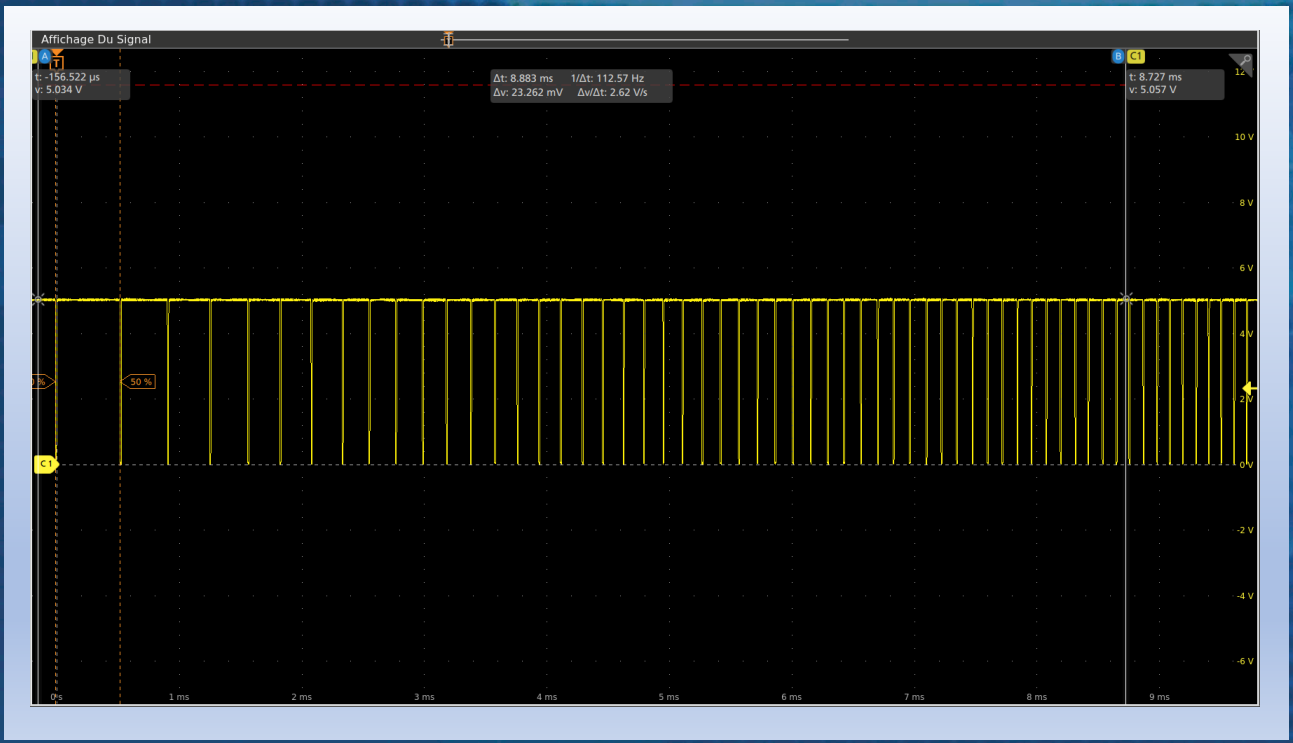
Multi-Axis PSO Rev. 1- 03/26

User Guide

XPS-D

Multi-Axis Pulse Synchronized Output (PSO)



1771 Deere Avenue
Irvine, CA 92606 United States
+1 (949)-863-3144

FAX: +1 (949)-253-1680

WWW.NEWPORT.COM

Table of Contents

1 Introduction

Position Synchronized Output (PSO) can generate trigger events along a trajectory using two primary operating modes: **Absolute Mode** and **Relative Mode**. Each mode provides a different mechanism for determining when trigger events occur during motion, and each offers advantages depending on the requirements of the application.

Absolute PSO Mode generates trigger events when the stage position crosses a predefined set of absolute coordinates along the trajectory. These coordinates are typically prepared before motion begins and stored internally as a list of trigger locations. Because the trigger points are stored explicitly, this mode is sometimes referred to as **array-based PSO** in the documentation.

Absolute mode provides deterministic placement of trigger events along the path. When the same trajectory is executed multiple times, trigger events occur at the same absolute positions, enabling highly repeatable multi-pass processing and precise alignment to known coordinates along the motion path. Absolute mode also integrates naturally with trajectory descriptions such as LineArc segments, where pulse pitch or other process parameters can be defined per segment. However, because trigger locations are stored explicitly, the number of trigger points available for a given trajectory is limited by the controller's internal storage capacity.

Relative PSO Mode, in contrast, generates trigger events based on the **incremental distance traveled** along the trajectory rather than predefined positions. A pitch value defines the distance between successive trigger events, and each time the accumulated travel distance reaches this increment, a new trigger is generated.

Because trigger events are derived from distance accumulation rather than a predefined list of points, relative mode does not require storage of trigger positions and therefore does not impose a practical limit on the number of pulses that can be generated along a path. This mode can also be more tolerant of small trajectory deviations or following error, since trigger events are based on measured travel distance rather than detection of specific coordinate crossings.

The trade-off is that relative mode does not inherently associate trigger events with specific positions along the trajectory. As a result, when a trajectory is repeated multiple times, trigger locations may not fall exactly on the same absolute coordinates due to small variations in path length or accumulated motion error.

In practice, the choice between absolute and relative PSO modes depends on the application requirements. Absolute mode is typically preferred when deterministic placement of trigger events and repeatability across passes are critical. Relative mode is often advantageous when very large numbers of pulses are required along long paths, or when the triggering process should follow the actual distance traveled rather than predetermined coordinates.

The new multi-axis PSO feature allows the following functionalities:

- perform multi-axis absolute PSO from a "point-to-point" file or a line-arc trajectory with and without gates (meaning some portion of the trajectory have no pulses).
- Perform multi-axis relative PSO according:
  - GroupMoveRelative API
  - GroupMoveAbsolute API
  - LineArc trajectory execution
  - PT trajectory execution
  - PVT trajectory execution
- Perform multi-axis PSO up to 3 axes and 1 000 000 points.
- Perform multi-axis PSO on these types of groups:
  - XY
  - MultipleAxes

2 Set-Up

The new multi-axis PSO feature requires specific configuration and may require existing equipment to be updated to comply.
Refer to “XPS-D – User Interface Manual” section “Controller – General Information” to check Hardware and Firmware versions installed.

2.1 Controller

The new multi-axis PSO feature is available for XPS-D controllers complying with Hardware and Firmware described below.

2.1.1 Hardware Version (CIE)

Compatible CIE boards: from version E5362F3

2.1.2 Firmware Version

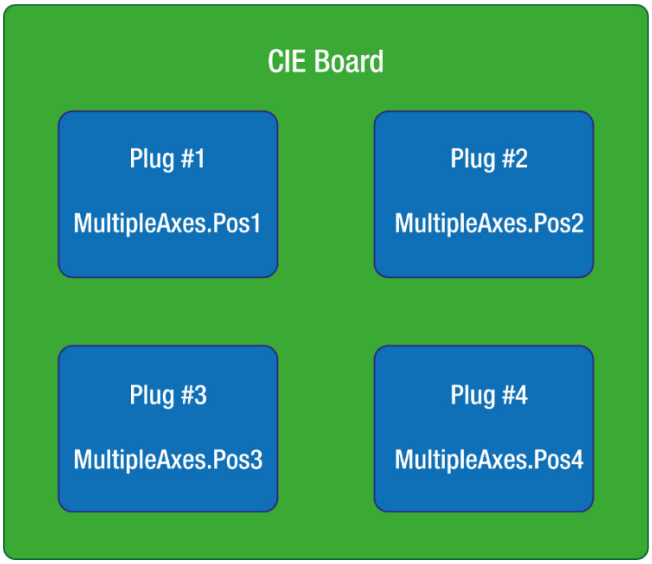
The new multi-axis PSO feature is implemented since XPS Unified 1.9.x:

- Firmware : version 1.9.1
- Snapshot : N13106.

Refer to “XPS-D – User Interface Manual” section “Controller – Firmware Update” to install the compatible firmware, available for download at www.newport.com.

2.2 System.ini

Positioners assigned to perform multi-axis PSO must be the first positioners in the group.
Also, the N positioner of the group must be plugged to the N encoder of the CIE board.



Example of configuration compatible with multi-axis PSO

2.3 System.ref

Maximum number of multi-axis PSO points

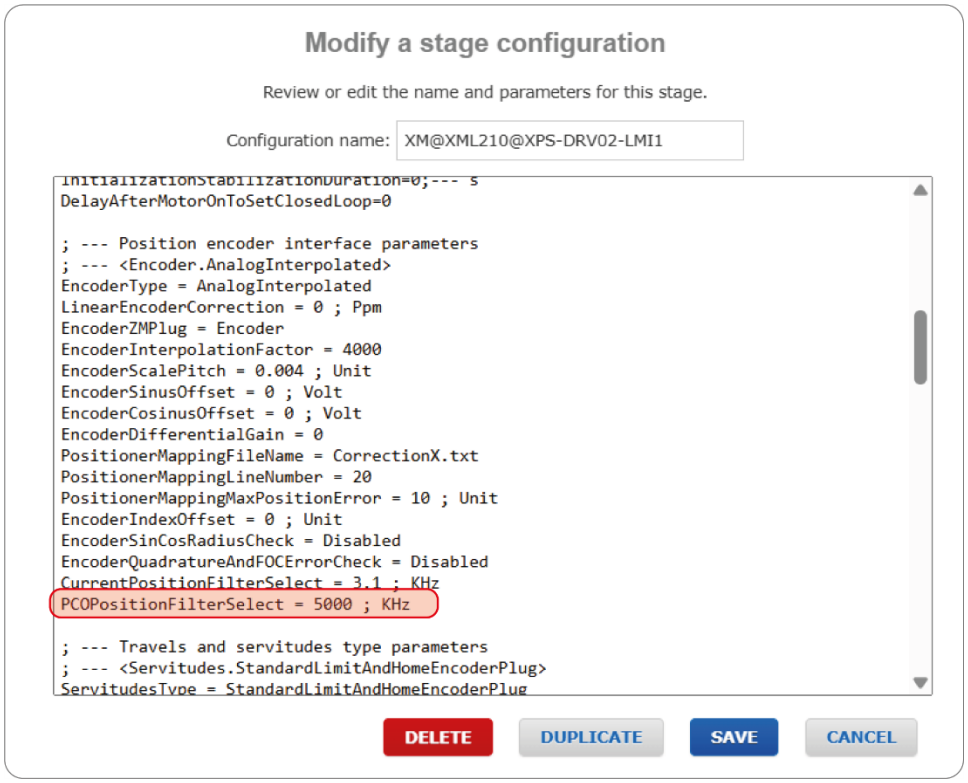
system.ref

```
[GENERAL]
OptionalModuleNames =
SharedLibraryModuleNames =
CorrectorISRPeriod = 100e-6                                    ; seconds
ProfileGeneratorISRRatio = 4
ServitudesISRRatio = 10
GatheringBufferSize = 1200000
DelayBeforeStartup = 0                                                 ; seconds
DebugTraceCommunicationBufferSize = 0                  ; characters, if 0 => no trace
CIEFastCompensatedPCOMaximumDataNumber = 1000000 ;--- data numbers
TrajectoryBufferSize = 100
PCOPlug = {1,2,3,4},{5,6,7,8}
CustomIRQDelaySwitch = Disabled;
IRQDelay = 20

[GPIO]
ExtendedGpioDacOutputRange = +/-10V

[FEATURES]             ; Only for STD or default PP firmware
CompensationSystemPreFeedForwardMode = Disabled ; Disabled or Enabled
CompensationSystemPostFeedForwardMode = Disabled ; Disabled or Enabled

[DEBUG]
PCI_DEBUG = Disabled                                   ; Disabled or Enabled
```

The maximum number of multi-axis absolute PSO points is configured by the **CIEFastCompensatedPCOMaximumDataNumber** parameter and set by default to the maximum value (1 000 000 points), If this value is exceeded, the following error message is returned: "*You are Trying to configure too much PSO points. Please refer to CIEFastCompensatedPCOMaximumDataNumber parameter from general section from /Admin/Config/system.ref file*".
The minimum value is 0.
The maximum value depends on the number of groups, and which features are enabled in the configuration.
If this value is too high, an error will be displayed at startup to indicate that the controller does not have enough memory "*temporary buffer creation failed: not enough RAM memory*" to perform PSO.

