = 250)
–	FileName	string	Trajectory file name (maximum size = 250)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **XYLineArcVerification** (int SocketID, char *GroupName, char *FileName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	XY group name
– FileName	char *	Trajectory file name (maximum size = 250)

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **XYLineArcVerification** (ByVal SocketID As Long, ByVal GroupName As String, ByVal FileName As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	XY group name
– FileName	string	Trajectory file name (maximum size = 250)

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **XYLineArcVerification** (int32 SocketID, cstring GroupName, cstring FileName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	XY group name
– FileName	cstring	Trajectory file name (maximum size = 250)

Return

– Function error code

Python

Prototype

integer **XYLineArcVerification** (integer SocketID, string GroupName, string FileName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	XY group name
– FileName	string	Trajectory file name (maximum size = 250)

Return

– Function error code

2.8.3.6 XYLineArcVerificationResultGet

Name

XYLineArcVerificationResultGet – Returns the results of the previous "XYLineArcVerification" function.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the group type (must be a XY group): ERR_WRONG_OBJECT_TYPE (-8)
- Check the last XY LineArc verification (must be done): ERR_NOT_ALLOWED_ACTION (-22)
- Check the positioner name length: ERR_STRING_TOO_LONG (-3)
- Check the positioner (must not be a gantry): ERR_POSITIONER_NAME (-18)
- Valid output parameter type: ERR_WRONG_TYPE_CHAR (-13), ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function returns the results of the previous "XYLineArcVerification" function, positioner by positioner. The results are the travel requirements (min and max values), the possible maximum velocity and the possible maximum acceleration.

If no verification was previously done then ERR_NOT_ALLOWED_ACTION (-22) is returned.

For a more thorough description of the line-arc trajectory capability, please refer to the XPS Motion Tutorial section Trajectories/Line-Arc Trajectories.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_POSITIONER_NAME (-18)
- ERR_STRING_TOO_LONG (-3)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13),
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

XYLineArcVerificationResultGet $SocketID $ PositionerName FileName MinimumPosition MaximumPosition MaximumVelocity MaximumAcceleration

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	PositionerName	string	XY positioner name (maximum size = 250)

Output parameters

–	FileName	string	Examined trajectory file name (maximum size = 250)
–	MinimumPosition	double	Minimum position (units)
–	MaximumPosition	double	Maximum position (units)
–	MaximumVelocity	double	Maximum trajectory velocity (units/seconds)
–	MaximumAcceleration	double	Maximum trajectory acceleration (units/seconds²)

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **XYLineArcVerificationResultGet** (int SocketID, char * PositionerName, char * FileName, double * MinimumPosition, double * MaximumPosition, double * MaximumVelocity, double * MaximumAcceleration)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– PositionerName	char *	XY positioner name

Output parameters

– FileName	char *	Examined trajectory file name (maximum size = 250)
– MinimumPosition	double *	Minimum position (units)
– MaximumPosition	double *	Maximum position (units)
– MaximumVelocity	double *	Maximum trajectory velocity (units/seconds)
– MaximumAcceleration	double *	Maximum trajectory acceleration (units/seconds²)

Return

– Function error code

Visual Basic

Prototype

Long **XYLineArcVerificationResultGet** (ByVal SocketID As Long, ByVal PositionerName As String, ByVal FileName As String, MinimumPosition As Double, MaximumPosition As Double, MaximumVelocity As Double, MaximumAcceleration As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– PositionerName	string	XY positioner name

Output parameters

– FileName	string	Examined trajectory file name (maximum size = 250)
– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– MaximumVelocity	double	Maximum trajectory velocity (units/seconds)
– MaximumAcceleration	double	Maximum trajectory acceleration (units/seconds²)

Return

– Function error code

Matlab

Prototype

[Error, FileName, MinimumPosition, MaximumPosition, MaximumVelocity, MaximumAcceleration] **XYLineArcVerificationResultGet** (int32 SocketID, cstring PositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– PositionerName	cstring	XY positioner name

Return

– Error	int32	Function error code
– FileName	cstring	Examined trajectory file name (maximum size = 250)
– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– MaximumVelocity	double	Trajectory trajectory velocity (units/seconds)
– MaximumAcceleration	double	Trajectory trajectory acceleration (units/seconds²)

Python

Prototype

[Error, FileName, MinimumPosition, MaximumPosition, MaximumVelocity, MaximumAcceleration] **XYLineArcVerificationResultGet** (integer SocketID, string PositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– PositionerName	string	XY positioner name

Return

– Error	int	Function error code
– FileName	string	Examined trajectory file name (maximum size = 250)
– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– MaximumVelocity	double	Trajectory velocity (units/seconds)
– MaximumAcceleration	double	Trajectory acceleration (units/seconds²)

2.8.3.7 XYPVTExecution

Name

XYPVTExecution – Executes a PVT trajectory.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Group state must be "READY": ERR_NOT_ALLOWED_ACTION (-22)
- Check backlash (must not be enabled): ERR_NOT_ALLOWED_BACKLASH (-46)
- Check the base velocity value (must be null): ERR_BASE_VELOCITY (-48)
- Check the group: ERR_POSITIONER_NAME (-18)
- Check the group type (must be an XYZ group): ERR_WRONG_OBJECT_TYPE (-8)
- Check the input parameter type: ERR_WRONG_TYPE_INT (-15)
- Check the input value (Number of executions > 0): ERR_PARAMETER_OUT_OF_RANGE (-17)
- Check the trajectory file name length: ERR_STRING_TOO_LONG (-3)
- Check the trajectory file existence or file reading: ERR_READ_FILE (-61)
- Check the message queue filling: ERR_MSG_QUEUE (-71)

Description

This function executes a PVT (Position Velocity Time) trajectory. The trajectory file must be stored in the folder "\ADMIN\Public\Trajectory" of the XPS controller. If the trajectory cannot be initialized (message queue or task error) then ERR_TRAJ_INITIALIZATION (-72) is returned.

Before a trajectory execution, it is recommended to check whether the trajectory is within capability of the group parameters and can execute by using the "XYPVTVerification" and "XYPVTVerificationResultGet" functions.

For a more thorough description of the PVT trajectory capability, please refer to the XPS Motion Tutorial section Trajectories/PVT Trajectories.

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

In case of an ERR_GROUP_MOTION_DONE_TIMEOUT (-33) error, an ERR_FOLLOWING_ERROR (-25) error or ERR_SLAVE (-44) error, the group state becomes DISABLE. To determine the source of error, check the positioner errors, the hardware status and the driver status.

Error codes

- ERR_BASE_VELOCITY (-48)
- ERR_FATAL_INIT (-20)
- ERR_FOLLOWING_ERROR (-25)
- ERR_GROUP_MOTION_DONE_TIMEOUT (-33)
- ERR_IN_INITIALIZATION (-21)
- ERR_MSG_QUEUE (-71)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_NOT_ALLOWED_BACKLASH (-46)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_POSITIONER_NAME (-18)
- ERR_READ_FILE (-61)
- ERR_SLAVE (-44)
- ERR_STRING_TOO_LONG (-3)
- ERR_TRAJ_INITIALIZATION (-72)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

XYPVTExecution $SocketID $GroupName $FileName $Velocity $Acceleration

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	XY group name (maximum size = 250)
– FileName	string	Trajectory file name (maximum size = 250)
– ExecutionNumber	int	Number of trajectory executions

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **XYPVTExecution** (int SocketID, char *GroupName, char *FileName , double Velocity, double Acceleration)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	XY group name
– FileName	char *	Trajectory file name (maximum size = 250)
– ExecutionNumber	int	Number of trajectory executions

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **XYPVTExecution** (ByVal SocketID As Long, ByVal GroupName As String, ByVal FileName As String, ByVal Velocity As Double, ByVal Acceleration As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	XY group name
– FileName	string	Trajectory file name (maximum size = 250)
– ExecutionNumber	long	Number of trajectory executions

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **XYPVTExecution** (int32 SocketID, cstring GroupName, cstring FileName, double Velocity, double Acceleration)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	XY group name
– FileName	cstring	Trajectory file name (maximum size = 250)
– ExecutionNumber	int32	Number of trajectory executions

Return

– Function error code

Python

Prototype

integer **XYPVTExecution** (integer SocketID, string GroupName, string FileName, double Velocity, double Acceleration)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	XY group name
– FileName	string	Trajectory file name (maximum size = 250)
– ExecutionNumber	int	Number of trajectory executions

Return

– Function error code

2.8.3.8 XYPVTLoadToMemory

Name

XYPVTLoadToMemory – Loads an XY PVT trajectory's line.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the group (must not be a positioner): ERR_GROUP_NAME (-19)
- Check the gantry mode (must not be a gantry): ERR_POSITIONER_NAME (-18)
- Check the group type (must be an XY group): ERR_WRONG_OBJECT_TYPE (-8)
- Check the trajectory data length: ERR_STRING_TOO_LONG (-3)
- Check the input value (> 0): ERR_PARAMETER_OUT_OF_RANGE (-17)

Description

This function loads a line of PVT trajectory elements into the XPS memory. Each trajectory element must be separated by a comma. To verify or to execute the PVT trajectory loaded in memory, use the string "**FromMemory**" instead of a FileName.

For a more thorough description of the PVT trajectory capability, please refer to the XPS Motion Tutorial section Trajectories/PVT Trajectories.

NOTE

All previous PVT trajectory functions, when called with the string "FromMemory" instead of a FileName, will perform the same operation from RAM content as it does from a file.

Example:

XYPVTLoadToMemory(XY, 0.04,0,0,3.2,0)

XYPVTVerification (XY, FromMemory)

XYPVTExecution(XY, FromMemory, 1)

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_STRING_TOO_LONG (-3)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

XYPVTLoadToMemory $SocketID $GroupName $TrajectoryLine

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	XY group name (maximum size = 250)
– TrajectoryLine	string	Trajectory line (maximum size = 400)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **XYPVTLoadToMemory** (int SocketID, char *GroupName, char *TrajectoryLine)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	XY group name
– TrajectoryLine	char *	Trajectory line (maximum size = 400)

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **XYPVTLoadToMemory** (ByVal SocketID As Long, ByVal GroupName As String, ByVal TrajectoryLine As String, ByVal)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	XY group name
– TrajectoryLine	string	Trajectory line (maximum size = 400)

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **XYPVTLoadToMemory** (int32 SocketID, cstring GroupName, cstring TrajectoryLine)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	XY group name
– TrajectoryLine	cstring	Trajectory line (maximum size = 400)

Return

– Function error code

Python

Prototype

integer **XYPVTLoadToMemory** (integer SocketID, string GroupName, string TrajectoryLine)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	XY group name
– TrajectoryLine	string	Trajectory file name (maximum size = 400)

Return

– Function error code

2.8.3.9 XYPVTParametersGet

Name

XYPVTParametersGet – Returns the PVT trajectory parameters.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the group type (must be a XY group): ERR_WRONG_OBJECT_TYPE (-8)
- Check the trajectory type (PVT): ERR_NOT_ALLOWED_ACTION (-22)
- Valid output parameter type: ERR_WRONG_TYPE_CHAR (-13), ERR_WRONG_TYPE_INT (-15)

Description

This function returns the PVT trajectory parameters (trajectory name and current executing element number) of the executed PVT trajectory.

For a more thorough description of the PVT trajectory capability, please refer to the XPS Motion Tutorial section Trajectories/PVT Trajectories.

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

XYPVTParametersGet $SocketID $GroupName FileName ElementNumber

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	XY group name (maximum size = 250)

Output parameters

– FileName	string	Executing trajectory file name (maximum size = 250)
– ElementNumber	int	Current executing element number

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **XYPVTParametersGet** (int SocketID, char *GroupName, char * FileName, int * ElementNumber)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	XY group name

Output parameters

– FileName	char *	Executing trajectory file name (maximum size = 250)
– ElementNumber	int *	Current executing element number

Return

– Function error code

Visual Basic

Prototype

Long **XYPVTParametersGet** (ByVal SocketID As Long, ByVal GroupName As String, ByVal FileName As String, ElementNumber As Integer)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	XY group name

Output parameters

– FileName	string	Executing trajectory file name (maximum size = 250)
– ElementNumber	int	Current executing element number

Return

– Function error code

Matlab

Prototype

[Error, FileName, ElementNumber] **XYPVTParametersGet** (int32 SocketID, cstring GroupName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	XY group name

Return

– Error	int32	Function error code
– FileName	cstring	Executing trajectory file name (maximum size = 250)
– ElementNumber	int32	Current executing element number

Python

Prototype

[Error, FileName, ElementNumber] **XYPVTParametersGet** (integer SocketID, string GroupName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	XY group name

Return

– Error	int	Function error code
– FileName	string	Executing trajectory file name (maximum size = 250)
– ElementNumber	int	Current executing element number

2.8.3.10 XYPVTPulseOutputGet

Name

XYPVTPulseOutputGet – Returns the configuration of pulse generation of a PVT trajectory.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the group (must not be a positioner): ERR_POSITIONER_NAME (-18)
- Check the group type (must be a XY group): ERR_WRONG_OBJECT_TYPE (-8)
- Valid output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function returns the last configuration of pulse generation of a PVT trajectory.

The pulse output configuration is defined by a start element, an end element, and a time interval in seconds.

Example:

XYPVTPulseOutputGet (Group1, 3, 5, 0.01)

One pulse will be generated every 10 ms between the start of the 3rd element and the end of the 5th element.

Start element= 3

End element = 5

Time interval = 0.01 seconds

For a more thorough description of the PVT trajectory capability, please refer to the XPS Motion Tutorial sections Trajectories/PVT Trajectories and Output Triggers.

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

XYPVTPulseOutputGet $SocketID $GroupName StartElement EndElement TimeInterval

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	XY group name (maximum size = 250)

Output parameters

–	StartElement	int	Start Element number
–	EndElement	int	End Element number
–	TimeInterval	double	Time interval (seconds)

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **XYPVTPulseOutputGet** (int SocketID, char *GroupName, int * StartElement, int * EndElement, double * TimeInterval)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	char *	XY group name

Output parameters

–	StartElement	int *	Start Element number
–	EndElement	int *	End Element number
–	TimeInterval	double *	Time interval (seconds)

Return

– Function error code

Visual Basic

Prototype

Long **XYPVTPulseOutputGet** (ByVal SocketID As Long, ByVal GroupName As String, StartElement As Long, EndElement As Long, TimeInterval As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	XY group name

Output parameters

– StartElement	long	Start Element number
– EndElement	long	End Element number
– TimeInterval	double	Time interval (seconds)

Return

– Function error code

Matlab

Prototype

[Error, StartElement, EndElement, TimeInterval] **XYPVTPulseOutputGet** (int32 SocketID, cstring GroupName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	XY Group name

Return

– Error	int32	Function error code
– StartElement	int32	Start Element number
– EndElement	int32	End Element number
– TimeInterval	double	Time interval (seconds)

Python

Prototype

[Error, StartElement, EndElement, TimeInterval] **XYPVTPulseOutputGet** (integer SocketID, string GroupName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	XY group name

Return

– Error	int	Function error code
– StartElement	int	Start Element number
– EndElement	int	End Element number
– TimeInterval	double	Time interval (seconds)

2.8.3.11 XYPVTPulseOutputSet

Name

XYPVTPulseOutputSet – Sets the configuration of pulse generation of a PVT trajectory.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the group (must not be a positioner): ERR_POSITIONER_NAME (-18)
- Check the group type (must be a XY group): ERR_WRONG_OBJECT_TYPE (-8)
- Check the pulse generation must not be in progress: ERR_NOT_ALLOWED_ACTION (-22)
- Valid input parameter type: ERR_WRONG_TYPE_DOUBLE (-14), ERR_WRONG_TYPE_INT (-15)

Description

This function configures and activates the pulse generation of a PVT trajectory. The pulse generation is defined by a start element, an end element, and a time interval in seconds. If a pulse generation is already activated on the selected PVT trajectory then this function returns ERR_NOT_ALLOWED_ACTION error.

Please note, that the pulse output settings are automatically removed when the trajectory is over. Hence, with the execution of every new trajectory, it is also required to define the pulse output settings again.

This capability allows output of pulses at constant time intervals on a PVT trajectory. The pulses are generated between a first and a last trajectory element. The minimum possible time interval is 100 µs.

Example:

XYPVTPulseOutputSet (Group1, 3, 5, 0.01)

One pulse will be generated every 10 ms between the start of the 3rd element and the end of the 5th element.

For a more thorough description of the PVT trajectory capability, please refer to the XPS Motion Tutorial sections Trajectories/PVT Trajectories and Ouput Triggers.

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

XYPVTPulseOutputSet $SocketID $GroupName $StartElement $EndElement $TimeInterval

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	XY group name (maximum size = 250)
–	StartElement	int	Start Element number
–	EndElement	int	End Element number
–	TimeInterval	double	Time interval (seconds)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **XYPVTPulseOutputSet** (int SocketID, char *GroupName, int StartElement, int EndElement, double TimeInterval)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	XY group name
– StartElement	int	Start Element number
– EndElement	int	End Element number
– TimeInterval	double	Time interval (seconds)

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **XYPVTPulseOutputSet** (ByVal SocketID As Long, ByVal GroupName As String, ByVal StartElement As Long, ByVal EndElement As Long, ByVal TimeInterval As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	XY group name
– StartElement	long	Start Element number
– EndElement	long	End Element number
– TimeInterval	double	Time interval (seconds)

Output parameters

– None

Return

– Function error code

Matlab

Prototype

[Error] **XYPVTPulseOutputSet** (int32 SocketID, cstring GroupName, int32 StartElement, int32 EndElement, double TimeInterval)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	XY group name
– StartElement	int32	Start Element number
– EndElement	int32	End Element number
– TimeInterval	double	Time interval (seconds)

Return

– Error	int32	Function error code

Python

Prototype

[Error] **XYPVTPulseOutputSet** (integer SocketID, string GroupName, integer StartElement, integer EndElement, double TimeInterval)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	XY group name
– StartElement	int	Start Element number
– EndElement	int	End Element number
– TimeInterval	double	Time interval (seconds)

Return

– Error	int	Function error code

2.8.3.12 XYPVTResetInMemory

Name

XYPVTResetInMemory – Deletes the current content of the PVT trajectory buffer in memory.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the group (must not be a positioner): ERR_GROUP_NAME (-19)
- Check the group (must not be a gantry): ERR_POSITIONER_NAME (-18)
- Check the group type (must be an XY group): ERR_WRONG_OBJECT_TYPE (-8)

Description

This function deletes the PVT trajectory buffer in the memory, loaded with the "XYPVTLoadToMemory" function.

For a more thorough description of the PVT trajectory capability, please refer to the XPS Motion Tutorial section Trajectories/PVT Trajectories.

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

XYPVTResetInMemory $SocketID $GroupName

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	XY group name (maximum size = 250)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **XYPVTResetInMemory** (int SocketID, char *GroupName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	XY group name

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **XYPVTResetInMemory** (ByVal SocketID As Long, ByVal GroupName As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	XY group name

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **XYPVTResetInMemory** (int32 SocketID, cstring GroupName)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	cstring	XY group name

Return

– Function error code

Python

Prototype

integer **XYPVTResetInMemory** (integer SocketID, string GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	string	XY group name

Return

– Function error code

2.8.3.13 XYPVTVerification

Name

XYPVTVerification – Verifies a PVT trajectory data file.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the group type (must be an XY group): ERR_WRONG_OBJECT_TYPE (-8)
- Check the base velocity value (must be null): ERR_BASE_VELOCITY (-48)
- Check the trajectory file name length: ERR_STRING_TOO_LONG (-3)
- Check the trajectory file existence and the file format: ERR_READ_FILE (-61)
- Check the trajectory (number of elements > 0): ERR_TRAJ_EMPTY (-66)
- Check trajectory element types in file: ERR_WRONG_TYPE_DOUBLE (-14)
- Check the end output velocity (must be null): ERR_TRAJ_FINAL_VELOCITY (-70)
- Check the velocity (Minimum Velocity <= Velocity <= Maximum Velocity): ERR_TRAJ_VEL_LIMIT (-68)
- Check the acceleration (Minimum acc. <= acceleration <= Maximum acc.): ERR_TRAJ_ACC_LIMIT (-69)
- Check the delta time (Delta Time > 0): ERR_TRAJ_TIME (-75)

Description

This function verifies the possible execution of a PVT trajectory. The results of the verification can be gathered with the "XYPVTVerificationResultGet" function. The trajectory file must be stored in the folder "\Admin\Public\Trajectory" of the XPS controller. If the trajectory cannot be initialized (message queue or task error) then ERR_TRAJ_INITIALIZATION (-72) is returned.

This function can be executed at any time and is independent of trajectory execution. This function performs the following:

- Checks the trajectory file for data coherence.
- Calculates the trajectory limits, which are: the required travel per positioner, the maximum possible trajectory velocity and the maximum possible trajectory acceleration. This function helps define the parameters for the trajectory execution.
- If all is OK, it returns "SUCCESS" (0). Otherwise, it returns a corresponding error.

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

The "XYPVTVerification" function is independent of the "XYPVTExecution" function. It is recommended, but not necessary to execute this function before executing a PVT trajectory.

For a more thorough description of the PVT trajectory capability, please refer to the XPS Motion Tutorial section Trajectories/PVT Trajectories.

Error codes

- ERR_BASE_VELOCITY (-48)
- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_READ_FILE (-61)
- ERR_STRING_TOO_LONG (-3)
- ERR_TRAJ_EMPTY (-66)
- ERR_TRAJ_ACC_LIMIT (-69)
- ERR_TRAJ_FINAL_VELOCITY (-70)
- ERR_TRAJ_INITIALIZATION (-72)
- ERR_TRAJ_TIME (-75)
- ERR_TRAJ_VEL_LIMIT (-68)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

XYPVTVerification $SocketID $GroupName $FileName

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	XY group name (maximum size = 250)
–	FileName	string	Trajectory file name (maximum size = 250)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **XYPVTVerification** (int SocketID, char *GroupName, char *FileName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	XY group name
– FileName	char *	Trajectory file name (maximum size = 250)

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **XYPVTVerification** (ByVal SocketID As Long, ByVal GroupName As String, ByVal FileName As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	XY group name
– FileName	string	Trajectory file name (maximum size = 250)

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **XYPVTVerification** (int32 SocketID, cstring GroupName, cstring FileName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	XY group name
– FileName	cstring	Trajectory file name (maximum size = 250)

Return

– Function error code

Python

Prototype

integer **XYPVTVerification** (integer SocketID, string GroupName, string FileName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	XY group name
– FileName	string	Trajectory file name (maximum size = 250)

Return

– Function error code

2.8.3.14 XYPVTVerificationResultGet

Name

XYPVTVerificationResultGet – Returns the results of "**XYPVTVerification**" function.

Input tests

- - Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the group type (must be an XY group): ERR_WRONG_OBJECT_TYPE (-8)
- Check the positioner name length: ERR_STRING_TOO_LONG (-3)
- Check the positioner name: ERR_POSITIONER_NAME (-18)
- Check the last XY PVT verification (must be done): ERR_NOT_ALLOWED_ACTION (-22)
- Valid output parameter type: ERR_WRONG_TYPE_CHAR (-13), ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function returns the results of the previous "XYPVTVerification" function, positioner by positioner. The results are the travel requirements (min and max values), the possible maximum velocity and the possible maximum acceleration.

If no verification previously done then ERR_NOT_ALLOWED_ACTION (-22) is returned.

For a more thorough description of the PVT trajectory capability, please refer to the XPS Motion Tutorial section Trajectories/PVT Trajectories.

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_POSITIONER_NAME (-18)
- ERR_STRING_TOO_LONG (-3)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13),
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

XYPVTVerificationResultGet $SocketID $PositionerName FileName MinimumPosition MaximumPosition MaximumVelocity MaximumAcceleration

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– PositionerName	string	XY positioner name (maximum size = 250)

Output parameters

– FileName	string	Examined trajectory file name (maximum size = 250)
– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– MaximumVelocity	double	Maximum trajectory velocity (units/seconds)
– MaximumAcceleration	double	Maximum trajectory acceleration (units/seconds²)

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **XYPVTVerificationResultGet** (int SocketID, char *PositionerName, char * FileName, double * MinimumPosition, double * MaximumPosition, double * MaximumVelocity, double * MaximumAcceleration)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– PositionerName	char *	XY positioner name

Output parameters

– FileName	char *	Examined trajectory file name (maximum size = 250)
– MinimumPosition	double *	Minimum position (units)
– MaximumPosition	double *	Maximum position (units)
– MaximumVelocity	double *	Maximum trajectory velocity (units/seconds)
– MaximumAcceleration	double *	Maximum trajectory acceleration (units/seconds²)

Return

– Function error code

Visual Basic

Prototype

Long **XYPVTVerificationResultGet** (ByVal SocketID As Long, ByVal PositionerName As String, ByVal FileName As String, MinimumPosition As Double, MaximumPosition As Double, MaximumVelocity As Double, MaximumAcceleration As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– PositionerName	string	XY positioner name

Output parameters

– FileName	string	Examined trajectory file name (maximum size = 250)
– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– MaximumVelocity	double	Maximum trajectory velocity (units/seconds)
– MaximumAcceleration	double	Maximum trajectory acceleration (units/seconds²)

Return

– Function error code

Matlab

Prototype

[Error, FileName, MinimumPosition, MaximumPosition, MaximumVelocity, MaximumAcceleration] **XYPVTVerificationResultGet** (int32 SocketID, cstring PositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– PositionerName	cstring	XY positioner name

Return

– Error	int32	Function error code
– FileName	cstring	Examined trajectory file name (maximum size = 250)
– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– MaximumVelocity	double	Trajectory trajectory velocity (units/seconds)
– MaximumAcceleration	double	Trajectory trajectory acceleration (units/seconds²)

Python

Prototype

[Error, FileName, MinimumPosition, MaximumPosition, MaximumVelocity, MaximumAcceleration] **XYPVTVerificationResultGet** (integer SocketID, string PositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– PositionerName	string	XY positioner name

Return

– Error	int	Function error code
– FileName	string	Examined trajectory file name (maximum size = 250)
– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– MaximumVelocity	double	Trajectory velocity (units/seconds)
– MaximumAcceleration	double	Trajectory acceleration (units/seconds²)

2.8.4 Configuration Files

Below is an example of an XY group (named "XY") in the system.ini file. The group XY is built with two positioners named "XGantry" and "Y". The XY home search sequence is "XThenY " and no XY mapping is used.

The positioner "XGantry" uses the parameters of "MYSTAGE1" from the stages.ini file and is connected to the plug 1 of the XPS controller. The positioner "XGantry" has a secondary positioner that uses the parameters of "MYSTAGE1" from the stages.ini file and is connected to the plug 2 of the XPS controller.

The positioner "Y" uses the parameters of "MYSTAGE2" from the stages.ini file and is connected to the plug 3 of the XPS controller.

NOTE

PIDBaseFilter parameter in system.ini file is to be configured only when using the XPS-Qn Precision Platform controller.

System.ini file:

```
[GROUPS]
XYInUse = XY
[XY] ; XY group configuration
PositionerInUse = XGantry, Y
InitializationAndHomeSearchSequence = XThenY        ; Together,  XThenY or YThenX
; XY gantry motor force
XMotorForceBalance = Disabled          ; Enabled or Disabled
YOffsetForForceRatio =                 ; If XMotorForceBalance = Enabled
PrimaryYForceRatio =                   ; If XMotorForceBalance = Enabled
SecondaryYForceRatio =                 ; If XMotorForceBalance = Enabled
;--- Mapping XY
XMappingFileName =
XMappingLineNumber =
XMappingColumnNumber =
XMappingMaxPositionError=      ; must be same unit as positioner
YMappingFileName =
YMappingLineNumber =
YMappingColumnNumber =
YMappingMaxPositionError=      ; must be same unit as positioner
[XY.X]
PlugNumber = 1
StageName = MYSTAGE1
; --- Time flasher
TimeFlasherBaseFrequency =     ; default value 40e6, must be between 39.5e6 and
                               40.5e6 Hz
```

;--- PIDBase filter (take into account the base movements, effective only with PIDFFAcceleration corrector) ***To be added only when using XPS-Q8 Precision Platform controller***

PIDBaseFilter = ; Enabled or Disabled

MovingMass = ; f PIDBaseFilter = Enabled

StaticMass = ; If PIDBaseFilter = Enabled

Viscosity = ; If PIDBaseFilter = Enabled

Stiffness = ; If PIDBaseFilter = Enabled

***;--- If Gantry* (Secondary positioner X2)**

SecondaryPositionerGantry = ; Enabled or Disabled

SecondaryPlugNumber = 2 ; If SecondaryPositionerGantry = Enabled

SecondaryStageName = MYSTAGE1 ;If SecondaryPositionerGantry = Enabled

SecondaryPositionerGantryEndReferencingPosition = 0 ;If SecondaryPositionerGantry = Enabled

SecondaryPositionerGantryEndReferencingTolerance =1 ;If SecondaryPositionerGantry = Enabled

SecondaryPositionerGantryOffsetAfterInitialization = 0 ;If SecondaryPositionerGantry = Enabled

SecondaryPositionerGantryMotorEnableDelay = ;seconds,if SecondaryPositionerGantry = Enabled

[XY.Y]

PlugNumber = 3

StageName = MYSTAGE2

; --- Time flasher

TimeFlasherBaseFrequency = ; default value 40e6, must be between 39.5e6 and 40.5e6 Hz

;--- PIDBase filter (take into account the base movements, effective only with PIDFFAcceleration corrector)

PIDBaseFilter = ; Enabled or Disabled

MovingMass = ; If PIDBaseFilter = Enabled

StaticMass = ; If PIDBaseFilter = Enabled

Viscosity = ; If PIDBaseFilter = Enabled

Stiffness = ; If PIDBaseFilter = Enabled

;---- Secondary positioner (Y2)

; None

Stages.ini file:

[MYSTAGE1]

MYSTAGE positioner configuration => See § "Positioner: Configuration files"

[MYSTAGE2]

MYSTAGE positioner configuration => See § "Positioner: Configurationfiles"

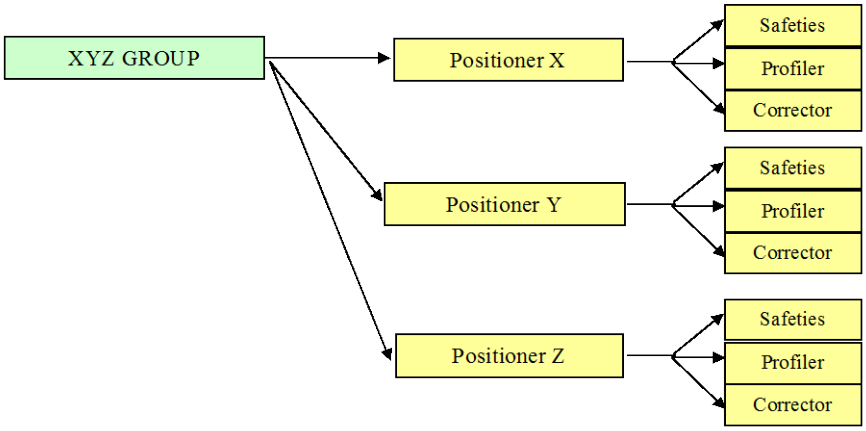
2.9 XYZ Group

2.9.1 Description

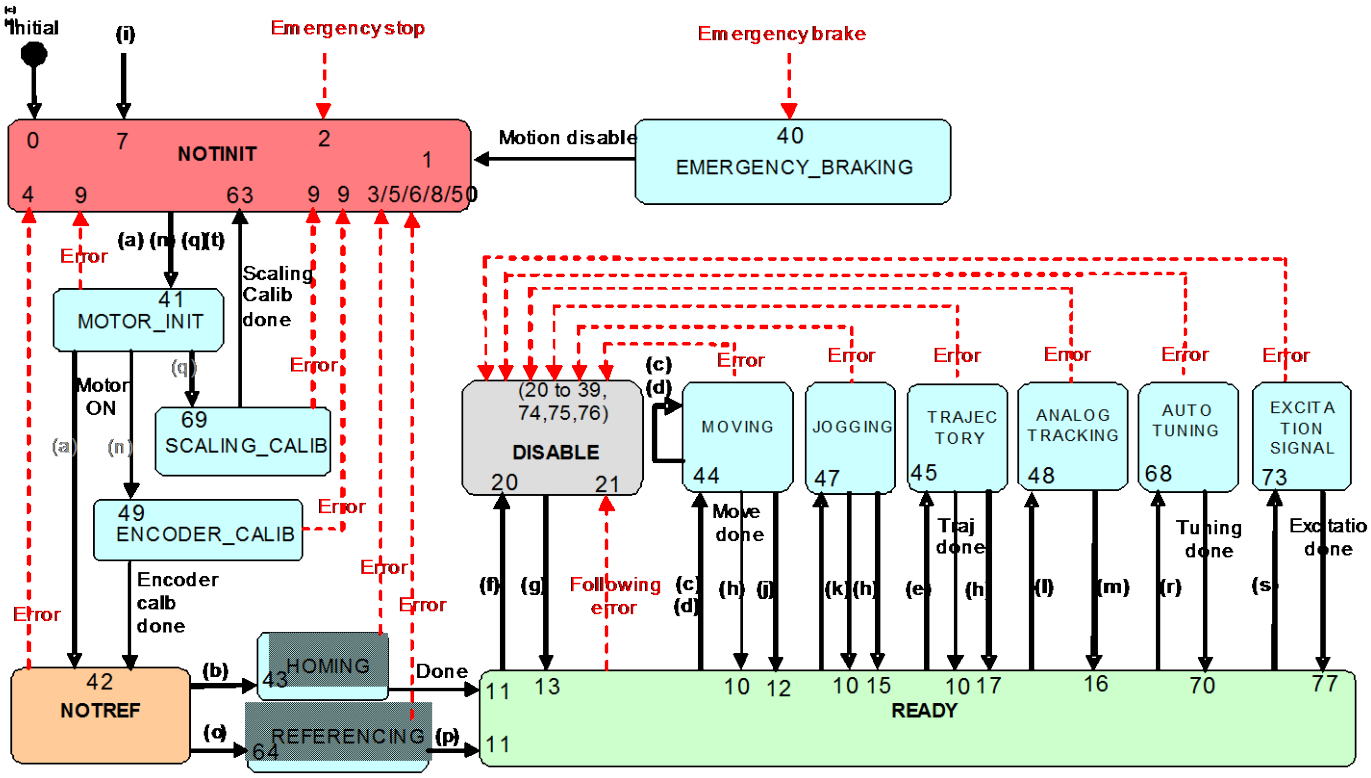
An XYZ group is a three positioner object, typically in an orthogonal XYZ configuration.

It includes an XYZ mapping feature: XYZ = f(XYZ)

It also includes 3D spline trajectories.



2.9.2 State Diagram



Called functions:

(a) GroupInitialize
(b) GroupHomeSearch
(c) GroupMoveAbsolute
(d) GroupMoveRelative
(e) XYZSplineExecution
(f) GroupMotionDisable
(g) GroupMotionEnable
(h) GroupMoveAbort
(i) GroupKill or KillAll
(j) GroupJogModeEnable
(k) GroupJogModeDisable
(l) GroupAnalogTrackingModeEnable
(m) GroupAnalogTrackingModeDisable
(n) GroupInitializeWithEncoderCalibration
(o) GroupReferencingStart
(p) GroupReferencingStop
(q) PositionerAccelerationAutoScaling
(r) PositionerCorrectorAutoTuning
(s) PositionerExcitationSignalSet
(t) GroupInitializeNoEncoderReset

2.9.3 Specific Function Description

2.9.3.1 XYZGroupPositionCorrectedProfilerGet

Name

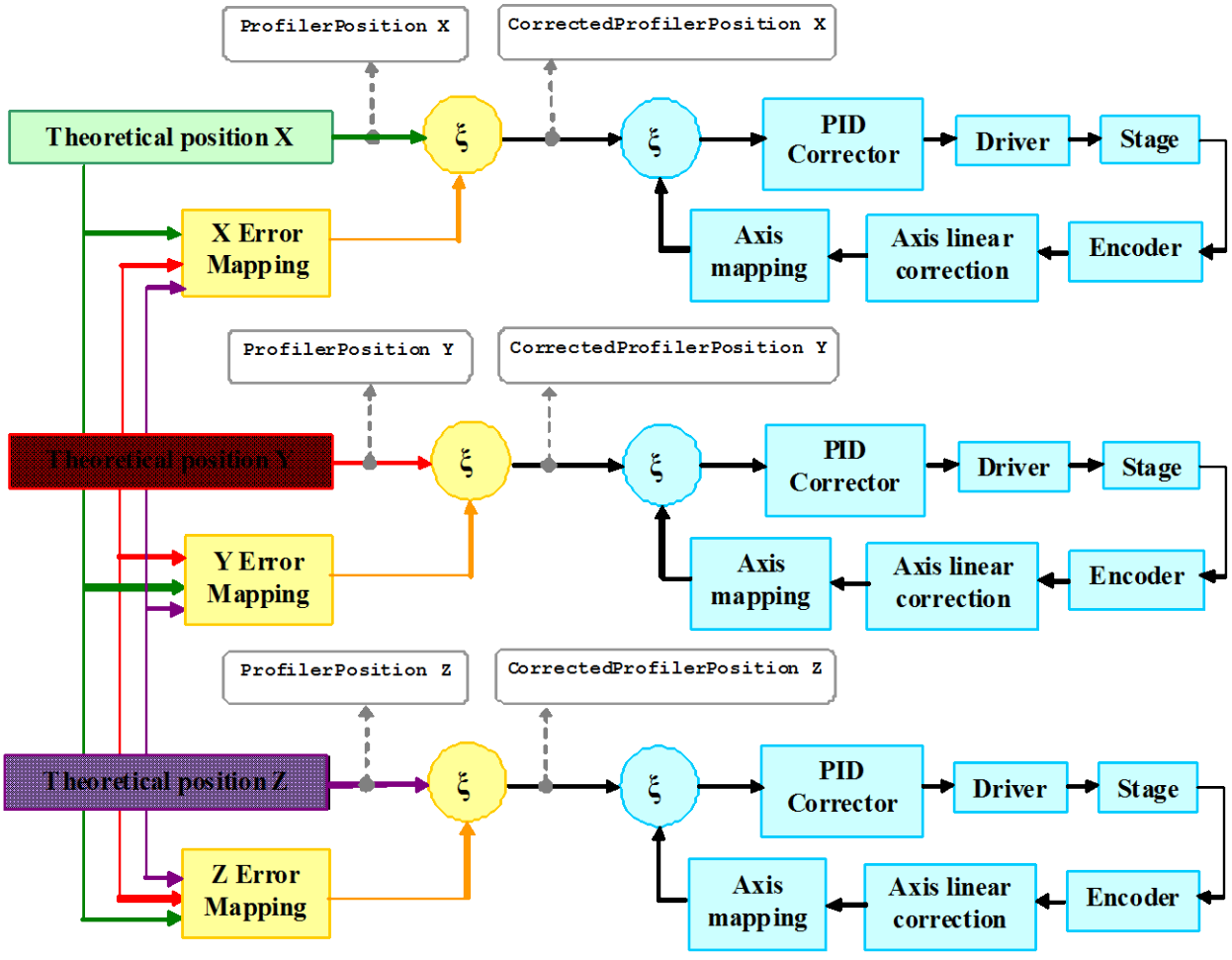
XYSGroupPositionCorrectedProfilerGet – Returns the corrected profiler position for all positioners of an XYZ group.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid object type: ERR_WRONG_OBJECT_TYPE (-8)
- Valid group type (must be an XYZ group): ERR_POSITIONER_NAME (-18)
- Valid group name: ERR_GROUP_NAME (-19)
- Valid output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function corrects a theoretical position wich is recalculated with the XYZ mapping correction.



This function applies the XYZ mapping on the theoretical user positions and returns the corrected positions. These corrected profiler positions (X, Y and Z) take the XYZ mapping correction into account.

NOTE

This function is only allowed with an XYZ group.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

XYZGroupPositionCorrectedProfilerGet SocketID GroupName PositionX PositionY PositionZ CorrectedPositionX CorrectedPositionY CorrectedPositionZ

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	XYZ group name (maximum size = 250)
– PositionX	floating point	Theoretical position X
– PositionY	floating point	Theoretical position Y
– PositionZ	floating point	Theoretical position Z

Output parameters

– CorrectedPositionX	floating point	Corrected theoretical position X
– CorrectedPositionY	floating point	Corrected theoretical position Y
– CorrectedPositionZ	floating point	Corrected theoretical position Z

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **XYZGroupPositionCorrectedProfilerGet** (int SocketID, char *GroupName, double PositionX, double PositionY, double PositionZ, double * CorrectedPositionX, double * CorrectedPositionY, double * CorrectedPositionZ)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	XYZ group name
– PositionX	double	Theoretical position X
– PositionY	double	Theoretical position Y
– PositionZ	double	Theoretical position Z

Output parameters

– CorrectedPositionX	double *	Corrected theoretical position X
– CorrectedPositionY	double *	Corrected theoretical position Y
– CorrectedPositionZ	double *	Corrected theoretical position Z

Return

– Function error code

Visual Basic

Prototype

Long **XYZGroupPositionCorrectedProfilerGet** (ByVal SocketID As Long, ByVal GroupName As String, ByVal PositionX As Double, ByVal PositionY As Double, ByVal PositionZ As Double, CorrectedPositionX As Double, CorrectedPositionY As Double, CorrectedPositionZ As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	XYZ group name
– PositionX	double	Theoretical position X
– PositionY	double	Theoretical position Y
– PositionZ	double	Theoretical position Z

Output parameters

– CorrectedPositionX	double	Corrected theoretical position X
– CorrectedPositionY	double	Corrected theoretical position Y
– CorrectedPositionZ	double	Corrected theoretical position Z

Return

– Function error code

Matlab

Prototype

[Error, CorrectedPositionX, CorrectedPositionY, CorrectedPositionZ]
XYZGroupPositionCorrectedProfilerGet (int32 SocketID, cstring GroupName, double PositionX, double PositionY, double PositionZ)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	XYZ group name
– PositionX	double	Theoretical position X
– PositionY	double	Theoretical position Y
– PositionZ	double	Theoretical position Z

Return

– Error	int32	Function error code
– CorrectedPositionX	doubletPtr	Corrected theoretical position X
– CorrectedPositionY	doubletPtr	Corrected theoretical position Y
– CorrectedPositionZ	doubletPtr	Corrected theoretical position Z

Python

Prototype

[Error, CorrectedPositionX, CorrectedPositionY, CorrectedPositionZ]
XYZGroupPositionCorrectedProfilerGet (integer SocketID, string GroupName, double PositionX, double PositionY, double PositionZ)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	XYZ group name
– PositionX	double	Theoretical position X
– PositionY	double	Theoretical position Y
– PositionZ	double	Theoretical position Z

Return

– Error	integer	Function error code
– CorrectedPositionX	doubletPtr	Corrected theoretical position X
– CorrectedPositionY	doubletPtr	Corrected theoretical position Y
– CorrectedPositionZ	doubletPtr	Corrected theoretical position Z

2.9.3.2 XYZGroupPositionPCORawEncoderGet

Name

XYZGroupPositionPCORawEncoderGet – Returns the PCO raw encoder positions of an XYZ group.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Valid object type: ERR_WRONG_OBJECT_TYPE (-8)
- Valid group type (must be a XYZ group): ERR_POSITIONER_NAME (-18)
- Valid group name: ERR_GROUP_NAME (-19)
- Valid output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function returns the X, Y and Z PCO raw encoder positions from the X, Y and Z user positions.

