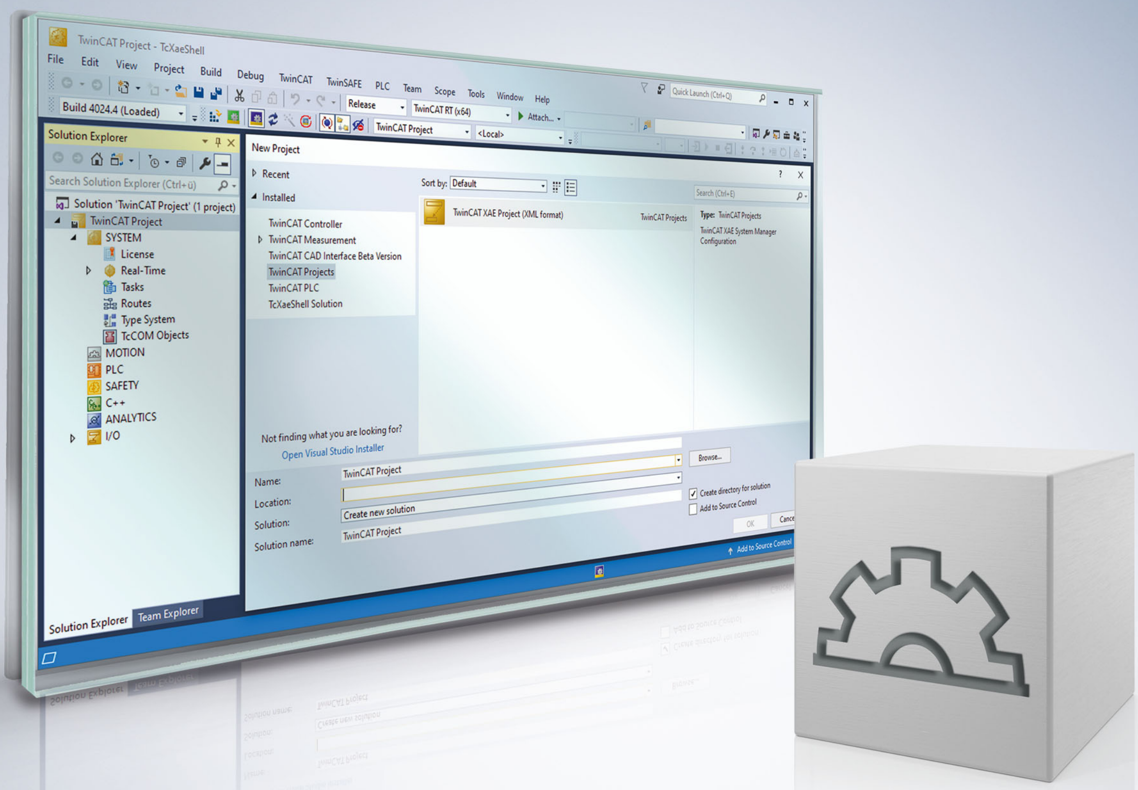
BECKHOFF New Automation Technology

Manual | EN

TF5055

TwinCAT 3 | NC Flying Saw



2024-04-08 | Version: 1.5.3

Table of contents

1 Foreword **5**
1.1 Notes on the documentation 5
1.2 For your safety 5
1.3 Notes on information security 7

2 Overview **8**

3 PLC Library Overview **9**

4 Flying saw **10**
4.1 MC_GearInVelo 10
4.2 MC_GearInPos 12
4.3 MC_ReadFlyingSawCharacteristics 14

5 Data types **16**
5.1 ST_SyncMode 16
5.2 E_GearInSyncMode 19
5.3 MC_FlyingSawCharacValues 19
5.4 ST_GearInPosOptions 20
5.5 ST_GearInVeloOptions 20

6 Sample program **22**

7 Support and Service **23**

8 Appendix **24**
8.1 Error Codes 24
8.2 Reversal of the master axis movement / backstop 33
8.3 Calculating the synchronisation phase 36

1 Foreword

1.1 Notes on the documentation

This description is intended exclusively for trained specialists in control and automation technology who are familiar with the applicable national standards.
For installation and commissioning of the components, it is absolutely necessary to observe the documentation and the following notes and explanations.
The qualified personnel is obliged to always use the currently valid documentation.