2.4 Stage configuration

The **PCOPositionFilterSelect** parameter of the stage configuration allows to set the frequency of the filter applied on the position used for PSO, External Gathering and AquadB output and is set by default to 5000 which means No Filter.



The ideal value for PSO Position Filter depends on the application.
In the event of a noisy signal, intermittent pulses may be generated.
Changing this setting may resolve these interferences but will add latency, see table above for more information.
Possible values for PSO Position Filter Frequency with associated latency:

Parameter Value	Filter Frequency	Position Latency
5000	No filter	
600	600 kHz	210 ns
230	230 kHz	540 ns
110	110 kHz	1.14 µs
49.74	49.74 kHz	2.5 µs
24.87	24.87 kHz	5 µs
12.43	12.43 kHz	10 µs
6.22	6.22 kHz	20 µs
3.11	3.11 kHz	40 µs
1.55	1.55 kHz	80 µs
0.78	0.78 kHz	160 µs

3 PSO configuration

There is one PSO connector for Axis 1 and Axis 2 and another for Axis 5 and Axis 6
(PSO is not available for Axis 3, 4, 7 and 8).
The signals provided on this plug depend on the configuration of the output triggers, see **XPS-D Features Manual** for more details.

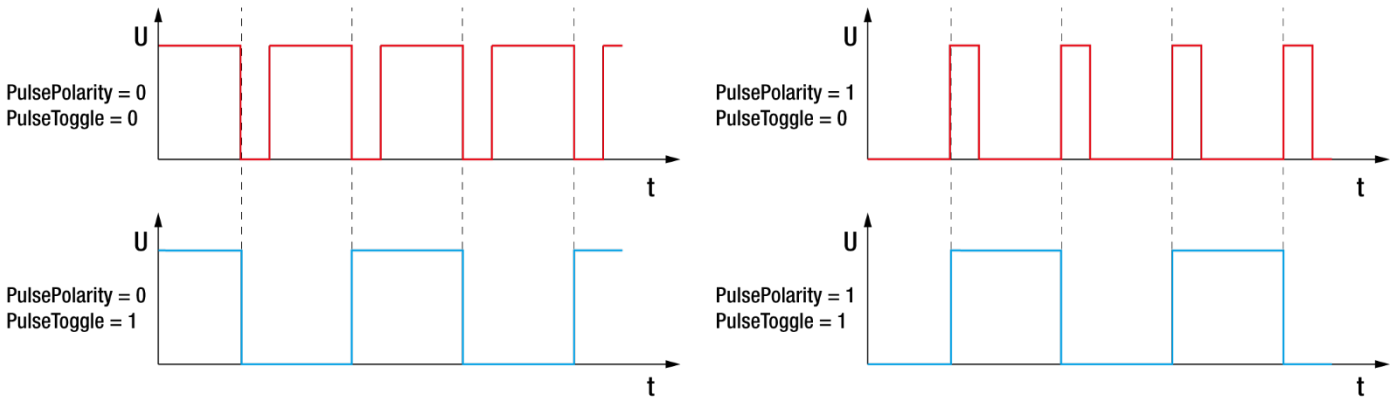
PSO output:

	Time Spaced Pulses	Distance Spaced Pulses	AquadB
Minimum pulse width	35 ns	35 ns	–
Minimum pulse frequency	0.05 Hz		
Maximum pulse frequency	20 MHz	1.6 MHz (5 MHz for less than 4096 pulses)	
Interpolation	–	x65536	AquadB encoder: x4 AnalogInterpolated encoder: x4, x256, x4096
Accuracy	10 ns	10 ns	10 ns
Position Window capability	YES	-	YES

3.1 Pulse parameters

The **PositionerCompensatedFastPCOPulseParametersSet** function allows to configure pulses parameters:

- **PulseWidth**: width of pulse in µs, possible values are: 35 ns to 327.68 µs (5 ns resolution).
  default value is set to 2 µs.
- **PulsePolarity**: 0 for negative pulse, 1 for positive pulse.
- **PulseToggle**: switch to Toggle mode instead of Pulse mode when set to 1.



Successive trigger pulses should have a minimum time lag of 625 ns (200 ns for less than 4096 pulses).

3.2 Maximum step distance

There is a maximum step distance limitation depending on the values of the interpolation factor and the encoder scale pitch which can be calculated as follows:

- Encoder resolution = Encoder scale pitch ÷ Interpolation factor.
- Maximum step distance = (2^18-1) * EncoderResolution.

The interpolation factor is configured for each positioner using **GroupMultiAxisPSOInterpolationFactorSet**.
The PSO interpolation factors available are:

- 4
- 16
- 256
- 4096
- 65536

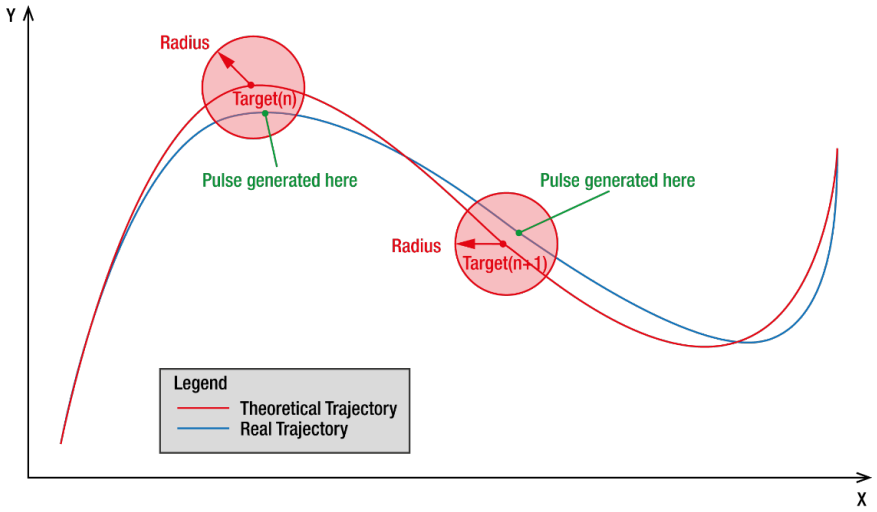
The step distance must be inferior to (2^18-1) * EncoderResolution.
If this value is exceeded, the following error message is returned: "*The step value for relative PSO exceeds the maximum allowed step value: for each axis the result of the formula [(EncoderScalePich / InterpolationFactor) * (2^18-1)] should be bigger than the requested step*".

Example:

Encoder scale pitch = 0.004 mm with interpolation factor @ 4096
Encoder resolution = Encoder scale pitch ÷ 4096
Maximum step distance = (2^18-1) * EncoderResolution = 0.255mm

3.3 Radius setting

In multi-axis absolute PSO mode, PSO points are strict positions, unlike relative multi-axis PSO.
Therefore, if the positioners do not go through the exact configured positions there will be no pulses.
The **GroupMultiAxisPSOAbsoluteRadiusSet** API allows to define a tolerance around the multi-axis PSO points.
This tolerance area is defined by a radius around the target point and depends on the performance of the different axes.



The controller checks if the axes radiuses are correctly set by calculating a global scaling coefficient as follow:

1. Calculating the radius scaled value for each axis:
   radiusScaled_Axis1 = radius_Axis1 * interpolationFactor_Axis1 ÷ encoderScalePitch_Axis1
   radiusScaled_Axis2 = radius_Axis2 * interpolationFactor_Axis2 ÷ encoderScalePitch_Axis2
   radiusScaled_Axis3 = radius_Axis3 * interpolationFactor_Axis3 ÷ encoderScalePitch_Axis3
   radiusScaled_Axis4 = radius_Axis4 * interpolationFactor_Axis4 ÷ encoderScalePitch_Axis4

2. Calculating an intermediate coefficient for each axis
   C_Value_Axis1 = 256 * radiusScaled_Axis2 * radiusScaled_Axis3 * radiusScaled_Axis4
   C_Value_Axis2 = 256 * radiusScaled_Axis1 * radiusScaled_Axis3 * radiusScaled_Axis4
   C_Value_Axis3 = 256 * radiusScaled_Axis1 * radiusScaled_Axis2 * radiusScaled_Axis4
   C_Value_Axis4 = 256 * radiusScaled_Axis1 * radiusScaled_Axis2 * radiusScaled_Axis3

3. Calculating the global scaling coefficient:
   The global scaling coefficient is based on the maximum intermediate coefficient value of the different axes.
   ScalingCoeff = 2^17 ÷ max_C_Value

4. Evaluation of the axes radius performance:
   ScalingCoeff must be > 1, else an error message is returned (ERR_PARAMETER_OUT_OF_RANGE).
   In this case, to keep the current interpolation factors and encoder scale pitch, the radius value for 1 or several axes must be reduced.
   If ScalingCoeff > 1, radiuses defined for each axis are accepted.