NOTE

This function is only allowed with a XYZ group.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

XYZGroupPositionPCORawEncoderGet SocketID GroupName PositionX PositionY PositionZ PCORawPositionX PCORawPositionY PCORawPositionZ

Input parameters

SocketID integer Socket identifier gets by the "TCP_ConnectToServer" function

GroupName String XYZ group name (maximum size = 250)

- PositionX floating point User position X
- PositionY floating point User position Y
- PositionZ floating point User position Z

Output parameters

- PCORawPositionX floating point PCO Raw position X
- PCORawPositionY floating point PCO Raw position Y
- PCORawPositionZ floating point PCO Raw position Z

Return

- TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **XYZGroupPositionPCORawEncoderGet** (int SocketID, char *GroupName, double PositionX, double PositionY, double PositionZ, double * PCORawPositionX, double * PCORawPositionY, double * *PCORawPositionZ)

Input parameters

SocketID int Socket identifier gets by the "TCP_ConnectToServer"function

GroupName char * XYZ group name

- PositionX double User position X
- PositionY double User position Y
- PositionZ double User position Z

Output parameters

- PCORawPositionX double * PCO Raw position X
- PCORawPositionY double * PCO Raw position Y
- PCORawPositionZ double * PCO Raw position Z

Return

- Function error code

Visual Basic

Prototype

Long **XYZGroupPositionPCORawEncoderGet** (ByVal SocketID As Long, ByVal GroupName As String, ByVal PositionX As Double, ByVal PositionY As Double, ByVal PositionZ As Double, PCORawPositionX As Double, PCORawPositionY As Double, PCORawPositionZ As Double)

Input parameters

SocketID Long Socket identifier gets by the "TCP_ConnectToServer" function

GroupNameString XYZ group name

- PositionX double User position X
- PositionY double User position Y
- PositionZ double User position Z

Output parameters

- PCORawPositionX double PCO Raw position X
- PCORawPositionY double PCO Raw position Y
- PCORawPositionZ double PCO Raw position Z

Return

- Function error code

Matlab

Prototype

[Error,PCORawPositionX,PCORawPositionY,PCORawPositionZ] **XYZGroupPositionPCORawEncoderGet** (int32 SocketID, cstring GroupName, double PositionX, double PositionY, double PositionZ)

Input parameters

SocketID int32 Socket identifier gets by the "TCP_ConnectToServer"function

GroupName cstring XYZ group name

- PositionX double User position X
- PositionY double User position Y
- PositionZ double User position Z

Return

- Error int32 Function error code
- PCORawPositionX doubletPtr PCO Raw position X
- PCORawPositionY doubletPtr PCO Raw position Y
- PCORawPositionZ doubletPtr PCO Raw position Z

Python

Prototype

[Error,PCORawPositionX,PCORawPositionY,PCORawPositionZ]
XYZGroupPositionPCORawEncoderGet (integer SocketID, string GroupName, double PositionX, double PositionY, double PositionZ)

Input parameters

SocketID integerSocket identifier gets by the "TCP_ConnectToServer"function

GroupNameString XYZ group name

- PositionX double User position X
- PositionY double User position Y
- PositionZ double User position Z

Return

- Error int Function error code
- PCORawPositionX doubletPtr PCO Raw position X
- PCORawPositionY doubletPtr PCO Raw position Y
- PCORawPositionZ doubletPtr PCO Raw position Z

2.9.3.3 XYZSplineExecution

Name

XYZSplineExecution – Executes an XYZ spline trajectory.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Group state must be "READY": ERR_NOT_ALLOWED_ACTION (-22)
- Check the base velocity value (must be null): ERR_BASE_VELOCITY (-48)
- Check the group: ERR_POSITIONER_NAME (-18)
- Check the group type (must be an XYZ group): ERR_WRONG_OBJECT_TYPE (-8)
- Check backlash (must not be enabled): ERR_NOT_ALLOWED_BACKLASH (-46)
- Check the trajectory file name length: ERR_STRING_TOO_LONG (-3)
- Check the trajectory file existence or file reading: ERR_READ_FILE (-61)
- Check the message queue filling: ERR_MSG_QUEUE (-71)
- Check the input value (velocity and acceleration > 0): ERR_PARAMETER_OUT_OF_RANGE (-17)
- Check the velocity (0 < Velocity <= TrajectoryMaximumVelocity): ERR_TRAJ_VEL_LIMIT (-68)
- Check the acceleration (0 < Acceleration <= TrajectoryMaximumAcceleration): ERR_TRAJ_ACC_LIMIT (-69)
- Check input parameter types: ERR_WRONG_TYPE_DOUBLE (-14), ERR_WRONG_TYPE_INT (-15)

Description

This function executes an XYZ Spline trajectory. The trajectory file must be stored in the folder "\ADMIN\Public\Trajectory" of the XPS controller. If the trajectory cannot be initialized (message queue or task error) then ERR_TRAJ_INITIALIZATION (-72) is returned.

Before a trajectory execution, it is recommended to check whether the trajectory is within group positioner parameter capabilities and can execute by using the "XYZSplineVerification" and "XYZSplineVerificationResultGet" functions.

For a more thorough description of the spline trajectory capability, please refer to the XPS Motion Tutorial section Trajectories/Spline Trajectories.

NOTE

In case of an ERR_GROUP_MOTION_DONE_TIMEOUT (-33) error, an ERR_FOLLOWING_ERROR (-25) error or ERR_SLAVE (-44) error, the group state becomes DISABLE. To help dertermine the error source, check the positioner errors, the hardware status and the driver status.

Error codes

- ERR_BASE_VELOCITY (-48)
- ERR_FATAL_INIT (-20)
- ERR_FOLLOWING_ERROR (-25)
- ERR_GROUP_MOTION_DONE_TIMEOUT (-33)
- ERR_IN_INITIALIZATION (-21)
- ERR_MSG_QUEUE (-71)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_NOT_ALLOWED_BACKLASH (-46)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_POSITIONER_NAME (-18)
- ERR_READ_FILE (-61)
- ERR_SLAVE (-44)
- ERR_STRING_TOO_LONG (-3)
- ERR_TRAJ_ACC_LIMIT (-69)
- ERR_TRAJ_VEL_LIMIT (-68)
- ERR_TRAJ_INITIALIZATION (-72)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14),
- SUCCESS (0): no error

TCL

Prototype

XYZSplineExecution $SocketID $GroupName $FileName $Velocity $Acceleration

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	XYZ group name (maximum size = 250)
–	FileName	string	Trajectory file name (maximum size = 250)
–	Velocity	double	Trajectory velocity (units/seconds)
–	Acceleration	double	Trajectory acceleration (units/seconds²)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **XYZSplineExecution** (int SocketID, char *GroupName, char *FileName , double Velocity, double Acceleration)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	XYZ group name
– FileName	char *	Trajectory file name (maximum size = 250)
– Velocity	double	Trajectory velocity (units/seconds)
– Acceleration	double	Trajectory acceleration (units/seconds²)

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **XYZSplineExecution** (ByVal SocketID As Long, ByVal GroupName As String, ByVal FileName As String, ByVal Velocity As Double, ByVal Acceleration As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	XYZ group name
– FileName	string	Trajectory file name (maximum size = 250)
– Velocity	double	Trajectory velocity (units/seconds)
– Acceleration	double	Trajectory acceleration (units/seconds²)

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **XYZSplineExecution** (int32 SocketID, cstring GroupName, cstring FileName, double Velocity, double Acceleration)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	XYZ group name
– FileName	cstring	Trajectory file name (maximum size = 250)
– Velocity	double	Trajectory velocity (units/seconds)
– Acceleration	double	Trajectory acceleration (units/seconds²)

Return

– Function error code

Python

Prototype

integer **XYZSplineExecution** (integer SocketID, string GroupName, string FileName, double Velocity, double Acceleration)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	XYZ group name
– FileName	string	Trajectory file name (maximum size = 250)
– Velocity	double	Trajectory velocity (units/seconds)
– Acceleration	double	Trajectory acceleration (units/seconds²)

Return

– Function error code

2.9.3.4 XYZSplineParametersGet

Name

XYZSplineParametersGet – Returns the Spline trajectory parameters.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the group type (must be a XYZ group): ERR_WRONG_OBJECT_TYPE (-8)
- Check the trajectory type (Spline): ERR_NOT_ALLOWED_ACTION (-22)
- Valid output parameter type: ERR_WRONG_TYPE_CHAR (-13), ERR_WRONG_TYPE_DOUBLE (-14), ERR_WRONG_TYPE_INT (-15)

Description

This function returns the XYZ Spline trajectory parameters (trajectory name, trajectory velocity, trajectory acceleration, current executing element number) of the current executed XYZ Spline trajectory.

For a more thorough description of the Spline trajectory capability, please refer to the XPS Motion Tutorial section Trajectories/Spline Trajectories.

Error codes

ERR_FATAL_INIT (-20)

ERR_IN_INITIALIZATION (-21)

ERR_NOT_ALLOWED_ACTION (-22)

ERR_POSITIONER_NAME (-18)

ERR_WRONG_FORMAT (-7)

ERR_WRONG_OBJECT_TYPE (-8)

ERR_WRONG_PARAMETERS_NUMBER (-9)

ERR_WRONG_TYPE_CHAR (-13)

ERR_WRONG_TYPE_DOUBLE (-14)

ERR_WRONG_TYPE_INT (-15)

SUCCESS (0): no error

TCL

Prototype

XYZSplineParametersGet $SocketID $GroupName FileName Velocity Acceleration ElementNumber

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	XYZ Group name (maximum size = 250)

Output parameters

– FileName	string	Executing trajectory file name (maximum size = 250)
– Velocity	double	Trajectory velocity (units/seconds)
– Acceleration	double	Trajectory acceleration (units/seconds²)
– ElementNumber	int	Current executing element number

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **XYZSplineParametersGet** (int SocketID, char *GroupName, char * FileName, double * Velocity, double * Acceleration, int * ElementNumber)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	XYZ Group name

Output parameters

– FileName	char *	Executing trajectory file name (maximum size = 250)
– Velocity	double *	Trajectory velocity (units/seconds)
– Acceleration	double *	Trajectory acceleration (units/seconds²)
– ElementNumber	int *	Current executing element number

Return

– Function error code

Visual Basic

Prototype

Long **XYZSplineParametersGet** (ByVal SocketID As Long, ByVal GroupName As String, ByVal FileName As String, Velocity As Double, Acceleration As Double, ElementNumber As Integer)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	XYZ Group name

Output parameters

– FileName	string	Executing trajectory file name (maximum size = 250)
– Velocity	double	Trajectory velocity (units/seconds)
– Acceleration	double	Trajectory acceleration (units/seconds²)
– ElementNumber	int	Current executing element number

Return

– Function error code

Matlab

Prototype

[Error, FileName, Velocity, Acceleration, ElementNumber] **XYZSplineParametersGet** (int32 SocketID, cstring GroupName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	XYZ Group name

Return

– Error	int32	Function error code
– FileName	cstring	Executing trajectory file name (maximum size = 250)
– Velocity	double	Trajectory velocity (units/seconds)
– Acceleration	double	Trajectory acceleration (units/seconds²)
– ElementNumber	int32	Current executing element number

Python

Prototype

[Error, FileName, Velocity, Acceleration, ElementNumber] **XYZSplineParametersGet** (integer SocketID, string GroupName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	XYZ Group name

Return

– Error	int	Function error code
– FileName	string	Executing trajectory file name (maximum size = 250)
– Velocity	double	Trajectory velocity (units/seconds)
– Acceleration	double	Trajectory acceleration (units/seconds²)
– ElementNumber	int	Current executing element number

2.9.3.5 XYZSplineVerification

Name

XYZSplineVerification – Verifies an XYZ Spline trajectory data file.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the group type (must be an XYZ group): ERR_WRONG_OBJECT_TYPE (-8)
- Check the base velocity value (must be null): ERR_BASE_VELOCITY (-48)
- Check the trajectory file name length: ERR_STRING_TOO_LONG (-3)
- Check the trajectory file existence and the file format: ERR_READ_FILE (-61)
- Check the trajectory (number of elements > 0): ERR_TRAJ_EMPTY (-66)
- Check trajectory element types in file: ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function verifies the execution of an XYZ Spline trajectory. The results can be gathered with the "XYZSplineVerificationResultGet" function. The trajectory file must be stored in the folder "\ADMIN\Public\Trajectory" of the XPS controller. If the trajectory cannot be initialized (message queue or task error) then ERR_TRAJ_INITIALIZATION (-72) is returned.

This function can be executed at any time and is independent of the trajectory execution. This function performs the following:

- Checks the trajectory file for data coherence.
- Calculates the trajectory limits, which are: the required travel per positioner, the maximum possible trajectory velocity and the maximum possible trajectory acceleration. This function helps define the parameters for the trajectory execution.
- If all is OK, it returns "SUCCESS" (0). Otherwise, it returns a corresponding error.

NOTE

The "XYZSplineVerification" function is independent of the "XYZSplineExecution" function. It is recommended, but not necessary, to execute this function before executing a Spline trajectory.

For a more thorough description of the Spline trajectory capability, please refer to the XPS Motion Tutorial section Trajectories/Spline Trajectories.

Error codes

- ERR_BASE_VELOCITY (-48)
- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_READ_FILE (-61)
- ERR_STRING_TOO_LONG (-3)
- ERR_TRAJ_EMPTY (-66)
- ERR_TRAJ_INITIALIZATION (-72)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

XYZSplineVerification $SocketID $GroupName $FileName

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	XYZ group name (maximum size = 250)
– FileName	string	Trajectory file name (maximum size = 250)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **XYZSplineVerification** (int SocketID, char *GroupName, char *FileName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	XYZ group name
– FileName	char *	Trajectory file name (maximum size = 250)

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **XYZSplineVerification** (ByVal SocketID As Long, ByVal GroupName As String, ByVal FileName As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	XYZ group name
– FileName	string	Trajectory file name (maximum size = 250)

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **XYZSplineVerification** (int32 SocketID, cstring GroupName, cstring FileName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	XYZ group name
– FileName	cstring	Trajectory file name (maximum size = 250)

Return

– Function error code

Python

Prototype

integer **XYZSplineVerification** (integer SocketID, string GroupName, string FileName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	XYZ group name
– FileName	string	Trajectory file name (maximum size = 250)

Return

– Function error code

2.9.3.6 XYZSplineVerificationResultGet

Name

XYZSplineVerificationResultGet – Returns the results of the previous "**XYZSplineVerification**" function.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the group type (must be a XYZ group): ERR_WRONG_OBJECT_TYPE (-8)
- Check the positioner name length: ERR_STRING_TOO_LONG (-3)
- Check the positioner name: ERR_POSITIONER_NAME (-18)
- Check the last XYZ Spline verification (must be done): ERR_NOT_ALLOWED_ACTION (-22)
- Valid output parameter type: ERR_WRONG_TYPE_CHAR (-13), ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function returns the results of the previous "XYZSplineVerification" function, positioner by positioner. The results are the travel requirements (min and max values), the possible maximum velocity and the possible maximum acceleration.

If no verification was previously done then ERR_NOT_ALLOWED_ACTION (-22) is returned.

For a more thorough description of the XYZ Spline trajectory capability, please refer to the XPS Motion Tutorial section Trajectories/Spline Trajectories.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_POSITIONER_NAME (-18)
- ERR_STRING_TOO_LONG (-3)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13),
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

XYZSplineVerificationResultGet $SocketID $PositionerName FileName MinimumPosition MaximumPosition MaximumVelocity MaximumAcceleration

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– PositionerName	string	XYZ positioner name (maximum size = 250)

Output parameters

– FileName	string	Examined trajectory file name (maximum size = 250)
– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– MaximumVelocity	double	Maximum trajectory velocity (units/seconds)
– MaximumAcceleration	double	Maximum trajectory acceleration (units/seconds²)

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **XYZSplineVerificationResultGet** (int SocketID, char * PositionerName, char * FileName, double * MinimumPosition, double * MaximumPosition, double * MaximumVelocity, double * MaximumAcceleration)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– PositionerName	char *	XYZ positioner name

Output parameters

– FileName	char *	Examined trajectory file name (maximum size = 250)
– MinimumPosition	double *	Minimum position (units)
– MaximumPosition	double *	Maximum position (units)
– MaximumVelocity	double *	Maximum trajectory velocity (units/seconds)
– MaximumAcceleration	double *	Maximum trajectory acceleration (units/seconds²)

Return

– Function error code

Visual Basic

Prototype

Long **XYZSplineVerificationResultGet** (ByVal SocketID As Long, ByVal PositionerName As String, ByVal FileName As String, MinimumPosition As Double, MaximumPosition As Double, MaximumVelocity As Double, MaximumAcceleration As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– PositionerName	string	XYZ positioner name

Output parameters

– FileName	string	Examined trajectory file name (maximum size = 250)
– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– MaximumVelocity	double	Maximum trajectory velocity (units/seconds)
– MaximumAcceleration	double	Maximum trajectory acceleration (units/seconds²)

Return

– Function error code

Matlab

Prototype

[Error, FileName, MinimumPosition, MaximumPosition, MaximumVelocity, MaximumAcceleration] **XYZSplineVerificationResultGet** (int32 SocketID, cstring PositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– PositionerName	cstring	XYZ positioner name

Return

– Error	int32	Function error code
– FileName	cstring	Examined trajectory file name (maximum size = 250)
– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– MaximumVelocity	double	Trajectory trajectory velocity (units/seconds)
– MaximumAcceleration	double	Trajectory trajectory acceleration (units/seconds²)

Python

Prototype

[Error, FileName, MinimumPosition, MaximumPosition, MaximumVelocity, MaximumAcceleration] **XYZSplineVerificationResultGet** (integer SocketID, string PositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– PositionerName	string	XYZ positioner name

Return

– Error	int	Function error code
– FileName	string	Examined trajectory file name (maximum size = 250)
– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– MaximumVelocity	double	Trajectory velocity (units/seconds)
– MaximumAcceleration	double	Trajectory acceleration (units/seconds²)

2.9.4 Configuration Files

Below is an example of an XYZ group (named "XYZ") in the system.ini file. The XYZ group is built with three positioners named "X" , "Y" and "Z". For example, positioner "X" uses the parameters of "MYSTAGE1" from the stages.ini file and is connected to the plug 1 of the XPS controller. The HomeSearchSequence is "Together", and an XYZ mapping is used.

System.ini file:

```
[GROUPS]
XYZInUse = XYZ
[XYZ]                                          ; XYZ group configuration
PositionerInUse = X, Y, Z
InitializationAndHomeSearchSequence = Together      ; Together or  XThenYThenZ
;--- Mapping XYZ
XMappingFileName = MatriceX.txt
XMappingXLineNumber = 5
XMappingYColumnNumber = 3
XMappingZDimNumber = 3
XMappingMaxPositionError = 2    ; must be same unit as positioner
YMappingFileName = MatriceY.txt
YMappingXLineNumber = 5
YMappingYColumnNumber = 3
YMappingZDimNumber = 3
YMappingMaxPositionError = 2    ; must be same unit as positioner
ZMappingFileName = MatriceZ.txt
ZMappingXLineNumber = 5
ZMappingYColumnNumber = 3
ZMappingZDimNumber = 3
ZMappingMaxPositionError = 2    ; must be same unit as positioner
[XYZ.X]
PlugNumber = 1
StageName = MYSTAGE1
X positioner configuration => See § "Positioner: Configuration files"
[XYZ.Y]
PlugNumbere = 2
StageName = MYSTAGE2
X positioner configuration => See § "Positioner: Configuration files"
[XYZ.Z]
PlugNumber = 3
StageName = MYSTAGE3
X positioner configuration => See § "Positioner: Configuration files"
```

Stages.ini file:

[MYSTAGE1]

MYSTAGE1 configuration => See § "Positioner: Configuration files"

[MYSTAGE2]

MYSTAGE2 configuration => See § "Positioner: Configuration files"

[MYSTAGE3]

MYSTAGE3 configuration => See § "Positioner: Configuration files"

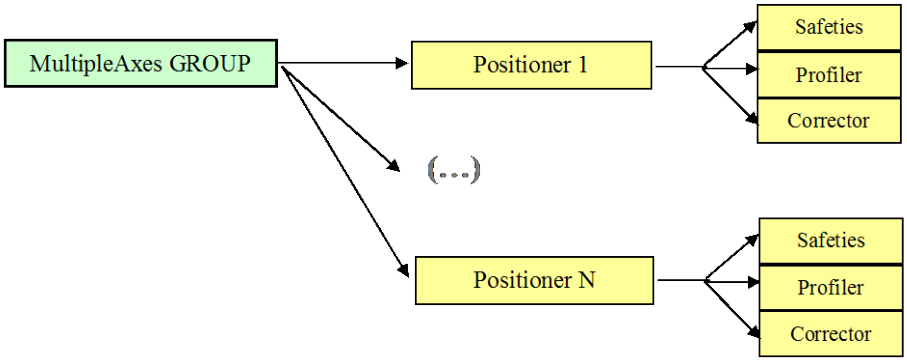
2.10 MultipleAxes Group

2.10.1 Description

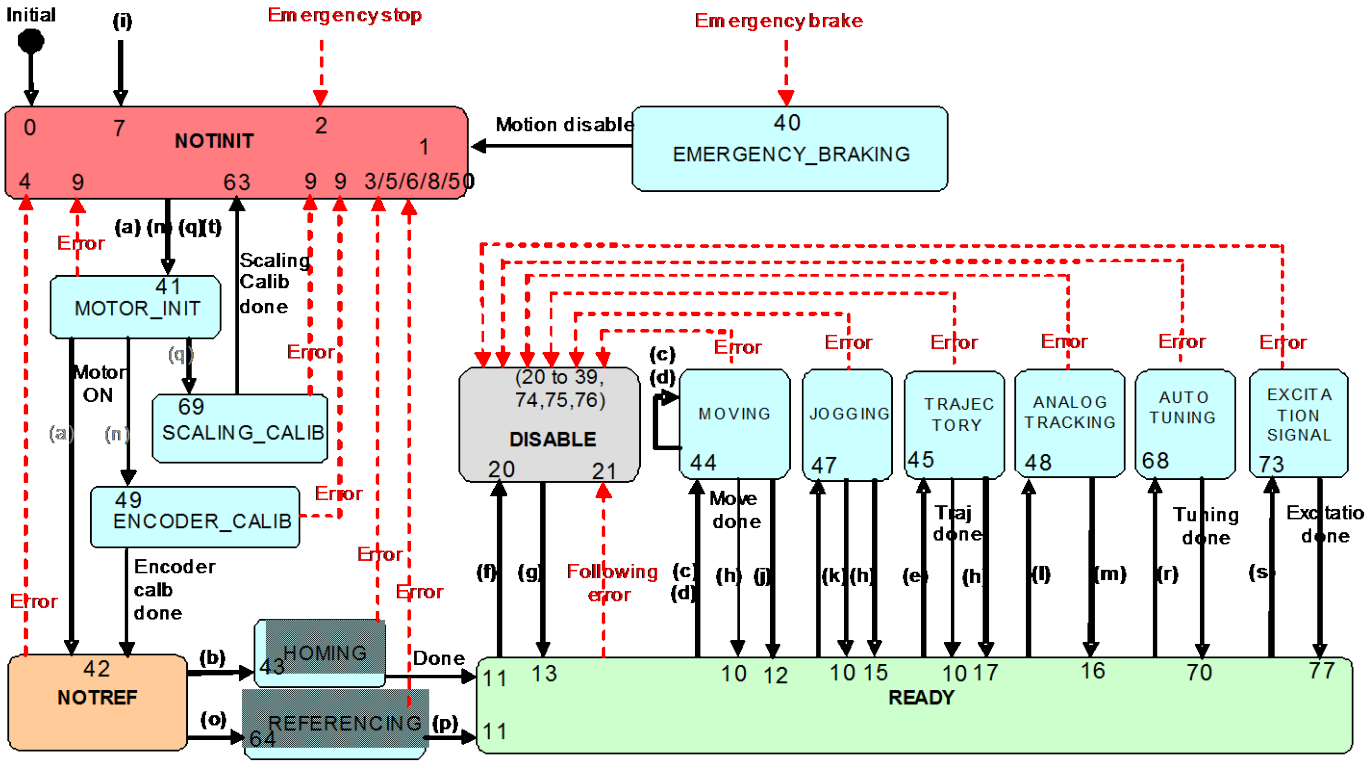
A MultipleAxes group is an n-positioner(s) object, where n can be any number from one to eight (N max = 8 or 12).

A MultipleAxes group can be used in GANTRY mode (dual positioner for one or some positioners).

It includes the PVT trajectory.



2.10.2 State Diagram



Called functions:

(a) GroupInitialize
(b) GroupHomeSearch
(c) GroupMoveAbsolute
(d) GroupMoveRelative
(e) MultipleAxesPVTExecution
(f) GroupMotionDisable
(g) GroupMotionEnable
(h) GroupMoveAbort
(i) GroupKill or KillAll
(j) GroupJogModeEnable
(k) GroupJogModeDisable
(l) GroupAnalogTrackingModeEnable
(m) GroupAnalogTrackingModeDisable
(n) GroupInitializeWithEncoderCalibration
(o) GroupReferencingStart
(p) GroupReferencingStop
(q) PositionerAccelerationAutoScaling
(r) PositionerCorrectorAutoTuning
(s) PositionerExcitationSignalSet
(t) GroupInitializeNoEncoderReset

2.10.3 Specific Function Description

2.10.3.1 MultipleAxesPVTExecution

Name

MultipleAxesPVTExecution – Executes a PVT trajectory.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Group state must be "READY": ERR_NOT_ALLOWED_ACTION (-22)
- Check backlash (must not be enabled): ERR_NOT_ALLOWED_BACKLASH (-46)
- Check the base velocity value (must be null): ERR_BASE_VELOCITY (-48)
- Check the group: ERR_POSITIONER_NAME (-18)
- Check the group type (must be an XYZ group): ERR_WRONG_OBJECT_TYPE (-8)
- Check the input parameter type: ERR_WRONG_TYPE_INT (-15)
- Check the input value (Number of executions > 0): ERR_PARAMETER_OUT_OF_RANGE (-17)
- Check the trajectory file name length: ERR_STRING_TOO_LONG (-3)
- Check the trajectory file existence or file reading: ERR_READ_FILE (-61)
- Check the message queue filling: ERR_MSG_QUEUE (-71)

Description

This function executes a PVT (Position Velocity Time) trajectory. The trajectory file must be stored in the folder "\Admin\Public\Trajectory" of the XPS controller. If the trajectory cannot be initialized (message queue or task error) then ERR_TRAJ_INITIALIZATION (-72) is returned.

Before a trajectory execution, it is recommended to check whether the trajectory is within group positioner parameter capabilities by using "MultipleAxesPVTVerification" and "MultipleAxesPVTVerificationResultGet" functions.

For a more thorough description of the PVT trajectory capability, please refer to the XPS Motion Tutorial section Trajectories/PVT Trajectories.

NOTE

In case of an ERR_GROUP_MOTION_DONE_TIMEOUT (-33) error, an ERR_FOLLOWING_ERROR (-25) error or ERR_SLAVE (-44) error, the group state becomes DISABLE. To help determine the error source, check the positioner errors, the hardware status and the driver status.

Error codes

- ERR_BASE_VELOCITY (-48)
- ERR_FATAL_INIT (-20)
- ERR_FOLLOWING_ERROR (-25)
- ERR_GROUP_MOTION_DONE_TIMEOUT (-33)
- ERR_IN_INITIALIZATION (-21)
- ERR_MSG_QUEUE (-71)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_NOT_ALLOWED_BACKLASH (-46)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_POSITIONER_NAME (-18)
- ERR_READ_FILE (-61)
- ERR_SLAVE (-44)
- ERR_STRING_TOO_LONG (-3)
- ERR_TRAJ_INITIALIZATION (-72)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

MultipleAxesPVTExecution $SocketID $GroupName $FileName $Velocity $Acceleration

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	MultipleAxes group name (maximum size = 250)
–	FileName	string	Trajectory file name (maximum size = 250)
–	ExecutionNumber	int	Number of trajectory executions

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **MultipleAxesPVTExecution** (int SocketID, char *GroupName, char *FileName , double Velocity, double Acceleration)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	MultipleAxes group name
– FileName	char *	Trajectory file name (maximum size = 250)
– ExecutionNumber	int	Number of trajectory executions

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **MultipleAxesPVTExecution** (ByVal SocketID As Long, ByVal GroupName As String, ByVal FileName As String, ByVal Velocity As Double, ByVal Acceleration As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	MultipleAxes group name
– FileName	string	Trajectory file name (maximum size = 250)
– ExecutionNumber	long	Number of trajectory executions

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **MultipleAxesPVTExecution** (int32 SocketID, cstring GroupName, cstring FileName, double Velocity, double Acceleration)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	MultipleAxes group name
– FileName	cstring	Trajectory file name (maximum size = 250)
– ExecutionNumber	int32	Number of trajectory executions

Return

– Function error code

Python

Prototype

integer **MultipleAxesPVTExecution** (integer SocketID, string GroupName, string FileName, double Velocity, double Acceleration)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	MultipleAxes group name
– FileName	string	Trajectory file name (maximum size = 250)
– ExecutionNumber	int	Number of trajectory executions

Return

– Function error code

2.10.3.2 MultipleAxesPVTLoadToMemory

Name

MultipleAxesPVTLoadToMemory – Loads a Multiple Axes PVT trajectory line.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the group (must not be a positioner): ERR_GROUP_NAME (-19)
- Check the gantry mode (must not be a gantry): ERR_POSITIONER_NAME (-18)
- Check the group type (must be an MultipleAxes group): ERR_WRONG_OBJECT_TYPE (-8)
- Check the trajectory data length: ERR_STRING_TOO_LONG (-3)
- Check the input value (> 0): ERR_PARAMETER_OUT_OF_RANGE (-17)

Description

This function loads a line of PVT trajectory into the XPS memory. Each trajectory element must be separated by a comma. To verify or to execute the PVT trajectory loaded in memory, use the string "**FromMemory**" instead of a FileName.

For a more thorough description of the PVT trajectory capability, please refer to the XPS Motion Tutorial section Trajectories/PVT Trajectories.

NOTE

All of the previous PVT functions, when called with the string "FromMemory" instead of a FileName, will perform the same operation as the PVT trajectory in RAM as it does from a disk.

Example:

MultipleAxesPVTLoadToMemory(myMultipleAxes, 0.5,1,2,2,4,1,2)

MultipleAxesPVTVerification (myMultipleAxes, FromMemory)

MultipleAxesPVTExecution(myMultipleAxes, FromMemory, 1)

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_STRING_TOO_LONG (-3)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

MultipleAxesPVTLoadToMemory $SocketID $GroupName $TrajectoryLine

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	MultipleAxes group name (maximum size = 250)
– TrajectoryLine	string	Trajectory line (maximum size = 400)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

Example in a TCL program:

```
    set code [catch "MultipleAxesPVTLoadToMemory $socketID MULTI
0.5,1,2,2,4,1,2"]
    if {$code != 0} {
        DisplayErrorAndClose $socketID $code "MultipleAxesPVTLoadToMemory"
        return
    }
```

C/C++

Prototype

int **MultipleAxesPVTLoadToMemory** (int SocketID, char *GroupName, char *TrajectoryLine)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	MultipleAxes group name
– TrajectoryLine	char *	Trajectory line (maximum size = 400)

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **MultipleAxesPVTLoadToMemory** (ByVal SocketID As Long, ByVal GroupName As String, ByVal TrajectoryLine As String, ByVal)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	MultipleAxes group name
– TrajectoryLine	string	Trajectory line (maximum size = 400)

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **MultipleAxesPVTLoadToMemory** (int32 SocketID, cstring GroupName, cstring TrajectoryLine)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	MultipleAxes group name
– TrajectoryLine	cstring	Trajectory line (maximum size = 400)

Return

– Function error code

Python

Prototype

integer **MultipleAxesPVTLoadToMemory** (integer SocketID, string GroupName, string TrajectoryLine)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	MultipleAxes group name
– TrajectoryLine	string	Trajectory file name (maximum size = 400)

Return

– Function error code

2.10.3.3 MultipleAxesPVTParametersGet

Name

MultipleAxesPVTParametersGet – Returns the PVT trajectory parameters.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the group type (must be a MultipleAxes group): ERR_WRONG_OBJECT_TYPE (-8)
- Check the trajectory type (PVT): ERR_NOT_ALLOWED_ACTION (-22)
- Valid output parameter type: ERR_WRONG_TYPE_CHAR (-13), ERR_WRONG_TYPE_INT (-15)

Description

This function returns the PVT trajectory parameters (trajectory name and current executing element number) of the current executed PVT trajectory.

For a more thorough description of the PVT trajectory capability, please refer to the XPS Motion Tutorial section Trajectories/PVT Trajectories.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

MultipleAxesPVTParametersGet $SocketID $GroupName FileName ElementNumber

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	MultipleAxes group name (maximum size = 250)

Output parameters

–	FileName	string	Executing trajectory file name (maximum size = 250)
–	ElementNumber	int	Current executing element number

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **MultipleAxesPVTParametersGet** (int SocketID, char *GroupName, char * FileName, int * ElementNumber)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	char *	MultipleAxes group name

Output parameters

–	FileName	char *	Executing trajectory file name (maximum size = 250)
–	ElementNumber	int *	Current executing element number

Return

– Function error code

Visual Basic

Prototype

Long **MultipleAxesPVTParametersGet** (ByVal SocketID As Long, ByVal GroupName As String, ByVal FileName As String, ElementNumber As Integer)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	MultipleAxes group name

Output parameters

– FileName	string	Executing trajectory file name (maximum size = 250)
– ElementNumber	int	Current executing element number

Return

– Function error code

Matlab

Prototype

[Error, FileName, ElementNumber] **MultipleAxesPVTParametersGet** (int32 SocketID, cstring GroupName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	MultipleAxes group name

Return

– Error	int32	Function error code
– FileName	cstring	Executing trajectory file name (maximum size = 250)
– ElementNumber	int32	Current executing element number

Python

Prototype

[Error, FileName, ElementNumber] **MultipleAxesPVTParametersGet** (integer SocketID, string GroupName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	MultipleAxes group name

Return

– Error	int	Function error code
– FileName	string	Executing trajectory file name (maximum size = 250)
– ElementNumber	int	Current executing element number

2.10.3.4 MultipleAxesPVTPulseOutputGet

Name

MultipleAxesPVTPulseOutputGet – Returns the configuration of pulse generation of a PVT trajectory.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the group (must not be a positioner): ERR_POSITIONER_NAME (-18)
- Check the group type (must be a MultipleAxes group): ERR_WRONG_OBJECT_TYPE (-8)
- Valid output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function returns the last configuration of pulse generation of a PVT trajectory.

The pulse output configuration is defined by a start element, an end element, and a time interval in seconds.

Example:

MultipleAxesPVTPulseOutputGet (Group1, 3, 5, 0.01)

One pulse will be generated every 10 ms between the start of the 3rd element and the end of the 5th element.

Start element= 3

End element = 5

Time interval = 0.01 seconds

For a more thorough description of the PVT trajectory capability, please refer to the XPS Motion Tutorial sections Trajectories/PVT Trajectories and Ouput triggers.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

MultipleAxesPVTPulseOutputGet $SocketID $GroupName StartElement EndElement TimeInterval

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	MultipleAxes group name (maximum size = 250)

Output parameters

– StartElement	int	Start Element number
– EndElement	int	End Element number
– TimeInterval	double	Time interval (seconds)

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **MultipleAxesPVTPulseOutputGet** (int SocketID, char *GroupName, int * StartElement, int * EndElement, double * TimeInterval)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	MultipleAxes group name

Output parameters

– StartElement	int *	Start Element number
– EndElement	int *	End Element number
– TimeInterval	double *	Time interval (seconds)

Return

– Function error code

Visual Basic

Prototype

Long **MultipleAxesPVTPulseOutputGet** (ByVal SocketID As Long, ByVal GroupName As String, StartElement As Long, EndElement As Long, TimeInterval As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	MultipleAxes group name

Output parameters

– StartElement	long	Start Element number
– EndElement	long	End Element number
– TimeInterval	double	Time interval (seconds)

Return

– Function error code

Matlab

Prototype

[Error, StartElement, EndElement, TimeInterval] **MultipleAxesPVTPulseOutputGet** (int32 SocketID, cstring GroupName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	MultipleAxes Group name

Return

– Error	int32	Function error code
– StartElement	int32	Start Element number
– EndElement	int32	End Element number
– TimeInterval	double	Time interval (seconds)

Python

Prototype

[Error, StartElement, EndElement, TimeInterval] **MultipleAxesPVTPulseOutputGet** (integer SocketID, string GroupName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	MultipleAxes group name

Return

– Error	int	Function error code
– StartElement	int	Start Element number
– EndElement	int	End Element number
– TimeInterval	double	Time interval (seconds)

2.10.3.5 MultipleAxesPVTPulseOutputSet

Name

MultipleAxesPVTPulseOutputSet – Sets the configuration of pulse generation of a PVT trajectory.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the group (must not be a positioner): ERR_POSITIONER_NAME (-18)
- Check the group type (must be a MultipleAxes group): ERR_WRONG_OBJECT_TYPE (-8)
- Check the pulse generation must not be in progress: ERR_NOT_ALLOWED_ACTION (-22)
- Valid input parameter type: ERR_WRONG_TYPE_DOUBLE (-14), ERR_WRONG_TYPE_INT (-15)

Description

This function configures and activates the pulse generation of a PVT trajectory. The pulse generation is defined by a start element, an end element, and a time interval in seconds. If a pulse generation is already activated on the selected PVT trajectory then this function returns ERR_NOT_ALLOWED_ACTION error.

Please note that the pulse output settings are automatically removed when the trajectory is over. Hence, with the execution of every new trajectory, it is required to define the pulse output settings again.

This capability allows output of pulses at constant time intervals on a PVT trajectory. The pulses are generated between the first and the last trajectory element. The minimum possible time interval is 100 µs.

Example:

MultipleAxesPVTPulseOutputSet (Group1, 3, 5, 0.01)

One pulse will be generated every 10 ms between the start of the 3rd element and the end of the 5th element.

For a more thorough description of the PVT trajectory capability, please refer to the XPS Motion Tutorial sections Trajectories/PVT Trajectories and Ouput triggers.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

MultipleAxesPVTPulseOutputSet $SocketID $GroupName $StartElement $EndElement $TimeInterval

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	MultipleAxes group name (maximum size = 250)
–	StartElement	int	Start Element number
–	EndElement	int	End Element number
–	TimeInterval	double	Time interval (seconds)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **MultipleAxesPVTPulseOutputSet** (int SocketID, char *GroupName, int StartElement, int EndElement, double TimeInterval)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	MultipleAxes group name
– StartElement	int	Start Element number
– EndElement	int	End Element number
– TimeInterval	double	Time interval (seconds)

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **MultipleAxesPVTPulseOutputSet** (ByVal SocketID As Long, ByVal GroupName As String, ByVal StartElement As Long, ByVal EndElement As Long, ByVal TimeInterval As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	MultipleAxes group name
– StartElement	long	Start Element number
– EndElement	long	End Element number
– TimeInterval	double	Time interval (seconds)

Output parameters

– None

Return

– Function error code

Matlab

Prototype

[Error] **MultipleAxesPVTPulseOutputSet** (int32 SocketID, cstring GroupName, int32 StartElement, int32 EndElement, double TimeInterval)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	MultipleAxes group name
– StartElement	int32	Start Element number
– EndElement	int32	End Element number
– TimeInterval	double	Time interval (seconds)

Return

– Error	int32	Function error code

Python

Prototype

[Error] **MultipleAxesPVTPulseOutputSet** (integer SocketID, string GroupName, integer StartElement, integer EndElement, double TimeInterval)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	MultipleAxes group name
– StartElement	int	Start Element number
– EndElement	int	End Element number
– TimeInterval	double	Time interval (seconds)

Return

– Error	int	Function error code

2.10.3.6 MultipleAxesPVTResetInMemory

Name

MultipleAxesPVTResetInMemory – Deletes the current content of the PVT trajectory buffer in memory.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the group (must not be a positioner): ERR_GROUP_NAME (-19)
- Check the group (must not be a gantry): ERR_POSITIONER_NAME (-18)
- Check the group type (must be a MultipleAxes group): ERR_WRONG_OBJECT_TYPE (-8)

Description

This function deletes the PVT trajectory buffer in memory, loaded with the "MultipleAxesPVTLoadToMemory" function.

For a more thorough description of the PVT trajectory capability, please refer to the XPS Motion Tutorial section Trajectories/PVT Trajectories.

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

MultipleAxesPVTResetInMemory $SocketID $GroupName

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	MultipleAxes group name (maximum size = 250)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **MultipleAxesPVTResetInMemory** (int SocketID, char *GroupName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	MultipleAxes group name

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **MultipleAxesPVTResetInMemory** (ByVal SocketID As Long, ByVal GroupName As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	MultipleAxes group name

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **MultipleAxesPVTResetInMemory** (int32 SocketID, cstring GroupName)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	cstring	MultipleAxes group name

Return

– Function error code

Python

Prototype

integer **MultipleAxesPVTResetInMemory** (integer SocketID, string GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	string	MultipleAxes group name

Return

– Function error code

2.10.3.7 MultipleAxesPVTVerification

Name

MultipleAxesPVTVerification – Verifies a PVT trajectory data file.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the group type (must be a MultipleAxes group): ERR_WRONG_OBJECT_TYPE (-8)
- Check the base velocity value (must be null): ERR_BASE_VELOCITY (-48)
- Check the trajectory file name length: ERR_STRING_TOO_LONG (-3)
- Check the trajectory file existence and the file format: ERR_READ_FILE (-61)
- Check the trajectory (number of elements > 0): ERR_TRAJ_EMPTY (-66)
- Check trajectory element types in file: ERR_WRONG_TYPE_DOUBLE (-14)
- Check the end output velocity (must be null): ERR_TRAJ_FINAL_VELOCITY (-70)
- Check the velocity (Minimum Velocity <= Velocity <= Maximum Velocity): ERR_TRAJ_VEL_LIMIT (-68)
- Check the acceleration (Minimum acc. <= acceleration <= Maximum acc.): ERR_TRAJ_ACC_LIMIT (-69)
- Check the delta time (Delta Time > 0): ERR_TRAJ_TIME (-75)

Description

This function verifies the execution of a PVT trajectory. The results of the verification can be gathered with the "MultipleAxesPVTVerificationResultGet" function. The trajectory file must be stored in the folder "\ADMIN\Public\Trajectory" of the XPS controller. If the trajectory cannot be initialized (message queue or task error) then ERR_TRAJ_INITIALIZATION (-72) is returned.

This function can be executed at any time and is independent of the trajectory execution. This function performs the following:

- Checks the trajectory file for data coherence.
- Calculates the trajectory limits, which are: the required travel per positioner, the maximum possible trajectory velocity and the maximum possible trajectory acceleration. This function helps define the parameters for the trajectory execution.
- If all is OK, it returns "SUCCESS" (0). Otherwise, it returns a corresponding error.

NOTE

The "MultipleAxesPVTVerification" function is independent from the "MultipleAxesPVTExecution" function. It is recommended, but not necessary, to execute this function before executing a PVT trajectory.

For a more thorough description of the PVT trajectory capability, please refer to the XPS Motion Tutorial section Trajectories/PVT Trajectories.

Error codes

- ERR_BASE_VELOCITY (-48)
- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_READ_FILE (-61)
- ERR_STRING_TOO_LONG (-3)
- ERR_TRAJ_EMPTY (-66)
- ERR_TRAJ_ACC_LIMIT (-69)
- ERR_TRAJ_FINAL_VELOCITY (-70)
- ERR_TRAJ_INITIALIZATION (-72)
- ERR_TRAJ_TIME (-75)
- ERR_TRAJ_VEL_LIMIT (-68)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

MultipleAxesPVTVerification $SocketID $GroupName $FileName

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	MultipleAxes group name (maximum size = 250)
– FileName	string	Trajectory file name (maximum size = 250)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **MultipleAxesPVTVerification** (int SocketID, char *GroupName, char *FileName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	MultipleAxes group name
– FileName	char *	Trajectory file name (maximum size = 250)

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **MultipleAxesPVTVerification** (ByVal SocketID As Long, ByVal GroupName As String, ByVal FileName As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	MultipleAxes group name
– FileName	string	Trajectory file name (maximum size = 250)

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **MultipleAxesPVTVerification** (int32 SocketID, cstring GroupName, cstring FileName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	MultipleAxes group name
– FileName	cstring	Trajectory file name (maximum size = 250)

Return

– Function error code

Python

Prototype

integer **MultipleAxesPVTVerification** (integer SocketID, string GroupName, string FileName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	MultipleAxes group name
– FileName	string	Trajectory file name (maximum size = 250)

Return

– Function error code

2.10.3.8 MultipleAxesPVTVerificationResultGet

Name

MultipleAxesPVTVerificationResultGet – Returns the results of the "MultipleAxesPVTVerification" function.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the group type (must be a MultipleAxes group): ERR_WRONG_OBJECT_TYPE (-8)
- Check the positioner name length: ERR_STRING_TOO_LONG (-3)
- Check the positioner name: ERR_POSITIONER_NAME (-18)
- Check the last MultipleAxes PVT verification (must be done): ERR_NOT_ALLOWED_ACTION (-22)
- Valid output parameter type: ERR_WRONG_TYPE_CHAR (-13), ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function returns the results of the previous "MultipleAxesPVTVerification" function, positioner by positioner. The results are the travel requirements (min and max values), the possible maximum velocity and the possible maximum acceleration.

If no verification was previously done then ERR_NOT_ALLOWED_ACTION (-22) is returned.