The responsible staff must ensure that the application or use of the products described satisfies all requirements for safety, including all the relevant laws, regulations, guidelines, and standards.

Disclaimer

The documentation has been prepared with care. The products described are, however, constantly under development.
We reserve the right to revise and change the documentation at any time and without notice.
No claims to modify products that have already been supplied may be made on the basis of the data, diagrams, and descriptions in this documentation.

Trademarks

Beckhoff®, TwinCAT®, TwinCAT/BSD®, TC/BSD®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC®, XTS® and XPlanar® are registered and licensed trademarks of Beckhoff Automation GmbH.
If third parties make use of designations or trademarks used in this publication for their own purposes, this could infringe upon the rights of the owners of the said designations.

Patents

The EtherCAT Technology is covered, including but not limited to the following patent applications and patents:
EP1590927, EP1789857, EP1456722, EP2137893, DE102015105702
and similar applications and registrations in several other countries.

EtherCAT® is registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany

Copyright

1.2 For your safety

Safety regulations

Read the following explanations for your safety.
Always observe and follow product-specific safety instructions, which you may find at the appropriate places in this document.

Exclusion of liability

All the components are supplied in particular hardware and software configurations which are appropriate for the application. Modifications to hardware or software configurations other than those described in the documentation are not permitted, and nullify the liability of Beckhoff Automation GmbH & Co. KG.

Personnel qualification

This description is only intended for trained specialists in control, automation, and drive technology who are familiar with the applicable national standards.

Signal words

The signal words used in the documentation are classified below. In order to prevent injury and damage to persons and property, read and follow the safety and warning notices.

Personal injury warnings

⚠ DANGER
Hazard with high risk of death or serious injury.

⚠ WARNING
Hazard with medium risk of death or serious injury.

⚠ CAUTION
There is a low-risk hazard that could result in medium or minor injury.

Warning of damage to property or environment

NOTICE
The environment, equipment, or data may be damaged.

Information on handling the product

This information includes, for example:
recommendations for action, assistance or further information on the product.

1.3 Notes on information security

The products of Beckhoff Automation GmbH & Co. KG (Beckhoff), insofar as they can be accessed online, are equipped with security functions that support the secure operation of plants, systems, machines and networks. Despite the security functions, the creation, implementation and constant updating of a holistic security concept for the operation are necessary to protect the respective plant, system, machine and networks against cyber threats. The products sold by Beckhoff are only part of the overall security concept. The customer is responsible for preventing unauthorized access by third parties to its equipment, systems, machines and networks. The latter should be connected to the corporate network or the Internet only if appropriate protective measures have been set up.

In addition, the recommendations from Beckhoff regarding appropriate protective measures should be observed. Further information regarding information security and industrial security can be found in our https://www.beckhoff.com/secguide.

Beckhoff products and solutions undergo continuous further development. This also applies to security functions. In light of this continuous further development, Beckhoff expressly recommends that the products are kept up to date at all times and that updates are installed for the products once they have been made available. Using outdated or unsupported product versions can increase the risk of cyber threats.

To stay informed about information security for Beckhoff products, subscribe to the RSS feed at https://www.beckhoff.com/secinfo.

2 Overview

In many plants workpieces undergo machining operations while being transported. For this purpose it is necessary to synchronise the position and the speed of tool and workpiece, so that the tool can then be applied as if to a stationary workpiece. One example of such an application is a saw that during the transport process cuts through the material that is being transported (flying saw). In order to implement this kind of application, TwinCAT provides the flying saw.