NOTE

**This evaluation method allows for the estimation of an overall satisfactory tolerance value but does not guarantee the generation of all pulses.
Indeed, if a radius is too small depending of the performances of the axes (< FollowingError for example), some pulses may be missed.**

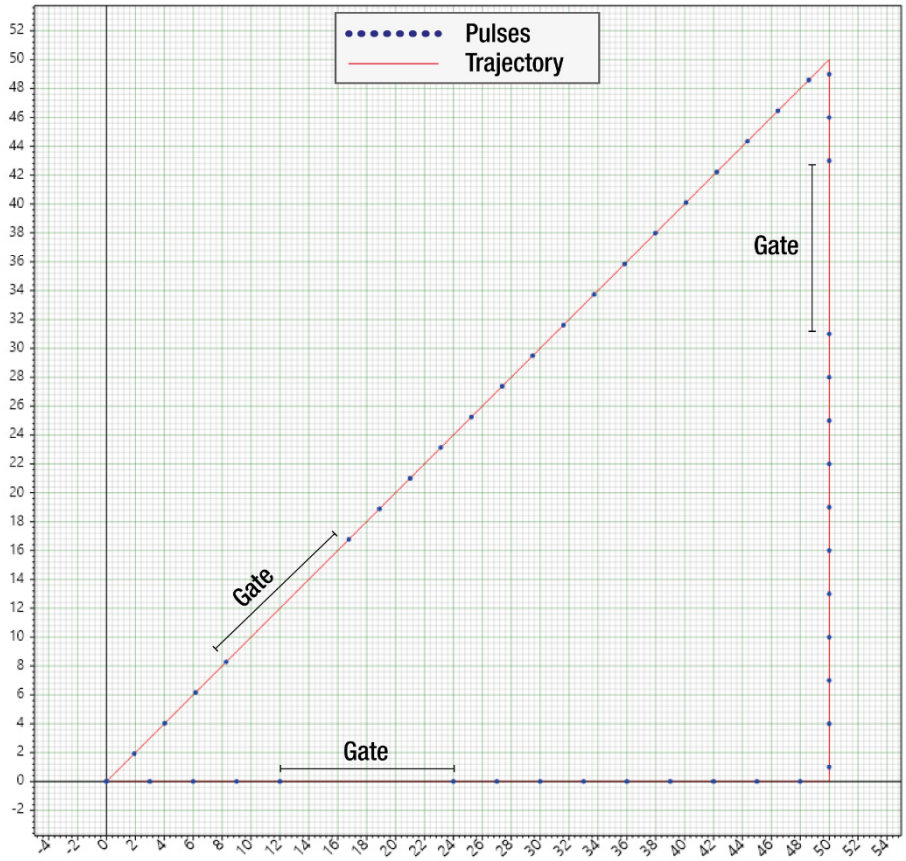
4 Absolute PSO Mode

Absolute PSO Mode generates trigger events when the stage position crosses a predefined set of absolute coordinates along the trajectory. These coordinates are typically prepared before motion begins and stored internally as a list of trigger locations. Because the trigger points are stored explicitly, this mode is sometimes referred to as **array-based PSO** in the documentation.

4.1 Trajectory PtoP combined with XY file

Description:

This mode is used to generate pulses along a trajectory from a PSO positions file.



In this example, pulses occur at the positions defined in the trajectory at a distance of 3mm between each other except at corner positions and gates positions.
The PSO position file contains the coordinates of the pulses programmed on the trajectory in a table.
Gates (move with no pulse) are defined by PSO point file.
Each line represents the coordinates of a pulse, with the position of each axis separated by a “tab” and the line ending with a carriage return (CR).

Example :

X_1	→	Y_1	↵		3.000 → 0.000 ↵
X_2	→	Y_2	↵		6.000 → 0.000 ↵
…		…			9.000 → 0.000 ↵
…		…			12.000 → 0.000 ↵
…		…			24.000 → 0.000 ↵
…		…			27.000 → 0.000 ↵
…		…			…
X_{n-1}	→	Y_{n-1}	↵		4.038 → 4.038 ↵
X_n	→	Y_n	↵		1.917 → 1.917 ↵

Application (based on above example):

2 axes in XY group or multiple axes group (Pulses generated every 3mm):

GroupMultiAxisPSORelativeDisable(XY,2,Immediate)

GroupMultiAxisPSOAbsoluteDisable(XY)

Disable any pulse generation to avoid pulse generation during initialization

PositionerCompensatedFastPCOPulseParametersSet(XY.X,10,0,0)

Configure pulses parameters: Pulse width, Pulse polarity and Pulse toggle.

GroupMultiAxisPSOInterpolationFactorSet(XY,2,256,256)

Configure the PSO interpolation factor for the specified number of positioners in the group.

GroupMultiAxisPSOAbsoluteRadiusSet(XY,2,0.004,0.004)

Define a tolerance around the multi-axis PSO points.

GroupMoveAbsolute(XY,0,0)

Initialization to position (0,0).

GroupMultiAxisPSOAbsoluteLoadFromFile(XY,2,TrianglePulses.trj)

GroupMultiAxisPSOAbsolutePrepare(XY,0,0)

Load the multi-axis PSO array from file "TrianglePulses.trj" and prepare controller taking into account the various position compensations/mappings to generate pulses.

GroupMultiAxisPSOAbsoluteEnable(XY)

Enable pulses generation.

GroupMoveAbsolute(XY,50,0)

Move to position (50, 0)

GroupMoveAbsolute(XY,50,50)

Move to position (50, 50)

GroupMoveAbsolute(XY,0,0)

Move to position (0, 0)

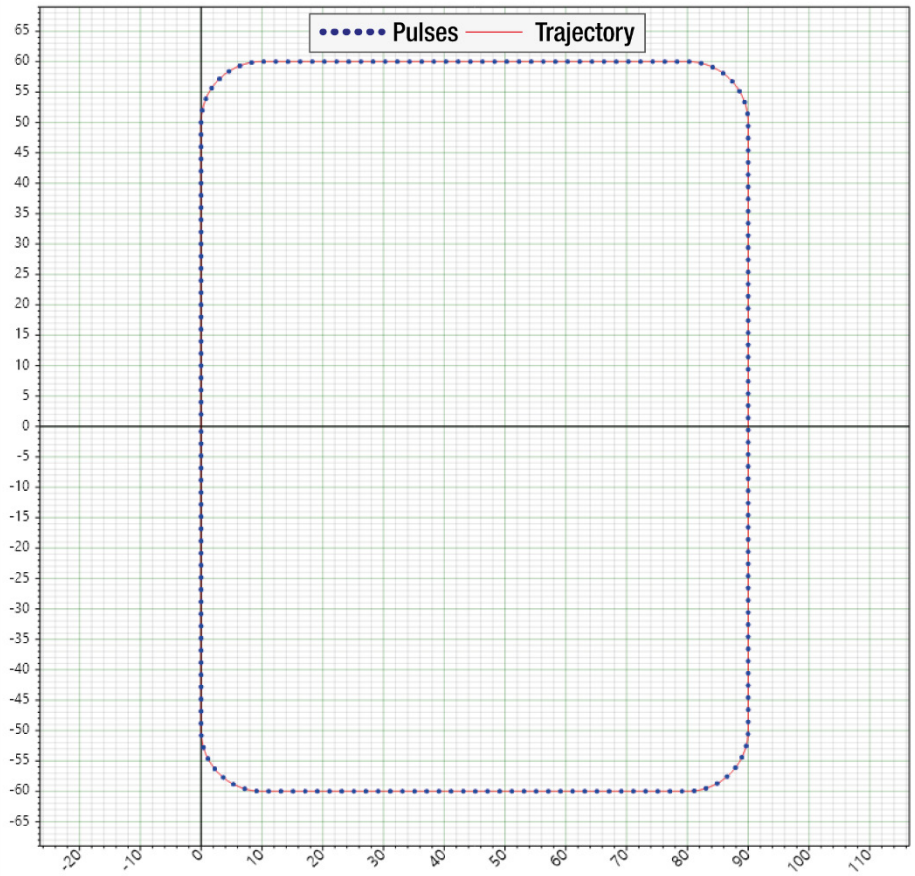
GroupMultiAxisPSOAbsoluteDisable(XY)

Disable pulse generation.

4.2 Trajectory LineArc without step values

Description:

This mode is used to generate regular pulses from a LineArc trajectory.



In this example, pulses occur along the LineArc trajectory at a distance of 2 mm between each other.
The distance between each pulse is defined once and for all by setting the step parameter of the **XYMultiAxisPSOAbsoluteLoadFromLineArc** API (Step parameter = 2 in the example).
The LineArc trajectory is defined in the following text file:

```
FirstTangent= 90; Degrees
DiscontinuityAngle= 0.01; Degrees

Line= 0, 50
Arc= 10, -90
Line= 80, 60
Arc= 10, -90
Line= 90, -50
Arc= 10, -90
Line= 10, -60
Arc= 10, -90
Line= 0, 0
```

Application (based on above example):

2 axes in XY group (Pulses generated every 2 mm):

GroupMultiAxisPSORelativeDisable(XY,2,Immediate)

GroupMultiAxisPSOAbsoluteDisable(XY)

Disable any pulse generation to avoid pulse generation during initialization

PositionerCompensatedFastPCOPulseParametersSet(XY.X,10,0,0)

Configure pulses parameters: Pulse width, Pulse polarity and Pulse toggle.

GroupMultiAxisPSOInterpolationFactorSet(XY,2,256,256)

Configure the PSO interpolation factor for the specified number of positioners in the group.

GroupMultiAxisPSOAbsoluteRadiusSet(XY,2,0.004,0.004)

Define a tolerance around the multi-axis PSO points.

GroupMoveAbsolute(XY,0,0)

Initialization to position (0,0).

XYMultiAxisPSOAbsoluteLoadFromLineArc(XY,Square120r10.linearc,2)

GroupMultiAxisPSOAbsolutePrepare(XY,0,0)

*Load the multi-axis PSO array from a LineArc trajectory file "Square120r10.linearc" and prepare controller taking into account the various position compensations/mappings to generate 2 mm pulses step distance (**step parameter set to 2**).*

XYLineArcVerification(XY,Square120r10.linearc)

XYLineArcVerificationResultGet(XY.X,char *,double *,double *,double *,double *)

Verify the execution of the LineArc trajectory file and return results

GroupMultiAxisPSOAbsoluteEnable(XY)

Enable pulses generation.

XYLineArcExecution(XY,Square120r10.linearc,50,1000,1)

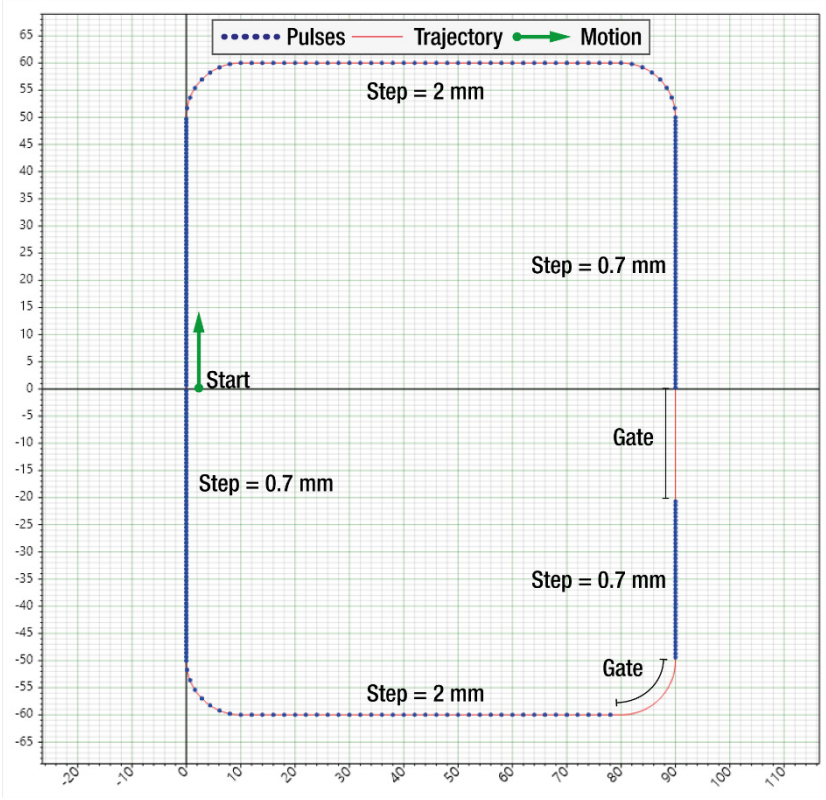
Execute the LineArc trajectory "Squrae120r10.linearc".