For a more thorough description of the PVT trajectory capability, please refer to the XPS Motion Tutorial section Trajectories/PVT Trajectories.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_POSITIONER_NAME (-18)
- ERR_STRING_TOO_LONG (-3)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13),
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

MultipleAxesPVTVerificationResultGet $SocketID $PositionerName FileName MinimumPosition MaximumPosition MaximumVelocity MaximumAcceleration

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– PositionerName	string	MultipleAxes positioner name (maximum size = 250)

Output parameters

– FileName	string	Examined trajectory file name (maximum size = 250)
– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– MaximumVelocity	double	Maximum trajectory velocity (units/seconds)
– MaximumAcceleration	double	Maximum trajectory acceleration (units/seconds²)

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **MultipleAxesPVTVerificationResultGet** (int SocketID, char *PositionerName, char * FileName, double * MinimumPosition, double * MaximumPosition, double * MaximumVelocity, double * MaximumAcceleration)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– PositionerName	char *	MultipleAxes positioner name

Output parameters

– FileName	char *	Examined trajectory file name (maximum size = 250)
– MinimumPosition	double *	Minimum position (units)
– MaximumPosition	double *	Maximum position (units)
– MaximumVelocity	double *	Maximum trajectory velocity (units/seconds)
– MaximumAcceleration	double *	Maximum trajectory acceleration (units/seconds²)

Return

– Function error code

Visual Basic

Prototype

Long **MultipleAxesPVTVerificationResultGet** (ByVal SocketID As Long, ByVal PositionerName As String, ByVal FileName As String, MinimumPosition As Double, MaximumPosition As Double, MaximumVelocity As Double, MaximumAcceleration As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– PositionerName	string	MultipleAxes positioner name

Output parameters

– FileName	string	Examined trajectory file name (maximum size = 250)
– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– MaximumVelocity	double	Maximum trajectory velocity (units/seconds)
– MaximumAcceleration	double	Maximum trajectory acceleration (units/seconds²)

Return

– Function error code

Matlab

Prototype

[Error, FileName, MinimumPosition, MaximumPosition, MaximumVelocity, MaximumAcceleration] **MultipleAxesPVTVerificationResultGet** (int32 SocketID, cstring PositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– PositionerName	cstring	MultipleAxes positioner name

Return

– Error	int32	Function error code
– FileName	cstring	Examined trajectory file name (maximum size = 250)
– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– MaximumVelocity	double	Trajectory trajectory velocity (units/seconds)
– MaximumAcceleration	double	Trajectory trajectory acceleration (units/seconds²)

Python

Prototype

[Error, FileName, MinimumPosition, MaximumPosition, MaximumVelocity, MaximumAcceleration] **MultipleAxesPVTVerificationResultGet** (integer SocketID, string PositionerName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– PositionerName	string	MultipleAxes positioner name

Return

– Error	int	Function error code
– FileName	string	Examined trajectory file name (maximum size = 250)
– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– MaximumVelocity	double	Trajectory velocity (units/seconds)
– MaximumAcceleration	double	Trajectory acceleration (units/seconds²)

2.10.4 Configuration Files

Below is an example of a MultipleAxes group (named "MULTI") in the system.ini file. The group MULTI is built with two positioners named "M1" and "M2". The positioner "M1" uses the parameters of "MYSTAGE1" from the stages.ini file and is connected to the plug 1 of the XPS controller. The secondary positioner of "M1" uses the parameters of "MYSTAGE1" from the stages.ini file and is connected to plug 2 of the XPS controller. The positioner "M2" uses the parameters of "MYSTAGE2" from the stages.ini file and is connected to the plug 3 of the XPS controller. The HomeSearchSequence is "*OneAfterAnother*".

NOTE

PIDBaseFilter parameter in system.ini file is to be configured only when using the XPS-Qn Precision Platform controller.

System.ini file:

```
[GROUPS]
MultipleAxesInUse = MULTI


[MULTI]          ; AXIS MultipleAxes group configuration
PositionerInUse = M1, M2
PositionerNumber = 2
InitializationAndHomeSearchSequence = OneAfterAnother     ;Together,
                                                          OneAfterAnother or
                                    ; OneAfterAnotherInReverseOrder

[MULTI.M1]
PLugNumber = 1
StageName = MYSTAGE1
; --- Time flasher
TimeFlasherBaseFrequency =        ; default value 40e6, must be between 39.5e6 and
                                  40.5e6 Hz
;--- PIDBase filter (take into account the base movements, effective only with
PIDFFAcceleration corrector) To be added only when using XPS-Q8 Precision
Platform controller
PIDBaseFilter =                   ; Enabled or Disabled
MovingMass =                      ; If PIDBaseFilter = Enabled
StaticMass =                      ; If PIDBaseFilter = Enabled
Viscosity =                       ; If PIDBaseFilter = Enabled
Stiffness =                       ; If PIDBaseFilter = Enabled
;--- If Gantry (secondary positioner)
SecondaryPositionerGantry =               ; Enabled or Disabled
SecondaryPlugNumber = 2                   ; If SecondaryPositionerGantry = Enabled
SecondaryStageName = MYSTAGE1             ;If SecondaryPositionerGantry = Enabled
SecondaryPositionerGantryEndReferencingPosition = 0       ;If
                                                          SecondaryPositionerGantry = Enabled
SecondaryPositionerGantryEndReferencingTolerance = 1      ;If
                                                          SecondaryPositionerGantry = Enabled
```

```
SecondaryPositionerGantryOffsetAfterInitialization = 0          ;If
                                                    SecondaryPositionerGantry = Enabled
SecondaryPositionerGantryMotorEnableDelay =          ;seconds,if
                                                    SecondaryPositionerGantry = Enabled
[MULTI.M2]
PLugNumber = 3
StageName = MYSTAGE2
; --- Time flasher
TimeFlasherBaseFrequency =      ; default value 40e6, must be between 39.5e6 and
                                40.5e6 Hz
;--- PIDBase filter (take into account the base movements, effective only with
PIDFFAcceleration corrector) To be added only when using XPS-Q8 Precision
Platform controller
PIDBaseFilter =             ; Enabled or Disabled
MovingMass =                ; If PIDBaseFilter = Enabled
StaticMass =                ; If PIDBaseFilter = Enabled
Viscosity =                 ; If PIDBaseFilter = Enabled
Stiffness =                 ; If PIDBaseFilter = Enabled
;--- If Gantry (secondary positioner)
SecondaryPositionerGantry =             ; Enabled or Disabled
SecondaryPlugNumber = 4                 ; If SecondaryPositionerGantry = Enabled
SecondaryStageName = MYSTAGE2           ;If SecondaryPositionerGantry = Enabled
SecondaryPositionerGantryEndReferencingPosition = 0          ;If
                                                    SecondaryPositionerGantry = Enabled
SecondaryPositionerGantryEndReferencingTolerance = 1                ;If
                                                    SecondaryPositionerGantry = Enabled
SecondaryPositionerGantryOffsetAfterInitialization = 0          ;If
                                                    SecondaryPositionerGantry = Enabled
SecondaryPositionerGantryMotorEnableDelay =          ;seconds,if
                                                    SecondaryPositionerGantry = Enabled
```

Stages.ini file:

```
[MYSTAGE1]
MYSTAGE1 configuration => See § "Positioner: Configuration files"

[MYSTAGE2]
MYSTAGE2 configuration => See § "Positioner: Configuration files"
```

2.11 TZ Group

2.11.1 Description

A TZ group is a 3-positioner object, like the XYZ group, but with the following differences:

- It supports the **PVT** trajectories.

- It does not support 3D **spline** trajectories.

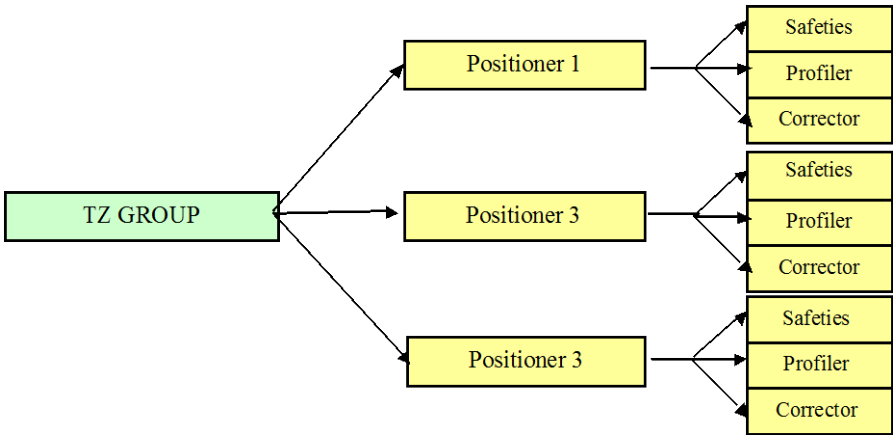
- It supports **TZDecoupling**, **XYtoZZZAccelerationFeedforward** and **TZTracking** functionalities.

- It supports **Focus process** via the XPS focus interface board and focus process module (focus.out).

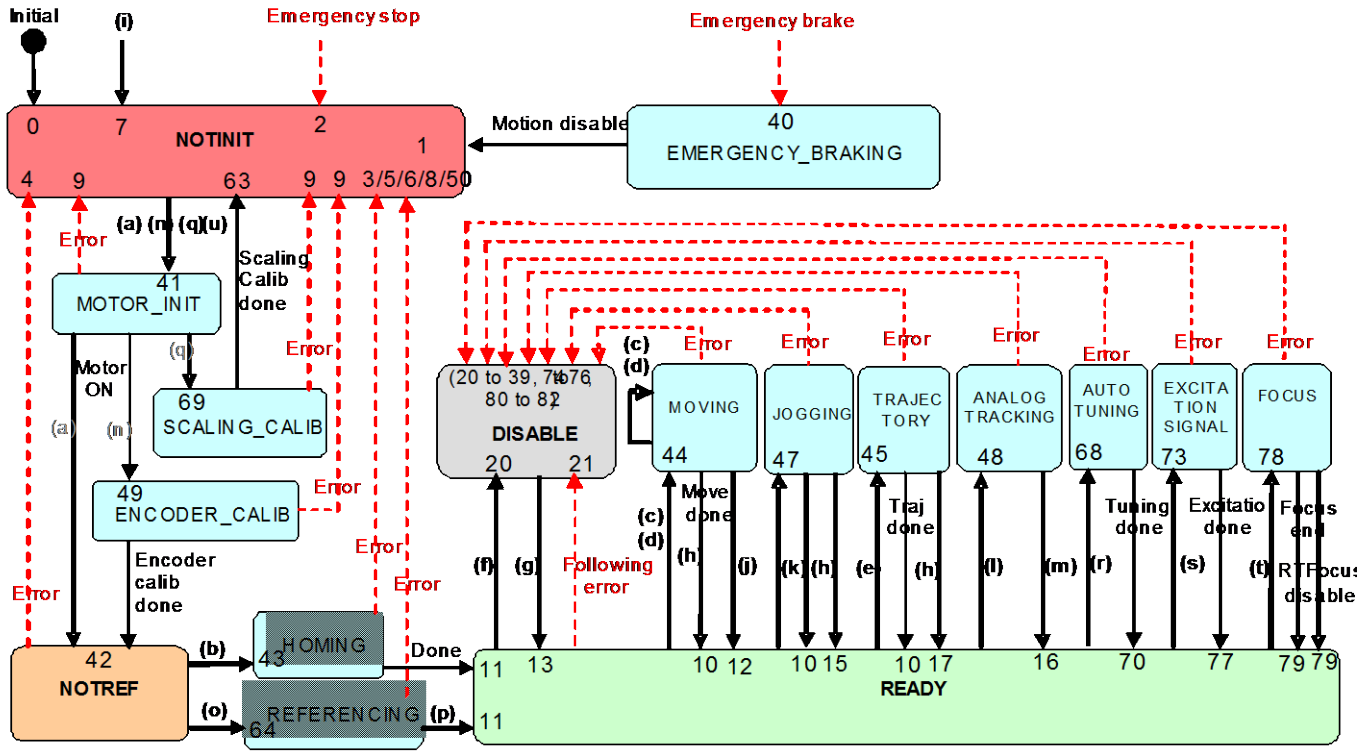
- It has a specific way of initialization (*MotorDriverInterface* = *AnalogAccelerationTZ*).

NOTE

This group can be used only with the XPS-Qn Precision Platform controller.



2.11.2 State Diagram



Called functions:

(a) GroupInitialize
(b) GroupHomeSearch
(c) GroupMoveAbsolute
(d) GroupMoveRelative
(e) MultipleAxesPVTExecution
(f) GroupMotionDisable
(g) GroupMotionEnable
(h) GroupMoveAbort
(i) GroupKill or KillAll
(j) GroupJogModeEnable
(k) GroupJogModeDisable
(l) GroupAnalogTrackingModeEnable
(m) GroupAnalogTrackingModeDisable
(n) GroupInitializeWithEncoderCalibration
(o) GroupReferencingStart
(p) GroupReferencingStop
(q) PositionerAccelerationAutoScaling
(r) PositionerCorrectorAutoTuning
(s) PositionerExcitationSignalSet
(t) TZFocusModeEnable
(u) GroupInitializeNoEncoderReset

2.11.3 Specific Function Description

2.11.3.1 TZPVTExecution

Name

TZPVTExecution – Executes a PVT trajectory.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Group state must be "READY": ERR_NOT_ALLOWED_ACTION (-22)
- Check backlash (must not be enabled): ERR_NOT_ALLOWED_BACKLASH (-46)
- Check the base velocity value (must be null): ERR_BASE_VELOCITY (-48)
- Check the group: ERR_POSITIONER_NAME (-18)
- Check the group type (must be an XYZ group): ERR_WRONG_OBJECT_TYPE (-8)
- Check the input parameter type: ERR_WRONG_TYPE_INT (-15)
- Check the input value (Number of executions > 0): ERR_PARAMETER_OUT_OF_RANGE (-17)
- Check the trajectory file name length: ERR_STRING_TOO_LONG (-3)
- Check the trajectory file existence or file reading: ERR_READ_FILE (-61)
- Check the message queue filling: ERR_MSG_QUEUE (-71)

Description

This function executes a PVT (Position Velocity Time) trajectory. The trajectory file must be stored in the folder "\ADMIN\Public\Trajectory" of the XPS controller. If the trajectory cannot be initialized (message queue or task error) then ERR_TRAJ_INITIALIZATION (-72) is returned.

Before a trajectory execution, it is recommended to check whether the trajectory is within group parameter postioner capability and can execute by using the "TZPVTVerification" and "TZPVTVerificationResultGet" functions.

For a more thorough description of the PVT trajectory capability, please refer to the XPS Motion Tutorial section Trajectories/PVT Trajectories.

NOTES

This function can be used only with the XPS-Qn Precision Platform controller.

In case of an ERR_GROUP_MOTION_DONE_TIMEOUT (-33) error, an ERR_FOLLOWING_ERROR (-25) error or ERR_SLAVE (-44) error, the group state becomes DISABLE. To help determine the error source, check the positioner errors, the hardware status and the driver status.

Error codes

- ERR_BASE_VELOCITY (-48)
- ERR_FATAL_INIT (-20)
- ERR_FOLLOWING_ERROR (-25)
- ERR_GROUP_MOTION_DONE_TIMEOUT (-33)
- ERR_IN_INITIALIZATION (-21)
- ERR_MSG_QUEUE (-71)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_NOT_ALLOWED_BACKLASH (-46)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_POSITIONER_NAME (-18)
- ERR_READ_FILE (-61)
- ERR_SLAVE (-44)
- ERR_STRING_TOO_LONG (-3)
- ERR_TRAJ_INITIALIZATION (-72)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

TZPVTExecution $SocketID $GroupName $FileName $Velocity $Acceleration

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	TZ group name (maximum size = 250)
– FileName	string	Trajectory file name (maximum size = 250)
– ExecutionNumber	int	Number of trajectory executions

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **TZPVTExecution** (int SocketID, char *GroupName, char *FileName , double Velocity, double Acceleration)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	TZ group name
– FileName	char *	Trajectory file name (maximum size = 250)
– ExecutionNumber	int	Number of trajectory executions

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **TZPVTExecution** (ByVal SocketID As Long, ByVal GroupName As String, ByVal FileName As String, ByVal Velocity As Double, ByVal Acceleration As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	TZ group name
– FileName	string	Trajectory file name (maximum size = 250)
– ExecutionNumber	long	Number of trajectory executions

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **TZPVTExecution** (int32 SocketID, cstring GroupName, cstring FileName, double Velocity, double Acceleration)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	TZ group name
– FileName	cstring	Trajectory file name (maximum size = 250)
– ExecutionNumber	int32	Number of trajectory executions

Return

– Function error code

Python

Prototype

integer **TZPVTExecution** (integer SocketID, string GroupName, string FileName, double Velocity, double Acceleration)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	TZ group name
– FileName	string	Trajectory file name (maximum size = 250)
– ExecutionNumber	int	Number of trajectory executions

Return

– Function error code

2.11.3.2 TZPVTLoadToMemory

Name

TZPVTLoadToMemory – Loads a Multiple Axes PVT trajectory's line.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the group (must not be a positioner): ERR_GROUP_NAME (-19)
- Check the gantry mode (must not be a gantry): ERR_POSITIONER_NAME (-18)
- Check the group type (must be an TZ group): ERR_WRONG_OBJECT_TYPE (-8)
- Check the trajectory data length: ERR_STRING_TOO_LONG (-3)
- Check the input value (> 0): ERR_PARAMETER_OUT_OF_RANGE (-17)

Description

This function loads a line of a PVT trajectory into the XPS memory. Each trajectory element must be separated by a comma. To verify or to execute the PVT trajectory loaded in memory, use the string "**FromMemory**" instead of a FileName.

For a more thorough description of the PVT trajectory capability, please refer to the XPS Motion Tutorial section Trajectories/PVT Trajectories.

NOTES

This function can be used only with the XPS-Qn Precision Platform controller.

All previous PVT functions, when called with the string "FromMemory" instead of a FileName, will perform the same operation on the PVT trajectory in RAM as it does in a disk file.

Example:

TZPVTLoadToMemory(myTZ, 0.04,0,0,3.2,0)

TZPVTVerification (myTZ, FromMemory)

TZPVTExecution(myTZ, FromMemory, 1)

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_STRING_TOO_LONG (-3)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

TZPVTLoadToMemory $SocketID $GroupName $TrajectoryLine

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	TZ group name (maximum size = 250)
– TrajectoryLine	string	Trajectory line (maximum size = 400)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **TZPVTLoadToMemory** (int SocketID, char *GroupName, char *TrajectoryLine)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	TZ group name
– TrajectoryLine	char *	Trajectory line (maximum size = 400)

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **TZPVTLoadToMemory** (ByVal SocketID As Long, ByVal GroupName As String, ByVal TrajectoryLine As String, ByVal)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	TZ group name
– TrajectoryLine	string	Trajectory line (maximum size = 400)

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **TZPVTLoadToMemory** (int32 SocketID, cstring GroupName, cstring TrajectoryLine)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	TZ group name
– TrajectoryLine	cstring	Trajectory line (maximum size = 400)

Return

– Function error code

Python

Prototype

integer **TZPVTLoadToMemory** (integer SocketID, string GroupName, string TrajectoryLine)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	TZ group name
– TrajectoryLine	string	Trajectory file name (maximum size = 400)

Return

– Function error code

2.11.3.3 TZPVTParametersGet

Name

TZPVTParametersGet – Returns the PVT trajectory parameters.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the group type (must be a TZ group): ERR_WRONG_OBJECT_TYPE (-8)
- Check the trajectory type (PVT): ERR_NOT_ALLOWED_ACTION (-22)
- Valid output parameter type: ERR_WRONG_TYPE_CHAR (-13), ERR_WRONG_TYPE_INT (-15)

Description

This function returns the PVT trajectory parameters (trajectory name and current executing element number) of the current executed PVT trajectory.

For a more thorough description of the PVT trajectory capability, please refer to the XPS Motion Tutorial section Trajectories/PVT Trajectories.

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

TZPVTParametersGet $SocketID $GroupName FileName ElementNumber

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	TZ group name (maximum size = 250)

Output parameters

– FileName	string	Executing trajectory file name (maximum size = 250)
– ElementNumber	int	Current executing element number

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **TZPVTParametersGet** (int SocketID, char *GroupName, char * FileName, int * ElementNumber)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	TZ group name

Output parameters

– FileName	char *	Executing trajectory file name (maximum size = 250)
– ElementNumber	int *	Current executing element number

Return

– Function error code

Visual Basic

Prototype

Long **TZPVTParametersGet** (ByVal SocketID As Long, ByVal GroupName As String, ByVal FileName As String, ElementNumber As Integer)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	TZ group name

Output parameters

– FileName	string	Executing trajectory file name (maximum size = 250)
– ElementNumber	int	Current executing element number

Return

– Function error code

Matlab

Prototype

[Error, FileName, ElementNumber] **TZPVTParametersGet** (int32 SocketID, cstring GroupName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	TZ group name

Return

– Error	int32	Function error code
– FileName	cstring	Executing trajectory file name (maximum size = 250)
– ElementNumber	int32	Current executing element number

Python

Prototype

[Error, FileName, ElementNumber] **TZPVTParametersGet** (integer SocketID, string GroupName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	TZ group name

Return

– Error	int	Function error code
– FileName	string	Executing trajectory file name (maximum size = 250)
– ElementNumber	int	Current executing element number

2.11.3.4 TZPVTPulseOutputGet

Name

TZPVTPulseOutputGet – Returns the configuration of pulse generation of a PVT trajectory.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the group (must not be a positioner): ERR_POSITIONER_NAME (-18)
- Check the group type (must be a TZ group): ERR_WRONG_OBJECT_TYPE (-8)
- Valid output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function returns the last configuration of pulse generation of a PVT trajectory.

The pulse output configuration is defined by a start element, an end element, and a time interval in seconds.

Example:

Add: TZPVTPulseOutputGet (Group1, 3, 5, 0.01)

One pulse will be generated every 10 ms between the start of the 3rd element and the end of the 5th element.

Start element= 3

End element = 5

Time interval = 0.01 seconds

For a more thorough description of the PVT trajectory capability, please refer to the XPS Motion Tutorial, sections named Trajectories/PVT Trajectories and Ouput triggers.

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

TZPVTPulseOutputGet $SocketID $GroupName StartElement EndElement TimeInterval

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	TZ group name (maximum size = 250)

Output parameters

– StartElement	int	Start Element number
– EndElement	int	End Element number
– TimeInterval	double	Time interval (seconds)

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **TZPVTPulseOutputGet** (int SocketID, char *GroupName, int * StartElement, int * EndElement, double * TimeInterval)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	TZ group name

Output parameters

– StartElement	int *	Start Element number
– EndElement	int *	End Element number
– TimeInterval	double *	Time interval (seconds)

Return

– Function error code

Visual Basic

Prototype

Long **TZPVTPulseOutputGet** (ByVal SocketID As Long, ByVal GroupName As String, StartElement As Long, EndElement As Long, TimeInterval As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	TZ group name

Output parameters

– StartElement	long	Start Element number
– EndElement	long	End Element number
– TimeInterval	double	Time interval (seconds)

Return

– Function error code

Matlab

Prototype

[Error, StartElement, EndElement, TimeInterval] **TZPVTPulseOutputGet** (int32 SocketID, cstring GroupName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	TZ group name

Return

– Error	int32	Function error code
– StartElement	int32	Start Element number
– EndElement	int32	End Element number
– TimeInterval	double	Time interval (seconds)

Python

Prototype

[Error, StartElement, EndElement, TimeInterval] **TZPVTPulseOutputGet** (integer SocketID, string GroupName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	TZ group name

Return

– Error	int	Function error code
– StartElement	int	Start Element number
– EndElement	int	End Element number
– TimeInterval	double	Time interval (seconds)

2.11.3.5 TZPVTPulseOutputSet

Name

TZPVTPulseOutputSet – Sets the configuration of pulse generation of a PVT trajectory.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the group (must not be a positioner): ERR_POSITIONER_NAME (-18)
- Check the group type (must be a TZ group): ERR_WRONG_OBJECT_TYPE (-8)
- Check the pulse generation must not be in progress: ERR_NOT_ALLOWED_ACTION (-22)
- Valid input parameter type: ERR_WRONG_TYPE_DOUBLE (-14), ERR_WRONG_TYPE_INT (-15)

Description

This function configures and activates the pulse generation of a PVT trajectory. The pulse generation is defined by a start element, an end element, and a time interval in seconds. If a pulse generation is already activated on the selected PVT trajectory then this function returns ERR_NOT_ALLOWED_ACTION.

Please note, that the pulse output settings are automatically removed when the trajectory is over. Hence, with the execution of every new trajectory, it is required to define the pulse output settings again.

This capability allows output of pulses at constant time intervals on a PVT trajectory. The pulses are generated between the first and the last trajectory element. The minimum possible time interval is 100 µs.

Example:

TZPVTPulseOutputSet (Group1, 3, 5, 0.01)

One pulse will be generated every 10 ms between the start of the 3rd element and the end of the 5th element.

For a more thorough description of the PVT trajectory capability, please refer to the XPS Motion Tutorial, sections named Trajectories/PVT Trajectories and Ouput triggers.

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

TZPVTPulseOutputSet $SocketID $GroupName $StartElement $EndElement $TimeInterval

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	TZ group name (maximum size = 250)
– StartElement	int	Start Element number
– EndElement	int	End Element number
– TimeInterval	double	Time interval (seconds)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **TZPVTPulseOutputSet** (int SocketID, char *GroupName, int StartElement, int EndElement, double TimeInterval)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	char *	TZ group name
–	StartElement	int	Start Element number
–	EndElement	int	End Element number
–	TimeInterval	double	Time interval (seconds)

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **TZPVTPulseOutputSet** (ByVal SocketID As Long, ByVal GroupName As String, ByVal StartElement As Long, ByVal EndElement As Long, ByVal TimeInterval As Double)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	TZ group name
–	StartElement	long	Start Element number
–	EndElement	long	End Element number
–	TimeInterval	double	Time interval (seconds)

Output parameters

– None

Return

– Function error code

Matlab

Prototype

[Error] **TZPVTPulseOutputSet** (int32 SocketID, cstring GroupName, int32 StartElement, int32 EndElement, double TimeInterval)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	TZ group name
– StartElement	int32	Start Element number
– EndElement	int32	End Element number
– TimeInterval	double	Time interval (seconds)

Return

– Error	int32	Function error code

Python

Prototype

[Error] **TZPVTPulseOutputSet** (integer SocketID, string GroupName, integer StartElement, integer EndElement, double TimeInterval)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	TZ group name
– StartElement	int	Start Element number
– EndElement	int	End Element number
– TimeInterval	double	Time interval (seconds)

Return

– Error	int	Function error code

2.11.3.6 TZPVTResetInMemory

Name

TZPVTResetInMemory – Deletes the current content of the PVT trajectory buffer in memory.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the group (must not be a positioner): ERR_GROUP_NAME (-19)
- Check the group (must not be a gantry): ERR_POSITIONER_NAME (-18)
- Check the group type (must be a TZ group): ERR_WRONG_OBJECT_TYPE (-8)

Description

This function deletes the PVT trajectory buffer in the memory, loaded with the "TZPVTLoadToMemory" function.

For a more thorough description of the PVT trajectory capability, please refer to the XPS Motion Tutorial section Trajectories/PVT Trajectories.

NOTE

This function can be used only with the XPS-Qn Precision Platform controller

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

TZPVTResetInMemory $SocketID $GroupName

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	TZ group name (maximum size = 250)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **TZPVTResetInMemory** (int SocketID, char *GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	char *	TZ group name

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **TZPVTResetInMemory** (ByVal SocketID As Long, ByVal GroupName As String)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	TZ group name

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **TZPVTResetInMemory** (int32 SocketID, cstring GroupName)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	cstring	TZ group name

Return

– Function error code

Python

Prototype

integer **TZPVTResetInMemory** (integer SocketID, string GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	string	TZ group name

Return

– Function error code

2.11.3.7 TZPVTVerification

Name

TZPVTVerification – Verifies a PVT trajectory data file.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the group type (must be a TZ group): ERR_WRONG_OBJECT_TYPE (-8)
- Check the base velocity value (must be null): ERR_BASE_VELOCITY (-48)
- Check the trajectory file name length: ERR_STRING_TOO_LONG (-3)
- Check the trajectory file existence and the file format: ERR_READ_FILE (-61)
- Check the trajectory (number of elements > 0): ERR_TRAJ_EMPTY (-66)
- Check trajectory element types in file: ERR_WRONG_TYPE_DOUBLE (-14)
- Check the end output velocity (must be null): ERR_TRAJ_FINAL_VELOCITY (-70)
- Check the velocity (Minimum Velocity <= Velocity <= Maximum Velocity): ERR_TRAJ_VEL_LIMIT (-68)
- Check the acceleration (Minimum acc. <= acceleration <= Maximum acc.): ERR_TRAJ_ACC_LIMIT (-69)
- Check the delta time (Delta Time > 0): ERR_TRAJ_TIME (-75)

Description

This function verifies the possible execution of a PVT trajectory. The results of the verification can be gathered with the "TZPVTVerificationResultGet" function. The trajectory file must be stored in the folder "\ADMIN\Public\Trajectory" of the XPS controller. If the trajectory cannot be initialized (message queue or task error) then ERR_TRAJ_INITIALIZATION (-72) is returned.

This function can be executed at any time and is independent from the trajectory execution. This function performs the following:

1. Checks the trajectory file for data coherence.
2. Calculates the trajectory limits, which are: the required travel per positioner, the maximum possible trajectory velocity and the maximum possible trajectory acceleration. This function helps define the parameters for the trajectory execution.
3. If all is OK, it returns "SUCCESS" (0). Otherwise, it returns a corresponding error.

NOTES

This function can be used only with the XPS-Q8 Precision Platform controller.

The "TZPVTVerification" function is independent from the "TZPVTExecution" function. It is recommended, but not necessary to execute this function before executing a PVT trajectory.

For a more thorough description of the PVT trajectory capability, please refer to the XPS Motion Tutorial section Trajectories/PVT Trajectories.

Error codes

- ERR_BASE_VELOCITY (-48)
- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_READ_FILE (-61)
- ERR_STRING_TOO_LONG (-3)
- ERR_TRAJ_EMPTY (-66)
- ERR_TRAJ_ACC_LIMIT (-69)
- ERR_TRAJ_FINAL_VELOCITY (-70)
- ERR_TRAJ_INITIALIZATION (-72)
- ERR_TRAJ_TIME (-75)
- ERR_TRAJ_VEL_LIMIT (-68)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

TZPVTVerification $SocketID $GroupName $FileName

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	TZ group name (maximum size = 250)
– FileName	string	Trajectory file name (maximum size = 250)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **TZPVTVerification** (int SocketID, char *GroupName, char *FileName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	TZ group name
– FileName	char *	Trajectory file name (maximum size = 250)

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **TZPVTVerification** (ByVal SocketID As Long, ByVal GroupName As String, ByVal FileName As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	TZ group name
– FileName	string	Trajectory file name (maximum size = 250)

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **TZPVTVerification** (int32 SocketID, cstring GroupName, cstring FileName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	TZ group name
– FileName	cstring	Trajectory file name (maximum size = 250)

Return

– Function error code

Python

Prototype

integer **TZPVTVerification** (integer SocketID, string GroupName, string FileName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	TZ group name
– FileName	string	Trajectory file name (maximum size = 250)

Return

– Function error code

2.11.3.8 TZPVTVerificationResultGet

Name

TZPVTVerificationResultGet – Returns the results of the "**TZPVTVerification**" function.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the group type (must be a TZ group): ERR_WRONG_OBJECT_TYPE (-8)
- Check the positioner name length: ERR_STRING_TOO_LONG (-3)
- Check the positioner name: ERR_POSITIONER_NAME (-18)
- Check the last TZ PVT verification (must be done): ERR_NOT_ALLOWED_ACTION (-22)
- Valid output parameter type: ERR_WRONG_TYPE_CHAR (-13), ERR_WRONG_TYPE_DOUBLE (-14)

Description

This function returns the results of the previous "TZPVTVerification" function, positioner by positioner. The results are travel requirements (min and max values), the possible maximum velocity and the possible maximum acceleration.

If no verification was previously done then ERR_NOT_ALLOWED_ACTION (-22) is returned.

For a more thorough description of the PVT trajectory capability, please refer to the XPS Motion Tutorial section Trajectories/PVT Trajectories.

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_POSITIONER_NAME (-18)
- ERR_STRING_TOO_LONG (-3)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13),
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

TZPVTVerificationResultGet $SocketID $PositionerName FileName MinimumPosition MaximumPosition MaximumVelocity MaximumAcceleration

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– PositionerName	string	TZ positioner name (maximum size = 250)

Output parameters

– FileName	string	Examined trajectory file name (maximum size = 250)
– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– MaximumVelocity	double	Maximum trajectory velocity (units/seconds)
– MaximumAcceleration	double	Maximum trajectory acceleration (units/seconds²)

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **TZPVTVerificationResultGet** (int SocketID, char *PositionerName, char * FileName, double * MinimumPosition, double * MaximumPosition, double * MaximumVelocity, double * MaximumAcceleration)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– PositionerName	char *	TZ positioner name

Output parameters

– FileName	char *	Examined trajectory file name (maximum size = 250)
– MinimumPosition	double *	Minimum position (units)
– MaximumPosition	double *	Maximum position (units)
– MaximumVelocity	double *	Maximum trajectory velocity (units/seconds)
– MaximumAcceleration	double *	Maximum trajectory acceleration (units/seconds²)

Return

– Function error code

Visual Basic

Prototype

Long **TZPVTVerificationResultGet** (ByVal SocketID As Long, ByVal PositionerName As String, ByVal FileName As String, MinimumPosition As Double, MaximumPosition As Double, MaximumVelocity As Double, MaximumAcceleration As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– PositionerName	string	TZ positioner name

Output parameters

– FileName	string	Examined trajectory file name (maximum size = 250)
– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– MaximumVelocity	double	Maximum trajectory velocity (units/seconds)
– MaximumAcceleration	double	Maximum trajectory acceleration (units/seconds²)

Return

– Function error code

Matlab

Prototype

[Error, FileName, MinimumPosition, MaximumPosition, MaximumVelocity, MaximumAcceleration] **TZPVTVerificationResultGet** (int32 SocketID, cstring PositionerName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– PositionerName	cstring	TZ positioner name

Return

– Error	int32	Function error code
– FileName	cstring	Examined trajectory file name (maximum size = 250)
– MinimumPosition	double	Minimum position (units)
– MaximumPosition	double	Maximum position (units)
– MaximumVelocity	double	Trajectory trajectory velocity (units/seconds)
– MaximumAcceleration	double	Trajectory trajectory acceleration (units/seconds²)

Python

Prototype

[Error, FileName, MinimumPosition, MaximumPosition, MaximumVelocity, MaximumAcceleration] **TZPVTVerificationResultGet** (integer SocketID, string PositionerName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	PositionerName	string	TZ positioner name

Return

–	Error	int	Function error code
–	FileName	string	Examined trajectory file name (maximum size = 250)
–	MinimumPosition	double	Minimum position (units)
–	MaximumPosition	double	Maximum position (units)
–	MaximumVelocity	double	Trajectory velocity (units/seconds)
–	MaximumAcceleration	double	Trajectory acceleration (units/seconds²)

2.11.3.9 TZFocusModeEnable

Name

TZFocusModeEnable – Enable the TZ group to go in FOCUS state.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Group state must be "READY": ERR_NOT_ALLOWED_ACTION (-22)
- Check the group name: ERR_GROUP_NAME (-19)
- Check the positioner name: ERR_POSITIONER_NAME (-18)
- Check the group type (must be a TZ group): ERR_WRONG_OBJECT_TYPE (-8)

Description

This function enables the focus mode for the TZ group.

To use this function, The TZ group must be in READY state. If it's not then ERR_NOT_ALLOWED_ACTION (-22) is returned.

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

TZFocusModeEnable $SocketID $GroupName

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	TZ group name (maximum size = 250)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **TZFocusModeEnable** (int SocketID, char *GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	char *	TZ group name

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **TZFocusModeEnable** (ByVal SocketID As Long, ByVal GroupName As String)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	TZ group name

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **TZFocusModeEnable** (int32 SocketID, cstring GroupName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	TZ group name

Return

– Function error code

Python

Prototype

integer **TZFocusModeEnable** (integer SocketID, string GroupName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	TZ group name

Return

– Function error code

2.11.3.10 TZFocusModeDisable

Name

TZFocusModeDisable – Disables the TZ group from out of the FOCUS state.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Group state must be "READY": ERR_NOT_ALLOWED_ACTION (-22)
- Check the group name: ERR_GROUP_NAME (-19)
- Check the positioner name: ERR_POSITIONER_NAME (-18)
- Check the group type (must be a TZ group): ERR_WRONG_OBJECT_TYPE (-8)

Description

This function disables the FOCUS mode of a TZ group. If it executes the group quits FOCUS state and goes into READY state.

To use this function, The group must be in a FOCUS state. If not then ERR_NOT_ALLOWED_ACTION (-22) is returned.

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GROUP_NAME (-19)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

TZFocusModeDisable $SocketID $GroupName

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	TZ group name (maximum size = 250)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **TZFocusModeDisable** (int SocketID, char *GroupName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	TZ group name

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **TZFocusModeDisable** (ByVal SocketID As Long, ByVal GroupName As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	TZ group name

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **TZFocusModeDisable** (int32 SocketID, cstring GroupName)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	cstring	TZ group name

Return

– Function error code

Python

Prototype

integer **TZFocusModeDisable** (integer SocketID, string GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	string	TZ group name

Return

– Function error code

2.11.3.11 TZTrackingUserMaximumZZZTargetDifferenceGet

Name

TZTrackingUserMaximumZZZTargetDifferenceGet – Get tracking maximum ZZZ target difference of TZ group.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the group (must not be a positioner): ERR_POSITIONER_NAME (-18)
- Check the group type (must be a TZ group): ERR_WRONG_OBJECT_TYPE (-8)

Description

This function gets the tracking maximum ZZZ target difference of TZ group.

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

TZTrackingUserMaximumZZZTargetDifferenceGet $SocketID $GroupName UserMaximumZZZTargetDifference

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	TZ group name (maximum size = 250)

Output parameters

– UserMaximumZZZTargetDifference	double	User maximum ZZZ target difference (units)

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **TZTrackingUserMaximumZZZTargetDifferenceGet** (int SocketID, char *GroupName, double * UserMaximumZZZTargetDifference)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	char *	TZ group name

Output parameters

– UserMaximumZZZTargetDifference	double *	User maximum ZZZ target difference (units)

Return

– Function error code

Visual Basic

Prototype

Long **TZTrackingUserMaximumZZZTargetDifferenceGet** (ByVal SocketID As Long, ByVal GroupName As String, UserMaximumZZZTargetDifference As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	TZ group name

Output parameters

– UserMaximumZZZTargetDifference	double	User maximum ZZZ target difference (units)

Return

– Function error code

Matlab

Prototype

[Error, UserMaximumZZZTargetDifference]
TZTrackingUserMaximumZZZTargetDifferenceGet (int32 SocketID, cstring GroupName)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	cstring	TZ group name

Return

–	Error	int32	Function error code
–	UserMaximumZZZTargetDifference	double	User maximum ZZZ target difference (units)

Python

Prototype

[Error, UserMaximumZZZTargetDifference]
TZTrackingUserMaximumZZZTargetDifferenceGet (integer SocketID, string GroupName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	string	TZ group name

Return

–	Error	int	Function error code
–	UserMaximumZZZTargetDifference	double	User maximum ZZZ target difference (units)

2.11.3.12 TZTrackingUserMaximumZZZTargetDifferenceSet

Name

TZTrackingUserMaximumZZZTargetDifferenceSet – Sets tracking maximum ZZZ target difference for a TZ group.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of command parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check the group (must not be a positioner): ERR_POSITIONER_NAME (-18)
- Check the group type (must be a TZ group): ERR_WRONG_OBJECT_TYPE (-8)
- Value is out of range: ERR_PARAMETER_OUT_OF_RANGE (-17)

Description

This function sets the tracking maximum ZZZ target difference for a TZ group.

NOTE

This function can be used only with the XPS-Qn Precision Platform controller.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_POSITIONER_NAME (-18)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

TZTrackingUserMaximumZZZTargetDifferenceSet $SocketID $GroupName $UserMaximumZZZTargetDifference

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	GroupName	string	TZ group name (maximum size = 250)
–	UserMaximumZZZTargetDifference	double	User maximum ZZZ target difference (units)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **TZTrackingUserMaximumZZZTargetDifferenceSet** (int SocketID, char *GroupName, int StartElement, int EndElement, double TimeInterval)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
–	GroupName	char *	TZ group name
–	UserMaximumZZZTargetDifference	double	User maximum ZZZ target difference (units)

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **TZTrackingUserMaximumZZZTargetDifferenceSet** (ByVal SocketID As Long, ByVal GroupName As String, ByVal StartElement As Long, ByVal EndElement As Long, ByVal TimeInterval As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– GroupName	string	TZ group name
– UserMaximumZZZTargetDifference	double	User maximum ZZZ target difference (units)

Output parameters

– None

Return

– Function error code

Matlab

Prototype

[Error] **TZTrackingUserMaximumZZZTargetDifferenceSet** (int32 SocketID, cstring GroupName, double UserMaximumZZZTargetDifference)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	cstring	TZ group name
– UserMaximumZZZTargetDifference	double	User maximum ZZZ target difference (units)

Return

– Error	int32	Function error code

Python

Prototype

[Error] **TZTrackingUserMaximumZZZTargetDifferenceSet** (integer SocketID, string GroupName, double UserMaximumZZZTargetDifference)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer"function
– GroupName	string	TZ group name
– UserMaximumZZZTargetDifference	double	User maximum ZZZ target difference (units)

Return

– Error	int	Function error code

2.11.4 Configuration Files

Below is an example of a TZ group (named "MYTZ") in the system.ini file. The group MYTZ is built with three positioners named "Z1", "Z2" and "Z3". The positioner "Z1" uses the parameters of "MYSTAGE1" from the stages.ini file and is connected to the plug 1 of the XPS controller. The HomeSearchSequence is "Together" or "OneAfterAnother", but "Together" only if the AnalogAccelerationTZ MotorDriverInterface is used (stages.ini - see § "Positioner: Configurationfiles" for details).

System.ini file:

[GROUPS]

TZInUse = MYTZ

[MYTZ] *; AXIS TZ group configuration*

PositionerInUse = Z1, Z2, Z3

InitializationAndHomeSearchSequence = Together *; Together or OneAfterAnother*

TZDecouplingGainMatrixFileName = *;TZ_decoupling_matrix_filename.txt*

MaximumZZZTargetDifference = *;Maximum difference between Z target positions (units)*

[MYTZ.Z1]

PLugNumber = 1

StageName = MYSTAGE1

STAGE configuration => See § "Positioner: Configurationfiles"

[MYTZ.Z2]

PLugNumber = 2

StageName = MYSTAGE2

STAGE configuration => See § "Positioner: Configuration files"

[MYTZ.Z3]

PLugNumber = 3

StageName = MYSTAGE3

STAGE configuration => See § "Positioner: Configuration files"

Stages.ini file:

[MYSTAGE1]

MYSTAGE1 configuration => See § "Positioner: Configuration files"

[MYSTAGE2]

MYSTAGE2 configuration => See § "Positioner: Configuration files"

[MYSTAGE3]

MYSTAGE3 configuration => See § "Positioner: Configuration files"

2.12 Analog and Nigital I/O

2.12.1 GPIO Name List

2.12.1.1 Digital inputs

GPIO1.DI	Digital Input of the I/O board connector # 1 (8 bits)
GPIO2.DI	Digital Input of the I/O board connector # 2 (6 bits)
GPIO3.DI	Digital Input of the I/O board connector # 3 (6 bits)
GPIO4.DI	Digital Input of the I/O board connector # 4 (16 bits)

2.12.1.2 Digital outputs

GPIO1.DO	Digital Output of the I/O board connector # 1 (8 bits)
GPIO3.DO	Digital Output of the I/O board connector # 3 (6 bits)
GPIO4.DO	Digital Output of the I/O board connector # 4 (16 bits)

2.12.1.3 Analog inputs

GPIO2.ADC1	Analog Input # 1 of the I/O board connector # 2
GPIO2.ADC2	Analog Input # 2 of the I/O board connector # 2
GPIO2.ADC3	Analog Input # 3 of the I/O board connector # 2
GPIO2.ADC4	Analog Input # 4 of the I/O board connector # 2

2.12.1.4 Analog outputs

GPIO2.DAC1	Analog Output # 1 of the I/O board connector # 2
GPIO2.DAC2	Analog Output # 2 of the I/O board connector # 2
GPIO2.DAC3	Analog Output # 3 of the I/O board connector # 2
GPIO2.DAC4	Analog Output # 4 of the I/O board connector # 2

2.12.2 Function Description

2.12.2.1 GPIOAnalogGainGet

Name

GPIOAnalogGainGet – Gets the gain for one or several analog inputs (ADC)

Input tests

- XPS Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Command format: ERR_WRONG_FORMAT (-7)
- Number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Output parameter type: ERR_WRONG_TYPE_INT (-15)
- Check board: ERR_WRONG_OBJECT_TYPE (-8)
- GPIO name (ADC): ERR_WRONG_OBJECT_TYPE (-8)
- Hardware compatibility or XPS initialization in progress: ERR_NOT_ALLOWED_ACTION (-22)

Description

Gets the gain value for one or several analog inputs. Please refer to Appendix *B.5 Analog I/O* of the XPS Motion Tutorial for further information about ADC gain.

The gain value must be 1, 2, 4 or 8.

The maximum number of INT boards that can be plugged inside the XPS controller is 2, increasing the number of analog outputs from 4 to 8 ADC inputs.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

GPIOAnalogGainGet SocketID GPIOName AnalogGainValue…

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– GPIOName	string	Analog input name (maximum size = 250)

Output parameters

– AnalogGainValue	int	Value of analog input gain

Return

– TCL error (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GPIOAnalogGainGet** (int SocketID, int NbElements, char* GPIONameList, double* AnalogGainValueArray)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– NbElements	int	Number of analog GPIO to read.
– GPIONameList	char *	List of analog input names – separator is comma

Output parameters

– AnalogGainValueArray	int *	Value of analog input gain

Return

– Function error code

Visual Basic

Prototype

Long **GPIOAnalogGainGet** (ByVal SocketID As Long, ByVal NbElements As Long, GPIONameList As String, AnalogGainValueArray As Long)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" Function
– NbElements	long	Number of analog GPIO to read.
– GPIONameList	string	List of analog input names – separator is comma

Output parameters

– AnalogGainValueArray	long	Value of analog input gain

Return

– Function error code

Matlab

Prototype

[Error, AnalogGainValueArray] **GPIOAnalogGainGet** (int32 SocketID, cstring GPIONameArray)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" Function
– GPIONameArray	cstring	Analog input name array (maximum size = 250)

Return

– Error	int32	Function error code
– AnalogGainValueArray	int32	Value of analog input gain

Python

Prototype

Prototype

[Error, AnalogGainValueArray] **GPIOAnalogGainGet** (integer SocketID, string GPIONameArray)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– GPIONameArray	string	Analog input name array (maximum size = 250)

Return

– Error	int	Function error code
– AnalogGainValueArray	int	Value of analog input gain

2.12.2.2 GPIOAnalogGainSet

Name

GPIOAnalogGainSet – Sets a gain for one or several analog inputs (ADC)

Input tests

- XPS Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Command format: ERR_WRONG_FORMAT (-7)
- Number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Input parameter type: ERR_WRONG_TYPE_INT (-15)
- Check board: ERR_WRONG_OBJECT_TYPE (-8)
- GPIO name (ADC): ERR_WRONG_OBJECT_TYPE (-8)
- Hardware compatibility or XPS initialization in progress: ERR_NOT_ALLOWED_ACTION (-22)
- Check outpout value (1, 2, 4 or 8): ERR_PARAMETER_OUT_OF_RANGE (-17)

Description

Sets a gain value for one or several analog inputs.