The TwinCAT PLC library Tc2_MC2_FlyingSaw, available as an *additional product*, provides easy management of the flying saw. An example program using the flying saw makes use of this library.

3 PLC Library Overview

Below there is an overview of the TF5055 TC3 MC Flying Saw PLC Library.

Flying Saw

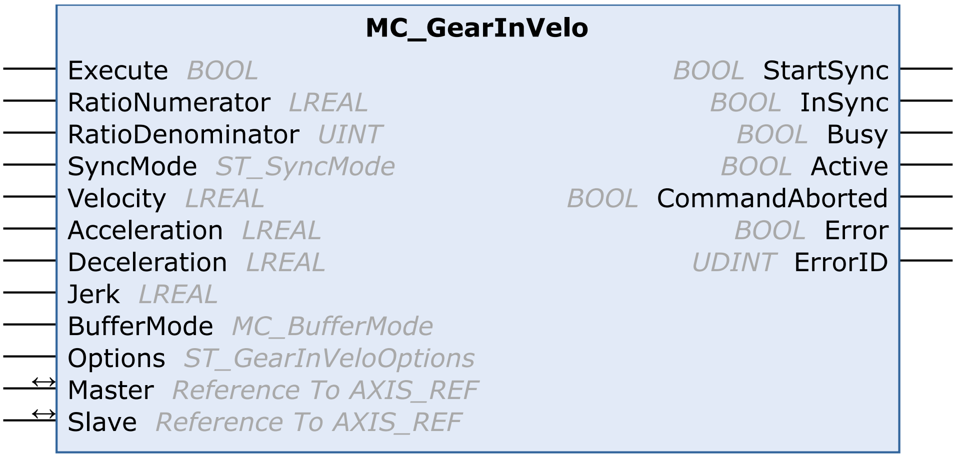
Function Block	Description
MC_GearInVelo [▶ 10]	Activates a linear master-slave coupling.
MC_GearInPos [▶ 12]	Synchronizes a slave axis precisely with a master axis (flying saw).
MC_ReadFlyingSawCharacteristics [▶ 14]	Allows the characteristic figures for the synchronization phase of the Universal Flying Saw to be read.

Data Types

Data Type	Description
ST_SyncMode [▶ 16]	Operation of the individual bits.
MC_FlyingSawCharacValues [▶ 19]	Type definition for the characteristic parameters of a flying saw synchronization.

4 Flying saw

4.1 MC_GearInVelo



The function block *MC_GearInVelo* activates a linear master-slave coupling (gear coupling). If the master axis is already moving, the slave axis synchronizes to the master velocity. The function block accepts a fixed gear ratio in numerator/denominator format.

The slave axis can be uncoupled with the function block MC_GearOut. If the slave is decoupled while it is moving, then it retains its velocity and can be halted using MC_Stop or MC_Halt.

Inputs

```
VAR_INPUT
    Execute          : BOOL;
    RatioNumerator   : LREAL;
    RatioDenominator : UINT;
    SyncMode         : ST_SyncMode;
    Velocity         : LREAL;
    Acceleration     : LREAL;
    Deceleration     : LREAL;
    Jerk             : LREAL;
    BufferMode       : MC_BufferMode;
    Options          : ST_GearInVeloOptions;
END_VAR
```

Name	Type	Description
Execute	BOOL	The command is executed with a rising edge at *Execute* input.
RatioNumerator	LREAL	Gear ratio numerator. Alternatively, the gear ratio can be specified in the enumerator as a floating point value, if the denominator is 1.
RatioDenominator	UINT	Gear ratio denominator
SyncMode	ST_SyncMode	In the data structure SyncMode [▸ 16] boundary conditions for the synchronization process are specified via individual flags.
Velocity	LREAL	Maximum slave velocity in the synchronization phase. If no velocity is specified, the default velocity of the axis from the System Manager data is used. **Info**: the velocity specified here is only checked if this check is activated through the SyncMode [▸ 16] variable.
Acceleration	LREAL	Maximum slave acceleration in the synchronization phase. If no acceleration is specified, the default acceleration of the axis from the System Manager data is