GroupMultiAxisPSOAbsoluteDisable(XY)

Disable pulse generation.

4.3 Trajectory LineArc with step values

This mode is used to generate different pulses from a LineArc trajectory.



In this example, pulses occur along the LineArc trajectory at a distance defined by step values in each command line of the trajectory file:

- 2 mm between each pulse from (0,50) to (90,50) and (80,-60) to (0,-50).
- 0.7 mm between each pulse from (0,-50) to (0,50), (90,50) to (90,0) and (90,-20) to (90,-50).
- No pulse (**Gates**) from (90,0) to (90,-20) and (90,-50) to (80,-60).

Gates are defined using **Line / Arc** commands instead of **LinePulse / ArcPulse** commands.
The step parameter of **XYMultiAxisPSOAbsoluteLoadFromLineArc API is set to 0**, allowing customized pulses.
The LineArc trajectory and steps values are defined in the following text file:

	FirstTangent= 90; Degrees DiscontinuityAngle= 0.01; Degrees
Pulse Step = 0.7 mm →	LinePulse = 0, 50, 0.7
Pulse Step = 2 mm →	ArcPulse = 10, -90, 2
…	LinePulse = 80, 60, 2
…	ArcPulse = 10, -90, 2
Pulse Step = 0.7 mm →	LinePulse = 90, 0, 0.7
Gate →	Line = 90, -20
Pulse Step = 0.7 mm →	LinePulse = 90, -50, 0.7
Gate →	Arc = 10, -90
Pulse Step = 2 mm →	LinePulse = 10, -60, 2
…	ArcPulse = 10, -90, 2
Pulse Step = 0.7 mm →	LinePulse = 0, 0, 0.7

Application (based on above example):

2 axes in XY group (Pulses generated alternately at 2 mm and 0.7 mm):

GroupMultiAxisPSORelativeDisable(XY,2,Immediate)

GroupMultiAxisPSOAbsoluteDisable(XY)

Disable any pulse generation to avoid pulse generation during initialization

PositionerCompensatedFastPCOPulseParametersSet(XY.X,10,0,0)

Configure pulses parameters: Pulse width, Pulse polarity and Pulse toggle.

GroupMultiAxisPSOInterpolationFactorSet(XY,2,256,256)

Configure the PSO interpolation factor for the specified number of positioners in the group.

GroupMultiAxisPSOAbsoluteRadiusSet(XY,2,0.004,0.004)

Define a tolerance around the multi-axis PSO points.

GroupMoveAbsolute(XY,0,0)

Initialization to position (0,0).

XYMultiAxisPSOAbsoluteLoadFromLineArc(XY,Square120r10.linearc,0)

GroupMultiAxisPSOAbsolutePrepare(XY,0,0)

*Load the multi-axis PSO array from the LineArc trajectory file "Square120r10_Step.linearc" and prepare controller taking into account the various position compensations/mappings to generate pulses step distance triggers defined in trajectory file (**step parameter set to 0**).*

XYLineArcVerification(XY,Square120r10.linearc)

XYLineArcVerificationResultGet(XY.X,char *,double *,double *,double *,double *)

Verify the execution of the LineArc trajectory file and return results

GroupMultiAxisPSOAbsoluteEnable(XY)

Enable pulses generation.

XYLineArcExecution(XY,Square120r10.linearc,50,1000,1)

Execute the LineArc trajectory "Squrae120r10_Step.linearc".

GroupMultiAxisPSOAbsoluteDisable(XY)

Disable pulse generation.

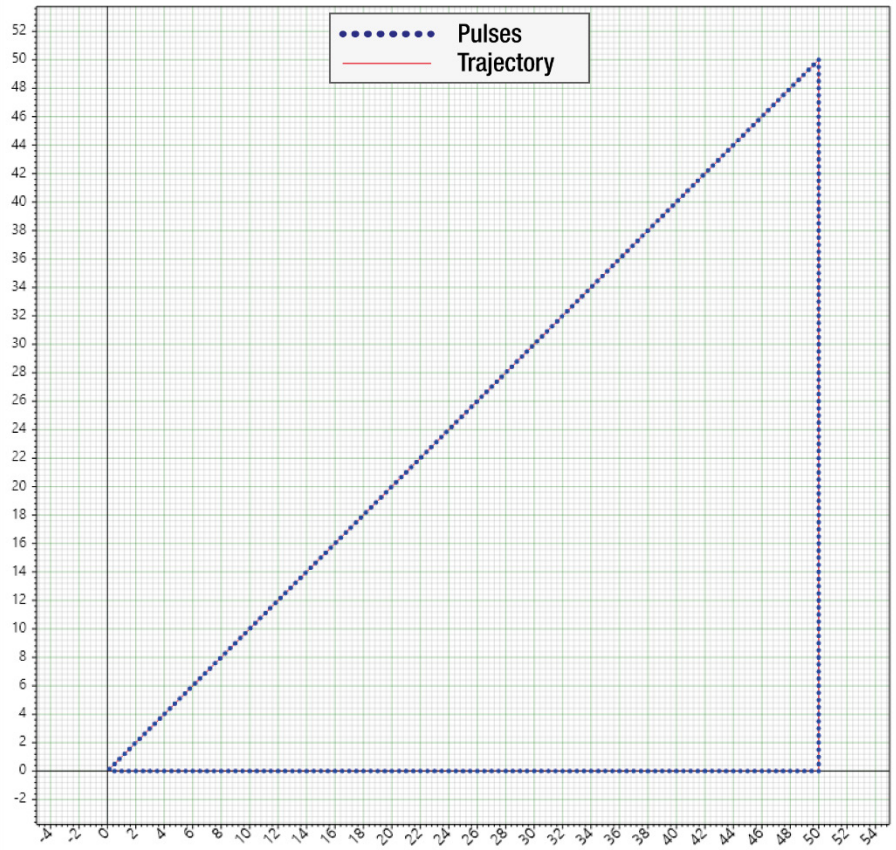
5 Relative PSO Mode

Relative PSO Mode generates trigger events based on the **incremental distance traveled** along the trajectory.

A pitch value defines the distance between successive trigger events, and each time the accumulated travel distance reaches this increment, a new trigger is generated.

5.1 On Absolute move

This mode is used to generate pulses on position along a trajectory or path with fixed distance intervals.



In this example, pulses occur along the trajectory at a distance of 0.5 mm between each other.

The distance between each pulse is defined once and for all by setting the step parameter of the **GroupMultiAxisPSORelativeEnable** API (Step parameter = 0.5 in the example).

Application (based on above example):

Script example for 2 axes in XY group or multiple axes group (Pulses generated every 0.5 mm):

GroupMultiAxisPSORelativeDisable(XY,2,Immediate)

GroupMultiAxisPSOAbsoluteDisable(XY)

Disable any pulse generation to avoid pulse generation during initialization

GroupMoveAbsolute(XY,0,0)

Initialization to position (0,0).

PositionerCompensatedFastPCOPulseParametersSet(XY.X,10,0,0)

Configure pulses parameters: Pulse width, Pulse polarity and Pulse toggle.

GroupMultiAxisPSOInterpolationFactorSet(XY,2,256,256)

Configure the PSO interpolation factor for the specified number of positioners in the group.

GroupMultiAxisPSORelativeEnable(XY,2,Immediate,0.5)

Enable the multi-axis relative PSO for specified number of positioners, mode “Immediate”, distance between pulses set to 0.5 mm)

GroupMoveAbsolute(XY,50,0)

Move to position (50, 0)

GroupMoveAbsolute(XY,50,50)

Move to position (50, 50)

GroupMoveAbsolute(XY,0,0)

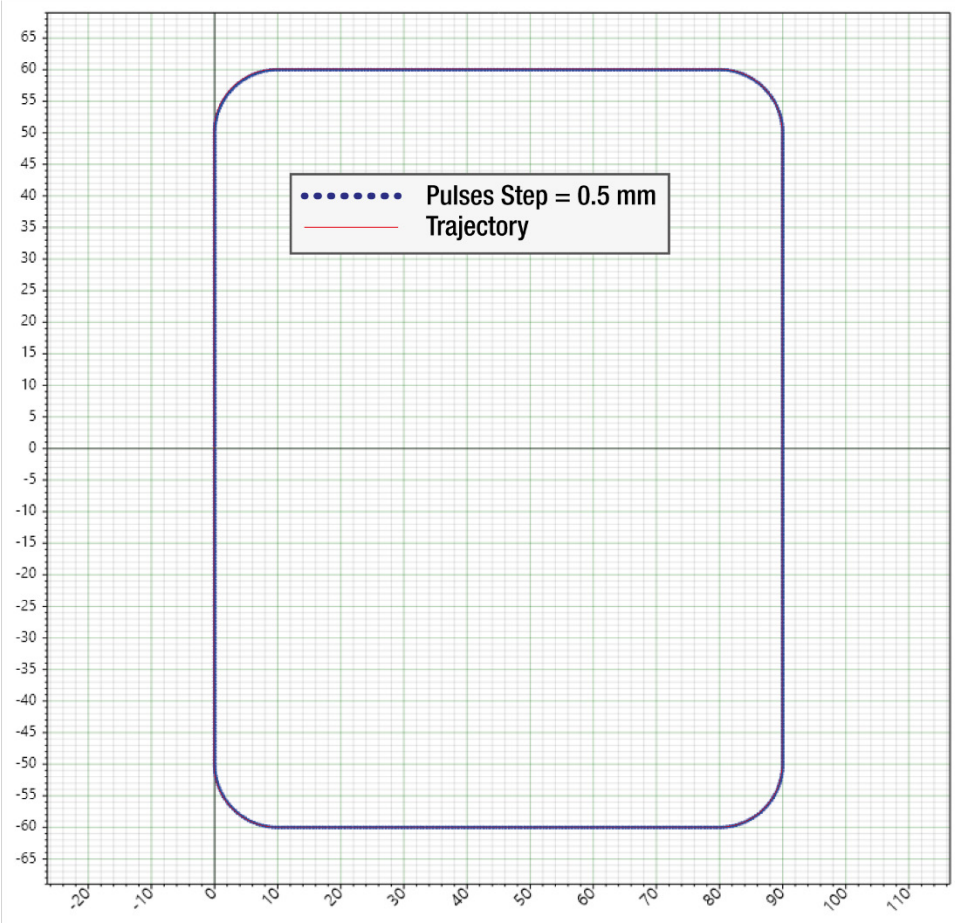
Move to position (0, 0)

GroupMultiAxisPSORelativeDisable(XY,2,Immediate)

Disable multi-axis PSO relative.

5.2 On Trajectory LineArc

This mode is used to generate pulses on XY Linear Arc trajectory.