The gain value can be: 1, 2, 4 or 8

If the conversion of the gain value to bits fails then ERR_NOT_ALLOWED_ACTION is returned.

The maximum number of INT boards, that can be plugged inside the XPS controller, is 2, increasing the number of analog outputs from 4 to 8 ADC inputs.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

GPIOAnalogGainSet SocketID GPIOName AnalogGainValue…

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– GPIOName	string	Analog input name (maximum size = 250)
– AnalogGainValue	int	Value of analog input gain

Output parameters

– None

Return

– TCL error (0 = success or 1 = syntax error) or Function error

C/C++

Prototype

int **GPIOAnalogGainSet** (int SocketID, int NbElements, char* GPIONameList, double* AnalogGainValueArray)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– NbElements	int	Number of analog GPIO to read.
– GPIONameList	char *	List of analog input names – separator is comma
– AnalogGainValueArray	int *	Value of analog input gain

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **GPIOAnalogGainSet** (ByVal SocketID As Long, ByVal NbElements As Long, GPIONameList As String, AnalogGainValueArray As Long)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" Function
– NbElements	long	Number of analog GPIO to read.
– GPIONameList	string	List of analog input names (maximum size = 250)
– AnalogGainValueArray	long	Value of analog input gain

Output parameters

– None

Return

– Function error code

Matlab

Prototype

[Error] **GPIOAnalogGainSet** (int32 SocketID, cstring GPIONameArray, int32 AnalogGainValueArray)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" Function
– GPIONameArray	cstring	Analog input name array (maximum size = 250)
– AnalogGainValueArray	int32	Value of analog input gain

Return

– Error	int32	Function error code

Python

Prototype

[Error] **GPIOAnalogGainSet** (integer SocketID, string GPIONameArray, integer AnalogGainValueArray)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– GPIONameArray	string	Analog input name array (maximum size = 250)
– AnalogGainValueArray	int	Value of analog input gain

Return

– Error	int	Function error code

2.12.2.3 GPIOAnalogGet

Name

GPIOAnalogGet – Reads one or several analog GPIO (DAC or ADC)

Input tests

- XPS Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Command format: ERR_WRONG_FORMAT (-7)
- Number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)
- GPIO name (ADC or DAC): ERR_WRONG_OBJECT_TYPE (-8)
- Hardware compatibility or XPS initialization in progress: ERR_NOT_ALLOWED_ACTION (-22)

Description

Reads one or several analog IO and returns the value(s) in an array.

The GPIO must be one or several analog inputs (ADC) and/or analog outputs (DAC) of GPIO2 connector.

See analog input list §2.12.1.3 and analog output list §2.12.1.4

NOTE

The GPIO2 connector is on the INT board of the controller. The maximum number of INT boards, that can be plugged inside the XPS controller is 2, increasing the number of analog IOs from 4 to 8 ADC and 4 to 8 DAC.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

GPIOAnalogGet SocketID GPIOName AnalogValue …

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– GPIOName	string	Analog GPIO name (maximum size = 250)

Output parameters

– AnalogValue	floating point	Value of analog GPIO (DAC or ADC)

Return

– TCL error (0 = success or 1 = syntax error) or Function error

C/C++

Prototype

int **GPIOAnalogGet** (int SocketID, int NbElements, char* GPIONameList, double* AnalogValueArray)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– NbElements	int	Number of analog GPIO to read.
– GPIONameList	char *	List of analog GPIO names – separator is comma

Output parameters

– AnalogValueArray	double *	Analog GPIO value array (DAC or ADC)

Return

– Function error

Visual Basic

Prototype

Long **GPIOAnalogGet** (ByVal SocketID As Long, ByVal NbElements As Long, GPIONameList As String, AnalogValueArray As Double)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" Function
– NbElements	long	Number of analog GPIO to read.
– GPIONameList	string	List of analog GPIO names (maximum size = 250)

Output parameters

– AnalogValueArray	double	Analog GPIO value array (DAC or ADC)

Return

– Function error

Matlab

Prototype

[Error, AnalogValueArray]**GPIOAnalogGet** (int32 SocketID, cstring GPIONameArray)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" Function
– GPIONameArray	cstring	Analog GPIO name array (maximum size = 250)

Return

– Function error		
– AnalogValueArray	double	Analog GPIO value array (DAC or ADC)

Python

Prototype

[Error, AnalogValueArray]**GPIOAnalogGet** (integer SocketID, string GPIONameArray)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– GPIONameArray	string	Analog GPIO name array (maximum size = 250)

Return

– Error	int	Function error
– AnalogValueArray	double	Analog GPIO value array (DAC or ADC)

2.12.2.4 GPIOAnalogSet

Name

GPIOAnalogSet – Sets one or several analog output (DAC)

Input tests

- XPS Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Command format: ERR_WRONG_FORMAT (-7)
- Number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Output parameter type: ERR_WRONG_TYPE_DOUBLE (-14)
- Check board: ERR_WRONG_OBJECT_TYPE (-8)
- GPIO name (DAC): ERR_WRONG_OBJECT_TYPE (-8)
- Hardware compatibility or XPS initialization in progress: ERR_NOT_ALLOWED_ACTION (-22)
- Check outpout value [-10V..10V]: ERR_PARAMETER_OUT_OF_RANGE (-17)

Description

Sets the analog value for one or several analog outputs (DAC) of the GPIO2 connector.

See analog output list §2.12.1.4

NOTE

The GPIO2 connector is on the INT board in the controller. The maximum number of INT boards, that can be plugged inside the XPS controller is 2, increasing the number of analog outputs from 4 to 8 DAC.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_DOUBLE (-14)
- SUCCESS (0): no error

TCL

Prototype

GPIOAnalogSet SocketID GPIOName AnalogValue …

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
–	GPIOName	string	Analog GPIO name (maximum size = 250)
–	AnalogValue	floating point	Value of analog GPIO (DAC)

Output parameters

– None

Return

– TCL error (0 = success or 1 = syntax error) or Function error

C/C++

Prototype

int **GPIOAnalogSet** (int SocketID, int NbElements, char* GPIONameList, double* AnalogValueArray)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
–	NbElements	int	Number of analog GPIO to read.
–	GPIONameList	char *	List of analog GPIO names – separator is comma
–	AnalogValueArray	double *	Analog GPIO value array (DAC)

Output parameters

– None

Return

– Function error

Visual Basic

Prototype

Long **GPIOAnalogSet** (ByVal SocketID As Long, ByVal NbElements As Long, GPIONameList As String, AnalogValueArray As Double)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" Function
–	NbElements	long	Number of analog GPIO to read.
–	GPIONameList	string	List of analog GPIO names (maximum size = 250)
–	AnalogValueArray	double	Analog GPIO value array (DAC)

Output parameters

– None

Return

– Function error

Matlab

Prototype

[Error] **GPIOAnalogSet** (int32 SocketID, cstring GPIONameArray, double AnalogValueArray)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" Function
–	GPIONameArray	cstring	Analog GPIO name array (maximum size = 250)
–	AnalogValueArray	double	Analog GPIO value array (DAC)

Return

– Function error

Python

Prototype

[Error] **GPIOAnalogSet** (integer SocketID, string GPIONameArray, double AnalogValueArray)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
–	GPIONameArray	string	Analog GPIO name array (maximum size = 250)
–	AnalogValueArray	double	Analog GPIO value array (DAC)

Return

–	Error	int	Function error

2.12.2.5 GPIODigitalGet

Name

GPIODigitalGet – Reads one digital input or output.

Input tests

- XPS Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Command format: ERR_WRONG_FORMAT (-7)
- Number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Output parameter type: ERR_WRONG_TYPE_UNSIGNEDINT (-16)
- GPIO name (DI or DO): ERR_WRONG_OBJECT_TYPE (-8)
- Hardware compatibility or XPS initialization in progress: ERR_NOT_ALLOWED_ACTION (-22)

Description

Returns the value of the digital input (DI) or of the digital output (DO).

See digital output list §2.12.1.2 and digital input list §2.12.1.1

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_UNSIGNEDINT (-16)
- SUCCESS (0): no error

TCL

Prototype

GPIODigitalGet SocketID GPIOName DigitalValue

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
–	GPIOName	string	Digital GPIO name (maximum size = 250)

Output parameters

–	DigitalValue	interger	Digital value (DI or DO)

Return

– TCL error (0 = success or 1 = syntax error) or Function error

C/C++

Prototype

int **GPIODigitalGet** (int SocketID, char* GPIOName, unsigned int* DigitalValue)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– GPIOName	char *	Digital GPIO name (maximum size = 250)

Output parameters

– DigitalValue	uint *	Digital value (DI or DO)

Return

– Function error

Visual Basic

Prototype

Long **GPIODigitalGet** (ByVal SocketID As Long, GPIOName As String, DigitalValue As Integer)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" Function
– GPIOName	string	Digital GPIO name (maximum size = 250)

Output parameters

– DigitalValue	int	Digital value (DI or DO)

Return

– Function error

Matlab

Prototype

[Error, DigitalValue] **GPIODigitalGet** (int32 SocketID, cstring GPIOName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" Function
– GPIOName	cstring	Digital GPIO name (maximum size = 250)

Return

– Function error		
– DigitalValue	uint16Ptr	Digital value (DI or DO)

Python

Prototype

[Error, DigitalValue] **GPIODigitalGet** (integer SocketID, string GPIOName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– GPIOName	string	Digital GPIO name (maximum size = 250)

Return

– Error	int	Function error
– DigitalValue	ushort *	Digital value (DI or DO)

2.12.2.6 GPIODigitalSet

Name

GPIODigitalSet – Sets one digital output.

Input tests

- XPS Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Command format: ERR_WRONG_FORMAT (-7)
- Number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Output parameter type: ERR_WRONG_TYPE_UNSIGNEDINT (-16)
- GPIO name (DO): ERR_WRONG_OBJECT_TYPE (-8)
- Hardware compatibility or XPS initialization in progress: ERR_NOT_ALLOWED_ACTION (-22)

Description

Sets the value of the selected digital output (DO).

See digital output list §2.12.1.2

Error codes

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_NOT_ALLOWED_ACTION (-22)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_UNSIGNEDINT (-16)
- SUCCESS (0): no error

TCL

Prototype

GPIODigitalSet SocketID GPIOName Mask DigitalOutputValue

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– GPIOName	string	Digital output name (maximum size = 250)
– Mask	int	Mask
– DigitalOutputValue	int	Digital output value

Output parameters

– None

Return

– TCL error (0 = success or 1 = syntax error) or Function error

C/C++

Prototype

int **GPIODigitalSet** (int SocketID, char* GPIOName, unsigned short Mask, unsigned short DigitalOutputValue)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– GPIOName	char *	Digital output name (maximum size = 250)
– Mask	ushort	Mask
– DigitalOutputValue	ushort	Digital output value

Output parameters

– None

Return

– Function error

Visual Basic

Prototype

Long **GPIODigitalSet** (ByVal SocketID As Long, GPIOName As String, ByVal Mask As Integer, ByVal DigitalOutputValue As Integer)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" Function
– GPIOName	string	Digital output name (maximum size = 250)
– Mask	int	Mask
– DigitalOutputValue	int	Digital output value

Output parameters

– None

Return

– Function error

Matlab

Prototype

[Error] **GPIODigitalSet** (int32 SocketID, cstring GPIOName, uint16 Mask, uint16 DigitalOutputValue)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" Function
– GPIOName	cstring	Digital output name (maximum size = 250)
– Mask	uint16	Mask
– DigitalOutputValue	uint16	Digital output value

Return

– Error	int32	Function error code

Python

Prototype

[Error] **GPIODigitalSet** (integer SocketID, string GPIOName, unsigned short Mask, unsigned short DigitalOutputValue)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– GPIOName	string	Digital output name (maximum size = 250)
– Mask	ushort	Mask
– DigitalOutputValue	ushort	Digital output value

Return

– Error	int	Function error code

2.13 Gathering

2.13.1 Function Description

2.13.1.1 GatheringConfigurationGet

Name

GatheringConfigurationGet – Returns the current configuration of the internally triggered data gathering.

Input tests

- XPS Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Command format: ERR_WRONG_FORMAT (-7)
- Number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Output parameter type: ERR_WRONG_TYPE_ CHAR (-13)
- Gathering must be configured: ERR_GATHERING_NOT_CONFIGURED (-32)

Description

This function returns the current configuration of the internally triggered data gathering.

Use the "GatheringListGet" function to retrieve a complete list of allowed gathering types.

For a more thorough description of the internal data gathering capability, please refer to the XPS Motion Tutorial section Data Gathering/Internal Data Gathering.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GATHERING_NOT_CONFIGURED (-32)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_TYPE_CHAR (-13)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

GatheringConfigurationGet $SocketID TypeList

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Output parameters

–	TypeList	string	List of configured gathering types (separator is semicolon)

Return

– TCL error (0 = success or 1 = syntax error) or Function error

C/C++

Prototype

int **GatheringConfigurationGet** (int SocketID, char * TypeList)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

–	TypeList	char *	List of configured gathering types (separator is semicolon)

Return

– Function error

Visual Basic

Prototype

Long **GatheringConfigurationGet** (ByVal SocketID As Long, ByVal TypeList As String)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" Function

Output parameters

–	TypeList	string	List of configured gathering types (separator is semicolon)

Return

– Function error

Matlab

Prototype

[Error, TypeList] **GatheringConfigurationGet** (int32 SocketID)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" Function

Return

– Error	int32	Function error
– TypeList	cstring	List of configured gathering types (separator is semicolon)

Python

Prototype

[Error, TypeList] **GatheringConfigurationGet** (integer SocketID)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Return

– Error	int	Function error
– TypeList	string	List of configured gathering types (separator is semicolon)

2.13.1.2 GatheringConfigurationSet

Name

GatheringConfigurationSet – Configures a data gathering action.

Input tests

- XPS Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Command format: ERR_WRONG_FORMAT (-7)
- Number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check input gathering mnemonic: ERR_MNEMOTYPEGATHERING (-29)
- Gathering must not be in progress: ERR_GATHERING_RUNNING (-43)

Description

Defines one or several types of data gathered during the internal triggered data gathering.

Maximum of 1000000 points can be acquired.

Maximum of 25 data types can be configured in a gathering.

Gathering data types listed below:

PositionerName.CorrectorOutput
PositionerName.CurrentAcceleration
PositionerName.CurrentPosition
PositionerName.CurrentVelocity
PositionerName.FollowingError
PositionerName.SetpointAcceleration
PositionerName.SetpointPosition
PositionerName.SetpointVelocity
PositionerName.ExcitationSignalInput
GPIO1.DI
GPIO1.DO
GPIO2.DI
GPIO3.DI
GPIO3.DO
GPIO4.DI
GPIO4.DO
GPIO2.ADC1
GPIO2.ADC2
GPIO2.ADC3
GPIO2.ADC4
GPIO2.DAC1
GPIO2.DAC2
GPIO2.DAC3
GPIO2.DAC4
F_Delta_Z (*for focus process only*)

F_diff (*for focus process only*)

F_diff_nfr (*for focus process only*)

F_an_diff (*for focus process only*)

F_dig_diff (*for focus process only*)

Z_Avr_CurrPos (*for focus process only*)

XPos (*for focus process only*)

XAcc (*for focus process only*)

YPos (*for focus process only*)

YAcc (for focus process only)

ISRCorrectorTimePeriod

ISRCorrectorTimeUsage

ISRProfilerTimeUsage

ISRServitudesTimeUsage

CPUTotalLoadRatio

The "GatheringListGet" function can be used to retrieve a complete list of gathering types.

For a more thorough description of the internal data gathering capability, please refer to the XPS Motion Tutorial section Data Gathering/Internal Data Gathering.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GATHERING_RUNNING (-43)
- ERR_IN_INITIALIZATION (-21)
- ERR_MNEMOTYPEGATHERING (-29)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_TYPE_CHAR (-13)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

GatheringConfigurationSet $SocketID $TypeList

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
–	TypeList	string	List of configured gathering types

Output parameters

– None

Return

– TCL error (0 = success or 1 = syntax error) or Function error

C/C++

Prototype

int **GatheringConfigurationSet** (int SocketID, int NbElements, char * TypeArray)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– NbElements	int	Number of types
– TypeArray	char *	Array of configured gathering types

Output parameters

– None

Return

– Function error

Visual Basic

Prototype

Long **GatheringConfigurationSet** (ByVal SocketID As Long, ByVal NbElements As Long, ByVal TypeNameArray As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" Function
– NbElements	long	Number of types
– TypeNameArray	string	Array of configured gathering types

Output parameters

– None

Return

– Function error

Matlab

Prototype

[Error] **GatheringConfigurationSet** (int32 SocketID, cstring TypeArray)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" Function
– TypeArray	cstring	Array of configured gathering types

Return

– Error	int32	Function error

Python

Prototype

[Error] **GatheringConfigurationSet** (integer SocketID, string TypeArray)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
–	TypeArray	string	Array of configured gathering types

Return

–	Error	int	Function error

2.13.1.3 GatheringCurrentNumberGet

Name

GatheringCurrentNumberGet – Returns the current and maximum number of gathered data points.

Input tests

- XPS Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Command format: ERR_WRONG_FORMAT (-7)
- Number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Output parameter type: ERR_WRONG_TYPE_INT (-15)
- Gathering must be configured: ERR_GATHERING_NOT_CONFIGURED (-32)

Description

This function returns the current and maximum number of data points gathered during the internal triggered data gathering.

For a more thorough description of the internal data gathering capability, please refer to the XPS Motion Tutorial section Data Gathering/Internal Data Gathering.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GATHERING_NOT_CONFIGURED (-32)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_TYPE_INT (-15)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

GatheringCurrentNumberGet $SocketID CurrentNumber MaxSamplesNumber

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Output parameters

–	CurrentNumber	int	Current number during acquisition
–	MaxSamplesNumber	int	Maximum number of samples

Return

– TCL error (0 = success or 1 = syntax error) or Function error

C/C++

Prototype

int **GatheringCurrentNumberGet** (int SocketID, int * CurrentNumber, int * MaxSamplesNumber)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– CurrentNumber	int *	Current number during acquisition
– MaxSamplesNumber	int *	Maximum number of samples

Return

– Function error

Visual Basic

Prototype

Long **GatheringCurrentNumberGet** (ByVal SocketID As Long, CurrentNumber As Long, MaxSamplesNumber As Long)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" Function

Output parameters

– CurrentNumber	long	Current number during acquisition
– MaxSamplesNumber	long	Maximum number of samples

Return

– Function error

Matlab

Prototype

[Error, CurrentNumber, MaxSamplesNumber] **GatheringCurrentNumberGet** (int32 SocketID)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" Function

Return

– Error	int32	Function error
– CurrentNumber	int32	Current number during acquisition
– MaxSamplesNumber	int32	Maximum number of samples

Python

Prototype

[Error, CurrentNumber, MaxSamplesNumber] **GatheringCurrentNumberGet** (integer SocketID)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
–	Return		
–	Error	int	Function error
–	CurrentNumber	int	Current number during acquisition
–	MaxSamplesNumber	int	Maximum number of samples

2.13.1.4 GatheringDataAcquire

Name

GatheringDataAcquire – Acquires one data set manually.

Input tests

- XPS Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Command format: ERR_WRONG_FORMAT (-7)
- Number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Gathering must be configured: ERR_GATHERING_NOT_CONFIGURED (-32)
- Gathering must not be in progress: ERR_GATHERING_RUNNING (-43)
- Check gathering buffer size: ERR_GATHERING_BUFFER_FULL (-111)

Description

This function acquires manually, one data set (configured by the "GatheringConfigurationSet" function).

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GATHERING_BUFFER_FULL (-111)
- ERR_GATHERING_NOT_CONFIGURED (-32)
- ERR_GATHERING_RUNNING (-43)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

GatheringDataAcquire $SocketID

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Output parameters

– None

Return

– TCL error (0 = success or 1 = syntax error) or Function error

C/C++

Prototype

int **GatheringDataAcquire** (int SocketID)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– None

Return

– Function error

Visual Basic

Prototype

Long **GatheringDataAcquire** (ByVal SocketID As Long)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" Function

Output parameters

– None

Return

– Function error

Matlab

Prototype

[Error] **GatheringDataAcquire** (int32 SocketID)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" Function

Return

–	Error	int32	Function error

Python

Prototype

[Error] **GatheringDataAcquire** (integer SocketID)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Return

–	Error	int	Function error

2.13.1.5 GatheringDataGet

Name

GatheringDataGet – Reads one data line from the current gathering buffer.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Check command format: ERR_WRONG_FORMAT (-7)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check input parameter type: ERR_WRONG_TYPE_INT (-15)
- Check output parameter type: ERR_WRONG_TYPE_CHAR (-13)
- Check gathering state: ERR_GATHERING_NOT_CONFIGURED (-32)
- Check index number: ERR_PARAMETER_OUT_OF_RANGE (-17)
  - IndexPoint ≥ 0
  - IndexPoint < currently gathered data number

Description

This function reads a line of data from the current gathering buffer. The buffer line number is defined by the index of an acquired point.

The separator is ";" in the returned data line.

A gathering must be configured to use this function, else ERR_GATHERING_NOT_CONFIGURED (-32) is returned.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GATHERING_NOT_CONFIGURED (-32)
- ERR_IN_INITIALIZATION (-21)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

GatheringDataGet $SocketID $IndexPoint DataBufferLine

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– IndexPoint	int	Index of an acquired data from the current gathering buffer.

Output parameters

– DataBufferLine	string	String contains values from the current buffer at the selected index.

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GatheringDataGet** (int SocketID, int IndexPoint, char *DataBufferLine)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– IndexPoint	int	Index of an acquired data from the current gathering buffer.

Output parameters

– DataBufferLine	char *	String contains values from the current buffer at the selected index.

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **GatheringDataGet** (ByVal SocketID As Long, ByVal IndexPoint As Long, DataBufferLine As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– IndexPoint	long	Index of an acquired data from the current gathering buffer.

Output parameters

– DataBufferLine	string	String contains values from the current buffer at the selected index.

Return

– Error	long	Function error code

Matlab

Prototype

[Error, DataBufferLine] **GatheringDataGet** (int32 SocketID, cstring IndexPoint)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– IndexPoint	int32	Index of an acquired data from the current gathering buffer.

Return

– Error	int32	Function error code
– DataBufferLine	cstring	String contains values from the current buffer at the selected index.

Python

Prototype

[Error, DataBufferLine] **GatheringDataGet** (integer SocketID, string UserName, string Password)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– IndexPoint	int	Index of an acquired data from the current gathering buffer.

Return

– Error	int	Function error code
– DataBufferLine	string	String contains values from the current buffer at the selected index.

2.13.1.6 GatheringDataMultipleLinesGet

Name

GatheringDataMultipleLinesGet – Reads several data lines from the current gathering buffer in memory.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Check command format: ERR_WRONG_FORMAT (-7)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check input parameter type: ERR_WRONG_TYPE_INT (-15)
- Check output parameter type: ERR_WRONG_TYPE_CHAR (-13)
- Check gathering state: ERR_GATHERING_NOT_CONFIGURED (-32)
- Check index number: ERR_PARAMETER_OUT_OF_RANGE (-17)
  - IndexPoint ≥ 0 (**Note:** index #0 = line #1)
  - IndexPoint < currently gathered data number

Description

This function reads one or several data lines from the current gathering buffer. The buffer line number is defined by the index of an acquired point.

The separator is ";" in the returned data line and the end of each line is carriage return "\n".

A gathering must be configured to use this function, else ERR_GATHERING_NOT_CONFIGURED (-32) is returned.

Example of gathering buffer in memory:

index	Data1	Data2	Data3	Data4	Data5
0 →	1	10	0.1	21	100
1 →	2	20	0.2	22	102
2 →	3	30	0.3	23	103
3 →	4	40	0.4	24	104
5 →	5	50	0.5	25	105

GatheringDataMultipleLinesGet(0, 3, myString)

=> 0 = the start line is #1

=> 3 = the number of lines to read is 3

=> myString = buffer to get the part of buffer (32767 characters maximum)

index	Data1	Data2	Data3	Data4	Data5
0 →	1	10	0.1	21	100
1 →	2	20	0.2	22	102
2 →	3	30	0.3	23	103
3 →	4	40	0.4	24	104
5 →	5	50	0.5	25	105

"myString" result:

```
1;10;0.1;21;100
2;20;0.2;22;102
3;30;0.3;23;103
```

GatheringDataMultipleLinesGet(1, 4, myString)

=> 1 = the start line is #2

=> 4 = the number of lines to read is 4

=> myString = buffer to get the part of buffer (65536 characters maximum)

index	Data1	Data2	Data3	Data4	Data5
0 →	1	10	0.1	21	100
1 →	2	20	0.2	22	102
2 →	3	30	0.3	23	103
3 →	4	40	0.4	24	104
5 →	5	50	0.5	25	105

"myString" result:

```
2;20;0.2;22;102
3;30;0.3;23;103
4;40;0.4;24;104
5;50;0.5;25;105
```

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GATHERING_NOT_CONFIGURED (-32)
- ERR_IN_INITIALIZATION (-21)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

GatheringDataMultipleLinesGet $SocketID $IndexPoint DataBufferLine

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	IndexPoint	int Index of an acquired data from the current gathering buffer.
– NbLines	int	Number of lines to get.

Output parameters

– DataBufferLine	string	String contains lines from the current buffer at the selected index.

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GatheringDataMultipleLinesGet** (int SocketID, int IndexPoint, char *DataBufferLine)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– IndexPoint	int	Index of an acquired data from the current gathering buffer.
– NbLines	int	Number of lines to get.

Output parameters

– DataBufferLine	char *	String contains lines from the current buffer at the selected index.

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **GatheringDataMultipleLinesGet** (ByVal SocketID As Long, ByVal IndexPoint As Long, DataBufferLine As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– IndexPoint	long	Index of an acquired data from the current gathering buffer.
– NbLines	long	Number of lines to get.

Output parameters

– DataBufferLine	string	String contains lines from the current buffer at the selected index.

Return

– Error	long	Function error code

Matlab

Prototype

[Error, DataBufferLine] **GatheringDataMultipleLinesGet** (int32 SocketID, cstring IndexPoint)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– IndexPoint	int32	Index of an acquired data from the current gathering buffer.
– NbLines	int32	Number of lines to get.

Return

– Error	int32	Function error code
– DataBufferLine	cstring	String contains lines from the current buffer at the selected index.

Python

Prototype

[Error, DataBufferLine] **GatheringDataMultipleLinesGet** (integer SocketID, string UserName, string Password)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– IndexPoint	int	Index of an acquired data from the current gathering buffer.
– NbLines	int	Number of lines to get.

Return

– Error	int	Function error code
– DataBufferLine	string	String contains lines from the current buffer at the selected index.

2.13.1.7 GatheringExternalConfigurationGet

Name

GatheringExternalConfigurationGet – Returns the current configuration of an externally triggered data gathering.

Input tests

- XPS Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Command format: ERR_WRONG_FORMAT (-7)
- Number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Output parameter type: ERR_WRONG_TYPE_ CHAR (-13)
- Gathering must be configured: ERR_GATHERING_NOT_CONFIGURED (-32)

Description

This function returns the current configuration of an externally triggered data gathering.

Use the "GatheringExternalListGet" function to retrieve a complete list of external gathering types.

For a more thorough description of the external data gathering capability, please refer to the XPS Motion Tutorial section Data Gathering/External Data Gathering.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GATHERING_NOT_CONFIGURED (-32)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_TYPE_CHAR (-13)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

GatheringExternalConfigurationGet $SocketID TypeList

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Output parameters

–	TypeList	string	List of configured gathering types (separator is semicolon)

Return

– TCL error (0 = success or 1 = syntax error) or Function error

C/C++

Prototype

int **GatheringExternalConfigurationGet** (int SocketID, char * TypeList)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– TypeList	char *	List of configured gathering types (separator is semicolon)

Return

– Function error

Visual Basic

Prototype

Long **GatheringExternalConfigurationGet** (ByVal SocketID As Long, ByVal TypeList As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" Function

Output parameters

– TypeList	string	List of configured gathering types (separator is semicolon)

Return

– Function error

Matlab

Prototype

[Error, TypeList] GatheringExternalConfigurationGet (int32 SocketID)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" Function

Return

– Error	int32	Function error
– TypeList	cstring	List of configured gathering types (separator is semicolon)

Python

Prototype

[Error, TypeList] **GatheringExternalConfigurationGet** (integer SocketID)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Return

– Error	int	Function error
– TypeList	string	List of configured gathering types (separator is semicolon)

2.13.1.8 GatheringExternalConfigurationSet

Name

GatheringExternalConfigurationSet – Configures an externally gathering.

Input tests

- XPS Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Command format: ERR_WRONG_FORMAT (-7)
- Number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check input external gathering mnemonic: ERR_MNEMOTYPEGATHERING (-29)
- Gathering must not be in progress: ERR_GATHERING_RUNNING (-43)

Description

Defines one or several types of data gathered during the external triggered data gathering.

Maximum of 1000000 points can be acquired.

Maximum of 25 data types can be configured in a gathering.

External gathering data types are listed below:

PositionerName.ExternalLatchPosition

GPIO2.ADC1

GPIO2.ADC2

GPIO2.ADC3

GPIO2.ADC4

GPIO2.DAC1

GPIO2.DAC2

GPIO2.DAC3

GPIO2.DAC4

Z_Avr_CurrPos (*for focus process only*)

The "GatheringExternalListGet" function can be used to retrieve a complete list of gathering types.

For a more thorough description of the external data gathering capability, please refer to the XPS Motion Tutorial section Data Gathering/External Data Gathering.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GATHERING_RUNNING (-43)
- ERR_IN_INITIALIZATION (-21)
- ERR_MNEMOTYPEGATHERING (-29)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_TYPE_CHAR (-13)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

GatheringExternalConfigurationSet $SocketID $TypeList

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
–	TypeList	string	List of configured gathering types

Output parameters

– None

Return

– TCL error (0 = success or 1 = syntax error) or Function error

C/C++

Prototype

int **GatheringExternalConfigurationSet** (int SocketID, int NbElements, char * TypeArray)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	NbElements	int	Number of types
–	TypeArray	char *	Array of configured gathering types

Output parameters

– None

Return

– Function error

Visual Basic

Prototype

Long **GatheringExternalConfigurationSet** (ByVal SocketID As Long, ByVal NbElements As Long, ByVal TypeNameArray As String)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" Function
–	NbElements	long	Number of types
–	TypeNameArray	string	Array of configured gathering types

Output parameters

– None

Return

– Function error

Matlab

Prototype

[Error] **GatheringExternalConfigurationSet** (int32 SocketID, cstring TypeArray)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" Function
– TypeArray	cstring	Array of configured gathering types

Return

– Error	int32	Function error

Python

Prototype

[Error] **GatheringExternalConfigurationSet** (integer SocketID, string TypeArray)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– TypeArray	string	Array of configured gathering types

Return

– Error	int	Function error

2.13.1.9 GatheringExternalCurrentNumberGet

Name

GatheringExternalCurrentNumberGet – Returns the current and maximum number of external gathered data points.

Input tests

- XPS Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Command format: ERR_WRONG_FORMAT (-7)
- Number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Output parameter type: ERR_WRONG_TYPE_INT (-15)
- External gathering must be configured: ERR_GATHERING_NOT_CONFIGURED (-32)

Description

This function returns the current and maximum number of data points gathered during an externally triggered data gathering.

For a more thorough description of the external data gathering capability, please refer to the XPS Motion Tutorial section Data Gathering/External Data Gathering.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GATHERING_NOT_CONFIGURED (-32)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_TYPE_INT (-15)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

GatheringExternalCurrentNumberGet $SocketID CurrentNumber MaxSamplesNumber

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Output parameters

–	CurrentNumber	int	Current number during acquisition
–	MaxSamplesNumber	int	Maximum number of samples

Return

– TCL error (0 = success or 1 = syntax error) or Function error

C/C++

Prototype

int **GatheringExternalCurrentNumberGet** (int SocketID, int * CurrentNumber, int *..... MaxSamplesNumber)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

–	CurrentNumber	int *	Current number during acquisition
–	MaxSamplesNumber	int *	Maximum number of samples

Return

– Function error

Visual Basic

Prototype

Long **GatheringExternalCurrentNumberGet** (ByVal SocketID As Long, CurrentNumber As Long, MaxSamplesNumber As Long)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" Function

Output parameters

–	CurrentNumber	long	Current number during acquisition
–	MaxSamplesNumber	long	Maximum number of samples

Return

– Function error

Matlab

Prototype

[Error, CurrentNumber, MaxSamplesNumber]
GatheringExternalCurrentNumberGet (int32 SocketID)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" Function

Return

–	Error	int32	Function error
–	CurrentNumber	int32	Current number during acquisition
–	MaxSamplesNumber	int32	Maximum number of samples

Python

Prototype

[Error, CurrentNumber, MaxSamplesNumber]
GatheringExternalCurrentNumberGet (integer SocketID)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Return

–	Error	int	Function error
–	CurrentNumber	int	Current number during acquisition
–	MaxSamplesNumber	int	Maximum number of samples

2.13.1.10 GatheringExternalDataGet

Name

GatheringExternalDataGet – Reads one line of data from the current external gathering buffer.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Check command format: ERR_WRONG_FORMAT (-7)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check input parameter type: ERR_WRONG_TYPE_INT (-15)
- Check output parameter type: ERR_WRONG_TYPE_CHAR (-13)
- Check gathering state: ERR_GATHERING_NOT_CONFIGURED (-32)
- Check index number: ERR_PARAMETER_OUT_OF_RANGE (-17)
  - IndexPoint ≥ 0
  - IndexPoint < currently gathered data number

Description

This function reads a line of data from the current gathering gathering buffer. The buffer line number is defined by the index of an acquired point.

The separator is ";" in the returned data line.

A gathering must be configured to use this function, else ERR_GATHERING_NOT_CONFIGURED (-32) is returned.

Error codes

ERR_FATAL_INIT (-20)

ERR_GATHERING_NOT_CONFIGURED (-32)

ERR_IN_INITIALIZATION (-21)

ERR_PARAMETER_OUT_OF_RANGE (-17)

ERR_WRONG_FORMAT (-7)

ERR_WRONG_PARAMETERS_NUMBER (-9)

ERR_WRONG_TYPE_CHAR (-13)

ERR_WRONG_TYPE_INT (-15)

SUCCESS (0): no error

TCL

Prototype

GatheringExternalDataGet $SocketID $IndexPoint DataBufferLine

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– IndexPoint	int	Index of an acquired data from the current gathering buffer.

Output parameters

– DataBufferLine	string	String contains values from the current buffer at the selected index.

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **GatheringExternalDataGet** (int SocketID, int IndexPoint, char *DataBufferLine)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– IndexPoint	int	Index of an acquired data from the current gathering buffer.

Output parameters

– DataBufferLine	char *	String contains values from the current buffer at the selected index.

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **GatheringExternalDataGet** (ByVal SocketID As Long, ByVal IndexPoint As Long, DataBufferLine As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– IndexPoint	long	Index of an acquired data from the current gathering buffer.

Output parameters

– DataBufferLine	string	String contains values from the current buffer at the selected index.

Return

– Error	long	Function error code

Matlab

Prototype

[Error, DataBufferLine] **GatheringExternalDataGet** (int32 SocketID, cstring IndexPoint)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– IndexPoint	int32	Index of an acquired data from the current gathering buffer.

Return

– Error	int32	Function error code
– DataBufferLine	cstring	String contains values from the current buffer at the selected index.

Python

Prototype

[Error, DataBufferLine] **GatheringExternalDataGet** (integer SocketID, string UserName, string Password)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– IndexPoint	int	Index of an acquired data from the current gathering buffer.

Return

– Error	int	Function error code
– DataBufferLine	string	String contains values from the current buffer at the selected index.

2.13.1.11 GatheringExternalStopAndSave

Name

GatheringExternalStopAndSave – Stops externally triggered data gathering and saves the data into the XPS controller.

Input tests

- XPS Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Command format: ERR_WRONG_FORMAT (-7)
- Number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check number of data (> 0): ERR_GATHERING_NOT_STARTED (-30)
- Check file opening: ERR_WRITE_FILE (-60)

Description

This function stops externally triggered data gathering and saves the data into the XPS controller. Gathered data are stored in the file "GatheringExternal.dat" in the "..\Public" folder of the XPS controller.

For a more thorough description of the external data gathering capability, please refer to the XPS Motion Tutorial section Data Gathering/External Data Gathering.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GATHERING_NOT_STARTED (-30)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRITE_FILE (-60)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error
- TCL

Prototype

GatheringExternalStopAndSave $SocketID

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Output parameters

– None

Return

– TCL error (0 = success or 1 = syntax error) or Function error

C/C++

Prototype

int GatheringExternalStopAndSave (int SocketID)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– None

Return

– Function error

Visual Basic

Prototype

Long **GatheringExternalStopAndSave** (ByVal SocketID As Long)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" Function

Output parameters

– None

Return

– Function error

Matlab

Prototype

[Error] GatheringExternalStopAndSave (int32 SocketID)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" Function

Return

–	Error	int32	Function error

Python

Prototype

[Error] GatheringExternalStopAndSave (integer SocketID)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Return

–	Error	int	Function error

2.13.1.12 GatheringReset

Name

GatheringReset – Resets gathered data to start new gathering from scratch.

Input tests

- XPS Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Command format: ERR_WRONG_FORMAT (-7)
- Number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Gathering must not be in progress: ERR_GATHERING_RUNNING (-43)

Description

This function resets to start a new gathering from scratch.

The number of gathered data is set to zero.

For a more thorough description of the internal data gathering capability, please refer to the XPS Motion Tutorial section Data Gathering/Internal Data Gathering.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GATHERING_RUNNING (-43)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

GatheringReset $SocketID

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Output parameters

– None

Return

– TCL error (0 = success or 1 = syntax error) or Function error

C/C++

Prototype

int **GatheringReset** (int SocketID)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– None

Return

– Function error

Visual Basic

Prototype

Long **GatheringReset** (ByVal SocketID As Long)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" Function

Output parameters

– None

Return

– Function error

Matlab

Prototype

[Error] **GatheringReset** (int32 SocketID)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" Function

Return

–	Error	int32	Function error

Python

Prototype

[Error] **GatheringReset** (integer SocketID)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Return

–	Error	int	Function error

2.13.1.13 GatheringRun

Name

GatheringRun – Starts to gather data.

Input tests

- XPS Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Command format: ERR_WRONG_FORMAT (-7)
- Number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Gathering must not be in progress: ERR_GATHERING_RUNNING (-43)
- Gathering must be configured: ERR_GATHERING_NOT_CONFIGURED (-32)

Description

This function starts a new data gathering.

The data gathering needs to be configured before using this function (See GatheringConfigurationSet)

The parameters are the number of data points to be gathered and the divisor of the frequency (servo frequency) at which the data gathering will be done.

For a more thorough description of the internal data gathering capability, please refer to the XPS Motion Tutorial section Data Gathering/Internal Data Gathering.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GATHERING_RUNNING (-43)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_GATHERING_NOT_CONFIGURED (-32)
- SUCCESS (0): no error

TCL

Prototype

GatheringRun $SocketID $DataNumber $Divisor

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– DataNumber	int	The number of data line to gather
– Divisor	int	The divisor of the servo frequency

Output parameters

– None

Return

– TCL error (0 = success or 1 = syntax error) or Function error

C/C++

Prototype

int **GatheringRun** (int SocketID, int DataNumber, int Divisor)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– DataNumber	int	The number of data line to gather
– Divisor	int	The divisor of the servo frequency

Output parameters

– None

Return

– Function error

Visual Basic

Prototype

Long **GatheringRun** (ByVal SocketID As Long, ByVal DataNumber As Long, ByVal Divisor As Long)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" Function
– DataNumber	long	The number of data line to gather
– Divisor	long	The divisor of the servo frequency

Output parameters

– None

Return

– Function error

Matlab

Prototype

[Error] **GatheringRun** (int32 SocketID, int32 DataNumber, int32 Divisor)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" Function
– DataNumber	int32	The number of data line to gather
– Divisor	int32	The divisor of the servo frequency

Return

– Error	int32	Function error

Python

Prototype

[Error] **GatheringRun** (integer SocketID, integer DataNumber, integer Divisor)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– DataNumber	int	The number of data line to gather
– Divisor	int	The divisor of the servo frequency

Return

– Error	int	Function error

2.13.1.14 GatheringRunAppend

Name

GatheringRunAppend – Repeat the gathering, continuing from the last gathered data.

Input tests

- XPS Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Command format: ERR_WRONG_FORMAT (-7)
- Number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Gathering must not be in progress: ERR_GATHERING_RUNNING (-43)
- Gathering must be configured: ERR_GATHERING_NOT_CONFIGURED (-32)

Description

This function repeats the gathering from the data point that was previously stopped, while the gathering current data number has not reached the *DataNumber* previously specified using the *GatheringRun()* function.

The gathering must to be configured, executed and stopped before using this function (see *GatheringConfigurationSet, GatheringRun, GatheringStop* functions)

For a more thorough description of the internal data gathering capability, please refer to the XPS Motion Tutorial section Data Gathering/Internal Data Gathering.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GATHERING_RUNNING (-43)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_GATHERING_NOT_CONFIGURED (-32)
- SUCCESS (0): no error

TCL

Prototype

GatheringRunAppend $SocketID

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Output parameters

– None

Return

– TCL error (0 = success or 1 = syntax error) or Function error

C/C++

Prototype

int **GatheringRunAppend** (int SocketID)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– None

Return

– Function error

Visual Basic

Prototype

Long **GatheringRunAppend** (ByVal SocketID As Long)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" Function

Output parameters

– None

Return

– Function error

Matlab

Prototype

[Error] **GatheringRunAppend** (int32 SocketID)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" Function

Return

–	Error	int32	Function error

Python

Prototype

[Error] **GatheringRunAppend** (integer SocketID)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Return

–	Error	int	Function error

2.13.1.15 GatheringStop

Name

GatheringStopAndSave – Stops internally and externally triggered data gathering.

Input tests

- XPS Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Command format: ERR_WRONG_FORMAT (-7)
- Number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check number of data (> 0): ERR_GATHERING_NOT_STARTED (-30)
- Check file opening: ERR_WRITE_FILE (-60)

Description

This function stops internally and externally triggered data gathering. To save to a file, use GatheringStopAndSave function.

For a more thorough description of the internal data gathering capability, please refer to the XPS Motion Tutorial section Data Gathering/Internal Data Gathering.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GATHERING_NOT_STARTED (-30)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRITE_FILE (-60)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

GatheringStop $SocketID

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Output parameters

– None

Return

– TCL error (0 = success or 1 = syntax error) or Function error

C/C++

Prototype

int **GatheringStop** (int SocketID)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– None

Return

– Function error

Visual Basic

Prototype

Long **GatheringStop** (ByVal SocketID As Long)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" Function

Output parameters

– None

Return

– Function error

Matlab

Prototype

[Error] **GatheringStop** (int32 SocketID)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" Function

Return

–	Error	int32	Function error

Python

Prototype

[Error] **GatheringStop** (integer SocketID)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Return

–	Error	int	Function error

2.13.1.16 GatheringStopAndSave

Name

GatheringStopAndSave – Stops internally triggered data gathering and saves data into the XPS controller.

Input tests

- XPS Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Command format: ERR_WRONG_FORMAT (-7)
- Number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check number of data (> 0): ERR_GATHERING_NOT_STARTED (-30)
- Check file opening: ERR_WRITE_FILE (-60)

Description

This function stops internally triggered data gathering and saves the data to the XPS controller. Data is stored in the file GATHERING.DAT in the "..\Public" folder of the XPS controller.

For a more thorough description of the internal data gathering capability, please refer to the XPS Motion Tutorial section Data Gathering/Internal Data Gathering.

Error codes

- ERR_FATAL_INIT (-20)
- ERR_GATHERING_NOT_STARTED (-30)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRITE_FILE (-60)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- SUCCESS (0): no error

TCL

Prototype

GatheringStopAndSave $SocketID

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Output parameters

– None

Return

– TCL error (0 = success or 1 = syntax error) or Function error

C/C++

Prototype

int **GatheringStopAndSave** (int SocketID)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– None

Return

– Function error

Visual Basic

Prototype

Long **GatheringStopAndSave** (ByVal SocketID As Long)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" Function

Output parameters

– None

Return

– Function error

Matlab

Prototype

[Error] **GatheringStopAndSave** (int32 SocketID)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" Function

Return

–	Error	int32	Function error

Python

Prototype

[Error] **GatheringStopAndSave** (integer SocketID)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Return

–	Error	int	Function error

2.14 Events and Actions

2.14.1 Functions Description

2.14.1.1 EventExtendedAllGet

Name

EventExtendedAllGet – Returns all "event and action" identifiers in progress.

