Table of contents

KinMoveDirectEx - Asynchronous point-to-point motion

KinMoveDirectEx - Asynchronous point-to-point motion

Basics

This command is an extension of the commands ↘ “KinMoveDirectAsyncAbs - Direct asynchronous absolute traversing of kinematics” and ↘ “KinMoveDirectAsyncRel - Direct asynchronous relative traversing of kinematics”.

It moves the axes of the kinematics asynchronously to the commanded absolute or relative target positions. Asynchronous means that although the axes start their respective motions at the same time, they reach their target position at different times depending on the travel distance and dynamics.

The coordinate system to which the specified target position refers can be selected as required (PCS, MCS, ACS). The target position is transformed internally in the ACS and the interpolation is carried out separately for each axis.

NOTICE!

As the interpolation is carried out separately for all axes involved at the axis level, the result is a curved path in space!

As the target position, enter the position value, the meaning and the desired attribute for each axis/coordinate of the kinematics to be moved at ↘ “Overview of”. Depending on the attribute, the position values are interpreted as absolute or relative values.

PLC:

Fig. 191: ML_KinMoveDirectAsyncEx

ML_KinMoveDirectAsyncEx

Name	Data type	Inherited from	Address	Initial value	Comment
Data	ML_KinMoveDirectAsyncExData				InOut: Command parameters
ML_KinMoveDirectAsyncEx	BOOL				Returns, whether it was possible to add the command to the kinematic object. ▪ TRUE: The ML_KinMoveDirectAsyncEx command was added successfully ▪ FALSE: An error occurred. Details in ML_oCmdResult

STRUCT ML_KinMoveDirectAsyncExData

STRUCT ML_KinMoveDirectAsyncExData

Name	Data type	Inherited from	Address	Initial value	Comment
In	ML_iKinMoveDirectAsyncEx				Input: Command data of the command
Out	ML_oCmdResult				Output: Command parameters

STRUCT ML_iKinMoveDirectEx EXTENDS ML_iKinCmdBase

Name	Data type	Inherited from	Address	Initial value	Comment
KinName	STRING(15)	ML_iKinCmdBase			Name of the kinematics used to execute the command
Source	STRING(50)	ML_iKinCmdBase			Specifies the command source, e.g PlcApplication1 is displayed in the diagnostic messages
SourceLine	ULINT	ML_iKinCmdBase			Specifies the source line of the command; is displayed in the diagnostic messages
CmdKinPose	ML_KinMotionCmdTarget				Commanded absolute target position with meanings
CoordSys	STRING(15)				Coordinate system of the commanded target position (PCS, WCS, MCS or ACS
Buffered	BOOL			TRUE	TRUE, if the command is to be executed as buffered command, otherwise FALSE
DynLimits	ML_iDynLimits				Limit values of the position command

STRUCT ML_KinMotionCmdTarget

Name	Data type	Inherited from	Address	Initial value	Comment
CmdKinTarget	ARRAY [0..24] OF ML_KinCmdPosePair				Commanded target position with meanings
CmdKinTargetSize	UINT				Number of entries in CmdKinTarget

STRUCT ML_KinCmdPosePair

STRUCT ML_KinCmdPosePair

Name	Data type	Inherited from	Address	Initial value	Comment
Value	LREAL				A value of the cmdKinPose
Meaning	ML_Meaning				Associated meaning of the value
Attribute	ML_Attributes			UNDEF	Additional attribute of the meaning

ENUM ML_Meaning

Name	Data type	Inherited from	Address	Initial value	Comment
UNDEF	INT			0	Error or undefined
X	INT				Absolute position of the X-axis
Y	INT				Absolute position of the Y-axis
Z	INT				Absolute position of the Z-axis
AX1	INT				Absolute position of axis 1
AX2	INT				Absolute position of axis 2
AX3	INT				Absolute position of axis 3
...					