In this example, pulses occur along the LineArc trajectory at a distance of 0.5 mm between each other.
The distance between each pulse is defined once and for all by setting the step parameter of the **GroupMultiAxisPSORelativeEnable** API (Step parameter = 0.5 in the example).
The LineArc trajectory is defined in the following text file:

```
FirstTangent= 90; Degrees
DiscontinuityAngle= 0.01; Degrees

Line =   0,   50
Arc  =  10,  -90
Line =  80,   60
Arc  =  10,  -90
Line =  90,  -50
Arc  =  10,  -90
Line =  10,  -60
Arc  =  10,  -90
Line =   0,    0
```

Application (based on above example):

Script example for pulses generated every 0.5 mm on XY Linear Arc trajectory:

GroupMultiAxisPSORelativeDisable(XY,2,Immediate)

GroupMultiAxisPSOAbsoluteDisable(XY)

Disable any pulse generation to avoid pulse generation during initialization

GroupMoveAbsolute(XY,0,0)

Initialization to position (0,0).

PositionerCompensatedFastPCOPulseParametersSet(XY.X,10,0,0)

Configure pulses parameters: Pulse width, Pulse polarity and Pulse toggle.

GroupMultiAxisPSOInterpolationFactorSet(XY,2,256,256)

Configure the PSO interpolation factor for the specified number of positioners in the group.

GroupMultiAxisPSORelativeEnable(XY,2,Immediate,0.5)

Enable the multi-axis relative PSO for specified number of positioners, mode “Immediate”, distance between pulses set to 0.5 mm)

XYLineArcVerification(XY,Square120r10.linearc)

XYLineArcVerificationResultGet(XY.X,char *,double *,double *,double *,double *)

XYLineArcVerificationResultGet(XY.Y,char *,double *,double *,double *,double *)

Verify the execution of the LineArc trajectory file and return results.

XYLineArcExecution(XY,Square120r10.linearc,50,1000,1)

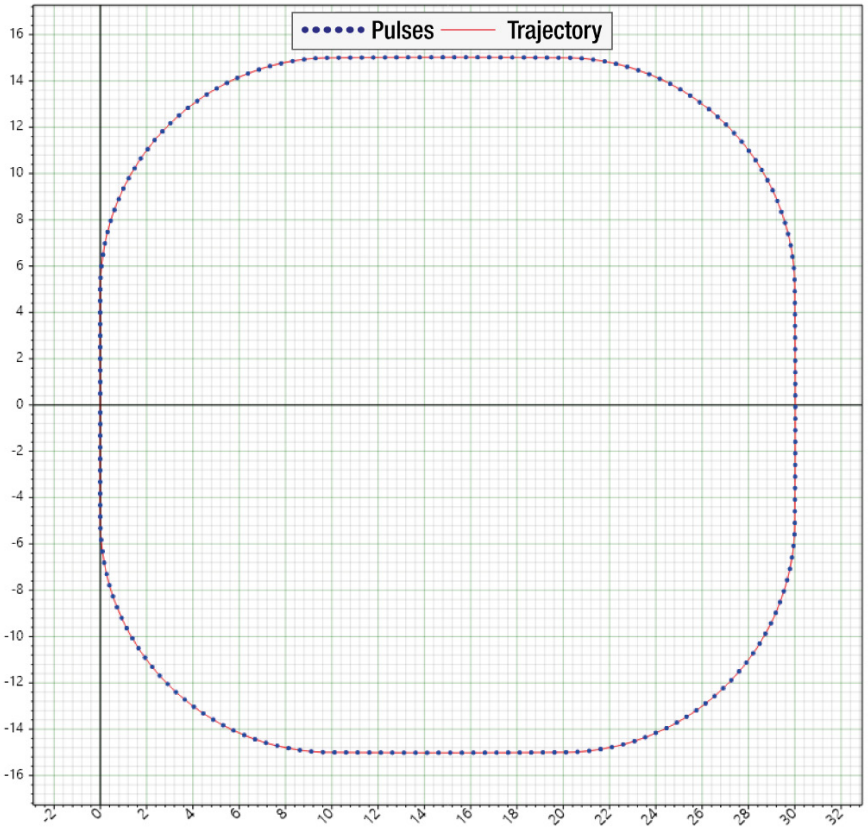
Execute the LineArc trajectory “Squrae120r10.linearc”.

GroupMultiAxisPSORelativeDisable(XY,2,Immediate)

Disable multi-axis PSO relative.

5.3 On Trajectory PT

This mode is used to generate pulses on XY PT trajectory.



In this example, pulses occur along the PT trajectory at a distance of 0.5 mm between each other.
The distance between each pulse is defined once and for all by setting the step parameter of the **GroupMultiAxisPSORelativeEnable** API (Step parameter = 0.5 in the example).
Refer to “XPS-D Features Manual” for description of the PT trajectory file structure.

Extract of PT trajectory file:

```
0.016722,  0.000000,  0.418060
0.091639,  0.000000,  4.581940
0.003452,  0.001490,  0.172606
0.003452,  0.004469,  0.172555
0.003452,  0.007447,  0.172452
0.003452,  0.010422,  0.172298
0.003452,  0.013395,  0.172092
0.003452,  0.016363,  0.171836
0.003452,  0.019327,  0.171527
0.003452,  0.022284,  0.171168
0.003452,  0.025235,  0.170758
0.003452,  0.028179,  0.170297
0.003452,  0.031114,  0.169785
0.003452,  0.034040,  0.169223
0.003452,  0.036956,  0.168610
0.003452,  0.039861,  0.167947
0.003452,  0.042754,  0.167234
0.003452,  0.045634,  0.166471
   …          …          …
```

Application (based on above example):

Script example for pulses generated every 0.5 mm on PT trajectory:

GroupMultiAxisPSORelativeDisable(XY,2,Immediate)

GroupMultiAxisPSOAbsoluteDisable(XY)

Disable any pulse generation to avoid pulse generation during initialization

GroupMoveAbsolute(XY,0,0)

Initialization to position (0,0).

PositionerCompensatedFastPCOPulseParametersSet(XY.X,10,0,0)

Configure pulses parameters: Pulse width, Pulse polarity and Pulse toggle.

GroupMultiAxisPSOInterpolationFactorSet(XY,2,256,256)

Configure the PSO interpolation factor for the specified number of positioners in the group.

GroupMultiAxisPSORelativeEnable(XY,2,Immediate,0.5)

Enable the multi-axis relative PSO for specified number of positioners, mode “Immediate”, distance between pulses set to 0.5 mm)

MultiplesAxesPTVerification(XY,trajectory.pt)

MultiplesAxesPTVerificationResultGet(XY.X,char *,double *,double *,double *,double *)

MultiplesAxesPTVerificationResultGet(XY.Y,char *,double *,double *,double *,double *)

Verify the execution of the PT trajectory file and return results.

MultiplesAxesPTExecution(XY,trajectory.pt,1)

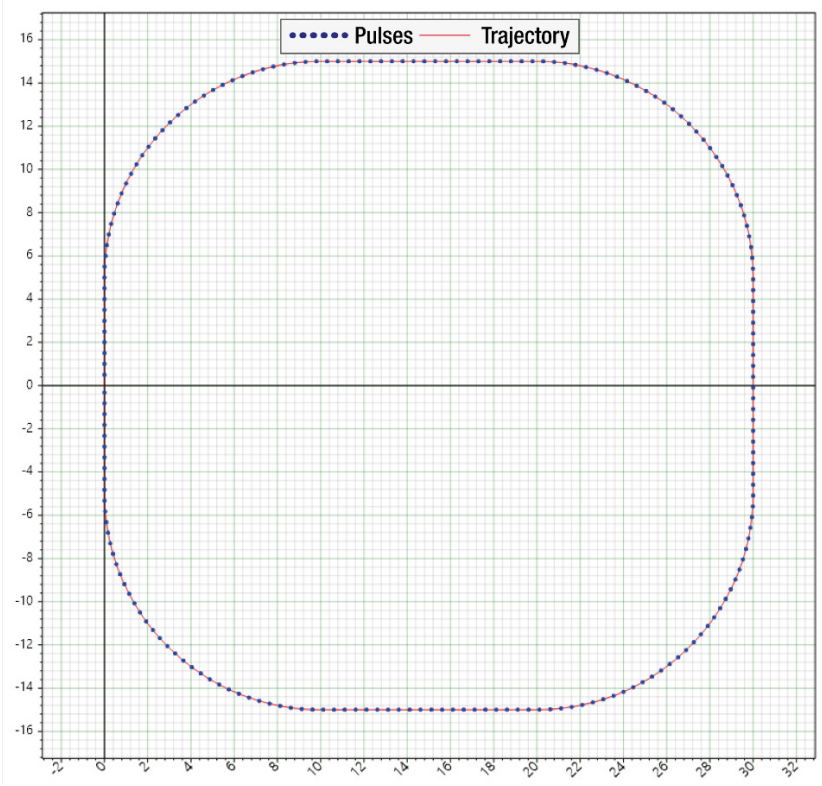
Execute once the PT trajectory “trajectory.pt"

GroupMultiAxisPSORelativeDisable(XY,2, Immediate)

Disable multi-axis PSO relative.

5.4 On Trajectory PVT

This mode is used to generate pulses on XY PVT trajectory.



In this example, pulses occur along the PVT trajectory at a distance of 0.5 mm between each other.
The distance between each pulse is defined once and for all by setting the step parameter of the **GroupMultiAxisPSORelativeEnable** API (Step parameter = 0.5 in the example).
Refer to “XPS-D Features Manual” for description of the PVT trajectory file structure.

Extract of PVT trajectory file:

```
0.016722,  0.000000,  0.000000,  0.418060,  50.000000
0.091639,  0.000000,  0.000000,  4.581940,  50.000000
0.003452,  0.001490,  0.863032,  0.172606,  49.992551
0.003452,  0.004469,  1.725807,  0.172555,  49.970207
0.003452,  0.007447,  2.588068,  0.172452,  49.932974
0.003452,  0.010422,  3.449557,  0.172298,  49.880864
0.003452,  0.013395,  4.310019,  0.172092,  49.813891
0.003452,  0.016363,  5.169197,  0.171836,  49.732076
0.003452,  0.019327,  6.026834,  0.171527,  49.635444
0.003452,  0.022284,  6.882676,  0.171168,  49.524022
0.003452,  0.025235,  7.736467,  0.170758,  49.397845
0.003452,  0.028179,  8.587953,  0.170297,  49.256949
0.003452,  0.031114,  9.436880,  0.169785,  49.101378
0.003452,  0.034040,  10.282995,  0.169223,  48.931176
0.003452,  0.036956,  11.126047,  0.168610,  48.746396
0.003452,  0.039861,  11.965783,  0.167947,  48.547091
   …          …          …          …          …
```

Application (based on above example):

Script example for pulses generated every 0.5 mm on PVT trajectory:

GroupMultiAxisPSORelativeDisable(XY,2,Immediate)

GroupMultiAxisPSOAbsoluteDisable(XY)

Disable any pulse generation to avoid pulse generation during initialization

GroupMoveAbsolute(XY,0,0)

Initialization to position (0,0).

PositionerCompensatedFastPCOPulseParametersSet(XY.X,10,0,0)

Configure pulses parameters: Pulse width, Pulse polarity and Pulse toggle.

GroupMultiAxisPSOInterpolationFactorSet(XY,2,256,256)

Configure the PSO interpolation factor for the specified number of positioners in the group.