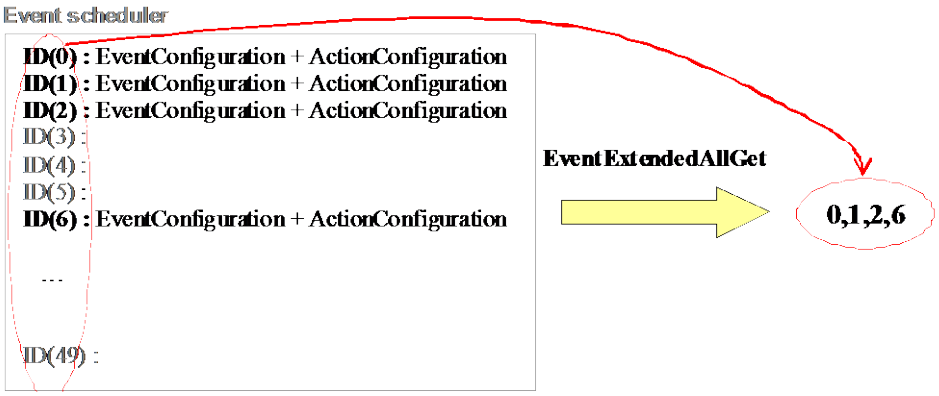
Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Output parameter type: ERR_WRONG_TYPE_CHAR (-13)

Description

Gets the list of all "event and action" combination identifiers from the event scheduler (filled by the **ExtendedEventStart** or **ExtendedEventWait** function).

The list separator is a comma. If no "event and action" combination is in progress (in the event scheduler) then the error ERR_EVENT_ID_UNDEFINED (-83) is returned.



Errors

- ERR_EVENT_ID_UNDEFINED (-83)
- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13)
- SUCCESS (0): no error

TCL

Prototype

EventExtendedAllGet $SocketID $EventID EventIdentifiersList

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– EventID	int	"Event and action" identifier from "ExtendedEventStart"

Output parameters

– EventIdentifiersList	string	List of "event and action" identifiers in scheduler

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **EventExtendedAllGet** (int SocketID, int EventID, char * EventIdentifiersList)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– EventID	int	"Event and action" identifier from "ExtendedEventStart"

Output parameters

– EventIdentifiersList	char *	List of "event and action" identifiers in scheduler

Return

– Function error code

Visual Basic

Prototype

Long **EventExtendedAllGet** (ByVal SocketID As Long, ByVal EventID As Long, EventIdentifiersList As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" Function
– EventID	long	"Event and action" identifier from "ExtendedEventStart"

Output parameters

– EventIdentifiersList	string	List of "event and action" identifiers in scheduler

Return

– Function error code

Matlab

Prototype

[Error, EventIdentifiersList] **EventExtendedAllGet** (int32 SocketID, int32 EventID)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" Function
– EventID	int32	"Event and action" identifier from "ExtendedEventStart"

Return

– Error	int	Function error code
– EventIdentifiersList	cstring	List of "event and action" identifiers in scheduler

Python

Prototype

[Error, EventIdentifiersList] **EventExtendedAllGet** (integer SocketID, integer EventID)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– EventID	int	"Event and action" identifier from "ExtendedEventStart"

Return

– Error	int	Function error code
– EventIdentifiersList	string	List of "event and action" identifiers in scheduler

2.14.1.2 EventExtendedConfigurationActionGet

Name

EventExtendedConfigurationActionGet – Returns the action combination defined in buffer.

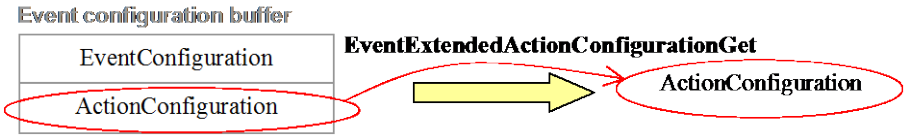
Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Output parameter type: ERR_WRONG_TYPE_CHAR (-13)
- Last action configuration in memory: ERR_ACTIONS_NOT_CONFIGURED (-81)

Description

Returns the combination of action(s) defined by "EventExtendedConfigurationActionSet" function.

If no action is configured in the buffer, ERR_ACTIONS_NOT_CONFIGURED (-81) is returned.



NOTE

This function doesn't return the last activated action. A combination of action(s) can be defined in the buffer but not activated.

Errors

- ERR_ACTIONS_NOT_CONFIGURED (-81)
- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13)
- SUCCESS (0): no error

TCL

Prototype

Event**ExtendedConfigurationActionGet** $SocketID ActionConfiguration

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Output parameters

– ActionConfigurationStringAction combination configured in buffer

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **EventExtendedConfigurationActionGet** (int SocketID, char * ActionConfiguration)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Output parameters

– ActionConfigurationStringAction combination configured in buffer

Return

– Function error code

Visual Basic

Prototype

Long **EventExtendedConfigurationActionGet** (ByVal SocketID As Long, ActionConfiguration As String)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" Function

Output parameters

– ActionConfigurationStringAction combination configured in buffer

Return

– Function error code

Matlab

Prototype

[Error, ActionConfiguration] **EventExtendedConfigurationActionGet** (int32 SocketID)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" Function

Return

– Error	int	Function error code
– ActionConfiguration	cstring	Action combination configured in buffer

Python

Prototype

[Error, ActionConfiguration] **EventExtendedConfigurationActionGet** (integer SocketID)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Return

– Error	int	Function error code
– ActionConfiguration	string	Action combination configured in buffer

2.14.1.3 EventExtendedConfigurationActionSet

Name

EventExtendedConfigurationActionSet – Defines a combination of one or several actions in buffer.

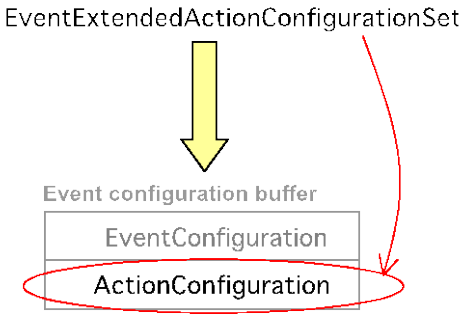
Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of arguments: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Output parameter type: ERR_WRONG_TYPE_CHAR (-13)
- Last action configured in memory: ERR_ACTIONS_NOT_CONFIGURED (-81)
- Action name: ERR_MNEMO_ACTION (-39)
- Action parameters: ERR_PARAMETER_OUT_OF_RANGE (-17), ERR_WRONG_OBJECT_TYPE (-8)
- Action to execute: ERR_GATHERING_NOT_CONFIGURED (-32) "Gathering" action.

Description

Defines a combination of one or several actions but does not activate the actions. Use the "EventExtendedStart" function to activate these defined actions. For each action, 4 parameters can be configured … see event specification to see which are necessary. The actions are defined in § "Events and Actions" in the XPS user's manual.

The number of actions in a combination is limited to 10 actions.



Action list

1. GPIOName.DOToggle
2. GPIOName.DOPulse
3. GPIOName.DOSet
4. GPIOName.DACSet.SetpointPosition
5. GPIOName.DACSet.SetpointVelocity
6. GPIOName.DACSet.SetpointAcceleration
7. GPIOName.DACSet.CurrentPosition
8. GPIOName.DACSet.CurrentVelocity
9. GPIOName.DACSet.Value
10. ExecuteTCLScript
11. KillTCLScript

12. ExternalGatheringRun
13. GatheringRun
14. GatheringOneData
15. GatheringStop
16. GatheringRunAppend
17. GroupName.MoveAbort
18. GroupName.MoveAbortFast
19. GlobalArrayDoubleSet
20. GlobalArrayStringSet
21. ExecuteCommand
22. EventRemove
23. SynchronizeProfiler

Action parameters

Actor					Parameters			
Group	Positioner	GPIO	Timer#	Action name	1	2	3	4
		•		DOToggle	Mask	0	0	0
		•		DOPulse	Mask	0	0	0
		•		DOSet	Mask	Value	0	0
		•		DACSet.SetpointPosition	Positioner name	Gain	Offset	0
		•		DACSet.SetpointVelocity	Positioner name	Gain	Offset	0
		•		DACSet.SetpointAcceleration	Positioner name	Gain	Offset	0
		•		DACSet.CurrentPosition	Positioner name	Gain	Offset	0
		•		DACSet.CurrentVelocity	Positioner name	Gain	Offset	0
		•		DACSet.Value	Value	0	0	0
				ExecuteTCLScript	TCL file name	Task name	Arguments	0
				KillTCLScript	Task name	0	0	0
				GatheringOneData	0	0	0	0
				GatheringRun	Nb of points	Divisor	0	0
				GatheringRunAppend	0	0	0	0
				GatheringStop	0	0	0	0
				ExternalGatheringRun	Nb of points	Divisor	0	0
•				MoveAbort	0	0	0	0
•				MoveAbortFast	Deceleration Multiplier	0	0	0
				GlobalArrayDoubleSet	Global variable number	Numeric value	0	0
				GlobalArrayStringSet	Global variable number	String value	0	0
				ExecuteCommand	Function name	Arguments string (Between {} and separator is the semi-column.)	Task name	0
				EventRemove	Identifier (-1 for itself)	0	0	0
				SynchronizeProfiler	0	0	0	0

NOTE

Before activating the defined actions, you must configure the events. Only then, can you use the "EventExtendedStart" or "EventExtendedWait" function.

For the "ExecuteTCLScript" action, the "ActionParameter3" represents a list of arguments, which must be separated with a semicolon (;).

Errors

- ERR_ACTIONS_NOT_CONFIGURED (-1)
- ERR_FATAL_INIT (-20)
- ERR_GATHERING_NOT_CONFIGURED (-32)
- ERR_IN_INITIALIZATION (-21)
- ERR_MNEMO_ACTION (-39)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13)
- SUCCESS (0): no error

TCL

Prototype

EventExtendedConfigurationActionSet $SocketID **{$**ExtendedActionName $ActionParameter1 $ActionParameter2 $ActionParameter3 $ActionParameter4**} …**

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
–	ExtendedActionName	String	Event full name (maximum size = 250) - see § Events -
–	ActionParameter1	String	Optional action's parameter #1 (maximum size = 250)
–	ActionParameter2	String	Optional action's parameter #2 (maximum size = 250)
–	ActionParameter3	String	Optional action's parameter #3 (maximum size = 250)
–	ActionParameter4	String	Optional action's parameter #4 (maximum size = 250)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **EventExtendedConfigurationActionSet** (int SocketID, int NbElements, char* ExtendedActionName, char* ActionParameter1, char* ActionParameter2, char* ActionParameter3, char* ActionParameter4)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
–	NbElements	int	Number of events in configuration.
–	ExtendedAction	Name	char*Event full name (maximum size = 250) - see § Events -
–	ActionParameter1	char*	optional action's parameter #1 (maximum size = 250)
–	ActionParameter2	char*	optional action's parameter #2 (maximum size = 250)
–	ActionParameter3	char*	optional action's parameter #3 (maximum size = 250)
–	ActionParameter4	char*	optional action's parameter #4 (maximum size = 250)

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **EventExtendedConfigurationActionSet** (ByVal SocketID As Long, ByVal NbElements As Long, ByVal ExtendedActionName As String, ByVal ActionParameter1 As String, ByVal ActionParameter2 As String, ByVal ActionParameter3 As String, ByVal ActionParameter4 As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" Function
– NbElements	long	Number of events in configuration.
– ExtendedAction	Name	StringEvent full name (maximum size = 250) - see § Events -
– ActionParameter1	string	optional action's parameter #1 (maximum size = 250)
– ActionParameter2	string	optional action's parameter #2 (maximum size = 250)
– ActionParameter3	string	optional action's parameter #3 (maximum size = 250)
– ActionParameter4	string	optional action's parameter #4 (maximum size = 250)

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **EventExtendedConfigurationActionSet** (int32 SocketID, cstring ExtendedActionName, cstring ActionParameter1, cstring ActionParameter2, cstring ActionParameter3, cstring ActionParameter4)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" Function
– ExtendedAction	Name	cstringEvent full name (maximum size = 250) - see § Events -
– ActionParameter1	cstring	optional action's parameter #1 (maximum size = 250)
– ActionParameter2	cstring	optional action's parameter #2 (maximum size = 250)
– ActionParameter3	cstring	optional action's parameter #3 (maximum size = 250)
– ActionParameter4	cstring	optional action's parameter #4 (maximum size = 250)

Return

– Function error code

Python

Prototype

integer **EventExtendedConfigurationActionSet** (integer SocketID, string ExtendedActionName, string ActionParameter1, string ActionParameter2, string ActionParameter3, string ActionParameter4)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– ExtendedAction	Name	StringEvent full name (maximum size = 250) - see § Events -
– ActionParameter1	string	optional action's parameter #1 (maximum size = 250)
– ActionParameter2	string	optional action's parameter #2 (maximum size = 250)
– ActionParameter3	string	optional action's parameter #3 (maximum size = 250)
– ActionParameter4	string	optional action's parameter #4 (maximum size = 250)

Return

– Function error code

2.14.1.4 EventExtendedConfigurationTriggerGet

Name

EventExtendedConfigurationTriggerGet – Returns the trigger defined in the buffer.

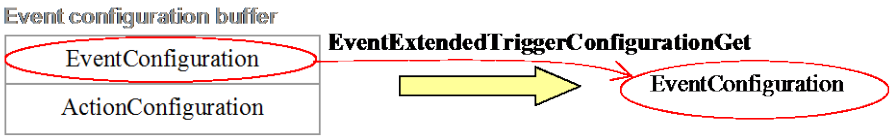
Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Parameter type: ERR_WRONG_TYPE_CHAR (-13)
- Last event configuration in memory: ERR_EVENTS_NOT_CONFIGURED (-80)

Description

Returns the last event defined in buffer by "EventExtendedConfigurationTriggerSet" function.

If no event is defined in buffer, ERR_EVENTS_NOT_CONFIGURED (-80) is returned.



NOTE

This function doesn't return the last activated event. An event can be configured but not activated.

Errors

- SUCCESS (0): no error
- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13)
- ERR_EVENTS_NOT_CONFIGURED (-80)

TCL

Prototype

Event**ExtendedConfigurationTriggerGet** $SocketID EventTriggerConfiguration

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Output parameters

– EventTriggerConfiguration	string	Event combination configured in buffer

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **EventExtendedConfigurationTriggerGet** (int SocketID, char * EventTriggerConfiguration)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Output parameters

– EventTriggerConfiguration	string	Event combination configured in buffer

Return

– Function error code

Visual Basic

Prototype

Long **EventExtendedConfigurationTriggerGet** (ByVal SocketID As Long, EventTriggerConfiguration As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" Function

Output parameters

– EventTriggerConfiguration	string	Event combination configured in buffer

Return

– Function error code

Matlab

Prototype

[Error, EventTriggerConfiguration] **EventExtendedConfigurationTriggerGet** (int32 SocketID)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" Function

Return

– Error	int	Function error code
– EventTriggerConfiguration	cstring	Event combination configured in buffer

Python

Prototype

[Error, EventTriggerConfiguration] **EventExtendedConfigurationTriggerGet** (integer SocketID)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Return

– Error	int	Function error code
– EventTriggerConfiguration	string	Event combination configured in buffer

2.14.1.5 EventExtendedConfigurationTriggerSet

Name

EventExtendedConfigurationTriggerSet - Defines a combination of one or several events in buffer.

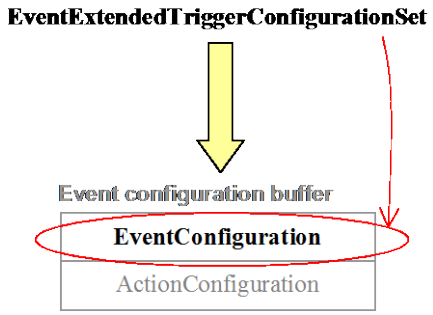
Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Event name: ERR_MNEMO_EVENT (-40)
- Event actor: ERR_WRONG_OBJECT_TYPE (-8)

Description

Defines one trigger (combination of one or several events). To activate the trigger, use the "EventExtendedStart" function. For each event, 4 parameters can be configured… see event specification to see the necessary parameters. The events are defined in § "Events and Actions" in the XPS user's manual.

The number of events in a combination is limited to 10 events.



Each full event name is defined as **[actor].[category].event** (see Event list):

[actor] - Optional actor name (Group name, Positioner name, GPIO name or Nothing)

[category] - Optional category name (Event category or Nothing)

event - Event name

Event list

1. Always
2. Immediate
3. Timer1
4. Timer2
5. Timer3
6. Timer4
7. Timer5
8. PositionerName.MotionDone
9. PositionerName.WarningFollowingError
10. PositionerName.PositionerError
11. PositionerName.PositionerHardwareStatus

12. PositionerName.Category.ConstantVelocityStart
13. PositionerName.Category.ConstantVelocityEnd
14. PositionerName.Category.ConstantVelocityState
15. PositionerName.Category.ConstantAccelerationStart
16. PositionerName.Category.ConstantAccelerationEnd
17. PositionerName.Category.ConstantAccelerationState
18. PositionerName.Category.ConstantDecelerationStart
19. PositionerName.Category.ConstantDecelerationEnd
20. PositionerName.Category.ConstantDecelerationState
21. PositionerName.Category.MotionStart
22. PositionerName.Category.MotionEnd
23. PositionerName.Category.MotionState
24. GroupName.Category.TrajectoryStart
25. GroupName.Category.TrajectoryEnd
26. GroupName.Category.TrajectoryState
27. GroupName.Category.TrajectoryPulse
28. GroupName.Category.TrajectoryPulseState
29. GroupName.Category.ElementNumberStart
30. GroupName.Category.ElementNumberState
31. GPIOName.ADCHighLimit
32. GPIOName.ADCLowLimit
33. GPIOName.DILowHigh
34. GPIOName.DIHighLow
35. GPIOName.DIToggle
36. GPIOName.DILowState
37. GPIOName.DIHighState
38. FocusSyncTrigger *(for focus process only)*
39. WarningFollowingError
40. DoubleGlobalArrayEqual
41. DoubleGlobalArrayDifferent
42. DoubleGlobalArrayInferiorOrEqual
43. DoubleGlobalArraySuperiorOrEqual
44. DoubleGlobalArrayInferior
45. DoubleGlobalArraySuperior
46. DoubleGlobalArrayInWindow
47. DoubleGlobalArrayOutWindow

Category list for "profile" positioner events

- SGamma
- Slave
- Spin
- Jog
- TrackingPosition
- TrackingVelocity

Category list for "trajectory" group events

- XYLineArc
- Spline
- PVT

NOTE

The events listed below are allowed only for the XPS-Qn Precision Platform controller:

FocusSyncTrigger

WarningFollowingError

GPIOName.DILowState

GPIOName.DIHighState

DoubleGlobalArrayEqual

DoubleGlobalArrayDifferent

DoubleGlobalArrayInferiorOrEqual

DoubleGlobalArraySuperiorOrEqual

DoubleGlobalArrayInferior

DoubleGlobalArraySuperior

DoubleGlobalArrayInWindow

DoubleGlobalArrayOutWindow

Event parameters

Actor				Category					Event name	Parameters			
Group	Positioner	GPIO	Timer #	SGamma	Jog	XY LineArc	XYZ Spline	PVT		1	2	3	4
									Immediate	0	0	0	0
									Always	0	0	0	0
			•						Timer	0	0	0	0
	•			•	•				MotionStart	0	0	0	0
	•			•	•				MotionStop	0	0	0	0
	•			•	•				MotionState	0	0	0	0
	•			•	•				ConstantVelocityStart	0	0	0	0
	•			•					ConstantVelocityEnd	0	0	0	0
	•			•	•				ConstantVelocityState	0	0	0	0
	•			•					ConstantAccelerationStart	0	0	0	0
	•			•					ConstantAccelerationEnd	0	0	0	0
	•			•					ConstantAccelerationState	0	0	0	0
	•			•					ConstantDecelerationStart	0	0	0	0
	•			•					ConstantDecelerationEnd	0	0	0	0
	•			•					ConstantDecelerationState	0	0	0	0
•						•	•	•	TrajectoryStart	0	0	0	0
•						•	•	•	TrajectoryEnd	0	0	0	0
•						•	•	•	TrajectoryState	0	0	0	0
•						•	•	•	ElementNumberStart	Element #	0	0	0
•						•	•	•	ElementNumberState	Element #	0	0	0
•	•								MotionDone	0	0	0	0
•						•		•	TrajectoryPulse	0	0	0	0
•						•		•	TrajectoryPulseOutputState	0	0	0	0
		•							DILowState	Bit index	0	0	0
		•							DIHighState	Bit index	0	0	0
		•							DILowHigh	Bit index	0	0	0
		•							DIHighLow	Bit index	0	0	0
		•							DIToggle	Bit index	0	0	0
		•							ADCHighLimit	Value	0	0	0
		•							ADCLowLimit	Value	0	0	0
	•								PositionerError	Mask	0	0	0
	•								PositionerHardwareStatus	Mask	0	0	0
	•								WarningFollowingError	0	0	0	0
									FocusSyncTrigger	0	0	0	0
									DoubleGlobalArrayEqual	Global variable number	Value to check	0	0
									DoubleGlobalArrayDifferent	Global variable number	Value to check	0	0
									DoubleGlobalArrayInferiorOrEqual	Global variable number	Value to check	0	0
									DoubleGlobalArraySuperiorOrEqual	Global variable number	Value to check	0	0
									DoubleGlobalArrayInferior	Global variable number	Value to check	0	0
									DoubleGlobalArraySuperior	Global variable number	Value to check	0	0
									DoubleGlobalArrayInWindow	Global variable number	Min value	Max value	0
									DoubleGlobalArrayOutWindow	Global variable number	Min value	Max value	0

NOTE

Before activating this event combination, you must define one or several action(s) with the "EventExtendedConfigurationTriggerSet" function. Next, use the "EventExtendedStart" or "EventExtendedWait" function to launch these defined "event and action".

Errors

ERR_FATAL_INIT (-20)

ERR_IN_INITIALIZATION (-21)

ERR_MNEMO_EVENT (-40)

ERR_WRONG_FORMAT (-7)

ERR_WRONG_OBJECT_TYPE (-8)

ERR_WRONG_PARAMETERS_NUMBER (-9)

SUCCESS (0): no error

TCL

Prototype

EventExtendedConfigurationTriggerSet $**SocketID **{$FullEventName $EventParameter1 $EventParameter2 $EventParameter3 $EventParameter4**} …**

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– ExtendedEvent	Name	StringEvent full name (maximum size = 250) - see § Events -
– EventParameter1	string	optional event's parameter #1 (maximum size = 250)
– EventParameter2	string	optional event's parameter #2 (maximum size = 250)
– EventParameter3	string	optional event's parameter #3 (maximum size = 250)
– EventParameter4	string	optional event's parameter #4 (maximum size = 250)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **EventExtendedConfigurationTriggerSet** (int SocketID, int NbElements, char* ExtendedEventName, char* EventParameter1, char* EventParameter2 , char* EventParameter3, char* EventParameter4)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
–	NbElements	int	Number of events in configuration.
–	ExtendedEvent	Name	char* list of event full names (maximum size = 250) – separator is ';'
–	EventParameter1	char*	list of optional event's parameter #1 (maximum size = 250)
–	EventParameter2	char*	list of optional event's parameter #2 (maximum size = 250)
–	EventParameter3	char*	list of optional event's parameter #3 (maximum size = 250)
–	EventParameter4	char*	list of optional event's parameter #4 (maximum size = 250)

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **EventExtendedConfigurationTriggerSet** (ByVal SocketID As Long, ByVal NbElements As Long, ByVal ExtendedEventName As String, ByVal EventParameter1 As String, ByVal EventParameter2 As String, ByVal EventParameter3 As String, ByVal EventParameter4 As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" Function
– NbElements	long	Number of events in configuration.
– ExtendedEvent	Name	string array of event full names (maximum size = 250) - see § Events
– EventParameter1	string	array of optional event's parameter #1 (maximum size = 250)
– EventParameter2	string	array of optional event's parameter #2 (maximum size = 250)
– EventParameter3	string	array of optional event's parameter #3 (maximum size = 250)
– EventParameter4	string	array of optional event's parameter #4 (maximum size = 250)

Output parameters

– None

Return

– Function error code

Matlab

Prototype

int32 **EventExtendedConfigurationTriggerSet** (int32 SocketID, cstring ExtendedEventName, cstring EventParameter1, cstring EventParameter2, cstring EventParameter3, cstring EventParameter4)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" Function
–	ExtendedEvent	Name	cstring array of event full names (maximum size = 250) - see § Events
–	EventParameter1	cstring	array of optional event's parameter #1 (maximum size = 250)
–	EventParameter2	cstring	array of optional event's parameter #2 (maximum size = 250)
–	EventParameter3	cstring	array of optional event's parameter #3 (maximum size = 250)
–	EventParameter4	cstring	array of optional event's parameter #4 (maximum size = 250)

Return

– Function error code

Python

Prototype

integer **EventExtendedConfigurationTriggerSet** (integer SocketID, string ExtendedEventName, string EventParameter1, string EventParameter2, string EventParameter3, string EventParameter4)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
–	ExtendedEvent	Name	string array of event full names (maximum size = 250) - see § Events
–	EventParameter1	string	array of optional event's parameter #1 (maximum size = 250)
–	EventParameter2	string	array of optional event's parameter #2 (maximum size = 250)
–	EventParameter3	string	array of optional event's parameter #3 (maximum size = 250)
–	EventParameter4	string	array of optional event's parameter #4 (maximum size = 250)

Return

– Function error code

2.14.1.6 EventExtendedGet

Name

EventExtendedGet – Returns the details of "event and action" combinations in scheduler defined by an identifier.

Input tests

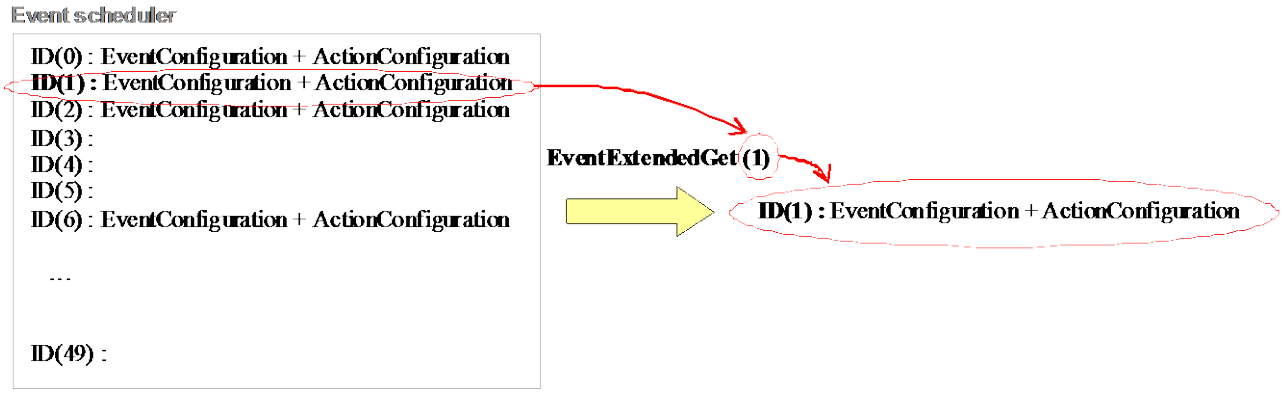
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Parameter type: ERR_WRONG_TYPE_INT (-15), ERR_WRONG_TYPE_CHAR (-13)
- Event identifier [0:49]: ERR_EVENT_ID_UNDEFINED (-83)

Description

Returns the composition of events and actions in progress defined by an identifier. This identifier is defined in the "EventExtendedStart" function.

The identifier must be defined between 0 and 49, if its value is "–1" then it's not defined.

If the configured event is already deleted, ERR_EVENT_ID_UNDEFINED (-83) is returned.



Errors

SUCCESS (0): no error

ERR_FATAL_INIT (-20)

ERR_IN_INITIALIZATION (-21)

ERR_WRONG_FORMAT (-7)

ERR_WRONG_PARAMETERS_NUMBER (-9)

ERR_WRONG_TYPE_INT (-15)

ERR_WRONG_TYPE_CHAR (-13)

ERR_EVENT_ID_UNDEFINED (-83)

TCL

Prototype

EventExtendedGet $SocketID $EventID EventConfiguration ActionConfiguration

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– EventID	int	"Event and action" identifier from "ExtendedEventStart"

Output parameters

– EventConfiguration	string	Event combination defined in scheduler
– ActionConfiguration	string	Action combination defined in scheduler

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **EventExtendedGet** (int SocketID, int EventID, char * EventConfiguration, char * ActionConfiguration)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– EventID	int	"Event and action" identifier from "ExtendedEventStart"

Output parameters

– EventConfiguration	char *	Event combination defined in scheduler
– ActionConfiguration	char *	Action combination defined in scheduler

Return

– Function error code

Visual Basic

Prototype

Long **EventExtendedGet** (ByVal SocketID As Long, ByVal EventID As Long, EventConfiguration As String, ActionConfiguration As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" Function
– EventID	long	"Event and action" identifier from "ExtendedEventStart"

Output parameters

– EventConfigurationStringEvent combination defined in scheduler

– ActionConfigurationStringAction combination defined in scheduler

Return

– Function error code

Matlab

Prototype

[Error, EventConfiguration, ActionConfiguration] **EventExtendedGet** (int32 SocketID, int32 EventID)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" Function
– EventID	int32	"Event and action" identifier from "ExtendedEventStart"

Return

– Error	int	Function error code
– EventConfiguration	cstring	Event combination defined in scheduler
– ActionConfiguration	cstring	Action combination defined in scheduler

Python

Prototype

[Error, EventConfiguration, ActionConfiguration] **EventExtendedGet** (integer SocketID, integer EventID)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– EventID	int	"Event and action" identifier from "ExtendedEventStart"

Return

– Error	int	Function error code
– EventConfiguration	string	Event combination defined in scheduler
– ActionConfiguration	string	Action combination defined in scheduler

2.14.1.7 EventExtendedRemove

Name

EventExtendedRemove – Removes an "event and action" combination in the scheduler defined by an identifier.

Input tests

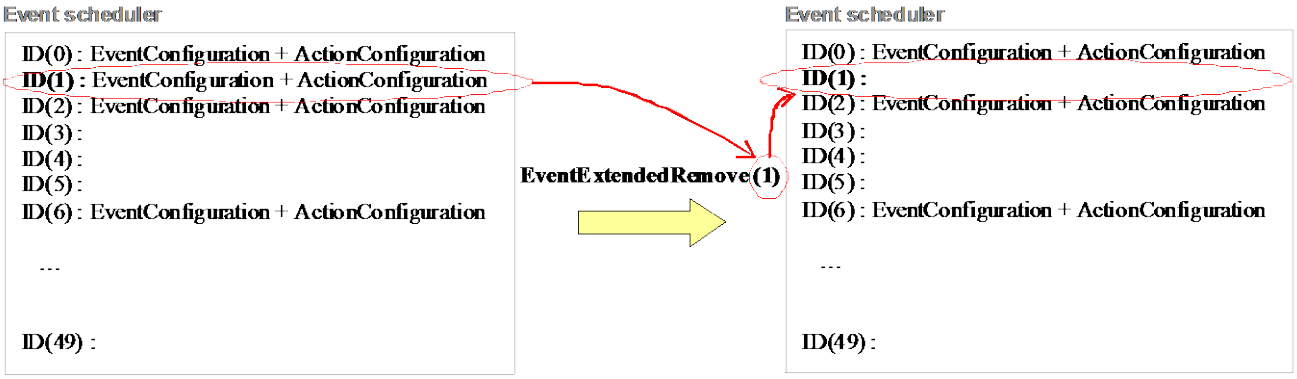
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of parameters [1]: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Parameter type: ERR_WRONG_TYPE_INT (-15)
- Event identifier [0:49]: ERR_EVENT_ID_UNDEFINED (-83), ERR_PARAMETER_OUT_OF_RANGE (-17)
- Actor event: ERR_WRONG_OBJECT_TYPE (-8)

Description

Deletes the "event(s) and action(s)" combination in the scheduler defined by an event identifier. This identifier is defined in the "EventExtendedStart" function.

The identifier must be defined between 0 and 49, or -1. If the identifier is equal to "–1", the EventExtendedRemove function removes all current "event and action" combinations.

If the configured event is already deleted, ERR_EVENT_ID_UNDEFINED (-83) is returned.



Errors

- ERR_EVENT_ID_UNDEFINED (-83)
- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

EventExtendedRemove $SocketID $EventID

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– EventID	int	"Event and action" identifier

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **EventExtendedRemove** (int SocketID, int EventID)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– EventID	int	"Event and action" identifier

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **EventExtendedRemove** (ByVal SocketID As Long, ByVal EventID As Long)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" Function
– EventID	long	"Event and action" identifier

Output parameters

– None

Return

– Function error code

Matlab

Prototype

[Error] **EventExtendedRemove** (int32 SocketID, int32 EventID)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" Function
– EventID	int32	"Event and action" identifier

Return

– Error	int32	Function error code

Python

Prototype

[Error] **EventExtendedRemove** (integer SocketID, integer EventID)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– EventID	int	"Event and action" identifier

Return

– Error	int	Function error code

2.14.1.8 EventExtendedStart

Name

EventExtendedStart – Activates the "event and action" defined in the buffer.

Input tests

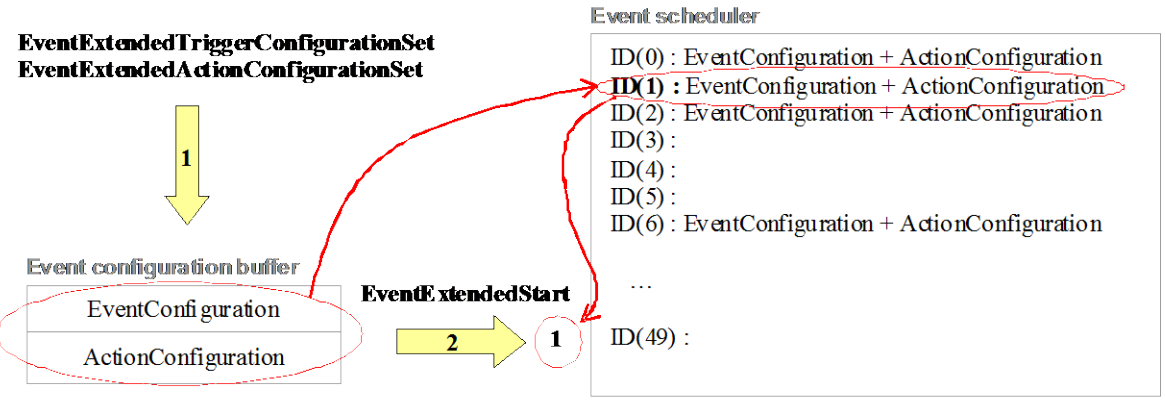
- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of arguments: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Output parameter type: ERR_WRONG_TYPE_INT (-15)
- Number of compositions in execution: ERR_EVENT_BUFFER_FULL (-82)
- Last event configuration in memory: ERR_EVENTS_NOT_CONFIGURED (-80)
- Last action configuration in memory: ERR_ACTIONS_NOT_CONFIGURED (-81)
- Event name to execute: ERR_MNEMO_EVENT (-40), ERR_WRONG_TYPE (-10), ERR_WRONG_OBJECT_TYPE (-8)

Description

Launches the configured event(s) and action(s) from the event configuration buffer into the event scheduler and gets an event identifier. The identifier must be defined between 0 and 49, if its value is "–1" then that means it's not defined.

If no event is configured in buffer, ERR_EVENTS_NOT_CONFIGURED (-80) is returned.

If no action is configured in buffer, ERR_ACTIONS_NOT_CONFIGURED is returned.



NOTE

In the event scheduler, when a configured event has occurred it is deleted from the event scheduler.

CAUTION

If the configured event is PERMANENT then it is not deleted after it occurs, and must use the "EventExtendedRemove" function to delete it.

Errors

- ERR_ACTIONS_NOT_CONFIGURED (-81)
- ERR_EVENT_BUFFER_FULL (-82)
- ERR_EVENTS_NOT_CONFIGURED (-80)
- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_MNEMO_EVENT (-40)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE (-10)
- ERR_WRONG_TYPE_INT (-15)
- SUCCESS (0): no error

TCL

Prototype

EventExtendedStart $SocketID EventID

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Output parameters

– EventID	int	"Event and action" identifier

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **EventExtendedStart** (int SocketID, int * EventID)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Output parameters

– EventID	int *	"Event and action" identifier

Return

– Function error code

Visual Basic

Prototype

Long **EventExtendedStart** (ByVal SocketID As Long, EventID As Long)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" Function

Output parameters

– EventID	long	"Event and action" identifier

Return

– Function error code

Matlab

Prototype

[Error, EventID] **EventExtendedStart** (int32 SocketID)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" Function

Return

– Error	int32	Function error code
– EventID	int32	"Event and action" identifier

Python

Prototype

[Error, EventID] **EventExtendedStart** (integer SocketID)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Return

– Error	int	Function error code
– EventID	int	"Event and action" identifier

2.14.1.9 EventExtendedWait

Name

EventExtendedWait – Activates the last "event" configuration in memory and wait until it occurs.

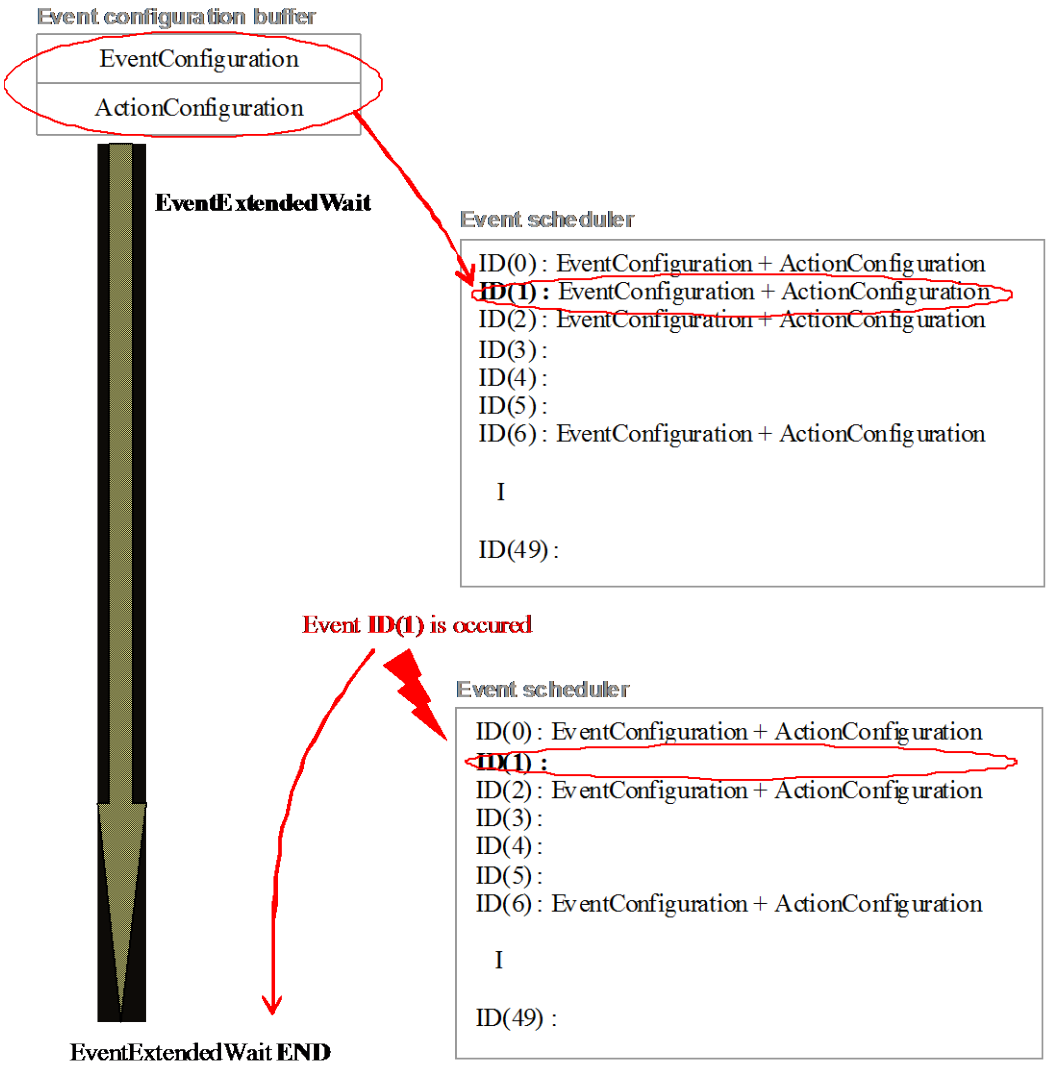
Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of arguments [0]: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Number of compositions in execution: ERR_EVENT_BUFFER_FULL (-82)
- Last event configuration in memory: ERR_EVENTS_NOT_CONFIGURED (-80)
- Event name to execute: ERR_MNEMO_EVENT (-40), ERR_WRONG_TYPE (-10)
- Event actor: ERR_WRONG_OBJECT_TYPE (-8)

Description

Launches the last configured event(s) into the event scheduler and wait until it occurs to unlock the socket.

If no "event and action" combination is configured in the event configuration buffer, ERR_EVENTS_NOT_CONFIGURED (-80) is returned.



Errors

- ERR_EVENT_BUFFER_FULL (-82)
- ERR_EVENTS_NOT_CONFIGURED (-80)
- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_MNEMO_EVENT (-40)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_OBJECT_TYPE (-8)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_ (-10)
- SUCCESS (0): no error

TCL

Prototype

EventExtendedWait $SocketID

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **EventExtendedWait** (int SocketID)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **EventExtendedWait** (ByVal SocketID As Long)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" Function

Output parameters

– None

Return

– Function error code

Matlab

Prototype

[Error] **EventExtendedWait** (int32 SocketID)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" Function

Return

–	Error	int32	Function error code

Python

Prototype

[Error] **EventExtendedWait** (integer SocketID)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Return

–	Error	int	Function error code

2.14.2 Obsolete Functions

Do not use for new projects! These Functions are only maintained for use in previously developed programs.

Use above described Functions.

2.14.2.1 EventAdd

TCL Prototype	int EventAdd (int SocketID, char FullPositionerName[250], char EventName[250], char EventParameter[250], char ActionName[250], char ActionParameter1[250], char ActionParameter2[250], char ActionParameter3[250])
Input parameters	int : SocketID (Socket identifier gets by the "TCP_ConnectToServer" Function) char [250]: FullPositionerName char [250]: EventName (see § Events) char [250]: EventParameter char [250]: ActionName (see § Actions) char [250]: ActionParameter1 char [250]: ActionParameter2 char [250]: ActionParameter3
Output parameters	None
Return	TCL error (0 = success or 1 = syntax error) or Function error
DLL Prototype	int EventAdd (int SocketID, char *FullPositionerName, char *EventName, char *EventParameter, char *ActionName, char *ActionParameter1, char *ActionParameter2, char *ActionParameter3)
Input parameters	int : SocketID (Socket identifier gets by the "TCP_ConnectToServer" Function) char *: FullPositionerName char *: EventName (see § Events) char *: EventParameter char *: ActionName (see § Actions) char *: ActionParameter1 char *: ActionParameter2 char *: ActionParameter3
Output parameters	None
Return	Function error
Function Input tests	Verify the number of parameters. Verify the full positioner name, the event name and the action name. Verify the type of all output parameters. Parameters coherence test.
Function Description	Adds an action associated to an event for the defined positioner. For a more thorough description and a complete list of possible events and the actions, please refer to the XPS Motion Tutorial section Event triggers.
Function Errors	0 -7 -8 -9 -13 -39 -40

2.14.2.2 EventGet

TCL Prototype	int EventGet (int SocketID, char FullPositionerName [250], char EventList[250])
Input parameters	int : SocketID (Socket identifier gets by the "TCP_ConnectToServer" Function) char [250]: FullPositionerName
Output parameters	char [250]: EventList
Return	TCL error (0 = success or 1 = syntax error) or Function error
DLL Prototype	int EventGet (int SocketID, char *FullPositionerName, char *EventList)
Input parameters	int : SocketID (Socket identifier gets by the "TCP_ConnectToServer" Function) char *: FullPositionerName
Output parameters	char *: EventList
Return	Function error

Function Input tests	Verify the number of parameters. Verify the positioner name and the event name. Verify the type of all output parameters. Parameters coherence test.
Function Description	Returns the list of events and actions in progress for the selected positioner. For a more thorough description and a complete list of possible events and the actions, please refer to the XPS Motion Tutorial section Event triggers.
Function Errors	0 -7 -8 -9 -13

2.14.2.3 EventRemove

TCL Prototype	int EventRemove (int SocketID, char FullPositionerName[250], char EventName[250] , char EventParameter[250])
Input parameters	int : SocketID (Socket identifier gets by the "TCP_ConnectToServer" Function) char [250]: FullPositionerName char [250]: EventName (see § Events) char [250]: EventParameter
Output parameters	None
Return	TCL error (0 = success or 1 = syntax error) or Function error

DLL Prototype	int EventRemove (int SocketID, char *FullPositionerName, char *EventName , char *EventParameter)
Input parameters	int : SocketID (Socket identifier gets by the "TCP_ConnectToServer" Function) char *: FullPositionerName char *: EventName (see § Events) char *: EventParameter
Output parameters	None
Return	Function error

Function Input tests	Verify the number of parameters. Verify the positioner name and the event name. Verify the event. Verify the type of all output parameters. Parameters coherence test.
Function Description	Deletes an action associated to an event for the defined positioner. For a more thorough description and a complete list of possible events and the actions, please refer to the XPS Motion Tutorial section Event triggers.
Function Errors	0 -7 -8 -9 -13 -40

2.14.2.4 EventWait

TCL Prototype	int EventWait (int SocketID, char FullPositionerName [250], char EventName[250] , char EventParameter[250])
Input parameters	int : SocketID (Socket identifier gets by the "TCP_ConnectToServer" Function) char [250]: FullPositionerName char [250]: EventName (see § Events) char [250]: EventParameter
Output parameters	None
Return	TCL error (0 = success or 1 = syntax error) or Function error
DLL Prototype	int EventWait (int SocketID, char *FullPositionerName, char *EventName , char *EventParameter)
Input parameters	int : SocketID (Socket identifier gets by the "TCP_ConnectToServer" Function) char *: FullPositionerName char *: EventName (see § Events) char *: EventParameter
Output parameters	None
Return	Function error
Function Input tests	Verify the number of parameters. Verify the positioner name and the event name. Verify the event. Verify the type of all output parameters. Parameters coherence test.
Function Description	Waits for an event for the selected positioner. The socket is locked. As soon as the event occurs, the socket gets unlocked. For a more thorough description and a complete list of possible events and the actions, please refer to the XPS Motion Tutorial section Event triggers.
Function Errors	0 -7 -8 -9 -13 -40

2.15 TCL Programming

2.15.1 Function Description

2.15.1.1 TCLScriptExecute

Name

TCLScriptExecute – Executes a TCL script.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of arguments: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check task name: ERR_WRONG_TCL_TASKNAME (-47)
- Check TCL file name: ERR_UNKNOWN_TCL_FILE (-36)
- Check TCL interpretor (task loading): ERR_TCL_INTERPRETOR (-37)

Description

This function executes a TCL script. The TCL script file must be saved in the folder "..\Public\Scripts" of the XPS controller.