Name	Type	Description
		used. **Info**: the acceleration specified here is only checked if this check is activated through the SyncMode [▸ 16] variable.
Deceleration	LREAL	Maximum slave deceleration in the synchronization phase. If no deceleration is specified, the default deceleration of the axis from the System Manager data is used. **Info**: the deceleration specified here is only checked if this check is activated through the SyncMode [▸ 16] variable.
Jerk	LREAL	Maximum slave jerk in the synchronization phase. If no jerk is specified, the default jerk of the axis from the System Manager data is used. **Info**: the jerk specified here is only checked if this check is activated through the SyncMode [▸ 16] variable.
BufferMode	MC_BufferMode	Currently not implemented
Options	ST_GearInVeloOptions	The ST_GearInVeloOptions [▸ 20] contains two position values as limits. The check for these position limits can be activated. To do this, two flags (`GearInSync_CheckMask_OptionalMinPos`, `GearInSync_CheckMask_OptionalMaxPos`) have to be set in the SyncMode [▸ 16] structure. The *RatioNumerator* must be 1 and the *RatioDenominator* must be 4 for a 1:4 ratio. Alternatively, the *RatioDenominator* may be 1, and the gear ratio can be specified as floating point number 0.25 at *RatioNumerator*. The *RatioNumerator* may be negative.

Inputs/outputs

The axis data structure of type AXIS_REF addresses an axis unambiguously within the system. Among other parameters, it contains the current axis status, including position, velocity or error state.

```
VAR_IN_OUT
    Master : AXIS_REF;
    Slave  : AXIS_REF;
END_VAR
```

Name	Type	Description
Master	AXIS_REF	Axis data structure of the master
Slave	AXIS_REF	Axis data structure of the slave

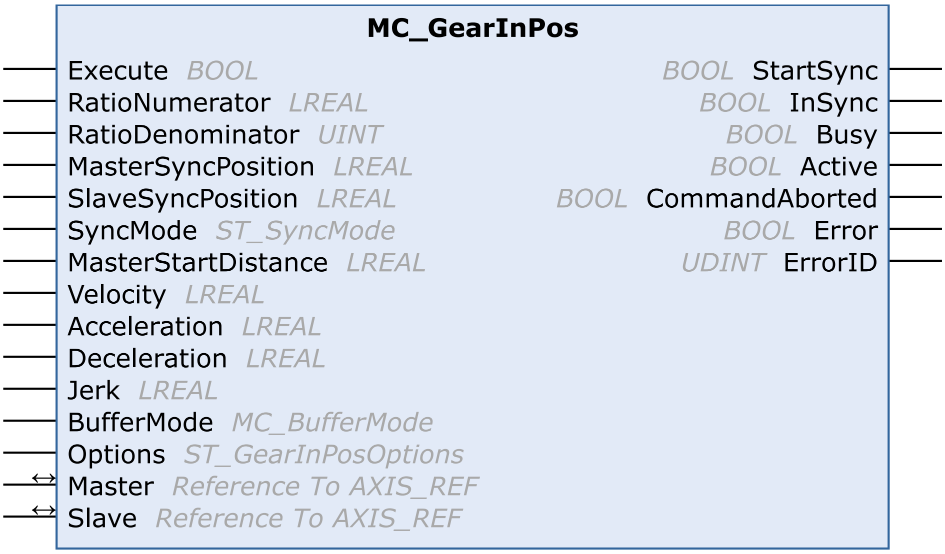
Outputs

```
VAR_OUTPUT
    StartSync       : BOOL;
    InSync          : BOOL;
    Busy            : BOOL;
    Active          : BOOL;
    CommandAborted  : BOOL;
    Error           : BOOL;
    ErrorID         : UDINT;
END_VAR
```

Name	Type	Description
StartSync	BOOL	Becomes TRUE when the synchronization with the master axis was started.
InSync	BOOL	Becomes TRUE, if the coupling was successfully completed and the slave axis is synchronized with the master axis.