GroupMultiAxisPSORelativeEnable(XY,2,Immediate,0.5)

Enable the multi-axis relative PSO for specified number of positioners, mode “Immediate”, distance between pulses set to 0.5 mm)

MultiplesAxesPVTVerification(XY,trajectory.pvt)

MultiplesAxesPVTVerificationResultGet(XY.X,char *,double *,double *,double *,double *)

MultiplesAxesPVTVerificationResultGet(XY.Y,char *,double *,double *,double *,double *)

Verify the execution of the PT trajectory file and return results.

MultiplesAxesPVTExecution(XY,trajectory.pvt,1)

Execute once the PVT trajectory “trajectory.pvt"

GroupMultiAxisPSORelativeDisable(XY,2, Immediate)

Disable multi-axis PSO relative.

6 New APIs

6.1 GroupMultiAxisPSOInterpolationFactorSet

Name

GroupMultiAxisPSOInterpolationFactorSet – *Set the multi-axis PSO interpolation factor for each positioner*.

Input tests

Refer to XPS Unified – Programmer's Manual section Input Tests Common to all XPS Functions.

- Check the group exists: ERR_GROUP_NAME.
- Check PCOExtended feature is enabled in configuration: ERR_NOT_ALLOWED_ACTION
- Check version of PSO module embedded in CIE Board is compatible (>=2.0.0): ERR_PSO_MODULE_VERSION_OF_CIE_BOARD_NOT_COMPATIBLE
- Check values for interpolation factor are correct: ERR_PARAMETER_OUT_OF_RANGE

Description

This API sets the PSO interpolation factor for the specified number of positioners in the group.
The PSO interpolation factors available are:

- 4
- 16
- 256
- 4096
- 65536

Prototype

```
int GroupMultiAxisPSOInterpolationFactorSet(
   int SocketID,
   char* GroupName,
   int PositionerNumber,
   int[PositionerNumber] PSOInterpolationFactor
   )
```

Input parameters

SocketID	int	Socket identifier gets by the TCP_ConnectToServer function.
GroupName	char*	The name of the group.
PositionerNumber	int	Number of positioners of the group to configure.
PSOInterpolationFactor	int[]	PSO Interpolation factor for each positioner. Number of elements should be equal to PositionerNumber parameter.

Output parameters

None

Return (In addition to the results of "Error Function List")

SUCCESS	0	
ERR_PARAMETER_OUT_OF_RANGE	-17	Parameter out of range or not expected
ERR_GROUP_NAME	-19	GroupName doesn't exist or unknown command
ERR_NOT_ALLOWED_ACTION	-22	Not allowed action
ERR_PSO_MODULE_VERSION_OF_CIE_BOARD_NOT_COMPATIBLE	-150	PSO module of the CIE board is not compatible with multi-axis PSO

6.2 GroupMultiAxisPSOInterpolationFactorGet

Name

GroupMultiAxisPSOInterpolationFactorGet – *Get the multi-axis PSO interpolation factor for each positioner*.

Input tests

Refer to XPS Unified – Programmer's Manual section Input Tests Common to all XPS Functions.

- Check the group exists: ERR_GROUP_NAME.
- Check PCOExtended feature is enabled in configuration: ERR_NOT_ALLOWED_ACTION
- Check version of PSO module embedded in CIE Board is compatible (>=2.0.0): ERR_PSO_MODULE_VERSION_OF_CIE_BOARD_NOT_COMPATIBLE

Description

This API gets the current PSO interpolation factor for the specified number of positioners in the group.

Prototype

```
int GroupMultiAxisPSOInterpolationFactorGet(
   int SocketID,
   char* GroupName,
   int PositionerNumber,
   int[PositionerNumber] *PSOInterpolationFactor
   )
```

Input parameters

SocketID	int	Socket identifier gets by the TCP_ConnectToServer function.
GroupName	char*	The name of the group.
PositionerNumber	int	Number of positioners of the group we want to configure.

Output parameters

PSOInterpolationFactor	int[]	PSO Interpolation factor for each positioner. Number of elements should be equal to PositionerNumber parameter.

Return (In addition to the results of "Error Function List")

SUCCESS	0	
ERR_GROUP_NAME	-19	GroupName doesn't exist or unknown command
ERR_NOT_ALLOWED_ACTION	-22	Not allowed action
ERR_PSO_MODULE_VERSION_OF_CIE_BOARD_NOT_COMPATIBLE	-150	PSO module of the CIE board is not compatible with multi-axis PSO

6.3 GroupMultiAxisPSOAbsoluteRadiusSet

Name

GroupMultiAxisPSOAbsoluteRadiusSet – *Set the multi-axis PSO absolute radius for each positioner*.

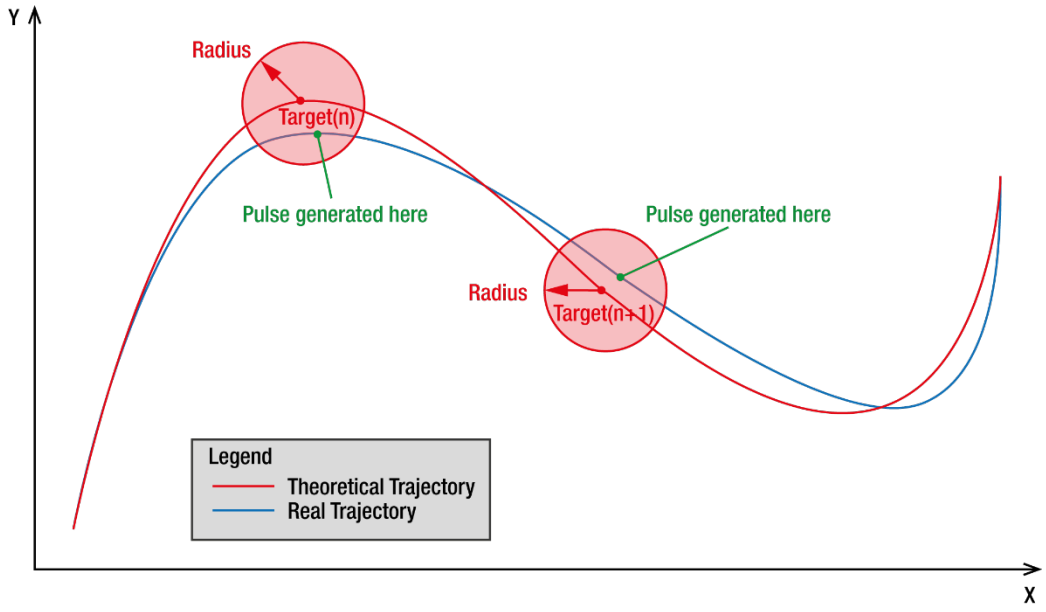
Input tests

Refer to XPS Unified – Programmer's Manual section Input Tests Common to all XPS Functions.

- Check the group exists: ERR_GROUP_NAME.
- Check PCOExtended feature is enabled in configuration: ERR_NOT_ALLOWED_ACTION
- Check version of PSO module embedded in CIE Board is compatible (>=2.0.0): ERR_PSO_MODULE_VERSION_OF_CIE_BOARD_NOT_COMPATIBLE
- Check values for radius are higher than 0 and lower than maximum value (see description to check if radius values are correct): ERR_PARAMETER_OUT_OF_RANGE

Description

Used only in case of multi-axis absolute PSO. Indeed, in that case unlike relative multi-axis PSO, multi-axis PSO points are strict positions. Therefore, if the positioners do not go through the exact configured positions there will be no pulses. This API is to define a tolerance around the multi-axis PSO points.



Below the method to check if the selected radiuses are ok:
radiusScaled_Axis1 = radius_Axis1 * interpolationFactor_Axis1 / encoderScalePitch_Axis1
radiusScaled_Axis2 = radius_Axis2 * interpolationFactor_Axis2 / encoderScalePitch_Axis2
radiusScaled_Axis3 = radius_Axis3 * interpolationFactor_Axis3 / encoderScalePitch_Axis3
radiusScaled_Axis4 = radius_Axis4 * interpolationFactor_Axis4 / encoderScalePitch_Axis4
C_Value_Axis1 = 256 * radiusScaled_Axis2 * radiusScaled_Axis3 * radiusScaled_Axis4
C_Value_Axis2 = 256 * radiusScaled_Axis1 * radiusScaled_Axis3 * radiusScaled_Axis4
C_Value_Axis3 = 256 * radiusScaled_Axis1 * radiusScaled_Axis2 * radiusScaled_Axis4
C_Value_Axis4 = 256 * radiusScaled_Axis1 * radiusScaled_Axis2 * radiusScaled_Axis3
Take the maximum value between C_Value_AxisN values.
ScalingCoeff = 2^17 / max_C_Value
ScalingCoeff has to be > 1 else to keep the current interpolation factors and encoder scale pitch the radius value for 1 or several axes must be reduced.

Prototype

```
int GroupMultiAxisPSOAbsoluteRadiusSet(
   int SocketID,
   char* GroupName,
   int PositionerNumber,
   double[PositionerNumber] Radius
   )
```

Input parameters

SocketID	int	Socket identifier gets by the TCP_ConnectToServer function.
GroupName	char*	The name of the group.
PositionerNumber	int	Number of positioners of the group we want to configure.
Radius	double[]	Radius for each positioner. Number of elements should be equal to PositionerNumber parameter.

Output parameters

None

Return (In addition to the results of “Error Function List”)

SUCCESS	0	
ERR_PARAMETER_OUT_OF_RANGE	-17	Parameter out of range or not expected
ERR_GROUP_NAME	-19	GroupName doesn't exist or unknown command
ERR_NOT_ALLOWED_ACTION	-22	Not allowed action
ERR_PSO_MODULE_VERSION_OF_CIE_BOARD_NOT_COMPATIBLE	-150	PSO module of the CIE board is not compatible with multi-axis PSO

6.4 GroupMultiAxisPSOAbsoluteRadiusGet

Name

GroupMultiAxisPSOAbsoluteRadiusGet – *Get the multi-axis absolute radius for each positioner*.

Input tests

Refer to XPS Unified – Programmer's Manual section Input Tests Common to all XPS Functions.

- Check the group exists: ERR_GROUP_NAME.
- Check PCOExtended feature is enabled in configuration: ERR_NOT_ALLOWED_ACTION
- Check version of PSO module embedded in CIE Board is compatible (>=2.0.0): ERR_PSO_MODULE_VERSION_OF_CIE_BOARD_NOT_COMPATIBLE

Description

This API gets the current PSO radius for the specified number of positioners in the group.

Prototype

```
int GroupMultiAxisPSOAbsoluteRadiusGet(
   int SocketID,
   char* GroupName,
   int PositionerNumber,
   double[PositionerNumber] *Radius
   )
```

Input parameters

SocketID	int	Socket identifier gets by the TCP_ConnectToServer function.
GroupName	char*	The name of the group.
PositionerNumber	int	Number of positioners of the group we want to configure.

Output parameters

Radius	double[]	Radius for each positioner. Number of elements should be equal to PositionerNumber parameter.