- *TaskName* is a user designation for the TCL script being executed. If two TCL scripts are executed at the same time with the same task name, The ERR_WRONG_TCL_TASKNAME (-47) is returned because having the same TaskName is not allowed.
- *InputArguments* represents the input arguments of the TCL script to be executed. The number of these input arguments is not limited but the string length is limited to 250 characters. The argument separator is a comma.

Errors

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_TCL_INTERPRETOR (-37)
- ERR_UNKNOWN_TCL_FILE (-36)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TCL_TASKNAME (-47)
- SUCCESS (0): no error

TCL

Prototype

TCLScriptExecute $SocketID $TCLFileName $TaskName $InputArguments

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– TCLFileName	string	File name contains the TCL script
– TaskName	string	Task name
– InputArguments	string	Input argument string (separator is a comma)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **TCLScriptExecute** (int SocketID, char *TCLFileName, char *TaskName, char *InputArguments)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– TCLFileName	char *	File name contains the TCL script
– TaskName	char *	Task name
– InputArguments	char *	Input argument string (separator is a comma)

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **TCLScriptExecute** (ByVal SocketID As Long, ByVal TCLFileName As String, ByVal TaskName As String, ByVal InputArguments As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– TCLFileName	string	File name contains the TCL script
– TaskName	string	Task name
– InputArguments	string	Input argument string (separator is a comma)

Output parameters

– None

Return

– Function error code

Matlab

Prototype

[Error] **TCLScriptExecute** (int32 SocketID, cstring TCLFileName, cstring TaskName, cstring InputArguments)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– TCLFileName	cstring	File name contains the TCL script
– TaskName	cstring	Task name
– InputArguments	cstring	Input argument string (separator is a comma)

Return

– Error	int32	Function error code

Python

Prototype

[Error] **TCLScriptExecute** (integer SocketID, string TCLFileName, string TaskName, string InputArguments)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– TCLFileName	string	File name contains the TCL script
– TaskName	string	Task name
– InputArguments	string	Input argument string (separator is a comma)

Return

– Error	int	Function error code

2.15.1.2 TCLScriptExecuteAndWait

Name

TCLScriptExecuteAndWait – Executes a TCL script and waits until the end of execution.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of arguments: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check task name: ERR_WRONG_TCL_TASKNAME (-47)
- Check TCL file name: ERR_UNKNOWN_TCL_FILE (-36)
- Check TCL interpretor (task loading): ERR_TCL_INTERPRETOR (-37)

Description

This function executes a TCL program. The "TCLScriptExecuteAndWait" function is different than the "TCLScriptExecute" function because it blocks the socket until the script teminates. The TCL script file must be saved in the folder "..\Public\Scripts" of the XPS controller. The file extension is ".tcl".

- *TaskName* is a user designation for the TCL script in execution. If two TCL scripts are executed at the same time with the same task name, The ERR_WRONG_TCL_TASKNAME (-47) is returned because having the same TaskName is not allowed.
- *InputArguments* represents the input arguments of the TCL script to be executed. The number of these input arguments is not limited but the string length is limited to 250 characters. The argument separator is a comma.
- *OutputArguments* represents the output arguments of the TCL script to be executed. The number of these output arguments is not limited but the string length is limited to 250 characters. The argument separator is a comma.

Errors

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_TCL_INTERPRETOR (-37)
- ERR_UNKNOWN_TCL_FILE (-36)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TCL_TASKNAME (-47)
- SUCCESS (0): no error

TCL

Prototype

TCLScriptExecuteAndWait $SocketID $TCLFileName $TaskName $InputArguments OutputArguments

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– TCLFileName	string	File name contains the TCL script
– TaskName	string	Task name
– InputArguments	string	Input argument string (separator is a comma)

Output parameters

– OutputArguments	string	Output argument string (separator is a comma)

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **TCLScriptExecuteAndWait** (int SocketID, char *TCLFileName, char *TaskName, char *InputArguments, char *OutputArguments)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– TCLFileName	char *	File name contains the TCL script
– TaskName	char *	Task name
– InputArguments	char *	Input argument string (separator is a comma)

Output parameters

– OutputArguments	char *	Output argument string (separator is a comma)

Return

– Function error code

Visual Basic

Prototype

Long **TCLScriptExecuteAndWait** (ByVal SocketID As Long, ByVal TCLFileName As String, ByVal TaskName As String, ByVal InputArguments As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– TCLFileName	string	File name contains the TCL script
– TaskName	string	Task name
– InputArguments	string	Input argument string (separator is a comma)

Output parameters

– OutputArguments	string	Output argument string (separator is a comma)

Return

– Function error code

Matlab

Prototype

[Error] **TCLScriptExecuteAndWait** (int32 SocketID, cstring TCLFileName, cstring TaskName, cstring InputArguments, cstring OutputArguments)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– TCLFileName	cstring	File name contains the TCL script
– TaskName	cstring	Task name
– InputArguments	cstring	Input argument string (separator is a comma)

Return

– Error	int32	Function error code
– OutputArguments	cstring	Output argument string (separator is a comma)

Python

Prototype

[Error,OutputArguments] **TCLScriptExecuteAndWait** (integer SocketID, string TCLFileName, string TaskName, string InputArguments)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– TCLFileName	string	File name contains the TCL script
– TaskName	string	Task name
– InputArguments	string	Input argument string (separator is a comma)

Return

– Error	int	Function error code
– OutputArguments	string	Output argument string (separator is a comma)

2.15.1.3 TCLScriptExecuteWithPriority

Name

TCLScriptExecuteWithPriority – Executes a TCL script with TCL task given priority.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of arguments: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check task name: ERR_WRONG_TCL_TASKNAME (-47)
- Priority mnemonic incorrect: ERR_PARAMETERS_OUT_OF_RANGE (-17)
- Check TCL file name: ERR_UNKNOWN_TCL_FILE (-36)
- Check TCL interpretor (task loading): ERR_TCL_INTERPRETOR (-37)

Description

This function executes a TCL script with a TCL task and a user-defined priority level. The TCL script file must be saved in the folder "..\Public\Scripts" of the XPS controller.

1. *TaskName* is a user designation for the TCL script in execution. If two TCL scripts are executed at the same time with the same task name, The ERR_WRONG_TCL_TASKNAME (-47) is returned because having the same TaskName is not allowed.
2. *InputArguments* represents the input arguments to the TCL script to be executed. The number of these input arguments is not limited but the string length is limited to 250 characters. The argument separator is a comma.
3. *PriorityLevel* has three possible values: "HIGH", "MEDIUM" and "LOW", with the order being HIGH > MEDIUM > LOW.

Errors

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_TCL_INTERPRETOR (-37)
- ERR_UNKNOWN_TCL_FILE (-36)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_PARAMETER_OUT_OF_RANGE (-17)
- ERR_WRONG_TCL_TASKNAME (-47)
- SUCCESS (0): no error

TCL

Prototype

TCLScriptExecuteWithPriority $SocketID $TCLFileName $TaskName $Priority $InputArguments

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– TCLFileName	string	File name contains the TCL script
– TaskName	string	Task name
– Priority	string	TCL task priority (HIGH, MEDIUM or LOW)
– InputArguments	string	Input argument string (separator is a comma)

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **TCLScriptExecuteWithPriority** (int SocketID, char *TCLFileName, char *TaskName, char *Priority, char *InputArguments)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– TCLFileName	char *	File name contains the TCL script
– TaskName	char *	Task name
– Priority	char *	TCL task priority (HIGH, MEDIUM or LOW)
– InputArguments	char *	Input argument string (separator is a comma)

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **TCLScriptExecuteWithPriority** (ByVal SocketID As Long, ByVal TCLFileName As String, ByVal TaskName As String, ByVal Priority As String , ByVal InputArguments As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– TCLFileName	string	File name contains the TCL script
– TaskName	string	Task name
– Priority	string	TCL task priority (HIGH, MEDIUM or LOW)
– InputArguments	string	Input argument string (separator is a comma)

Output parameters

– None

Return

– Function error code

Matlab

Prototype

[Error] **TCLScriptExecuteWithPriority** (int32 SocketID, cstring TCLFileName, cstring TaskName, cstring Priority, cstring InputArguments)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– TCLFileName	cstring	File name contains the TCL script
– TaskName	cstring	Task name
– Priority	cstring	TCL task priority (HIGH, MEDIUM or LOW)
– InputArguments	cstring	Input argument string (separator is a comma)

Return

– Error	int32	Function error code

Python

Prototype

[Error] **TCLScriptExecuteWithPriority** (integer SocketID, string TCLFileName, string TaskName, string Priority, string InputArguments)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– TCLFileName	string	File name contains the TCL script
– TaskName	string	Task name
– Priority	string	TCL task priority (HIGH, MEDIUM or LOW)
– InputArguments	string	Input argument string (separator is a comma)

Return

– Error	int	Function error code

2.15.1.4 TCLScriptKill

Name

TCLScriptKill – Kills a TCL script.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of arguments: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check TCL interpretor (task loading) and task name: ERR_TCL_SCRIPT_KILL (-38)
- Check semaphore to use the TCL interpretor: ERR_TCL_INTERPRETOR (-37)

Description

This function kills a running TCL script selected using its task name. The task name is a user designation for the TCL script in execution.

NOTE

For the boot script, the task name is "BootScript".

Errors

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_TCL_INTERPRETOR (-37)
- ERR_TCL_SCRIPT_KILL (-38)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TCL_TASKNAME (-47)
- SUCCESS (0): no error

TCL

Prototype

TCLScriptKill $SocketID $TaskName

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
–	TaskName	string	Task name to kill

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **TCLScriptKill** (int SocketID, char *TaskName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
–	TaskName	char *	Task name to kill

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **TCLScriptKill** (ByVal SocketID As Long, ByVal TaskName As String)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
–	TaskName	string	Task name to kill

Output parameters

– None

Return

– Function error code

Matlab

Prototype

[Error] **TCLScriptKill** (int32 SocketID, cstring TaskName)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
–	TaskName	cstring	Task name to kill

Return

–	Error	int32	Function error code

Python

Prototype

[Error] **TCLScriptKill** (integer SocketID, string TaskName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
–	TaskName	string	Task name to kill

Return

–	Error	int	Function error code

2.15.1.5 TCLScriptKillAll

Name

TCLScriptKillAll – Kills all running TCL scripts.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Check TCL interpretor (task loading) and task name: ERR_TCL_SCRIPT_KILL (-38)

Description

This function kills all running TCL scripts.

Errors

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_TCL_SCRIPT_KILL (-38)
- ERR_WRONG_FORMAT (-7)
- SUCCESS (0): no error

TCL

Prototype

TCLScriptKillAll $SocketID

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **TCLScriptKillAll** (int SocketID)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **TCLScriptKillAll** (ByVal SocketID As Long)

Input parameters

–	SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– None

Return

– Function error code

Matlab

Prototype

[Error] **TCLScriptKillAll** (int32 SocketID)

Input parameters

–	SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function

Return

–	Error	int32	Function error code

Python

Prototype

[Error] **TCLScriptKillAll** (integer SocketID)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Return

–	Error	int	Function error code

2.16 Optional Module Programming

The optional module programming manages (execute and stop execution) the written by user program blocks (optional modules) in the XPS controller, with the following conditions:

1. Every optional module is written in C language (GNU with QNX Momentics IDE).
2. The optional module file must be stored in the "/Admin/ UserOptionalModules /" in the XPS controller.

In case a user needs to execute the optional module at boot of the controller *OptionalModuleNames* feature is to be added in the *[GENERAL]* section of system.ref file followed by the name of the optional module created by the user:

system.ref:

```
[GENERAL]
HardwareType = XPS          ; XPS or SPS
FirmwareName = MainController
ExternalModuleNames = XPSRemoteControl
OptionalModuleNames = OptionalModule1Name, OptionalModule2Name
DelayBeforeStartup = 0       ; seconds
[…]
```

2.16.1 Function Description

2.16.1.1 OptionalModuleExecute

Name

OptionalModuleExecute – Executes an optional (user) external module.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of arguments: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check task name: ERR_OPTIONAL_EXTERNAL_MODULE_EXECUTE (-95)

Description

This function executes an optional (user) module. The optional module file must be saved in the folder "\Admin\UserOptionalModules" of the XPS controller.

Errors

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_OPTIONAL_EXTERNAL_MODULE_EXECUTE (-95)
- SUCCESS (0): no error

TCL

Prototype

OptionalModuleExecute $SocketID $ModuleFileName $TaskName

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– ModuleName	string	Module name

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **OptionalModuleExecute** (int SocketID, char * ModuleFileName, char *TaskName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– ModuleName	char *	Module name

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **OptionalModuleExecute** (ByVal SocketID As Long, ByVal ModuleFileName As String, ByVal TaskName As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– ModuleName	string	Module name

Output parameters

– None

Return

– Function error code

Matlab

Prototype

[Error] **OptionalModuleExecute** (int32 SocketID, cstring ModuleFileName, cstring TaskName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– ModuleName	cstring	Module name

Return

– Error	int32	Function error code

Python

Prototype

[Error] **OptionalModuleExecute** (integer SocketID, string ModuleFileName, string TaskName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
– ModuleName	string	Module name

Return

– Error	int	Function error code

2.16.1.2 OptionalModuleKill

Name

OptionalModuleKill – Kills the execution of a optional module.

Input tests

- Controller initialization failed: ERR_FATAL_INIT (-20)
- XPS initialization in progress: ERR_IN_INITIALIZATION (-21)
- Valid command format: ERR_WRONG_FORMAT (-7)
- Number of arguments: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check task name and task killing:
  ERR_OPTIONAL_EXTERNAL_MODULE_KILL (-96)

Description

This function kills a running optional module .

Errors

- ERR_FATAL_INIT (-20)
- ERR_IN_INITIALIZATION (-21)
- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_OPTIONAL_EXTERNAL_MODULE_KILL (-96)
- SUCCESS (0): no error

TCL

Prototype

OptionalModuleKill $SocketID $TaskName

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
–	ModuleName	string	Module name to kill

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **OptionalModuleKill** (int SocketID, char *TaskName)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– ModuleName	char *	Module name to kill

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **OptionalModuleKill** (ByVal SocketID As Long, ByVal TaskName As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– ModuleName	string	Module name to kill

Output parameters

– None

Return

– Function error code

Matlab

Prototype

[Error] **OptionalModuleKill** (int32 SocketID, cstring TaskName)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– ModuleName	cstring	Module name to kill

Return

– Error	int32	Function error code

Python

Prototype

[Error] **OptionalModuleKill** (integer SocketID, string TaskName)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
–	ModuleName	string	Module name to kill

Return

–	Error	int	Function error code

2.17 Hardware Date and Time Setting

2.17.1 Function Description

2.17.1.1 HardwareDateAndTimeGet

Name

HardwareDateAndTimeGet – Returns the current date and time.

Input tests

- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check output parameter type: ERR_WRONG_TYPE_CHAR (-13)
- Check output format: ERR_WRONG_FORMAT (-7)

Description

This function returns the current date and time of the XPS controller with the format "***WeekDay Month Day Hour:Minute:Second Year*** ", for example "*Tue Jan 15 10:28:06 2008*".

Errors

- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13)
- ERR_WRONG_FORMAT (-7)
- SUCCESS (0): no error

TCL

Prototype

HardwareDateAndTimeGet $SocketID DateAndTime

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– DateAndTime	string	Controller date and time

Return

– Error	int	TCL error code (0 = success or 1 = syntax error) or function error code

C/C++

Prototype

int **HardwareDateAndTimeGet** (int SocketID, char * DateAndTime)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– DateAndTime	char *	Controller date and time

Return

– Error	int	Function error code

Visual Basic

Prototype

Long **HardwareDateAndTimeGet** (ByVal SocketID As Long, ByVal DateAndTime As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– DateAndTime	string	Controller date and time

Return

– Error	long	Function error code

Matlab

Prototype

[Error, DateAndTime] **HardwareDateAndTimeGet** (int32 SocketID)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function

Return

– Error	int32	Function error code
– DateAndTime	cstring	Controller date and time

Python

Prototype

[Error, DateAndTime] **HardwareDateAndTimeGet** (integer SocketID)

Input parameters

SocketID.....integerSocket identifier gets by the "TCP_ConnectToServer" function

Return

– Error	int	Function error code
– DateAndTime	string	Controller date and time

2.17.1.2 HardwareDateAndTimeSet

Name

HardwareDateAndTimeSet – Sets the date and time.

Input tests

- Check command format: ERR_WRONG_FORMAT (-7)
- Verify the number of parameters: ERR_WRONG_PARAMETERS_NUMBER (-9)
- Check intput parameter type: ERR_WRONG_TYPE_CHAR (-13)

Description

This function sets the date and time of the XPS controller. The date format must be "***WeekDay Month Day Hour:Minute:Second Year*** ", for example "*Tue Jan 15 10:28:06 2008*".

Errors

- ERR_WRONG_FORMAT (-7)
- ERR_WRONG_PARAMETERS_NUMBER (-9)
- ERR_WRONG_TYPE_CHAR (-13)
- SUCCESS (0): no error

TCL

Prototype

HardwareDateAndTimeSet $SocketID $DateAndTime

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
–	DateAndTime	string	Date and time to set

Output parameters

– None

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

int **HardwareDateAndTimeSet** (int SocketID, char * DateAndTime)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function
– DateAndTime	char *	Date and time to set

Output parameters

– None

Return

– Function error code

Visual Basic

Prototype

Long **HardwareDateAndTimeSet** (ByVal SocketID As Long, ByVal DateAndTime As String)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function
– DateAndTime	string	Date and time to set

Output parameters

– None

Return

– Function error code

Matlab

Prototype

[Error] **HardwareDateAndTimeSet** (int32 SocketID, cstring DateAndTime)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function
– DateAndTime	cstring	Date and time to set

Return

– Error	int32	Function error code

Python

Prototype

[Error] **HardwareDateAndTimeSet** (integer SocketID, string DateAndTime)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function
–	DateAndTime	string	Date and time to set

Return

–	Error	int	Function error code

2.18 Version

2.18.1 Function Description

2.18.1.1 GetLibraryVersion

Name

GetLibraryVersion – Returns the version of the DLL library.

Input tests

- None

Description

This function returns the version of the DLL library.

The library version represents the firmware version that was used to build the library.

Errors

- None

TCL

Prototype

GetLibraryVersion $SocketID LibVersion

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Output parameters

–	LibVersion	string	DLL library version

Return

– TCL error code (0 = success or 1 = syntax error) or Function error code

C/C++

Prototype

– char * **GetLibraryVersion** (int SocketID)

Input parameters

–	SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– None

Return

–	LibVersion	char *	DLL library version

Visual Basic

Prototype

String **GetLibraryVersion** (ByVal SocketID As Long)

Input parameters

– SocketID	long	Socket identifier gets by the "TCP_ConnectToServer" function

Output parameters

– None

Return

– LibVersion	string	DLL library version

Matlab

Prototype

[LibVersion] **GetLibraryVersion** (int32 SocketID)

Input parameters

– SocketID	int32	Socket identifier gets by the "TCP_ConnectToServer" function

Return

– LibVersion	cstring	DLL library version

Python

Prototype

[LibVersion] **GetLibraryVersion** (integer SocketID)

Input parameters

– SocketID	int	Socket identifier gets by the "TCP_ConnectToServer" Function

Return

– LibVersion	string	DLL library version

2.19 Positioner Error List

code	Error description
0	
0x80000001	General inhibition detected
0x80000002	Fatal following error detected
0x80000004	Home search time out
0x80000008	Motion done time out
0x80000010	Requested position exceed travel limits in trajectory or slave mode
0x80000020	Requested velocity exceed maximum value in trajectory or slave mode
0x80000040	Requested acceleration exceed maximum value in trajectory or slave mode
0x80000100	Minus end of run activated
0x80000200	Plus end of run activated
0x80000400	Minus end of run glitch
0x80000800	Plus end of run glitch
0x80001000	Encoder quadrature error
0x80002000	Encoder frequency and coherancy error
0x80010000	Hard interpolator encoder error
0x80020000	Hard interpolator encoder quadrature error
0x80100000	First driver in fault
0x80200000	Second driver in fault
0x81000000	Home search mechanical zero inconsistency
0x82000000	Interferometer no signal error on axis or reference
0x84000000	Interferometer glitch error on axis or reference
0x88000000	Fatal internal error

NOTE

The most significant bit is always set to 1. So, all positioner errors are negative.

2.20 Positioner Hardware Status List

code	Error description
0x00000001	General inhibition detected
0x00000004	ZM high level
0x00000100	Minus end of run activated
0x00000200	Plus end of run activated
0x00000400	Minus end of run glitch
0x00000800	Plus end of run glitch
0x00001000	Encoder quadrature error
0x00002000	Encoder frequency or coherancy error
0x00010000	Hard interpolator encoder error
0x00020000	Hard interpolator encoder quadrature error
0x00100000	First driver in fault
0x00200000	Second driver in fault
0x00400000	First driver powered on
0x00800000	Second driver powered on
0x01000000	Interferometer no signal error on axis or reference
0x02000000	Interferometer glitch error on axis or reference

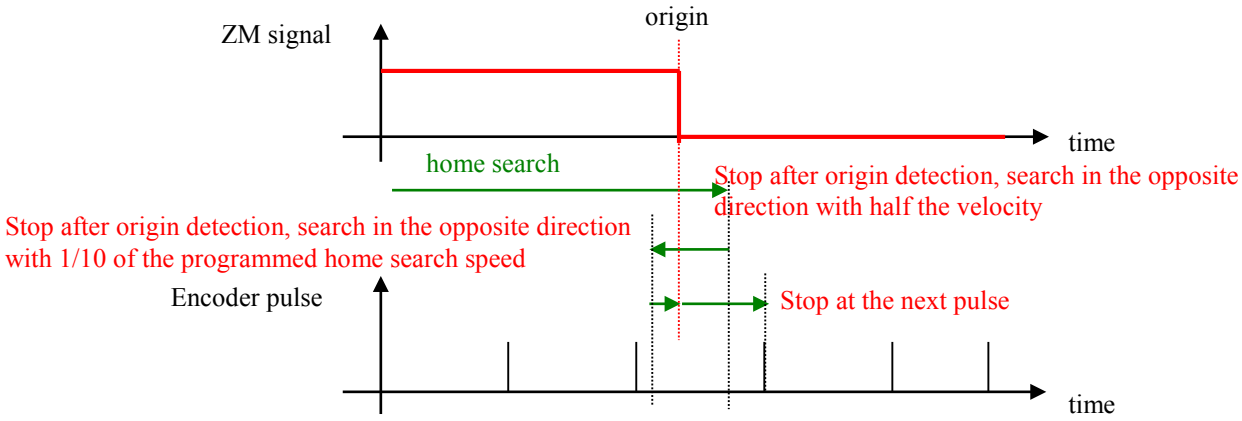
NOTE

Positioner errors are used to trigger consequences on the system, for instance disable, emergency break, etc. Positioner hardware status information is mainly provided for information purposes.

Explanation about positioner hardware status:

General inhibition detected: This refers to the General Inhibition connector at the rear panel or the Stop All button at the front panel of the XPS controller. The General Inhibiton connector is a safety feature and can be used for a custom STOP ALL emergency switch. Inhibition (pin#2), must always be connected to GND during normal operation of the controller. In this case, inhibition is not detected. An open circuit is equivalent to pressing STOP ALL on the front panel, in which case, inhibition is detected.

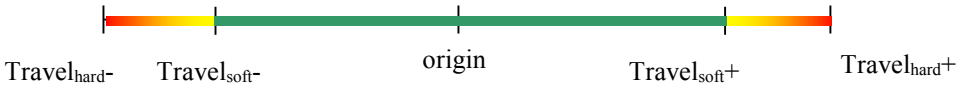
ZM high level: This refers to the mechanical zero signal used with some stages. The ZM signal is high during one part of the travel and low during the other part of the travel. The detection of the ZM high/low transition in combination with an encoder index pulse signal allows a fast and repeatable origin search (MechanicalZeroAndIndexHomeSearch).



Minus end of run activated: Refers to the hardware minus end of run limit switch. During normal operation, this end of run switch should never be activated and any motion will be stopped by the detection of the minus software limit.

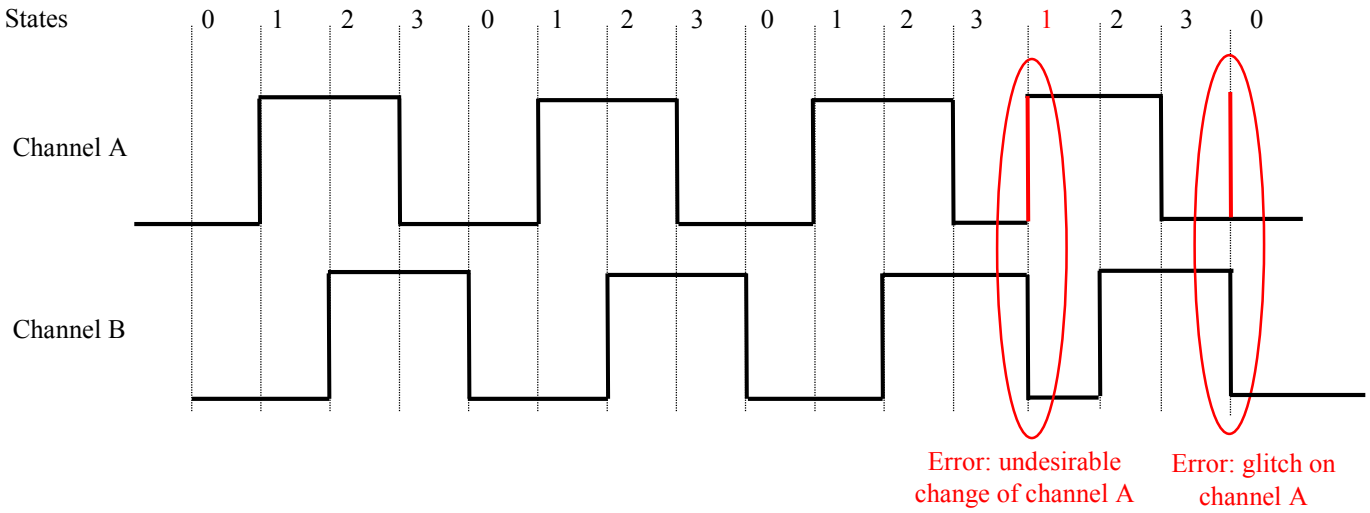
Plus end of run activated: Refers to the hardware positive end of run limit switch. During normal operation, this end of run switch should never be activated and any motion will be stopped by the detection of the positive software limit.

Minus end of run glitch: Undesirable, momentary instability of the hardware minus end of run signal, for instance can be generated by ripple or noise.



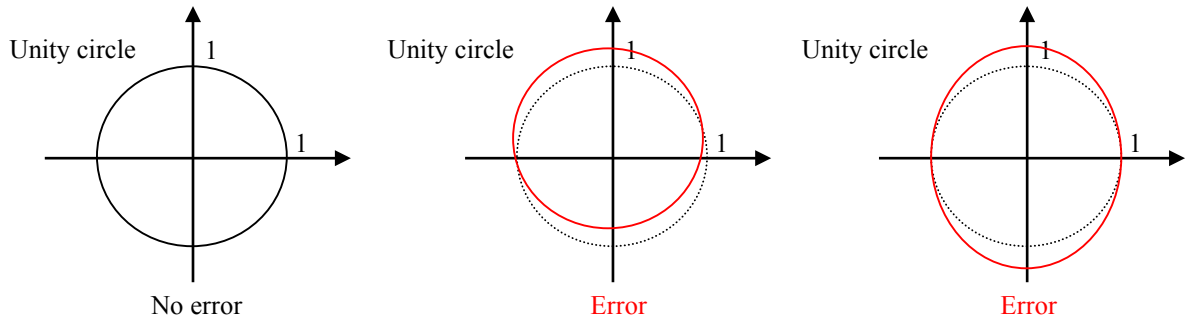
Plus end of run glitch: Undesirable, momentary instability of the hardware positive end of run signal, for instance can be generated by ripple or noise.

Encoder quadrature error: Error generated when the signals of both encoder channels simultaneously change. In normal operation, only one quadrature signal changes state at a time. This error can occur due to an undesirable level change or a glitch as illustrated below.



Encoder freq. and coherency error: Error generated when the frequency of the signals is too high. The maximum frequency of the encoder input is 25MHz.

Hard interpolator encoder error: Error generated when the difference of the sine/cosine encoder signals from a unity circle is too large (for instance when signals are phase shifted or amplitude modified).



Hard interpolator quad. encoder error: Error generated when the signals of both encoder channels of the hardware interpolated encoder output simultaneously change. Same error as *Encoder quadrature error* except that the quadrature signals are those converted from the sine/cosine signals of the hard interpolator. The hardware interpolator is used only with AnalogInterpolated encoders to trigger the position compare output and to gather positions during external data gathering.

First driver in fault: problem with the first driver.

Second driver in fault: problem with the second driver in case two drivers are connected to one axis.

First driver powered on: First driver with motor ON after initialization.

Second driver powered on: Second driver with motor ON after initialization, in case two drivers are connected to one axis.

2.21 Positioner Driver Status List

Bit	code	DRV00x	DRV01	DRV02x	D6U	DRV03	DRVP1	DRVM
0	a			Short-circuit	Short-circuit	Short-circuit		
1	b			Broken fuse	Broken fuse	Broken fuse	Voltage out of range	
2	c			Thermistor fault	Thermistor fault		Over temperature	
3	d			Initialization error	Initialization error	Initialization error	Initialization error	
4	e			I²T	I²T	I²T	Dynamic error	
5	f			Current limit	Current limit			
6	g					TG is opened	No stage connected	
7	h	Inhibition input	Inhibition input	Inhibition input	Inhibition input	Inhibition input	Inhibition input	Inhibition input
8	i	Driver in fault	Driver in fault	Driver in fault	Driver in fault	Driver in fault	Driver in fault	Driver in fault

2.22 Group Status List

Code	Description
0	NOTINIT state
1	NOTINIT state due to an emergency brake: see positioner status
2	NOTINIT state due to an emergency stop: see positioner status
3	NOTINIT state due to a following error during homing
4	NOTINIT state due to a following error
5	NOTINIT state due to an homing timeout
6	NOTINIT state due to a motion done timeout during homing
7	NOTINIT state due to a KillAll command
8	NOTINIT state due to an end of run after homing
9	NOTINIT state due to an encoder calibration error
10	Ready state due to an AbortMove command
11	Ready state from homing
12	Ready state from motion
13	Ready State due to a MotionEnable command
14	Ready state from slave
15	Ready state from jogging
16	Ready state from analog tracking
17	Ready state from trajectory
18	Ready state from spinning
19	Ready state due to a group interlock error during motion
20	Disable state
21	Disabled state due to a following error on ready state
22	Disabled state due to a following error during motion

23	Disabled state due to a motion done timeout during moving
24	Disabled state due to a following error on slave state
25	Disabled state due to a following error on jogging state
26	Disabled state due to a following error during trajectory
27	Disabled state due to a motion done timeout during trajectory
28	Disabled state due to a following error during analog tracking
29	Disabled state due to a slave error during motion
30	Disabled state due to a slave error on slave state
31	Disabled state due to a slave error on jogging state
32	Disabled state due to a slave error during trajectory
33	Disabled state due to a slave error during analog tracking
34	Disabled state due to a slave error on ready state
35	Disabled state due to a following error on spinning state
36	Disabled state due to a slave error on spinning state
37	Disabled state due to a following error on auto-tuning
38	Disabled state due to a slave error on auto-tuning
39	Disable state due to an emergency stop on auto-tuning state
40	Emergency braking
41	Motor initialization state
42	Not referenced state
43	Homing state
44	Moving state
45	Trajectory state
46	Slave state due to a SlaveEnable command
47	Jogging state due to a JogEnable command
48	Analog tracking state due to a TrackingEnable command
49	Analog interpolated encoder calibrating state
50	NOTINIT state due to a mechanical zero inconsistency during homing
51	Spinning state due to a SpinParametersSet command
52	NOTINIT state due to a clamping timeout
55	Clamped
56	Ready state from clamped
58	Disabled state due to a following error during clamped
59	Disabled state due to a motion done timeout during clamped
60	NOTINIT state due to a group interlock error on not reference state
61	NOTINIT state due to a group interlock error during homing
63	NOTINIT state due to a motor initialization error
64	Referencing state
65	Clamping initialization
66	NOTINIT state due to a perpendicularity error homing
67	NOTINIT state due to a master/slave error during homing
68	Auto-tuning state
69	Scaling calibration state

70	Ready state from auto-tuning
71	NOTINIT state from scaling calibration
72	NOTINIT state due to a scaling calibration error
73	Excitation signal generation state
74	Disable state due to a following error on excitation signal generation state
75	Disable state due to a master/slave error on excitation signal generation state
76	Disable state due to an emergency stop on excitation signal generation state
77	Ready state from excitation signal generation
78	Focus state
79	Ready state from focus
80	Disable state due to a following error on focus state
81	Disable state due to a master/slave error on focus state
82	Disable state due to an emergency stop on focus state
83	NOTINIT state due to a group interlock error
84	Disable state due to a group interlock error during moving
85	Disable state due to a group interlock error during jogging
86	Disable state due to a group interlock error on slave state
87	Disable state due to a group interlock error during trajectory
88	Disable state due to a group interlock error during analog tracking
89	Disable state due to a group interlock error during spinning
90	Disable state due to a group interlock error on ready state
91	Disable state due to a group interlock error on auto-tuning state
92	Disable state due to a group interlock error on excitation signal generation state
93	Disable state due to a group interlock error on focus state
94	Disabled state due to a motion done timeout during jogging
95	Disabled state due to a motion done timeout during spinning
96	Disabled state due to a motion done timeout during slave mode

2.23 Error List

Error mnemonic	code	Error description
ERR_TCL_INTERPRETOR_ERROR	1	TCL interpretor error: wrong syntax
SUCCESS	0	Successful command
ERR_BUSY_SOCKET	-1	Busy socket: previous command not yet finished
ERR_TCP_TIMEOUT	-2	TCP timeout
ERR_STRING_TOO_LONG	-3	String command too long
ERR_UNKNOWN_COMMAND	-4	Unknown command
ERR_POSITIONER_ERROR	-5	Not allowed due to a positioner error
ERR_WRONG_FORMAT	-7	Wrong format in the command string
ERR_WRONG_OBJECT_TYPE	-8	Wrong object type for this command
ERR_WRONG_PARAMETERS_NUMBER	-9	Wrong number of parameters in the command
ERR_WRONG_TYPE	-10	Wrong parameter type in the command string
ERR_WRONG_TYPE_BIT_WORD	-11	Wrong parameters type in the command string: word or word * expected
ERR_WRONG_TYPE_BOOL	-12	Wrong parameter type in the command string: bool or bool * expected
ERR_WRONG_TYPE_CHAR	-13	Wrong parameter type in the command string: char * expected
ERR_WRONG_TYPE_DOUBLE	-14	Wrong parameter type in the command string: double or double * expected
ERR_WRONG_TYPE_INT	-15	Wrong parameter type in the command string : int or int * expected
ERR_WRONG_TYPE_UNSIGNEDINT	-16	Wrong parameter type in the command string : unsigned int or unsigned int * expected
ERR_PARAMETER_OUT_OF_RANGE	-17	Parameter out of range
ERR_POSITIONER_NAME	-18	Positioner Name doesn't exist
ERR_GROUP_NAME	-19	GroupName doesn't exist or unknown command
ERR_FATAL_INIT	-20	Fatal Error during initialization, read the error.log file for more details
ERR_IN_INITIALIZATION	-21	Controller in initialization
ERR_NOT_ALLOWED_ACTION	-22	Not allowed action
ERR_POSITION_COMPARE_NOT_SET	-23	Position compare not set
ERR_UNCOMPATIBLE	-24	Not available in this configuration
ERR_FOLLOWING_ERROR	-25	Following Error
ERR_EMERGENCY_SIGNAL	-26	Emergency signal
ERR_GROUP_ABORT_MOTION	-27	Move Aborted
ERR_GROUP_HOME_SEARCH_TIMEOUT	-28	Home search timeout
ERR_MNEMOTYPEGATHERING	-29	Mnemonique gathering type doesn't exist
ERR_GATHERING_NOT_STARTED	-30	Gathering not started
ERR_HOME_OUT_RANGE	-31	Home position is out of user travel limits
ERR_GATHERING_NOT_CONFIGURED	-32	Gathering not configurated
ERR_GROUP_MOTION_DONE_TIMEOUT	-33	Motion done timeout
ERR_TRAVEL_LIMITS	-35	Not allowed: home preset outside travel limits
ERR_UNKNOWN_TCL_FILE	-36	Unknown TCL file
ERR_TCL_SCRIPT_KILL	-37	TCL interpretor doesn't run
ERR_TCL_INTERPRETOR	-38	TCL script can't be killed
ERR_MNEMO_ACTION	-39	Mnemonique action doesn't exist
ERR_MNEMO_EVENT	-40	Mnemonique event doesn't exist
ERR_SLAVE_CONFIGURATION	-41	Slave-Master mode not configurated
ERR_JOG_OUT_OF_RANGE	-42	Jog value out of range
ERR_GATHERING_RUNNING	-43	Gathering running
ERR_SLAVE	-44	Slave error disabling master
ERR_END_OF_RUN	-45	End of run activated
ERR_NOT_ALLOWED_BACKLASH	-46	Not allowed action due to backlash
ERR_WRONG_TCL_TASKNAME	-47	Wrong TCL task name: each TCL task name must be different

ERR_BASE_VELOCITY	-48	BaseVelocity must be null
ERR_GROUP_HOME_SEARCH_ZM_ERROR	-49	Inconsistent mechanical zero during home search
ERR_MOTOR_INITIALIZATION_ERROR	-50	Motor initialization error: check InitializationAcceleration
ERR_SPIN_OUT_OF_RANGE	-51	Spin value out of range
ERR_GROUP_INTERLOCK_ERROR	-52	Group interlock
ERR_NOT_ALLOWED_ACTION_GROUP_INTERLOCK	-53	Not allowed action due to a group interlock
ERR_WRITE_FILE	-60	Error during file writing or file doesn't exist
ERR_READ_FILE	-61	Error during file reading or file doesn't exist
ERR_TRAJ_ELEM_TYPE	-62	Wrong trajectory element type
ERR_TRAJ_ELEM_RADIUS	-63	Wrong XY trajectory element arc radius
ERR_TRAJ_ELEM_SWEEP	-64	Wrong XY trajectory element sweep angle
ERR_TRAJ_ELEM_LINE	-65	Trajectory line element discontinuity error or new element is too small
ERR_TRAJ_EMPTY	-66	Trajectory doesn't content any element or not loaded
ERR_TRAJ_VEL_LIMIT	-68	Velocity on trajectory is too high
ERR_TRAJ_ACC_LIMIT	-69	Acceleration on trajectory is too high
ERR_TRAJ_FINAL_VELOCITY	-70	Final velocity on trajectory is not zero
ERR_MSG_QUEUE	-71	Error write or read from message queue
ERR_TRAJ_INITIALIZATION	-72	Error during trajectory initialization
ERR_END_OF_FILE	-73	End of file
ERR_READ_FILE_PARAMETER_KEY	-74	Error file parameter key not found
ERR_TRAJ_TIME	-75	Time delta of trajectory element is negative or null
ERR_EVENTS_NOT_CONFIGURED	-80	Event not configured
ERR_ACTIONS_NOT_CONFIGURED	-81	Action not configured
ERR_EVENT_BUFFER_FULL	-82	Event buffer is full
ERR_EVENT_ID_UNDEFINED	-83	Event ID not defined
ERR_HOME_SEARCH_GANTRY_TOLERANCE_ERROR	-85	Secondary positioner index is too far from first positioner
ERR_FOCUS_RESERVED_SOCKET	-90	Focus socket not reseved or closed
ERR_FOCUS_BUSY_EVENT_SCHEDULER	-91	Focus event scheduler is busy
ERR_OPTIONAL_EXTERNAL_MODULE_EXECUTE	-95	Error of executing an optional module
ERR_OPTIONAL_EXTERNAL_MODULE_KILL	-96	Error of stopping an optional module
ERR_OPTIONAL_EXTERNAL_MODULE_UNLOAD	-98	Error of unloading an optional module
ERR_FATAL_EXTERNAL_MODULE_LOAD	-99	Fatal external module load: see error.log
ERR_INTERNAL_ERROR	-100	Internal error (memory allocation error, …)
ERR_RELAY_FEEDBACK_TEST_NO_OSCILLATION	-101	Relay Feedback Test failed: No oscillation
ERR_RELAY_FEEDBACK_TEST_SIGNAL_NOISY	-102	Relay Feedback Test failed: Signal too noisy
ERR_SIGNAL_POINTS_NOT_ENOUGH	-103	Relay Feedback Test failed: Signal data not enough for analyse
ERR_PID_TUNING_INITIALIZATION	-104	Error of tuning process initialization
ERR_SCALING_CALIBRATION	-105	Error of scaling calibration initialization
ERR_WRONG_USERNAME_OR_PASSWORD	-106	Wrong user name or password
ERR_NEED_ADMINISTRATOR_RIGHTS	-107	This function requires to be logged in with Administrator rights
ERR_SOCKET_CLOSED_BY_ADMIN	-108	The TCP/IP connection was closed by an administrator
ERR_NEED_TO_BE_HOMED_AT_LEAST_ONCE	-109	Group need to be homed at least once to use this function (distance mesured during home search)
ERR_NOT_ALLOWED_FOR_GANTRY	-110	Execution not allowed for Gantry configuration
ERR_GATHERING_BUFFER_FULL	-111	Gathering buffer is full
ERR_EXCITATION_SIGNAL_INITIALIZATION	-112	Error of excitation signal generation initialization
ERR_BOTH_ENDS_OF_RUN_ACTIVATED	-113	Both ends of run activated
ERR_GROUP_CLAMPING_TIMEOUT	-114	Clamping timeout
ERR_HARDWARE_FUNCTION_NOT_SUPPORTED	-115	Function is not supported by current hardware
ERR_EXTERNAL_DRIVER_INIT	-116	Error during external driver initialization, read error.log file for more details
ERR_FUNCTION_ONLY_ALLOWED_IN_DISABLED_STA	-117	Function is only allowed in DISABLED group

TE		state
ERR_NOT_ALLOWED_DRIVER_NOT_INITIALIZED	-118	Not allowed action driver not initialized
ERR_TRAVEL_LIMITS_ON_SECONDARY_POSITIONER	-119	Position is outside of travel limits on secondary positioner
ERR_WARNING_FOLLOWING_ERROR	-120	Warning following error during move with position compare enabled
ERR_NOT_ALLOWED_MODE_DISABLED	-121	Function is not allowed due to configuration disabled
ERR_CHECK_DATA_INCORRECT	-122	Data incorrect (wrong value, wrong format, wrong order or inexistent)
ERR_ADMINISTRATOR_LOGGED_IN	-123	Action not allowed, an Administrator is already logged in
ERR_MOVE_SECONDARY_POSITIONER	-124	Error during move of secondary positioner: check positioners errors for details
ERR_TCL_INTERPRETOR_SYNCRONIZATION	-125	Check tcl task name is not empty
ERR_WRONG_TYPE_SHORT	-126	Wrong parameter type in the command string : short or short * expected
ERR_WRONG_TYPE_LONG	-127	Wrong parameter type in the command string : long or long * expected
ERR_WRONG_TYPE_UNSIGNEDSHORT	-128	Wrong parameter type in the command string : unsigned short or unsigned short * expected
ERR_WRONG_TYPE_UNSIGNEDLONG	-129	Wrong parameter type in the command string : unsigned long or unsigned long * expected
ERR_WRONG_TYPE_FLOAT	-130	Wrong parameter type in the command string : float or float * expected
ERR_WRONG_TYPE_LONGLONG	-131	Wrong parameter type in the command string : long long int or long long int * expected
ERR_WRONG_TYPE_UNSIGNEDLONGLONG	-132	Wrong parameter type in the command string : unsigned long long or unsigned long long * expected

2.24 Controller Status List

Controller status code	code	Controller status description
CONTROLLER_STATUS_OK	0x00000000	Controller status OK
CONTROLLER_STATUS_INITIALIZATION_FAILED	0x00000001	Controller initialization failed
CONTROLLER_STATUS_NB_OPENED_SOCKETS_REACHED_MAXIMUM_ALLOWED	0x00000002	Number of currently opened sockets reached maximum allowed number
CONTROLLER_STATUS_CPU_OVERLOAD	0x00000004	Controller CPU is overloaded
CONTROLLER_STATUS_CORRECTOR_OVER_CALCULATED	0x00000008	Current measured corrector calculation time exceeds the corrector period
CONTROLLER_STATUS_PROFILER_OVER_CALCULATED	0x00000010	Profile generator calculating time exceeds ProfileGeneratorISRRatio * IRSCorrectorPeriod
CONTROLLER_STATUS_CORRECTOR_INTERRUPT_LOST	0x00000020	Controller has lost a corrector interrupt

NOTE

Within about 5 minutes after the controller startup, due to the hardware thermal stabilization, the CONTROLLER_STATUS_CORRECTOR_OVER_CALCULATED,

CONTROLLER_STATUS_CORRECTOR_INTERRUPT_LOST ,

CONTROLLER_STATUS_PROFILER_OVER_CALCULATED, CONTROLLER_STATUS_CPU_OVERLOAD or CONTROLLER_STATUS_NB_OPENED_SOCKETS_REACHED_MAXIMUM_ALLOWED status flags may be raised.