ENUM ML_MeaningAttribute

Name	Data type	Inherited from	Address	Initial value	Comment
UNDEF	INT			0	Error, undefined or not used
INCR_VALUE	INT				IC(): Incremental value
ABS_VALUE	INT				AC(): Absolute value
SHORT_WAY	INT				DC(): Shortest route
POS_ROT_DIR	INT				ACP(): Positive direction of rotation
NEG_ROT_DIR	INT				ACN(): Negative direction of rotation

STRUCT ML_iDynLimits

Name	Data type	Inherited from	Address	Initial value	Comment
Velocity	LREAL				Velocity limit value
Acceleration	LREAL				Acceleration limit value
Deceleration	LREAL				Deceleration limit value
JerkAcc	LREAL			0	Jerk limit value for acceleration phase
JerkDec	LREAL			0	Jerk limit value for deceleration phase

STRUCT ML_oCmdResult

Name	Data type	Inherited from	Address	Initial value	Comment
CmdID	ULINT				Command ID (set to 18446744073709551615 in case of error)
Error	BOOL				Error ID TRUE indicates that an error occurred
ErrorID	UDINT				Specifies the type of occurred error: The upper 4 bytes reflect the main diagnostics, the lower 4 bytes reflect the detailed diagnostics, 0 means no error

Data Layer

/motion/kin/<kin_name>/cmd/move-direct-async-ex

Python

```
<cmdID> = motion.kin_cmd_move_direct_async_ex( kin=<kinName>, kin_pose_pair=<targetPose>
[,coord_sys=<targetCoordSys>] [,vel=<velocity>] [,acc=<acceleration>]
[,dec=<deceleration>] [,jrk_acc=<jerkAcceleration>]
[,jrk_dec=<jerkDeceleration>] [,buffered=<buffered>])
```

Generates KinMoveDirectEx command for a kinematics (object has to be "attached"), see documentation Python Runtime App Python functions.

- <kinName> - string, kinematic name
- <targetPose> - Specify a target position of the command by pairing a meaning and a corresponding value for the coordinates to be offset.
  The pairs are collected in a dictionary data type { <meaning string>:<position>, ... ,<meaning string>:<position>}.
  <position>can be a <float>, <long> or a <sub dictionary> with {<attribute string>:<float|long>}.
  Example:
  kin_pose_pair={ "X":3.0, "Y":2, "Z":{"INCR_VALUE":-10}, "AX5":{"POS_ROT_DIR":20}}
- <targetCoordSys> - string ("PCS", "WCS", "MCS" or "ACS"), coordinate system, in which <targetPose> is defined (coordinate system is "PCS", if not programmed)
- <velocity> - double, velocity limit value of the command (the last programmed velocity limit value is used if this parameter is not programmed)

- <acceleration> - double, acceleration limit value of the command (the last programmed acceleration limit value is used if this parameter is not programmed)
- <deceleration> - double, deceleration limit value of the command (the last programmed deceleration limit value is used if this parameter is not programmed)
- <jerkAcceleration> - double, jerk limit value for the acceleration of the command (the last programmed jerk limit value for the acceleration is used if this parameter is not programmed)
- <jerkDeceleration> - double, jerk limit value for the deceleration of the command (the last programmed jerk limit value for the deceleration is used if this parameter is not programmed)
- <buffered> - bool, use TRUE if the command is to be buffered (TRUE by default)

BundleIF:

Command type for a linear motion with absolute target position:

```
/// @brief Create a point-to-point motion command for a kinematics in which all axes are moved independently and the target position is specified by v
alue pairs
///
/// @param[in]   sourceInfo           Command source information
/// @param[in]   kinName              Name of the motion object
/// @param[in]   targetPose           Target pose of the command
/// @param[in]   nofTargetPoseElements  Number of target pose elements
/// @param[in]   targetCOS            Target coordinate system for the targetPose
/// @param[in]   dynLim               Dynamic limits of the command (not yet supported)
/// @param[out]  cmdID                The ID of the added command
/// @param[in]   buffered             is buffered mode?
/// @return                       Status of the function call: STS_OK or an error code on failure.

virtual MotionResult kinMoveCmdDirectAsyncEx(const dia::CmdSourceInfo& sourceInfo, const char* kinName,
                            const cmd::CmdKinPosePair* targetPose, const uint32_t nofTargetPoseElements,
                            cmd::kin::CmdTargetCOS targetCOS, const cmd::DynamicLimitsT& dynLim,
                            uint64_t& cmdID, bool buffered = true) const = 0;
```