Return (In addition to the results of "Error Function List")

SUCCESS	0	
ERR_GROUP_NAME	-19	GroupName doesn't exist or unknown command
ERR_NOT_ALLOWED_ACTION	-22	Not allowed action
ERR_PSO_MODULE_VERSION_OF_CIE_BOARD_NOT_COMPATIBLE	-150	PSO module of the CIE board is not compatible with multi-axis PSO

6.5 GroupMultiAxisPSORelativeEnable

Name

GroupMultiAxisPSORelativeEnable – *Enable the multi-axis PSO relative*.

Input tests

Refer to XPS Unified – Programmer's Manual section Input Tests Common to all XPS Functions.

- Check the group exists: ERR_GROUP_NAME.
- Check PCOExtended feature is enabled in configuration: ERR_NOT_ALLOWED_ACTION
- Check version of PSO module embedded in CIE Board is compatible (>=2.0.0): ERR_PSO_MODULE_VERSION_OF_CIE_BOARD_NOT_COMPATIBLE
- Check interpolation factor and encoder scale pitch are not equal to 0: ERR_PARAMETER_OUT_OF_RANGE.

Description

This API enables the multi-axis relative PSO for specified number of positioners.
The distance between each multi-axis PSO pulses is defined by the step value.
This API work with mode set to "Immediate": start multi-axis PSO relative immediately.
Other modes are currently being developed and will be available later.

Prototype

```
int GroupMultiAxisPSORelativeEnable(
   int SocketID,
   char* GroupName,
   int PositionerNumber,
   char* Mode,
   double Step
   )
```

Input parameters

SocketID	int	Socket identifier gets by the TCP_ConnectToServer function.
GroupName	char*	The name of the group.
PositionerNumber	int	Number of positioners of the group we want to configure.
Mode	Char*	The mode.
Step	double	Distance between each multi-axis PSO pulses.

Output parameters

None

Return (In addition to the results of “Error Function List”)		
SUCCESS	0	
ERR_PARAMETER_OUT_OF_RANGE	-17	Parameter out of range or not expected
ERR_GROUP_NAME	-19	GroupName doesn't exist or unknown command
ERR_NOT_ALLOWED_ACTION	-22	Not allowed action
ERR_PSO_MODULE_VERSION_OF_CIE_BOARD_NOT_COMPATIBLE	-150	PSO module of the CIE board is not compatible with multi-axis PSO

6.6 GroupMultiAxisPSORelativeDisable

Name

GroupMultiAxisPSORelativeDisable – *Disable the multi-axis PSO relative*.

Input tests

Refer to XPS Unified – Programmer's Manual section Input Tests Common to all XPS Functions.

- Check the group exists: ERR_GROUP_NAME.
- Check PCOExtended feature is enabled in configuration: ERR_NOT_ALLOWED_ACTION
- Check version of PSO module embedded in CIE Board is compatible (>=2.0.0): ERR_PSO_MODULE_VERSION_OF_CIE_BOARD_NOT_COMPATIBLE

Description

This API disables the multi-axis relative PSO for specified number of positioners.
This API work with mode set to "Immediate": stop multi-axis PSO relative immediately.
Other modes are currently being developed and will be available later.

Prototype

```
int GroupMultiAxisPSORelativeDisable(
   int SocketID,
   char* GroupName,
   int PositionerNumber,
   char* Mode
   )
```

Input parameters

SocketID	int	Socket identifier gets by the TCP_ConnectToServer function.
GroupName	char*	The name of the group.
PositionerNumber	int	Number of positioners of the group we want to configure.
Mode	Char*	The mode.

Output parameters

None

Return (In addition to the results of "Error Function List")

SUCCESS	0	
ERR_GROUP_NAME	-19	GroupName doesn't exist or unknown command
ERR_NOT_ALLOWED_ACTION	-22	Not allowed action
ERR_PSO_MODULE_VERSION_OF_CIE_BOARD_NOT_COMPATIBLE	-150	PSO module of the CIE board is not compatible with multi-axis PSO

6.7 GroupMultiAxisPSOAbsoluteLoadFromFile

Name

GroupMultiAxisPSOAbsoluteLoadFromFile – *Load a multi-axis PSO array from a file*.

Input tests

Refer to XPS Unified – Programmer's Manual section Input Tests Common to all XPS Functions.

- Check the group exists: ERR_GROUP_NAME.
- Check PCOExtended feature is enabled in configuration: ERR_NOT_ALLOWED_ACTION
- Check version of PSO module embedded in CIE Board is compatible (>=2.0.0): ERR_PSO_MODULE_VERSION_OF_CIE_BOARD_NOT_COMPATIBLE
- Check number of positioners is lower or equal than maximum positioners for multi-axis PSO: ERR_PARAMETER_OUT_OF_RANGE
- Check multi axis PSO array file exists: ERR_READ_FILE
- Check format of the data in the file is correct: ERR_TRAJ_BAD_FORMAT
- Check the number of lines in the file is lower than the maximum number of PSO points: ERR_CHECK_DATA_INCORRECT

Description

This API loads the multi-axis PSO array from a file. The file should have as many columns as the specified number of positioners.

Example of file:

```
3.000       1.000
6.000       2.000
9.000       3.000
12.000      4.000
15.000      5.000
18.000      6.000
21.000      7.000
```

The 5th multi axis PSO point is at position (15,5).

Prototype

```
int GroupMultiAxisPSOAbsoluteLoadFromFile(
   int SocketID,
   char* GroupName,
   int PositionerNumber,
   char* FileName
   )
```

Input parameters

SocketID	int	Socket identifier gets by the TCP_ConnectToServer function.
GroupName	char*	The name of the group.
PositionerNumber	int	Number of positioners of the group we want to configure.
FileName	char*	The relative path to the file from /Admin/Public/trajectories/.

Output parameters

None

Return (In addition to the results of “Error Function List”)		
SUCCESS	0	
ERR_PARAMETER_OUT_OF_RANGE	-17	Parameter out of range or not expected
ERR_GROUP_NAME	-19	GroupName doesn't exist or unknown command
ERR_NOT_ALLOWED_ACTION	-22	Not allowed action
ERR_READ_FILE	-61	Error file corrupt or file doesn't exist
ERR_TRAJ_BAD_FORMAT	-77	Bad trajectory line format
ERR_CHECK_DATA_INCORRECT	-122	Data incorrect (wrong value, wrong format, wrong order or inexistent)
ERR_PSO_MODULE_VERSION_OF_CIE_BOARD_NOT_COMPATIBLE	-150	PSO module of the CIE board is not compatible with multi-axis PSO

6.8 XYMultiAxisPSOAbsoluteLoadFromLineArc

Name

XYMultiAxisPSOAbsoluteLoadFromLineArc – *Load a multi-axis PSO array from a LineArc trajectory*.

Input tests

Refer to XPS Unified – Programmer's Manual section Input Tests Common to all XPS Functions.

- Check the group exists: ERR_GROUP_NAME.
- Check PCOExtended feature is enabled in configuration: ERR_NOT_ALLOWED_ACTION
- Check version of PSO module embedded in CIE Board is compatible (>=2.0.0): ERR_PSO_MODULE_VERSION_OF_CIE_BOARD_NOT_COMPATIBLE
- Check step parameter is not lower than 0: ERR_PARAMETER_OUT_OF_RANGE
- Check LineArc trajectory file exists or wrong format: ERR_READ_FILE
- Check number of elements in the LineArc trajectory does not exceed 100 000: ERR_FILE_TOO_BIG
- In case of LinePulse or ArcPulse elements, checks that the step value (from the file) is not lower or equal 0: ERR_PARAMETER_OUT_OF_RANGE
- Check the number of lines in the file is lower than the maximum number of PSO points: ERR_CHECK_DATA_INCORRECT

Description

This API loads the multi-axis PSO array from a LineArc trajectory.

If the step parameter from the API prototype is higher than 0, then it will create a multi-axis PSO points array using this value as distance between each multi-axis PSO point that follows the trajectory path.

If the step parameter from the API prototype is equal to 0, it will use step values from the LineArc trajectory. To do that, there are 2 new elements from regular LineArc trajectory:

- LinePulse: the same as "Line" element but with a 3rd value for the step.
- ArcPulse: the same as "Arc" element but with a 3rd value for the step.

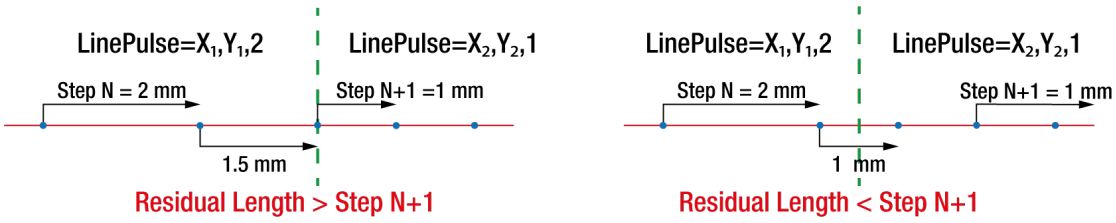
The step values defined in the LineArc trajectory allows to have different distance between pulse depending on the current element. If you don't want to pulse in the middle of your LineArc trajectory, you can simply use Line or Arc elements.

Example of file:

```
FirstTangent= 90; Degrees
DiscontinuityAngle= 0.01; Degrees

LinePulse=   0, 20, 2
Line=   0, 40
LinePulse= 0, 50, 2
Arc=   10, -90
LinePulse= 80, 60, 2
ArcPulse=   10, -90, 2
Line= 90, -50
ArcPulse=   10, -90, 2
Line= 10, -60
ArcPulse=   10, -90, 2
Line=   0, 0
```

It is possible that a trajectory's element can't contain an integer number of pulses. In that case there are 2 behaviors:

- If the step value of the next trajectory element is smaller than the residual length, then the next pulse will be at the start of the next trajectory element.
- If the step value of next trajectory element is bigger than the residual length, then the next pulse will be at position of the previous pulse + the new step value



Prototype

```
int XYMultiAxisPSOAbsoluteLoadFromLineArc(
   int SocketID,
   char* GroupName,
   int PositionerNumber,
   char* FileName,
   double Step
   )
```

Input parameters

SocketID	int	Socket identifier gets by the TCP_ConnectToServer function.
GroupName	char*	The name of the group.
PositionerNumber	int	Number of positioners of the group we want to configure.
FileName	char*	The relative path to the file from /Admin/Public/trajectories/.
Step	double	Distance between each multi-axis PSO pulses.

Output parameters

None

Return (In addition to the results of “Error Function List”)

SUCCESS	0	
ERR_PARAMETER_OUT_OF_RANGE	-17	Parameter out of range or not expected
ERR_GROUP_NAME	-19	GroupName doesn't exist or unknown command
ERR_NOT_ALLOWED_ACTION	-22	Not allowed action
ERR_READ_FILE	-61	Error file corrupt or file doesn't exist
ERR_TRAJ_BAD_FORMAT	-77	Bad trajectory line format
ERR_CHECK_DATA_INCORRECT	-122	Data incorrect (wrong value, wrong format, wrong order or inexistent)
ERR_PSO_MODULE_VERSION_OF_CIE_BOARD_NOT_COMPATIBLE	-150	PSO module of the CIE board is not compatible with multi-axis PSO

6.9 GroupMultiAxisPSOAbsolutePrepare

Name

GroupMultiAxisPSOAbsolutePrepare – *Prepare multi-axis absolute PSO from loaded datas.*

Input tests

Refer to XPS Unified – Programmer's Manual section Input Tests Common to all XPS Functions.