These flags are automatically reset after a controller status reading using the *ControllerStatusGet()* command.

Another way to avoid these flags during the 5 first minutes after boot is to set the following parameter in system.ref to 300 (seconds):

DelayBeforeStartup = 300 ; Controller boots completely after 300 seconds

2.25 User Defined Soft Motor Output DAC Offsets

The firmware supports user-defined soft motor output DAC offsets.

- PrimaryDAC1Offset
- PrimaryDAC2Offset
- SecondaryDAC1Offset
- SecondaryDAC2Offset

These parameters add an offset directly to the motor interface DAC outputs (inputs of motor driver board).

Their default values are zero (0) at controller boot, but they can be read or modified using *PositionerMotorOutputOffsetGet()/...Set()* functions. These functions are not documented, because they are used for internal tests only.

This feature is allowed only for the XPS-Qn Precision Platform controller.

2.26 Gantry Configuration for MultipleAxes Group

The "Precision Platform" firmware supports the gantry configuration for MultipleAxes group.

- Additional Gantry feature for MultipleAxes group with different possibilities, for example a 5-positioner MultipleAxes group (M1, M2, M3, M4, M5):
  - ⇨ One gantry positioner in any order:
    - M1_1(primary), M1_2(secondary), M2, M3, M4, M5
    - M1, M2_1(primary), M2_2(secondary), M3, M4, M5
    - M1, M2, M3, M4, M5_1(primary), M5_2(secondary)
    - …..
  - ⇨ Two gantry positioners in any order:
    - M1_1(primary), M1_2(secondary), M2_1(primary), M2_2(secondary), M3, M4, M5
    - M1, M2_1(primary), M2_2(secondary), M3, M4_1(primary),M4_2(secondary), M5
    - M1_1(primary), M1_2(secondary), M2, M3, M4, M5_1(primary), M5_2(secondary)
    - …
- Additional user command errors:
  - ⇨ ERR_FUNCTION_ONLY_ALLOWED_IN_DISABLED_STATE (-117)
  - ⇨ ERR_NOT_ALLOWED_DRIVER_NOT_INITIALIZED (-118)
  - ⇨ ERR_TRAVEL_LIMITS_ON_SECONDARY_POSITIONER (-119)

Configuration files

- System.ini: Additional type of motor initialization and home search order for MultipleAxes group:
  - ⇨ InitializationAndHomeSearchSequence = OneAfterAnotherInReverseOrder.

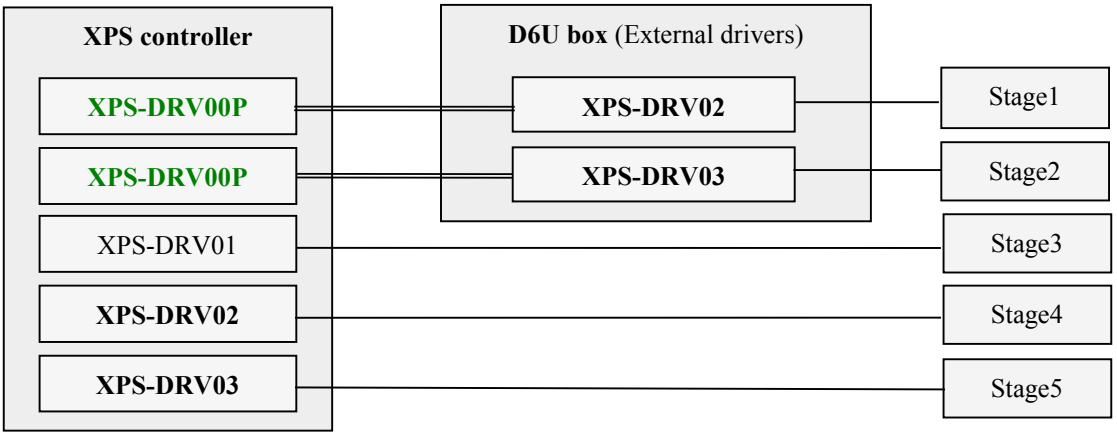
2.27 Driver Notes for Precision Platform Firmware

2.27.1 D6U Interface

The "Precision Platform" firmware is equipped with a D6U interface. This interface allows use of external drivers (DRV02 or DRV03).

How to use an external driver

An external driver must be connected to the XPS controller with an XPS-DRV00P board. An external driver interface can be only defined as an XPS-DRV02 or an XPS-DRV03 board.



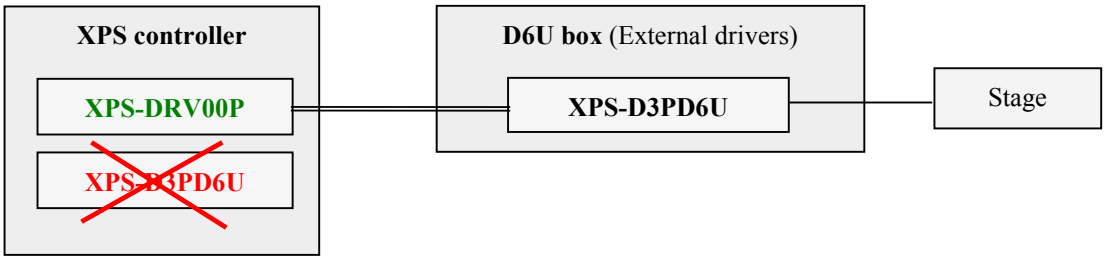
NOTE

In the stages.ini file, the "DriverName" in the stage configuration must be declared XPS-DRV02 or XPS-DRV03 and not XPS-DRV00P.

```
[Stage1]
DriverName = XPS-DRV02
[Stage2]
DriverName = XPS-DRV03
```

2.27.2 D3PD6U Driver / D3PD6U-15 Drivers

The "Precision Platform" firmware version integrates the D3PD6U (from V1.0.0) and D3PD6U-15 (from V1.2.7) drivers. These drivers are **external drivers**. So these external drivers must be connected to the XPS controller with an XPS-DRV00P board.



NOTE

In the stages.ini file, the "DriverName" of your stage configuration must be declared XPS-D3PD6U or XPS-D3PD6U-15 (not XPS-DRV00P).

```
[Stage]
DriverName = XPS-D3PD6U          ; or XPS-D3PD6U-15
```

2.27.3 NON_CONFIGURABLE_STAGE Driver

The "DriverNonConfigurable" driver is simulated by a loopback connector.

The declaration name is NON_CONFIGURABLE_DRV.

In the ***stages.ini*** file:

```
[MY_STAGE]
…
DriverName = NON_CONFIGURABLE_DRV
…
```

2.27.4 Driver Initialization Not at Boot

The controller checks and sets its external drivers not at boot, but at the beginning of the **GroupInitialize()** call.

2.27.5 DRVP1 Piezo Driver

The "Precision Platform" firmware supports the DRVP1 piezo driver.

Stages.ini:

; ---------- Piezo configuration ----------

; Note: Piezo driver works only with NoEncoderPosition or PIPosition CorrectorType

; Piezo motor driver interface

MotorDriverInterface = AnalogPositionPiezo

; Limit sensors input plug

ServitudesType = Piezo

; Piezo driver model

DriverName = XPS-DRVP1

; Piezo driver parameters

DriverNotchFrequency =

DriverNotchBandwidth =

DriverNotchGain =

DriverLowpassFrequency =

DriverKI =

DriverFatalFollowingError =

DriverStagePositionOffset =

DriverTravelCorrection =

Added functions:

- PositionerDriverFiltersGet()/…Set(): Set/get piezo driver filters parameters
- PositionerDriverPositionOffsetsGet(): Get piezo driver offsets

2.28 Function List Classed in Categories

General function

1. CloseAllOtherSockets
2. ControllerMotionKernelTimeLoadGet
3. ControllerRTTimeGet
4. ControllerSlaveStatusGet
5. ControllerSlaveStatusStringGet
6. ControllerStatusGet
7. ControllerStatusStringGet
8. ControllerSynchronizeCorrectorISR
9. DoubleGlobalArrayGet
10. DoubleGlobalArraySet
11. ElapsedTimeGet
12. ErrorStringGet
13. EventAdd
14. EventGet
15. EventRemove
16. EventWait
17. EventExtendedConfigurationTriggerGet
18. EventExtendedConfigurationTriggerSet
19. EventExtendedConfigurationActionGet
20. EventExtendedConfigurationActionSet
21. EventExtendedStart
22. EventExtendedAllGet
23. EventExtendedGet
24. EventExtendedRemove
25. EventExtendedWait
26. FirmwareVersionGet
27. GatheringConfigurationGet
28. GatheringConfigurationSet
29. GatheringCurrentNumberGet
30. GatheringDataAcquire
31. GatheringDataGet
32. GatheringDataMultipleLinesGet
33. GatheringExternalConfigurationGet
34. GatheringExternalConfigurationSet
35. GatheringExternalCurrentNumberGet
36. GatheringExternalDataGet
37. GatheringExternalStopAndSave
38. GatheringReset
39. GatheringRun
40. GatheringRunAppend
41. GatheringStop

42. GatheringStopAndSave
43. GlobalArrayGet
44. GlobalArraySet
45. GPIOAnalogGet
46. GPIOAnalogSet
47. GPIOAnalogGainGet
48. GPIOAnalogGainSet
49. GPIODigitalGet
50. GPIODigitalSet
51. HardwareDateAndTimeGet
52. HardwareDateAndTimeSet
53. KillAll
54. Login
55. OtionalModuleExecute
56. OtionalModuleKill
57. Reboot
58. TCLScriptExecute
59. TCLScriptExecuteAndWait
60. TCLScriptExecuteWithPriority
61. TCLScriptKill
62. TimerGet
63. TimerSet

Positioner functions

64. PositionerAccelerationAutoScaling
65. PositionerAnalogTrackingPositionParametersGet
66. PositionerAnalogTrackingPositionParametersSet
67. PositionerAnalogTrackingVelocityParametersGet
68. PositionerAnalogTrackingVelocityParametersSet
69. PositionerBacklashGet
70. PositionerBacklashSet
71. PositionerBacklashEnable
72. PositionerBacklashDisable
73. PositionerCompensatedPCOAbort
74. PositionerCompensatedPCOCurrentStatusGet
75. PositionerCompensatedPCOEnable
76. PositionerCompensatedPCOFromFile
77. PositionerCompensatedPCOLoadToMemory
78. PositionerCompensatedPCOMemoryReset
79. PositionerCompensatedPCOPrepare
80. PositionerCompensatedPCOSet
81. PositionerCompensationFrequencyNotchsGet
82. PositionerCompensationFrequencyNotchsSet
83. PositionerCompensationLowPassTwoFiltersGet
84. PositionerCompensationLowPassTwoFiltersSet

85. PositionerCompensationNotchModeFiltersGet
86. PositionerCompensationNotchModeFiltersSet
87. PositionerCompensationPhaseCorrectionFiltersGet
88. PositionerCompensationPhaseCorrectionFiltersSet
89. PositionerCompensationSpatialPeriodicNotchsGet
90. PositionerCompensationSpatialPeriodicNotchsSet
91. PositionerCorrectorAutoTuning
92. PositionerCorrectorNotchFiltersSet
93. PositionerCorrectorNotchFiltersGet
94. PositionerCorrectorPIDFFAccelerationGet
95. PositionerCorrectorPIDFFAccelerationSet
96. PositionerCorrectorSR1AccelerationGet
97. PositionerCorrectorSR1AccelerationSet
98. PositionerCorrectorSR1ObserverAccelerationGet
99. PositionerCorrectorSR1ObserverAccelerationSet
100. PositionerCorrectorSR1OffsetAccelerationGet
101. PositionerCorrectorSR1OffsetAccelerationSet
102. PositionerCorrectorPIDFFVelocityGet
103. PositionerCorrectorPIDFFVelocitySet
104. PositionerCorrectorPIDDualFFVoltageGet
105. PositionerCorrectorPIDDualFFVoltageSet
106. PositionerCorrectorPIPositionGet
107. PositionerCorrectorPIPositionSet
108. PositionerCorrectorTypeGet
109. PositionerCurrentVelocityAccelerationFiltersGet
110. PositionerCurrentVelocityAccelerationFiltersSet
111. PositionerDriverFiltersGet
112. PositionerDriverFiltersSet
113. PositionerDriverPositionOffsetsGet
114. PositionerDriverStatusGet
115. PositionerDriverStatusStringGet
116. PositionerEncoderAmplitudeValuesGet
117. PositionerEncoderCalibrationParametersGet
118. PositionerErrorGet
119. PositionerErrorRead
120. PositionerErrorStringGet
121. PositionerExcitationSignalGet
122. PositionerExcitationSignalSet
123. PositionerHardwareStatusGet
124. PositionerHardwareStatusStringGet
125. PositionerHardInterpolatorFactorGet
126. PositionerHardInterpolatorFactorSet
127. PositionerHardInterpolatorPositionGet
128. PositionerMaximumVelocityAndAccelerationGet

129. PositionerMotionDoneGet
130. PositionerMotionDoneSet
131. PositionerPositionCompareDisable
132. PositionerPositionCompareEnable
133. PositionerPositionCompareGet
134. PositionerPositionCompareSet
135. PositionerPositionComparePulseParametersGet
136. PositionerPositionComparePulseParametersSet
137. PositionerPositionCompareScanAccelerationLimitGet
138. PositionerPositionCompareScanAccelerationLimitSet
139. PositionerPositionCompareAquadBAlwaysEnable
140. PositionerPositionCompareAquadBWindowedGet
141. PositionerPositionCompareAquadBWindowedSet
142. PositionerPreCorrectorExcitationSignalGet
143. PositionerPreCorrectorExcitationSignalSet
144. PositionerRawEncoderPositionGet
145. PositionersEncoderIndexDifferenceGet
146. PositionerSGammaExactVelocityAjustedDisplacementGet
147. PositionerSGammaParametersGet
148. PositionerSGammaParametersSet
149. PositionerSGammaPreviousMotionTimesGet
150. PositionerStageParameterGet
151. PositionerStageParameterSet
152. PositionerTimeFlasherDisable
153. PositionerTimeFlasherEnable
154. PositionerTimeFlasherGet
155. PositionerTimeFlasherSet
156. PositionerUserTravelLimitsGet
157. PositionerUserTravelLimitsSet
158. PositionerWarningFollowingErrorGet
159. PositionerWarningFollowingErrorSet

Group functions

	SingleAxis	SingleAxisTheta	SingleAxisWithClamping	Spindle	XY	XYZ	MultipleAxes	TZ	POSITIONER
1. GroupAccelerationSetpointGet									×
2. GroupAnalogTrackingModeEnable	×	×	×					×	
3. GroupAnalogTrackingModeDisable									
4. GroupCorrectorOutputGet									×
5. GroupCurrentFollowingErrorGet									
6. GroupHomeSearch									
7. GroupHomeSearchAndRelativeMove									
8. GroupInitialize									
9. GroupInitializeNoEncoderReset									
10. GroupInitializeWithEncoderCalibration									
11. GroupInterlockDisable									
12. GroupInterlockEnable									
13. GroupJogParametersSet									
14. GroupJogParametersGet									
15. GroupJogCurrentGet									
16. GroupJogModeEnable									
17. GroupJogModeDisable									
18. GroupKill									
19. GroupMoveAbort									
20. GroupMoveAbortFast									
21. GroupMoveAbsolute									
22. GroupMoveRelative									
23. GroupMotionDisable									
24. GroupMotionEnable									
25. GroupPositionCorrectedProfilerGet									
26. GroupPositionCurrentGet									
27. GroupPositionSetpointGet									
28. GroupPositionTargetGet									
29. GroupPositionPCORawEncoderGet									
30. GroupReferencingActionExecute									
31. GroupReferencingStart									
32. GroupReferencingStop									
33. GroupStatusGet									
34. GroupStatusStringGet									
35. GroupVelocityCurrentGet									×

SingleAxes group functions

1. SingleAxisSlaveModeDisable
2. SingleAxisSlaveModeEnable
3. SingleAxisSlaveParametersGet
4. SingleAxisSlaveParametersSet

SingleAxesWithClamping group functions

SingleAxes Theta group functions

1. SingleAxisThetaClampEnable
2. SingleAxisThetaClampDisable
3. SingleAxisThetaPositionRawGet (extended mode)

Spindle group functions

1. GroupSpinCurrentGet
2. GroupSpinModeStop
3. GroupSpinParametersGet
4. GroupSpinParametersSet
5. SpindleSlaveModeDisable
6. SpindleSlaveModeEnable
7. SpindleSlaveParametersGet
8. SpindleSlaveParametersSet

XY group functions

1. XYLineArcExecution
2. XYLineArcParametersGet
3. XYLineArcPulseOutputGet
4. XYLineArcPulseOutputSet
5. XYLineArcVerification
6. XYLineArcVerificationResultGet

XYZ group functions

1. XYZSplineExecution
2. XYZSplineParametersGet
3. XYZSplineVerification
4. XYZSplineVerificationResultGet

MultipleAxes group functions

- MultipleAxesPVTExecution
- MultipleAxesPVTParametersGet
- MultipleAxesPVTPulseOutputGet
- MultipleAxesPVTPulseOutputSet
- MultipleAxesPVTVerification
- MultipleAxesPVTVerificationResultGet

TZ group functions

- TZPVTExecution
- TZPVTParametersGet
- TZPVTPulseOutputGet
- TZPVTPulseOutputSet
- TZPVTVerification
- TZPVTVerificationResultGet
- TZFocusModeDisable
- TZFocusModeEnable
- TZTrackingUserMaximumZZZTargetDifferenceGet
- TZTrackingUserMaximumZZZTargetDifferenceSet

3.0 Process Examples

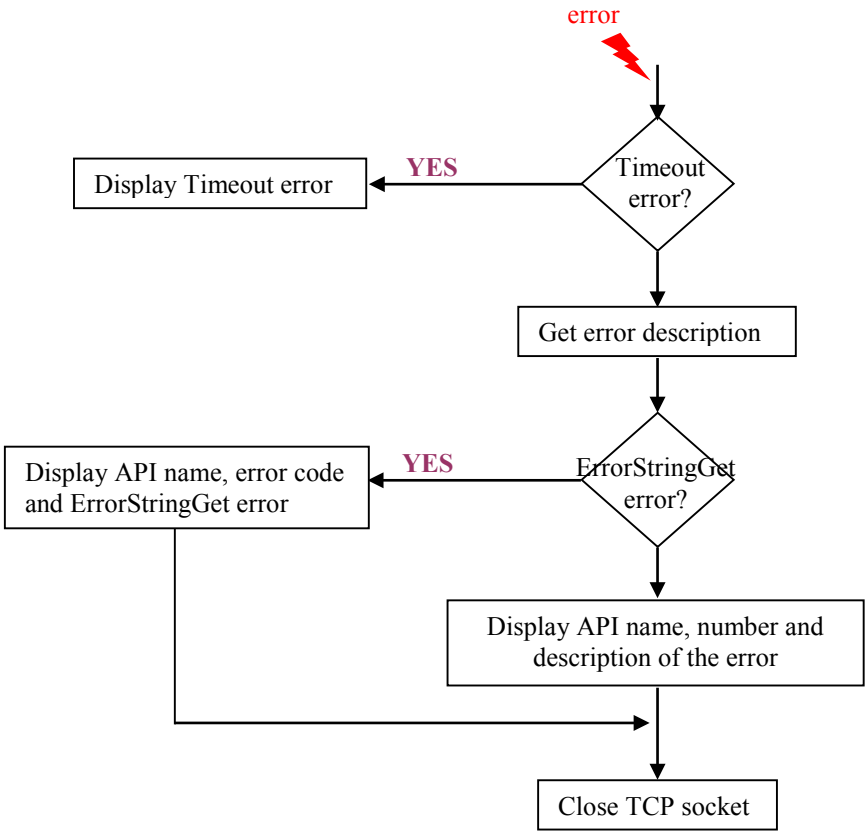
This section provides examples of programming sequences. The next 7 diagrams show the order of use of the different Functions. To see programming code examples, please refer to:

1. The TCL Manual for TCL scripts (part 7. Examples of TCL programs with XPS).
2. The Software Drivers Manual for C++ sequences (part 1.3 Example of C++ programs using the XPS DLL).

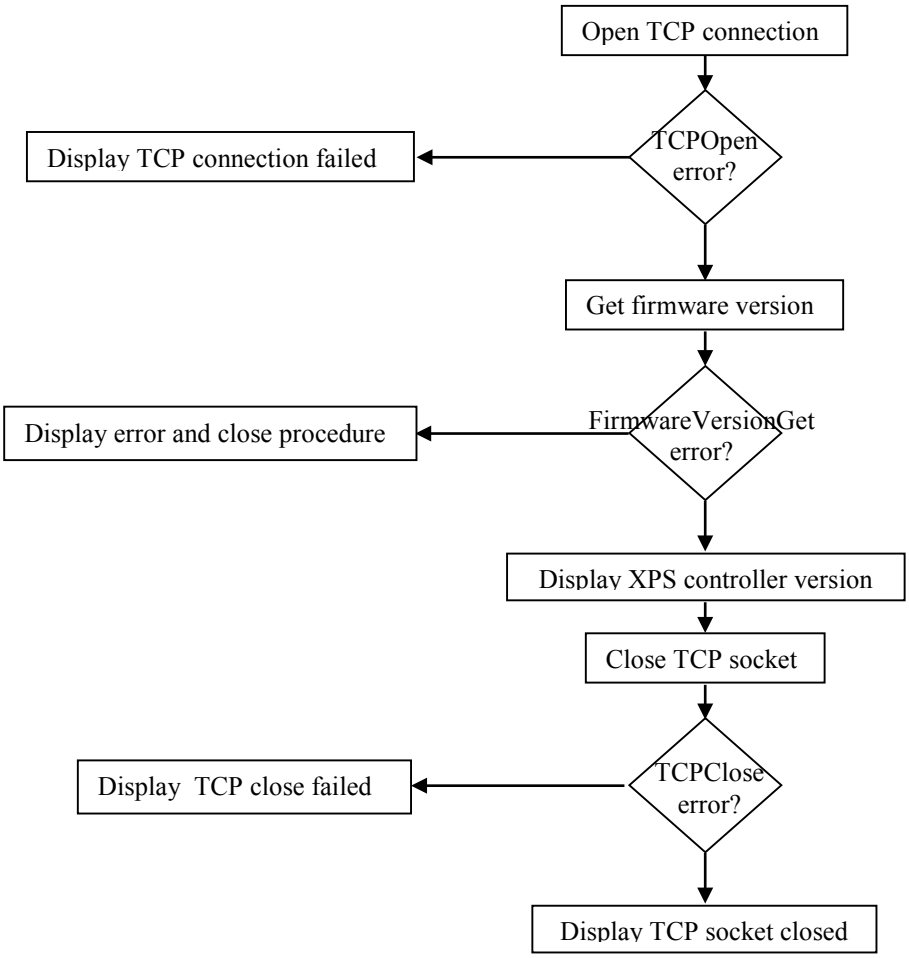
3.1 Management of the Errors

When an error occurs, it is desirable to analyze and treat the error. The following display error and close procedure is useful to detect and display the errors during the execution of a program. This sequence could be added to each program and called each time users need to test certain parts of a program.

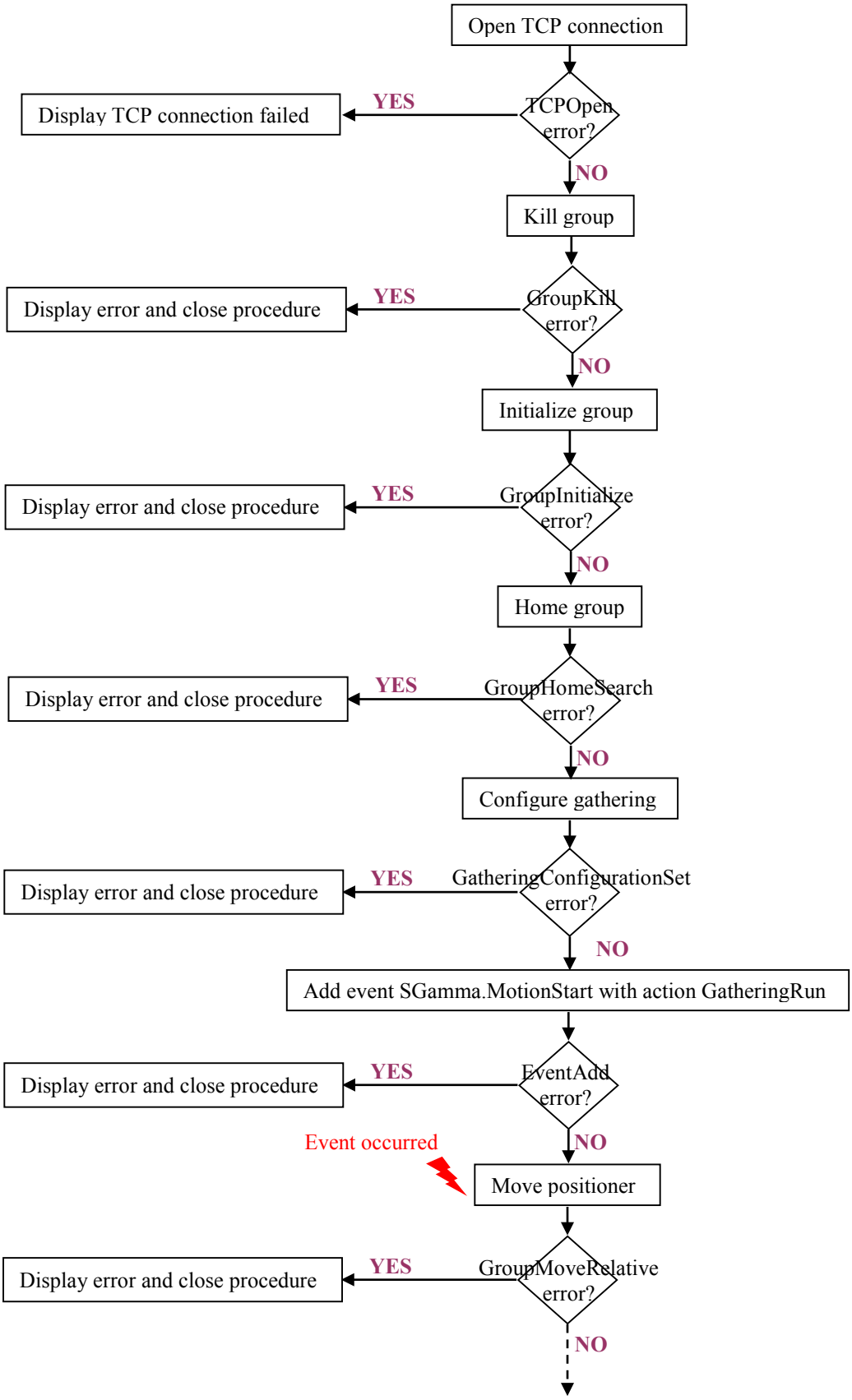
Display error and close procedure:

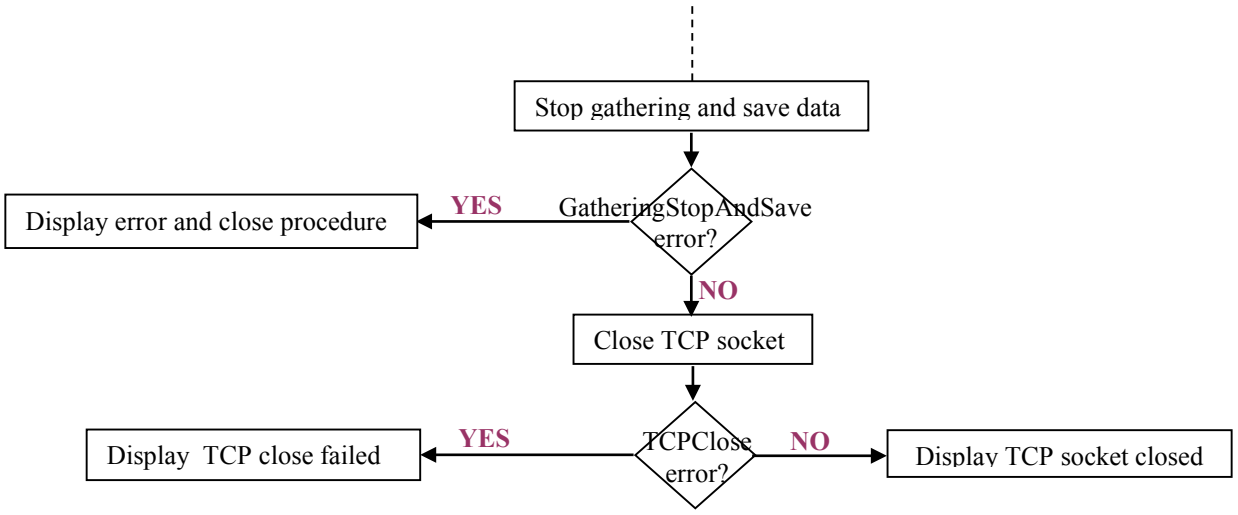


3.2 Firmware Version

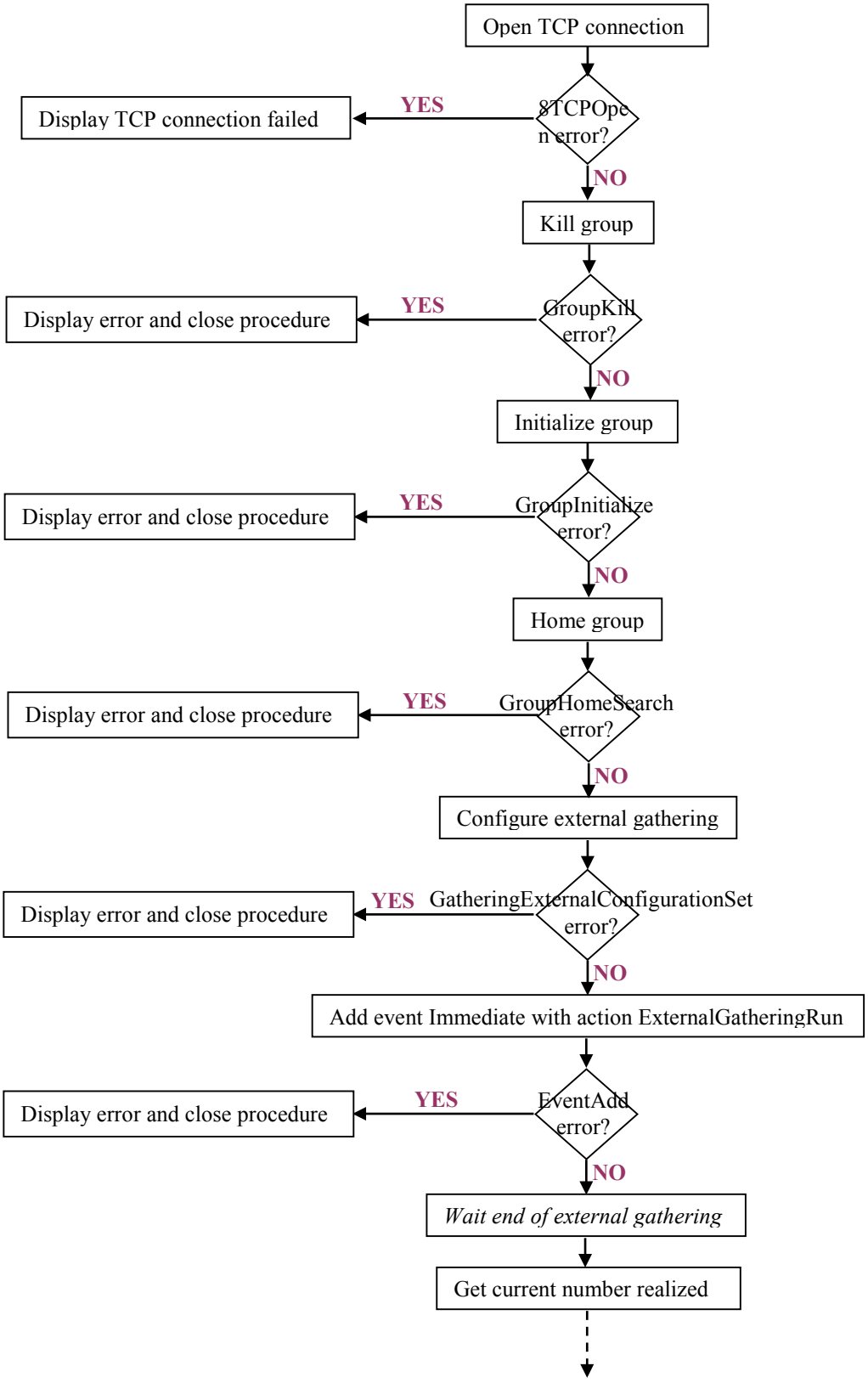


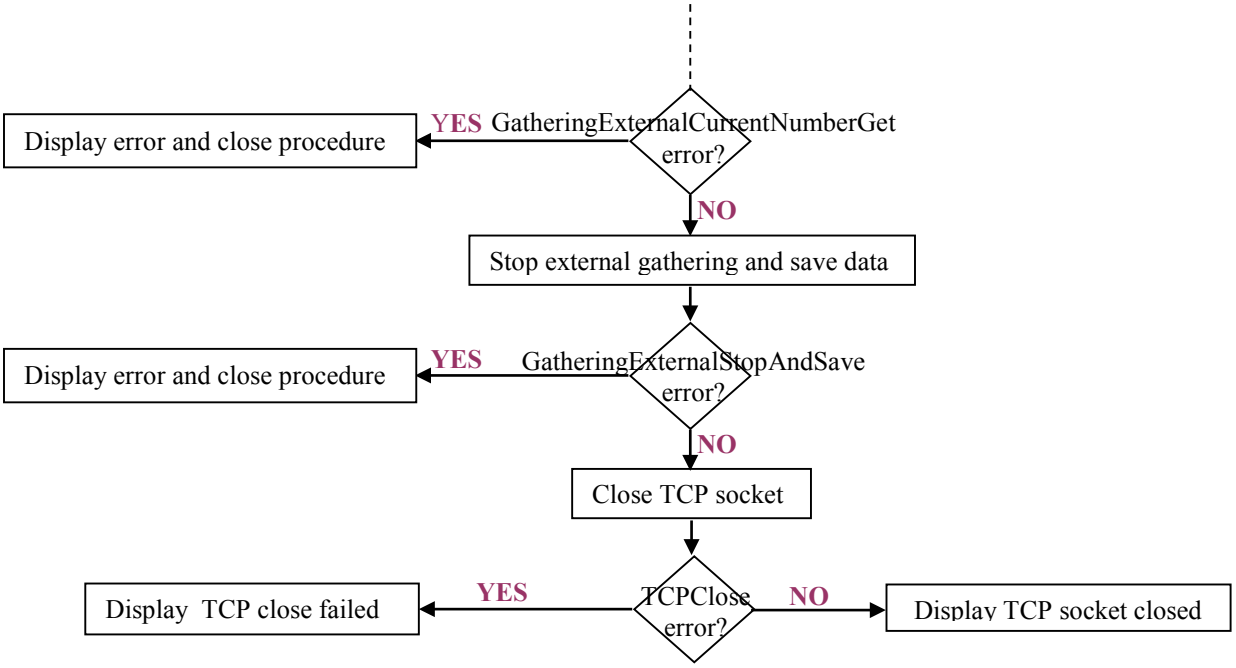
3.3 Gathering with Motion



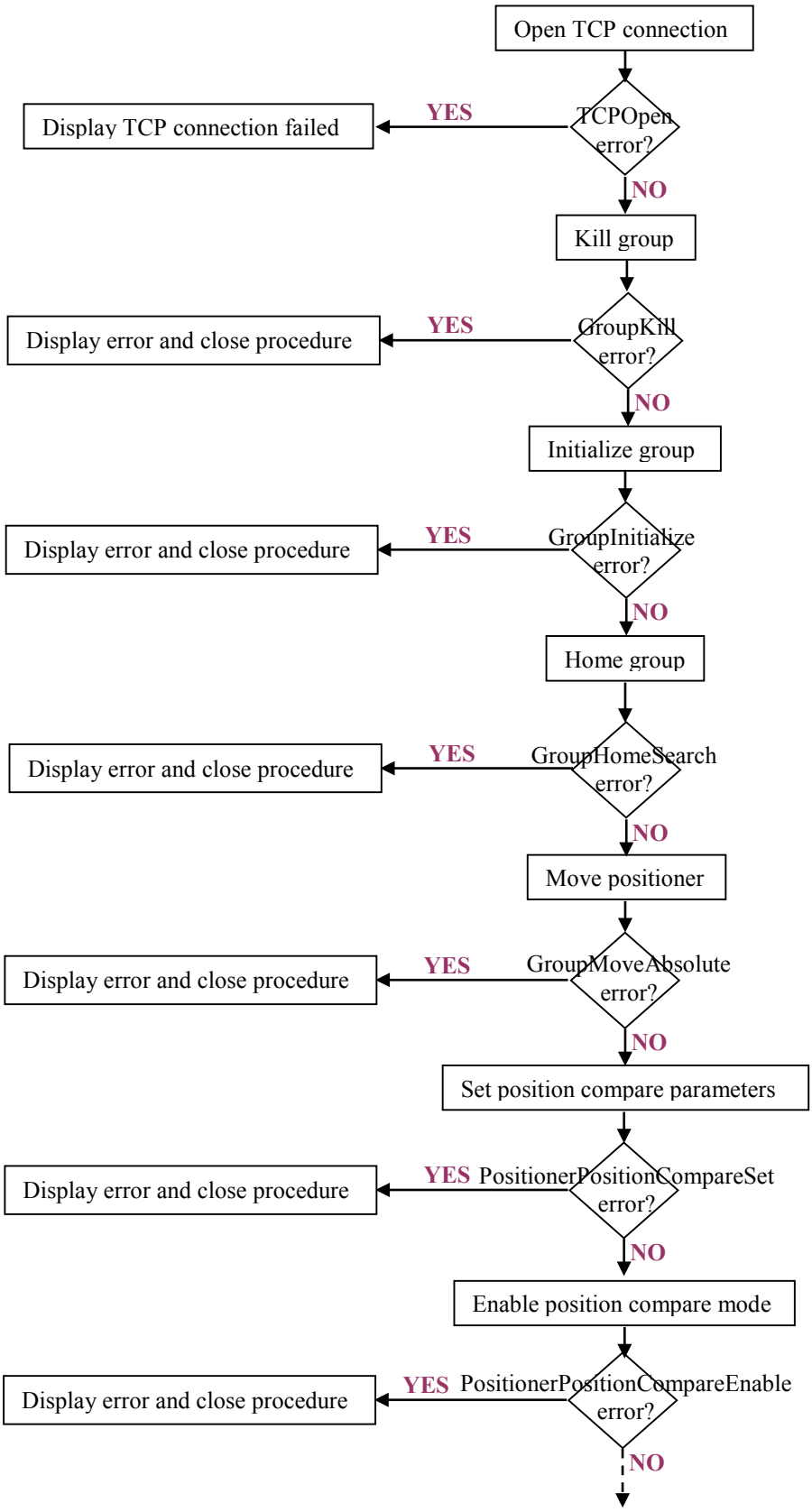

Stop gathering and save data
GatheringStopAndSave error?
YES
Display error and close procedure
NO
Close TCP socket
TCPClose error?
YES
Display TCP close failed
NO
Display TCP socket closed

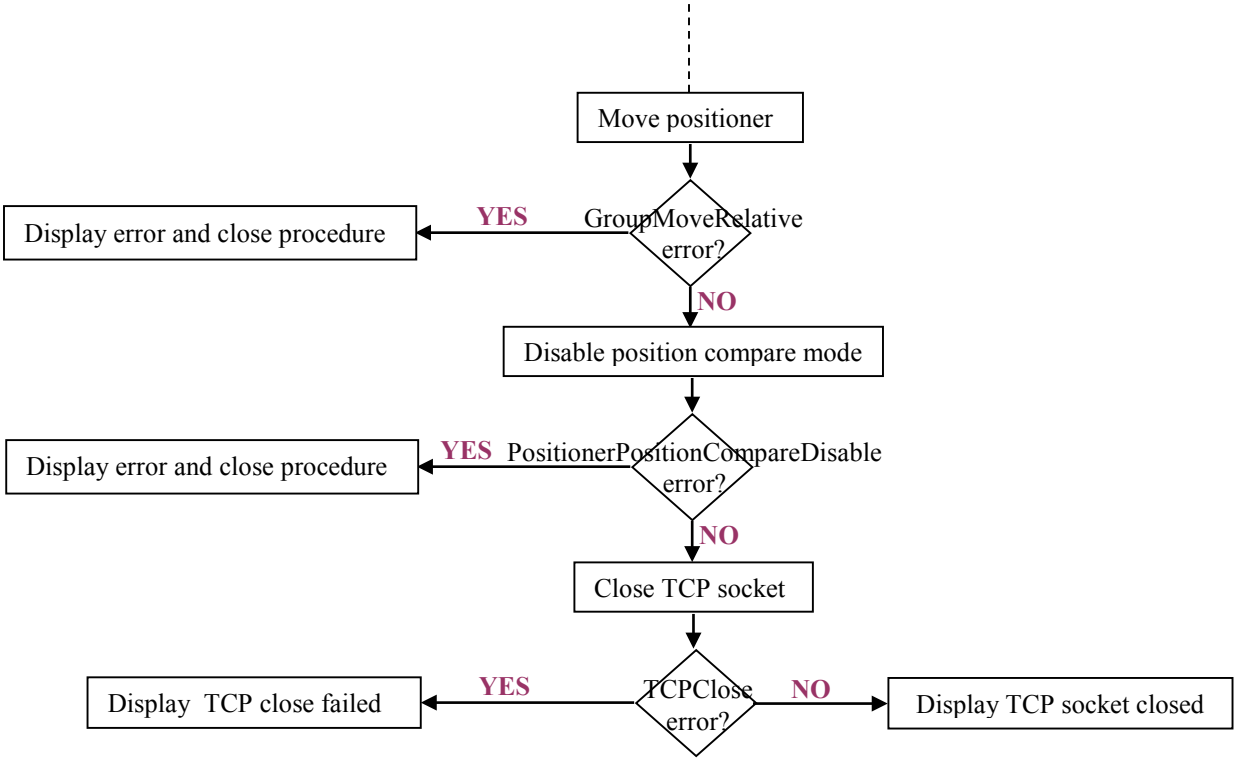
3.4 External Gathering



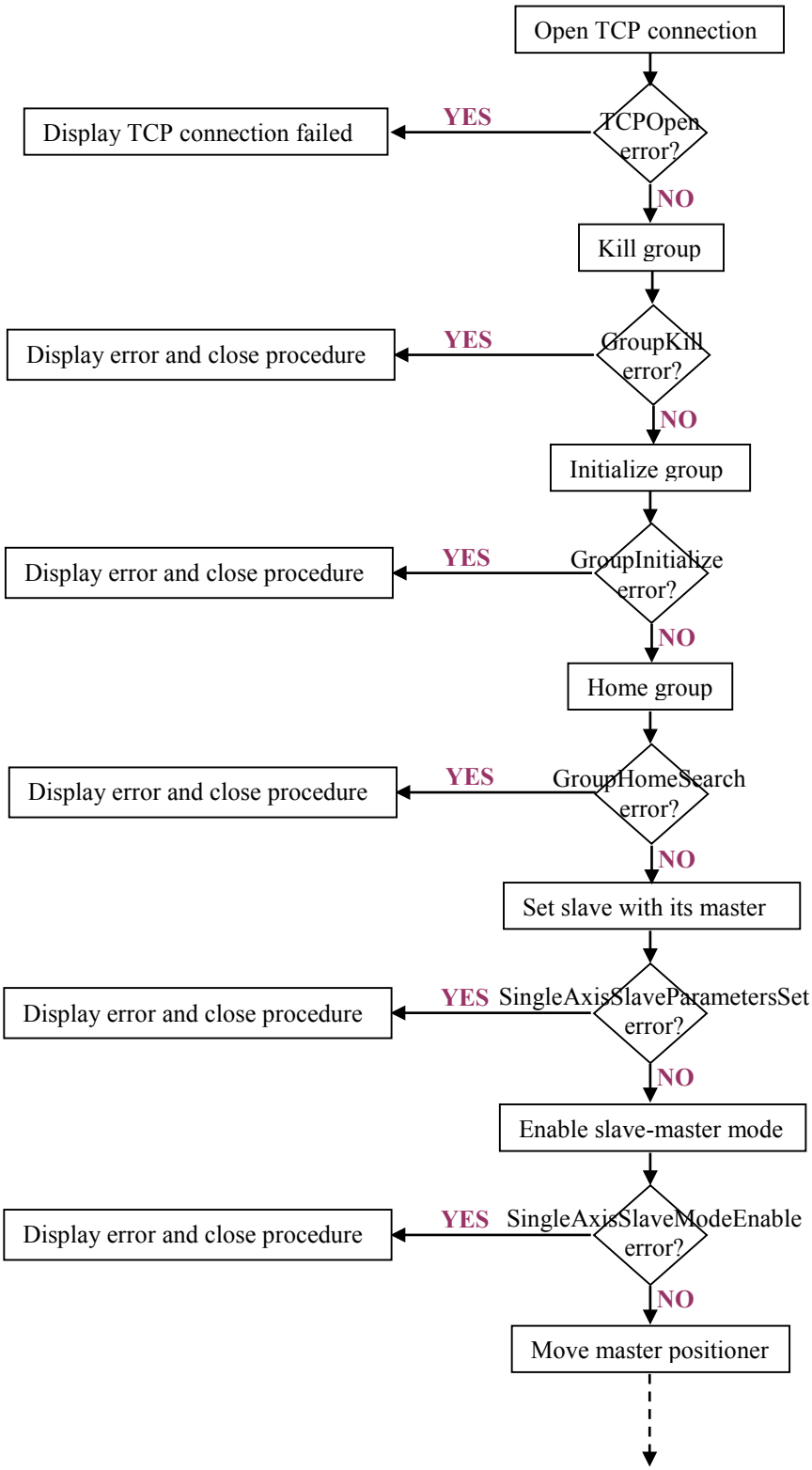

Display error and close procedure
YES
GatheringExternalCurrentNumberGet error?
NO
Stop external gathering and save data
Display error and close procedure
YES
GatheringExternalStopAndSave error?
NO
Close TCP socket
Display TCP close failed
YES
TCPClose error?
NO
Display TCP socket closed

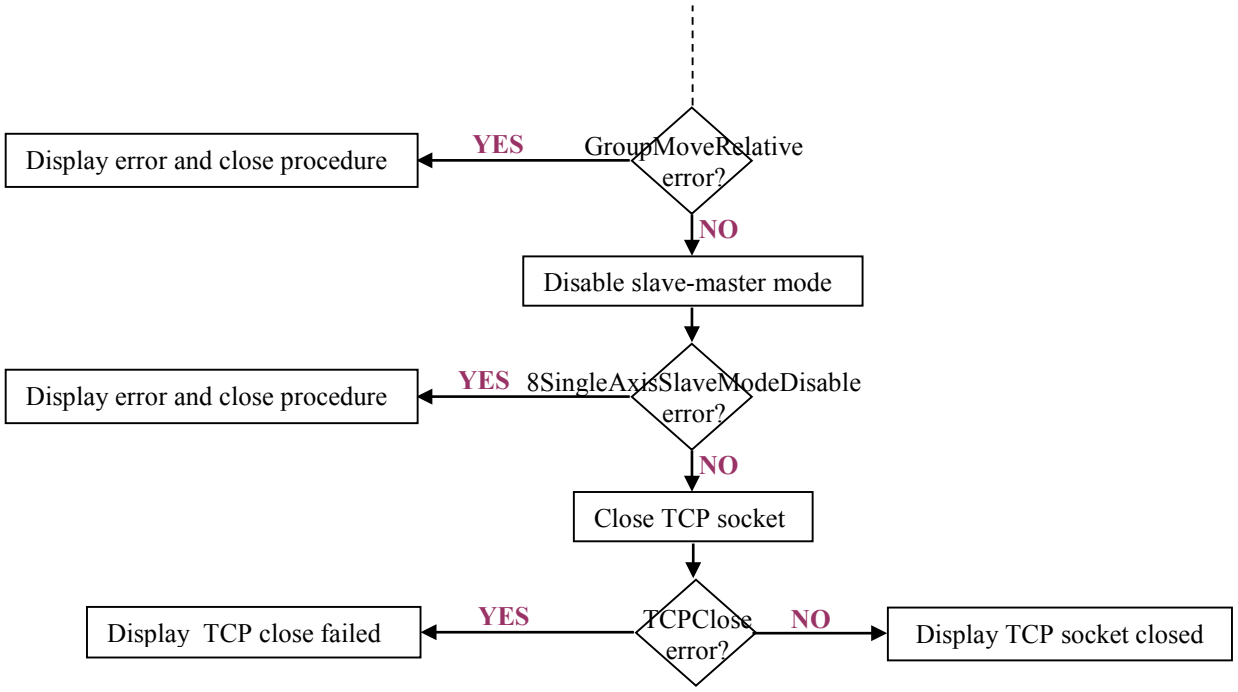
3.5 Output Compare



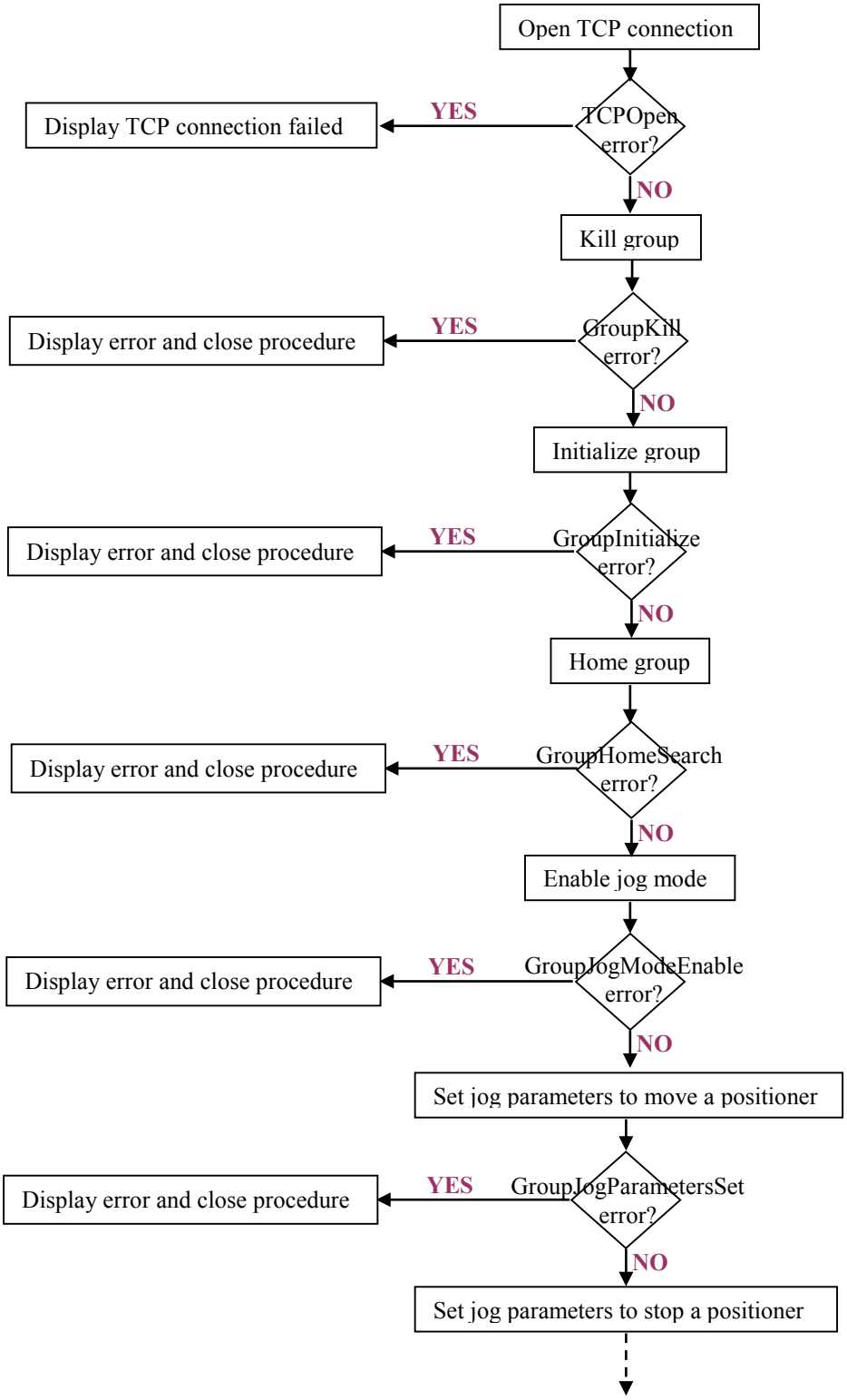

Move positioner
GroupMoveRelative error?
YES
Display error and close procedure
NO
Disable position compare mode
PositionerPositionCompareDisable error?
YES
Display error and close procedure
NO
Close TCP socket
TCPClose error?
YES
Display TCP close failed
NO
Display TCP socket closed

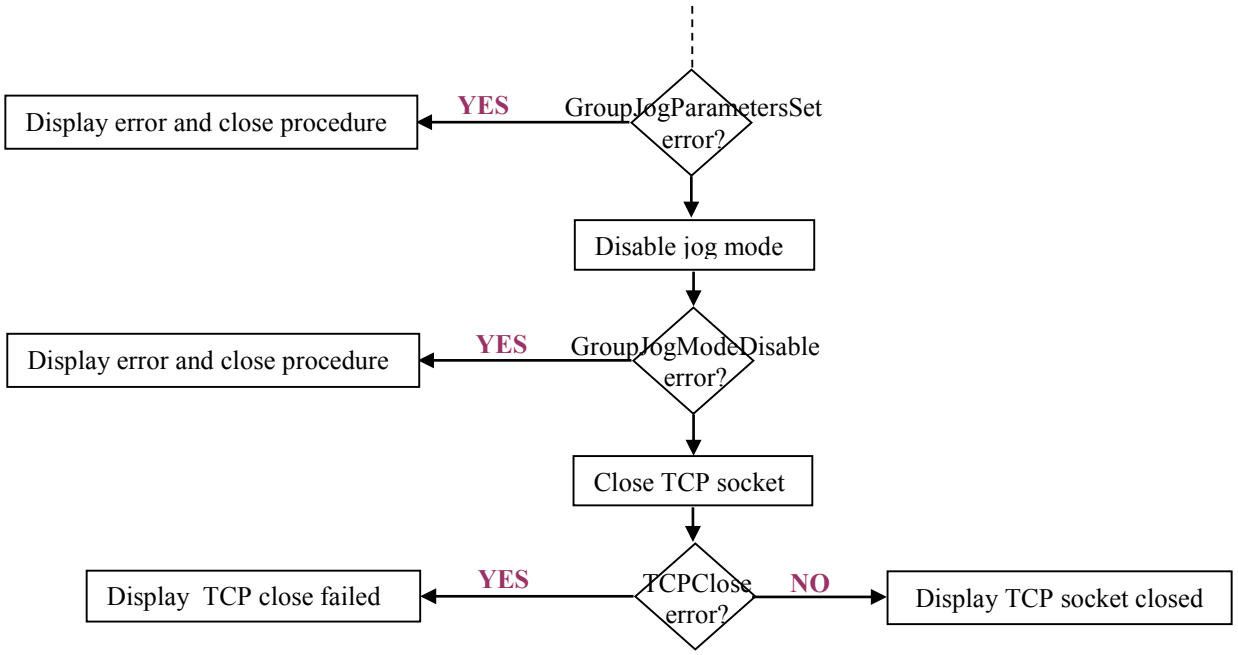
3.6 Slave-Master Mode



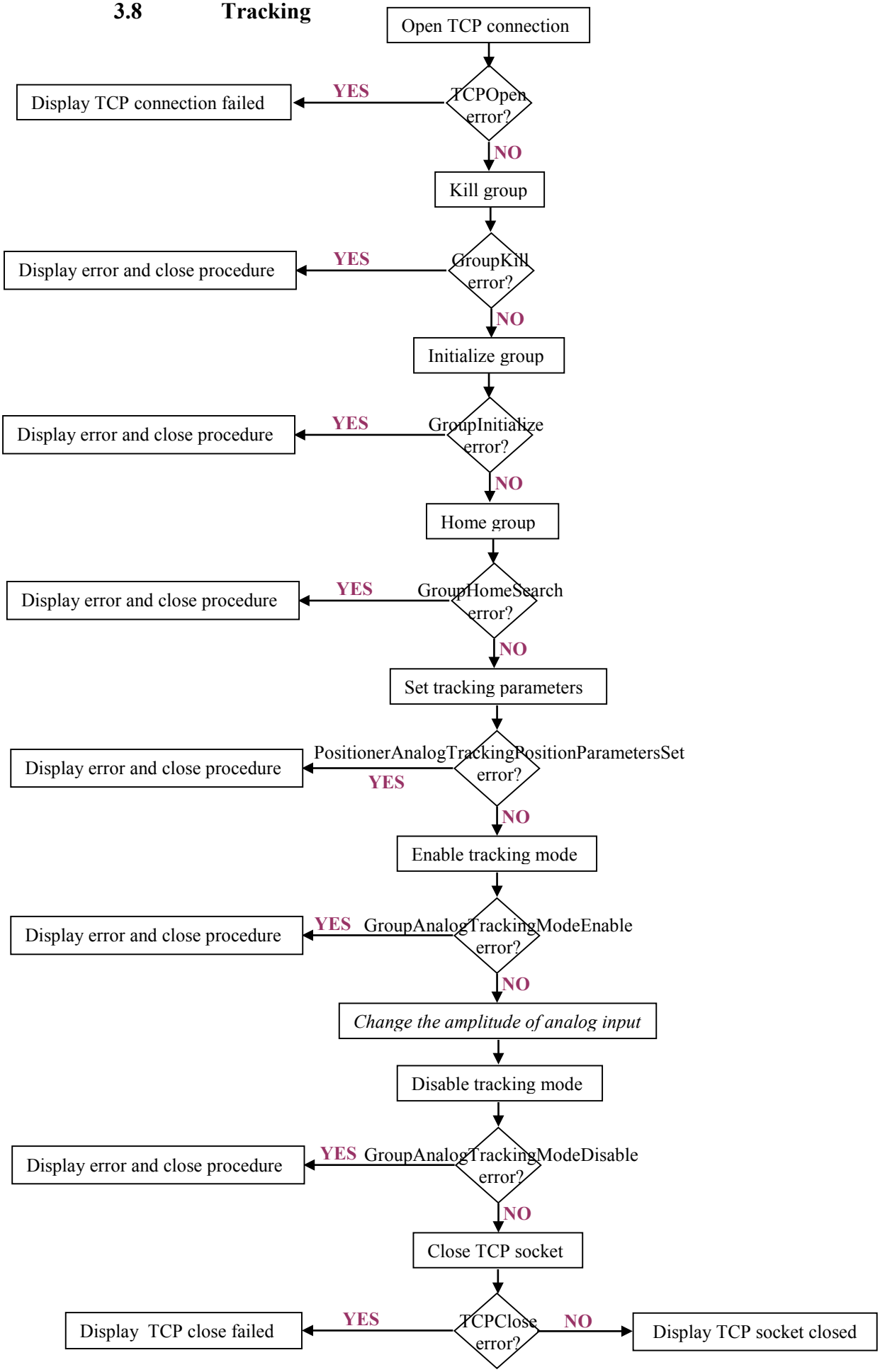

GroupMoveRelative error?
YES
Display error and close procedure
NO
Disable slave-master mode
8SingleAxisSlaveModeDisable error?
YES
Display error and close procedure
NO
Close TCP socket
TCPClose error?
YES
Display TCP close failed
NO
Display TCP socket closed

3.7 Jogging

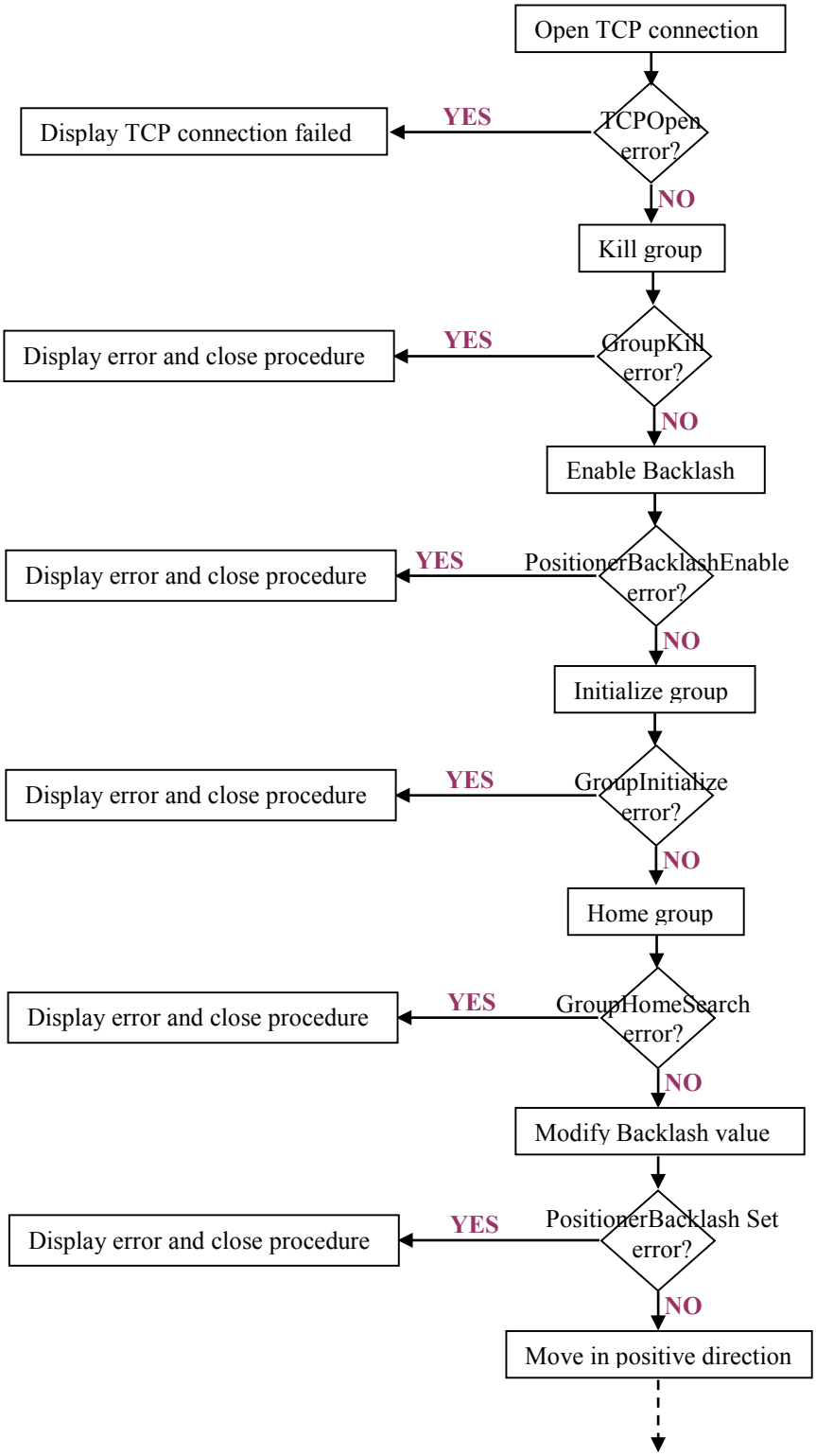

Open TCP connection
TCPOpen error?
YES
Display TCP connection failed
NO
Kill group
GroupKill error?
YES
Display error and close procedure
NO
Initialize group
GroupInitialize error?
YES
Display error and close procedure
NO
Home group
GroupHomeSearch error?
YES
Display error and close procedure
NO
Enable jog mode
GroupJogModeEnable error?
YES
Display error and close procedure
NO
Set jog parameters to move a positioner
GroupJogParametersSet error?
YES
Display error and close procedure
NO
Set jog parameters to stop a positioner

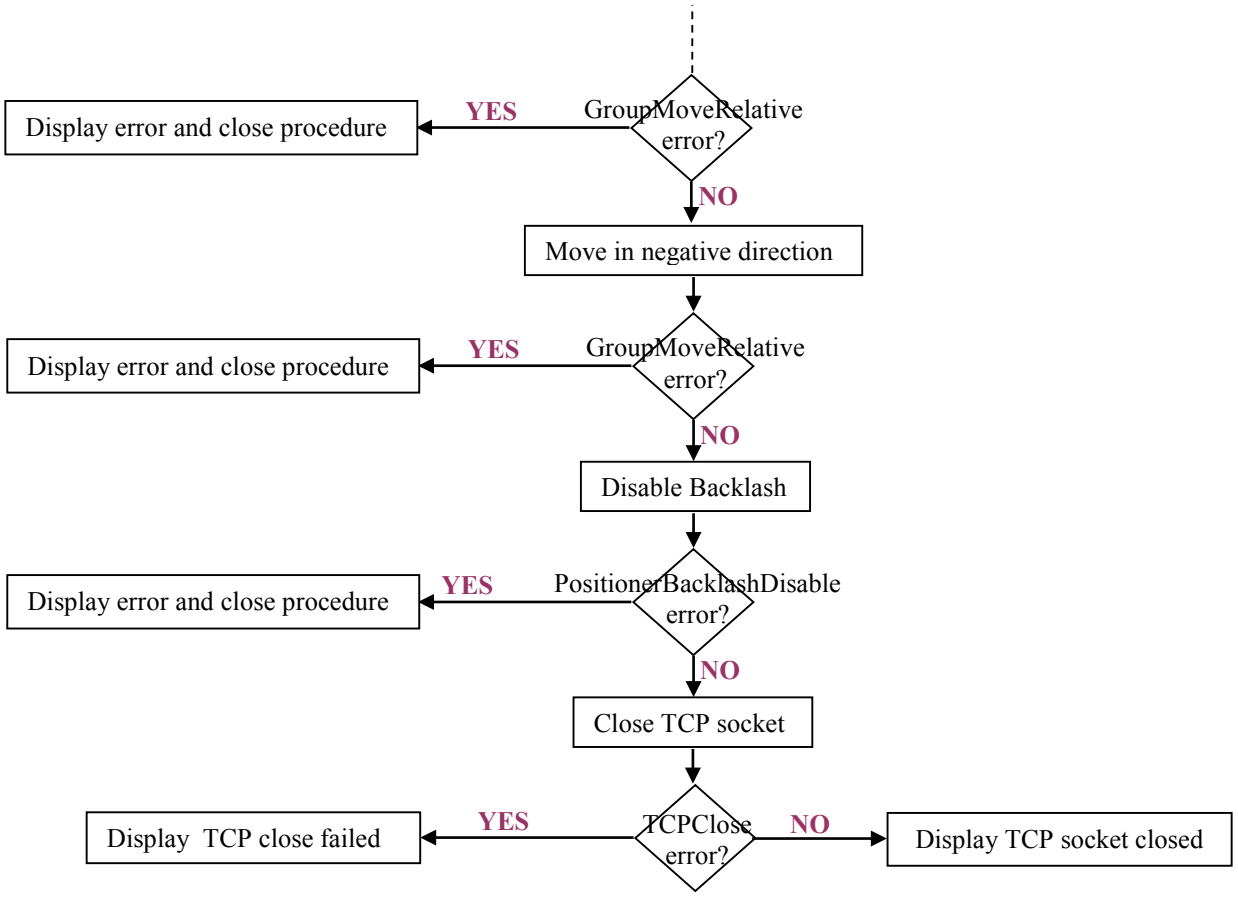

GroupJogParametersSet error?
YES
Display error and close procedure
Disable jog mode
GroupJogModeDisable error?
YES
Display error and close procedure
Close TCP socket
TCPClose error?
YES
Display TCP close failed
NO
Display TCP socket closed

3.8 Tracking

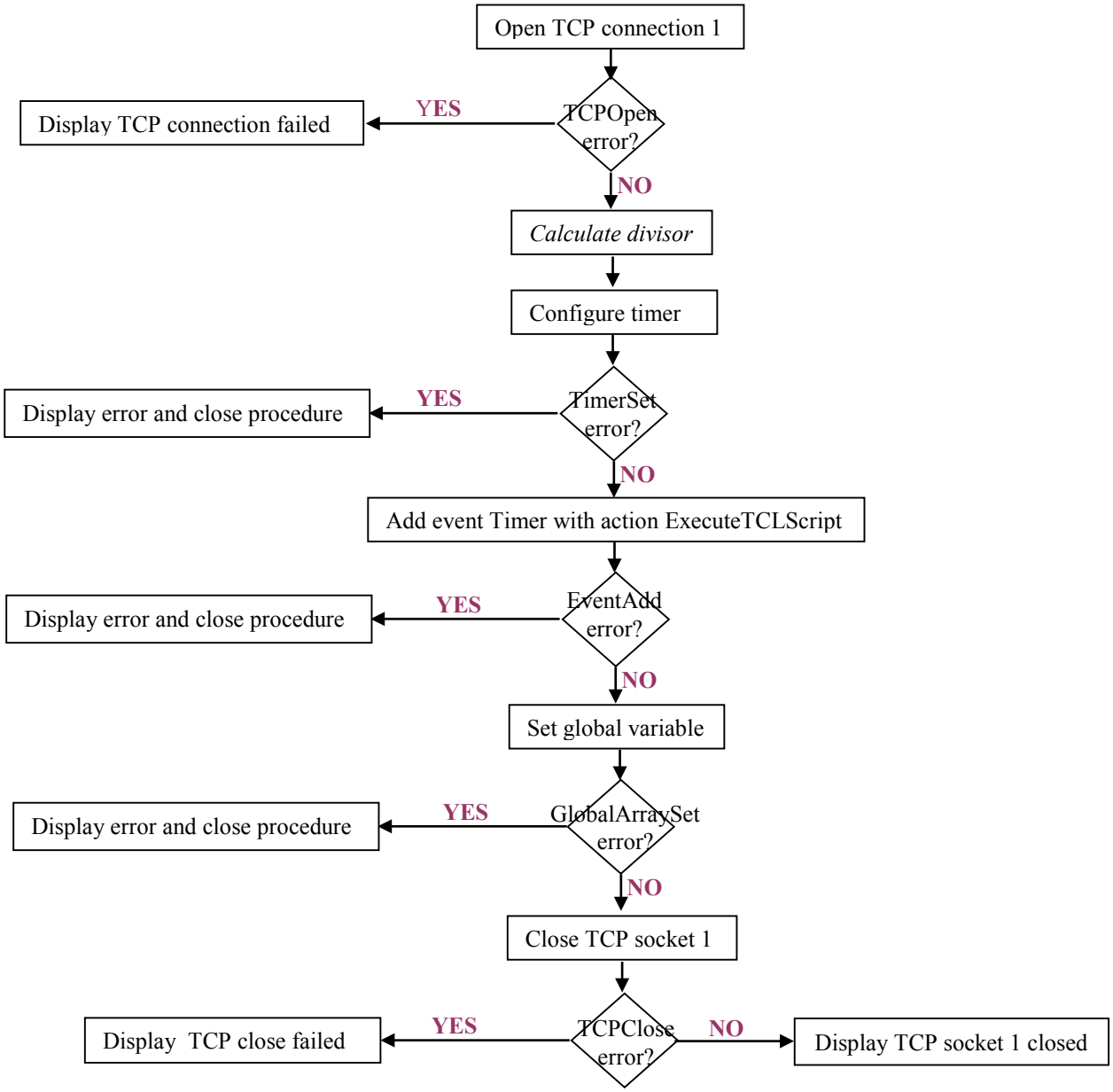

Open TCP connection
TCPOpen error?
YES
Display TCP connection failed
NO
Kill group
GroupKill error?
YES
Display error and close procedure
NO
Initialize group
GroupInitialize error?
YES
Display error and close procedure
NO
Home group
GroupHomeSearch error?
YES
Display error and close procedure
NO
Set tracking parameters
PositionerAnalogTrackingPositionParametersSet error?
YES
Display error and close procedure
NO
Enable tracking mode
GroupAnalogTrackingModeEnable error?
YES
Display error and close procedure
NO
Change the amplitude of analog input
Disable tracking mode
GroupAnalogTrackingModeDisable error?
YES
Display error and close procedure
NO
Close TCP socket
TCPClose error?
YES
Display TCP close failed
NO
Display TCP socket closed

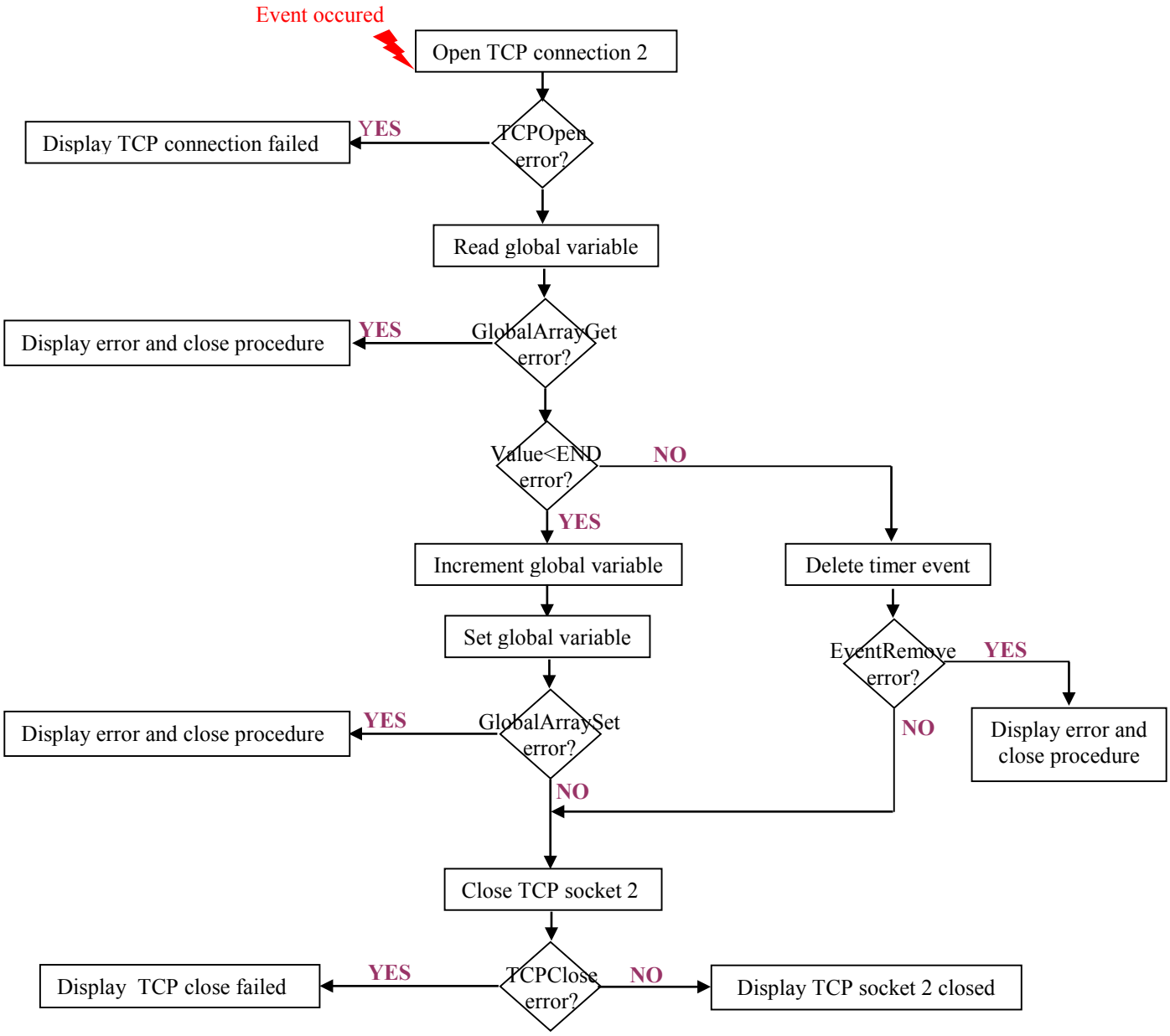
3.9 Backlash


Open TCP connection
TCPOpen error?
YES
Display TCP connection failed
NO
Kill group
GroupKill error?
YES
Display error and close procedure
NO
Enable Backlash
PositionerBacklashEnable error?
YES
Display error and close procedure
NO
Initialize group
GroupInitialize error?
YES
Display error and close procedure
NO
Home group
GroupHomeSearch error?
YES
Display error and close procedure
NO
Modify Backlash value
PositionerBacklash Set error?
YES
Display error and close procedure
NO
Move in positive direction


GroupMoveRelative error?
YES
Display error and close procedure
NO
Move in negative direction
GroupMoveRelative error?
YES
Display error and close procedure
NO
Disable Backlash
PositionerBacklashDisable error?
YES
Display error and close procedure
NO
Close TCP socket
TCPClose error?
YES
Display TCP close failed
NO
Display TCP socket closed

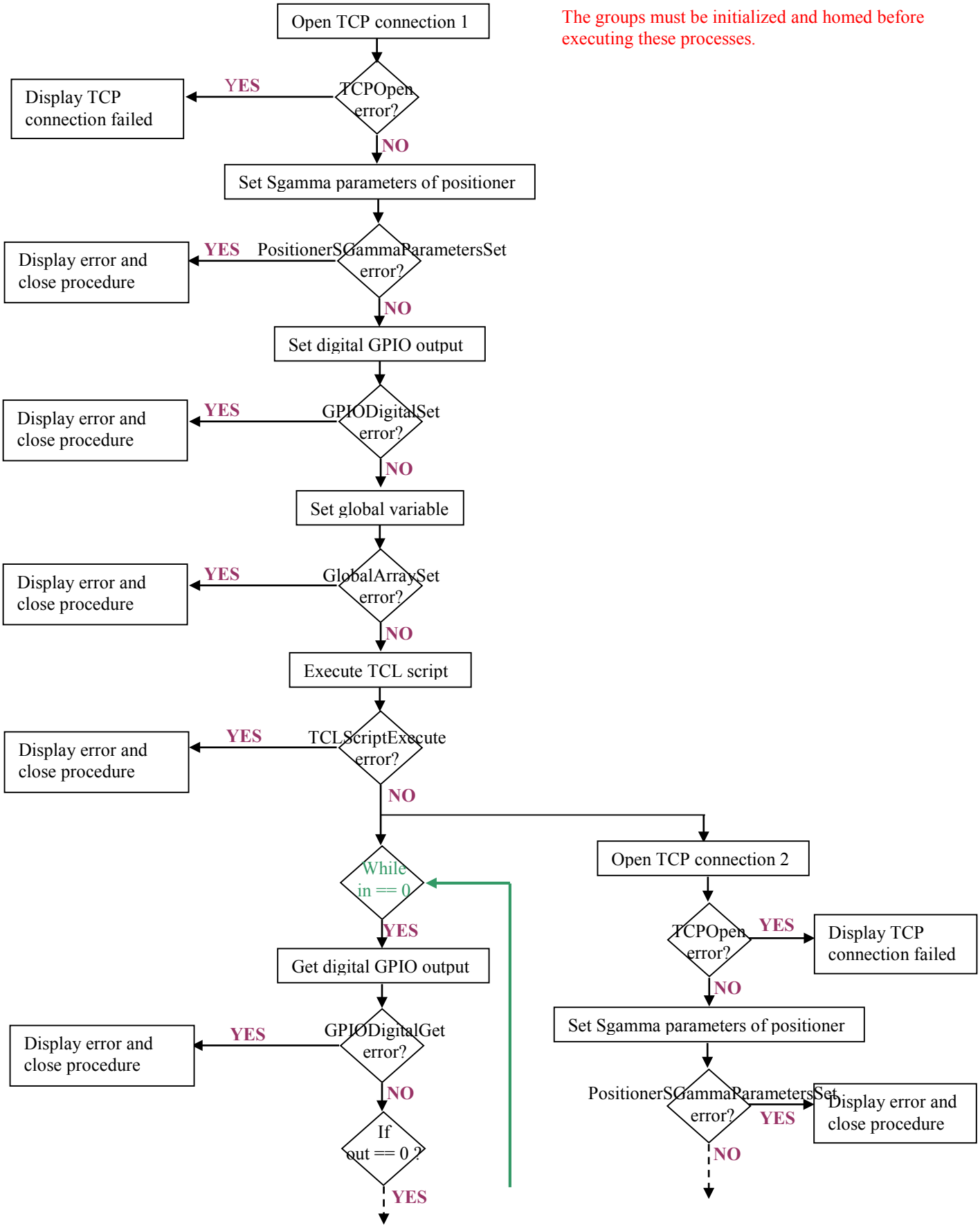
3.10 Timer Event and Global Variables

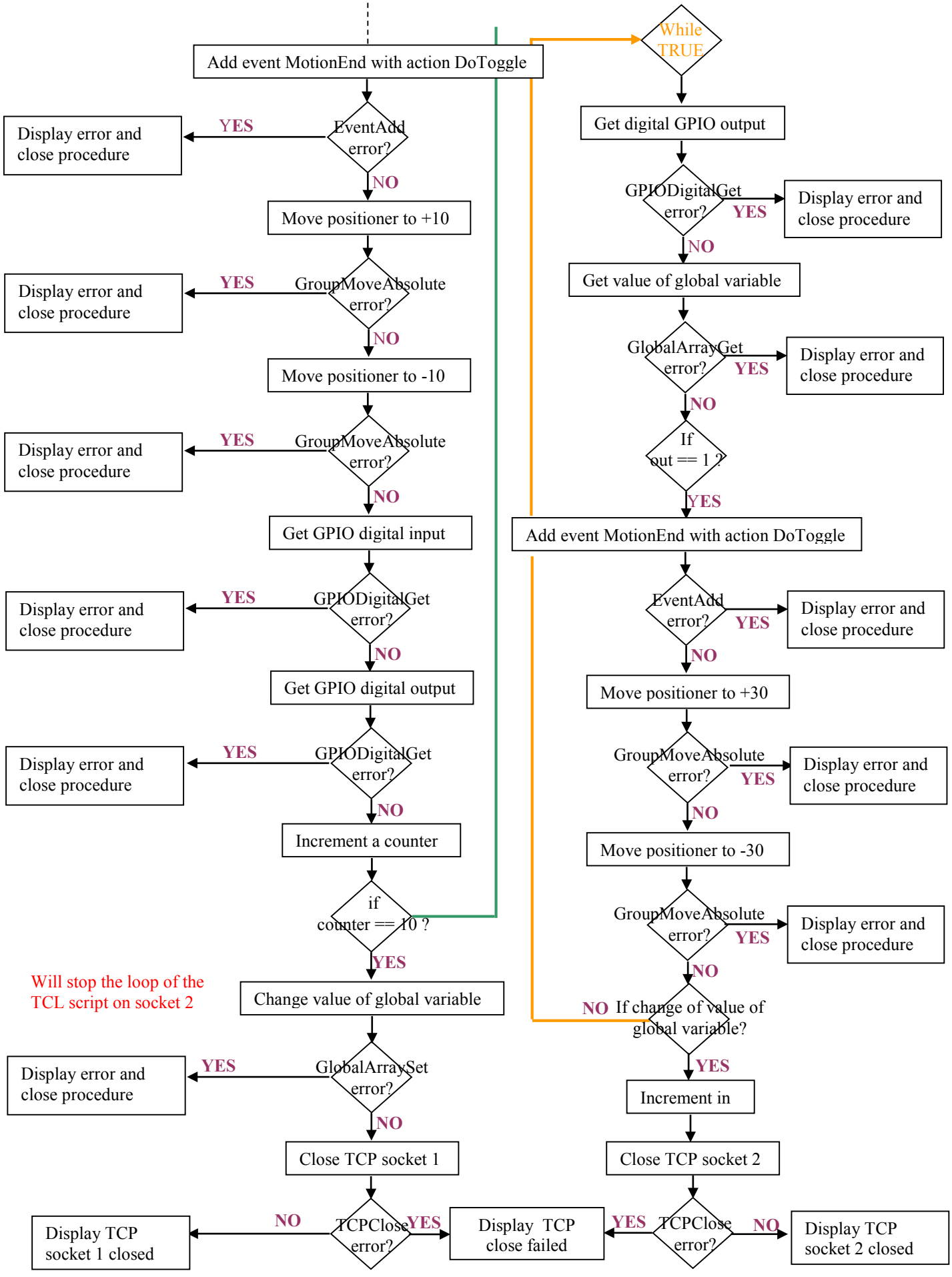



Event occured
Open TCP connection 2
TCPOpen error?
YES
Display TCP connection failed
Read global variable
GlobalArrayGet error?
YES
Display error and close procedure
Value<END error?
NO
YES
Increment global variable
Delete timer event
Set global variable
EventRemove error?
YES
Display error and close procedure
GlobalArraySet error?
YES
Display error and close procedure
NO
NO
Close TCP socket 2
TCPClose error?
YES
Display TCP close failed
NO
Display TCP socket 2 closed

3.11 Running Simultaneously Several Motion Processes

The groups must be initialized and homed before executing these processes.




Add event MotionEnd with action DoToggle
EventAdd error?
YES
Display error and close procedure
NO
Move positioner to +10
GroupMoveAbsolute error?
YES
Display error and close procedure
NO
Move positioner to -10
GroupMoveAbsolute error?
YES
Display error and close procedure
NO
Get GPIO digital input
GPIODigitalGet error?
YES
Display error and close procedure
NO
Get GPIO digital output
GPIODigitalGet error?
YES
Display error and close procedure
NO
Increment a counter
if counter == 10 ?
YES
Will stop the loop of the TCL script on socket 2
Change value of global variable
GlobalArraySet error?
YES
Display error and close procedure
NO
Close TCP socket 1
TCPClose error?
NO
Display TCP socket 1 closed
YES
Display TCP close failed
While TRUE
Get digital GPIO output
GPIODigitalGet error?
YES
Display error and close procedure
NO
Get value of global variable
GlobalArrayGet error?
YES
Display error and close procedure
NO
If out == 1 ?
YES
Add event MotionEnd with action DoToggle
EventAdd error?
YES
Display error and close procedure
NO
Move positioner to +30
GroupMoveAbsolute error?
YES
Display error and close procedure
NO
Move positioner to -30
GroupMoveAbsolute error?
YES
Display error and close procedure
NO
If change of value of global variable?
NO
YES
Increment in
Close TCP socket 2
TCPClose error?
YES
NO
Display TCP socket 2 closed

Service Form

Your Local Representative

Tel.: _____

Fax: _____

Name: _____ Return authorization #: _____

(Please obtain prior to return of item)

Company: _____

Address: _____ Date: _____

Country: _____ Phone Number: _____

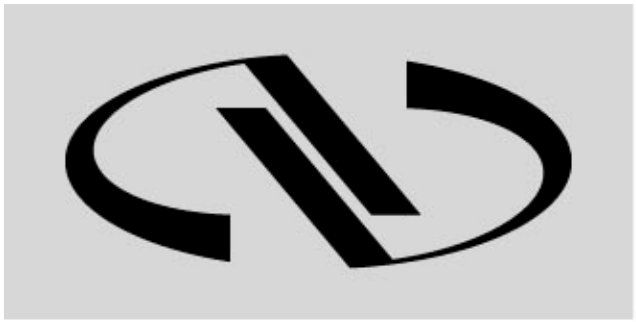
P.O. Number: _____ Fax Number: _____

Item(s) Being Returned: _____

Model#: _____ Serial #: _____

Description: _____

Reasons of return of goods (please list any specific problems): _____



Newport®

Experience | Solutions

Visit Newport Online at:
www.newport.com

North America & Asia
Newport Corporation
1791 Deere Ave.
Irvine, CA 92606, USA

Sales
Tel.: (800) 222-6440
e-mail: sales@newport.com

Technical Support
Tel.: (800) 222-6440
e-mail: tech@newport.com

Service, RMAs & Returns
Tel.: (800) 222-6440
e-mail: service@newport.com

Europe
MICRO-CONTROLE Spectra-Physics S.A.S
9, rue du Bois Sauvage
91055 Évry CEDEX
France

Sales
Tel.: +33 (0)1.60.91.68.68
e-mail: france@newport.com

Technical Support
e-mail: tech_europe@newport.com

Service & Returns
Tel.: +33 (0)2.38.40.51.55