- Check the group exists: ERR_GROUP_NAME.
- Check PCOExtended feature is enabled in configuration: ERR_NOT_ALLOWED_ACTION
- Check version of PSO module embedded in CIE Board is compatible (>=2.0.0): ERR_PSO_MODULE_VERSION_OF_CIE_BOARD_NOT_COMPATIBLE
- Check number of multi-axis PSO points is higher than 0: ERR_CHECK_DATA_INCORRECT

Description

This API sends loaded multi-axis absolute PSO points to the CIEBoard.
Therefore, this API must be used after **GroupMultiAxisPSOAbsoluteLoadFromFile** or **XYMultiAxisPSOAbsoluteLoadFromLineArc** APIs and before **GroupMultiAxisPSOAbsoluteEnable** API.
You have to define as much Start parameter values as you have defined the multi-axis PSO positioners.
These parameters allow to add an offset to the loaded multi-axis PSO point.

Prototype

```
int GroupMultiAxisPSOAbsolutePrepare(
   int SocketID,
   char* GroupName,
   double[NbPositioner] Start
   )
```

Input parameters

SocketID	int	Socket identifier gets by the TCP_ConnectToServer function.
GroupName	char*	The name of the group.
Start	double[]	Start position for each positioner.

Output parameters

None

Return (In addition to the results of "Error Function List")

SUCCESS	0	
ERR_GROUP_NAME	-19	GroupName doesn't exist or unknown command
ERR_NOT_ALLOWED_ACTION	-22	Not allowed action
ERR_CHECK_DATA_INCORRECT	-122	Data incorrect (wrong value, wrong format, wrong order or inexistent)
ERR_PSO_MODULE_VERSION_OF_CIE_BOARD_NOT_COMPATIBLE	-150	PSO module of the CIE board is not compatible with multi-axis PSO

6.10 GroupMultiAxisPSOAbsoluteEnable

Name

GroupMultiAxisPSOAbsoluteEnable – *Enable multi-axis absolute PSO.*

Input tests

Refer to XPS Unified – Programmer's Manual section Input Tests Common to all XPS Functions.

- Check the group exists: ERR_GROUP_NAME.
- Check PCOExtended feature is enabled in configuration: ERR_NOT_ALLOWED_ACTION
- Check version of PSO module embedded in CIE Board is compatible (>=2.0.0): ERR_PSO_MODULE_VERSION_OF_CIE_BOARD_NOT_COMPATIBLE

Description

This API enables the multi-axis absolute PSO.

Prototype

```
int GroupMultiAxisPSOAbsoluteEnable(
   int SocketID,
   char* GroupName
   )
```

Input parameters

SocketID	int	Socket identifier gets by the TCP_ConnectToServer function.
GroupName	char*	The name of the group.

Output parameters

None

Return (In addition to the results of "Error Function List")

SUCCESS	0	
ERR_GROUP_NAME	-19	GroupName doesn't exist or unknown command
ERR_NOT_ALLOWED_ACTION	-22	Not allowed action
ERR_PSO_MODULE_VERSION_OF_CIE_BOARD_NOT_COMPATIBLE	-150	PSO module of the CIE board is not compatible with multi-axis PSO

6.11 GroupMultiAxisPSOAbsoluteDisable

Name

GroupMultiAxisPSOAbsoluteDisable – *Enable multi-axis absolute PSO.*

Input tests

Refer to XPS Unified – Programmer's Manual section Input Tests Common to all XPS Functions.

- Check the group exists: ERR_GROUP_NAME.
- Check PCOExtended feature is enabled in configuration: ERR_NOT_ALLOWED_ACTION
- Check version of PSO module embedded in CIE Board is compatible (>=2.0.0): ERR_PSO_MODULE_VERSION_OF_CIE_BOARD_NOT_COMPATIBLE

Description

This API disables the multi-axis absolute PSO.

Prototype

```
int GroupMultiAxisPSOAbsoluteDisable(
   int SocketID,
   char* GroupName
   )
```

Input parameters

SocketID	int	Socket identifier gets by the TCP_ConnectToServer function.
GroupName	char*	The name of the group.

Output parameters

None

Return (In addition to the results of "Error Function List")

SUCCESS	0	
ERR_GROUP_NAME	-19	GroupName doesn't exist or unknown command
ERR_NOT_ALLOWED_ACTION	-22	Not allowed action
ERR_PSO_MODULE_VERSION_OF_CIE_BOARD_NOT_COMPATIBLE	-150	PSO module of the CIE board is not compatible with multi-axis PSO

6.12 PositionerCompensatedFastPCOPulseParametersSet

Name

PositionerCompensatedFastPCOPulseParametersSet – *Sets pulse configuration to PSO.*

Input tests

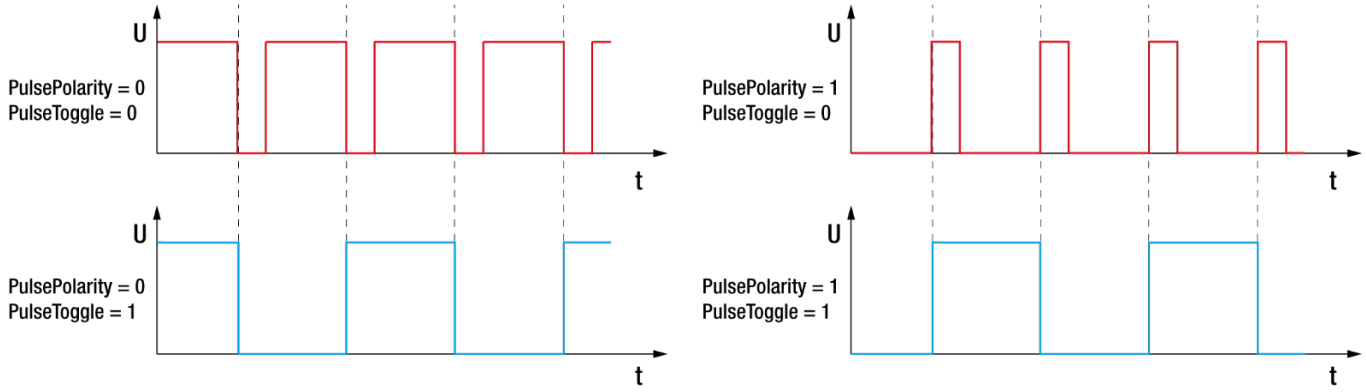
Refer to XPS Unified – Programmer's Manual section Input Tests Common to all XPS Functions.

- Checks the positioner name: (-18)
- Checks pulses generation status: (-22)
- Checks if CIEFAST compensated PCO pulses generation is supported: (-115)

Description

This function allows to configure pulses parameters:

- **PulseWidth**: width of pulse in µs, possible values are: 35 ns to 327.68 µs (5 ns resolution). default value is set to 2 µs.
- **PulsePolarity**: 0 for negative pulse, 1 for positive pulse.
- **PulseToggle**: switch to Toggle mode instead of Pulse mode when set to 1.



Successive trigger pulses should have a minimum time lag of 625 ns (200 ns for less than 4096 pulses).

Prototype

```
int PositionerCompensatedFastPCOPulseParametersSet(
   int SocketID,
   char * PositionerName,
   double * PulseWidth,
   int * PulsePolarity,
   bool * PulseToggle
   )
```

Input parameters

SocketID	int	Socket identifier gets by the TCP_ConnectToServer function.
PositionerName	char *	Positioner name
PulseWidth	double *	Width of pulse enable signal (µs).
PulsePolarity	int *	0 for negative pulse, 1 for positive pulse
PulseToggle	bool *	Toggle mode = 1 Pulse mode = 0

Output parameters

None.

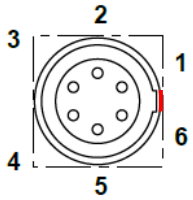
Return (In addition to the results of “Error Function List”)

SUCCESS	0	No error
ERR_WRONG_OBJECT_TYPE	-8	Wrong object type for this command
ERR_POSITIONER_NAME	-18	Positioner Name doesn't exist or unknown command
ERR_NOT_ALLOWED_ACTION	-22	Not allowed action
ERR_HARDWARE_FUNCTION_NOT_SUPPORTED	-115	Function is not supported by current hardware

7 PSO Connector description

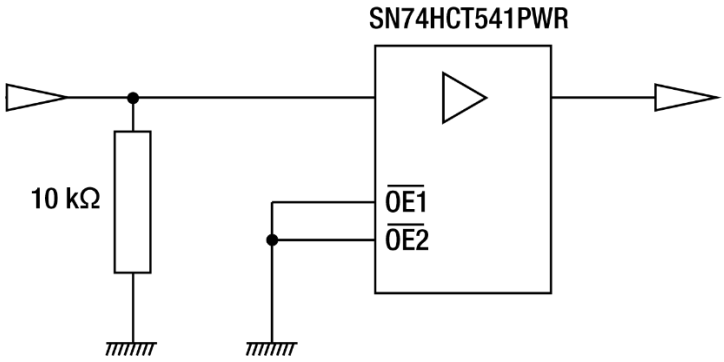
This chapter briefly describes the PSO (Pulse Synchronized Output) connector of the XPS.

NOTE **Mating connector: LEMO FGG0B306CLAD52 (or 42).**

PSO connector		
	**Pin #**	**Function**
	1	*+5 V (fused: max 63 mA)*
	2	*Axis 1/5 PCO Pulse/QuadA*
	3	*Axis 1/5 PCO Enable/QuadB*
	4	*Axis 2/6 PCO Pulse/QuadA*
	5	*Axis 2/6 PCO Enable/QuadB*
	6	*GND*

PSO connector description

The PSO signal is driven by SN74HCT541PWR line driver.



Output high level = 3 V minimum.

Service Form

Your Local Representative
Tel.:
Fax:

Name: _____
Company: _____
Address: _____
Country: _____
P.O. Number: _____
Item(s) Being Returned: _____
Model#: _____

Return authorization #: _____
(Please obtain prior to return of item)
Date: _____
Phone Number: _____
Fax Number: _____

Serial #: _____

Description: _____
Reasons of return of goods (please list any specific problems): _____

Visit MKS I Newport Online at:
www.newport.com

North America & Asia
Newport Corporation
1791 Deere Ave.
Irvine, CA 92606, USA

Sales
Tel.: +1 (949)-863-3144
e-mail: sales@newport.com

Technical Support
Tel.: +1 (949)-863-3144
e-mail: tech@newport.com

Service, RMAs & Returns
Tel.: +1 (949)-863-3144
e-mail: service@newport.com

Europe
MICRO-CONTROLE Spectra-Physics S.A.S
7 rue des Plantes
45340 Beaune-la-Rolande
France

Sales Europe (EMEAI)
Tel.: +49 (0) 6151-708-0
e-mail: germany@newport.com

Sales France
Tel.: +33 (0)1 60 91 68 68
e-mail: france@newport.com

Sales UK
Tel.: +44 (0)1235 432 710
e-mail: uk@newport.com

Technical Support
e-mail: tech_europe@newport.com

Service & Returns
Tel.: +33 (0)2 38 40 51 55
DST-BEA-RMA-service@newport.com