Name	Type	Description
Busy	BOOL	The *Busy* output becomes TRUE when the command is started with *Execute* and remains TRUE for as long as the command is processed. If *Busy* becomes FALSE again, the function block is ready for a new order. At the same time one of the outputs, *InSync*, *CommandAborted* or *Error*, is set.
Active	BOOL	Active indicates that the command is executed. (currently Active=Busy, see BufferMode)
CommandAborted	BOOL	Becomes TRUE, if the command could not be fully executed. The axis may have become decoupled during the coupling process (simultaneous command execution).
Error	BOOL	Becomes TRUE as soon as an error occurs.
ErrorID	UDINT	If the error output is set, this parameter supplies the error number.

4.2 MC_GearInPos



The function block *MC_GearInPos* synchronizes a slave axis precisely with a master axis (flying saw). The synchronous velocity is achieved exactly at the synchronous point of the master and slave.

The master axis must already be moving, otherwise synchronization is not possible.

The slave axis can be uncoupled with the function block MC_GearOut. If the slave is decoupled while it is moving, then it retains its velocity and can be halted using MC_Stop or MC_Halt.

Inputs

```
VAR_INPUT
    Execute             : BOOL;
    RatioNumerator      : LREAL;
    RatioDenominator    : UINT;
    MasterSyncPosition  : LREAL;
    SlaveSyncPosition   : LREAL;
    SyncMode            : ST_SyncMode;
    MasterStartDistance : LREAL;
    Velocity            : LREAL;
    Acceleration        : LREAL;
    Deceleration        : LREAL;
    Jerk                : LREAL;
    BufferMode          : MC_BufferMode;
    Options             : ST_GearInPosOptions;
END_VAR
```

Name	Type	Description
Execute	BOOL	The command is executed with a rising edge at *Execute* input.
RatioNumerator	LREAL	Gear ratio numerator. Alternatively, the gear ratio can be specified in the enumerator as a floating point value, if the denominator is 1.
RatioDenominator	UINT	Gear ratio denominator
MasterSyncPosition	LREAL	The master's synchronous position
SlaveSyncPosition	LREAL	The slave's synchronous position
SyncMode	ST_SyncMode	In the data structure SyncMode [▸ 16] boundary conditions for the synchronization process are specified via individual flags.
MasterStartDistance	LREAL	Currently not implemented.
Velocity	LREAL	Maximum slave velocity in the synchronization phase. If no velocity is specified, the default velocity of the axis from the System Manager data is used. **Info**: the velocity specified here is only checked if this check is activated through the SyncMode [▸ 16] variable.
Acceleration	LREAL	Maximum slave acceleration in the synchronization phase. If no acceleration is specified, the default acceleration of the axis from the System Manager data is used. **Info**: the acceleration specified here is only checked if this check is activated through the SyncMode [▸ 16] variable.
Deceleration	LREAL	Maximum slave deceleration in the synchronization phase. If no deceleration is specified, the default deceleration of the axis from the System Manager data is used. **Info**: the deceleration specified here is only checked if this check is activated through the SyncMode [▸ 16] variable.
Jerk	LREAL	Maximum slave jerk in the synchronization phase. If no jerk is specified, the default jerk of the axis from the System Manager data is used. **Info**: the jerk specified here is only checked if this check is activated through the SyncMode [▸ 16] variable.
BufferMode	MC_BufferMode	Currently not implemented
Options	ST_GearInPosOptions	The ST_GearInPosOptions [▸ 20] contains two position values as limits. The check for these position limits can be activated. To do this, two flags (`GearInSync_CheckMask_OptionalMinPos`, `GearInSync_CheckMask_OptionalMaxPos`) have to be set in the SyncMode [▸ 16] structure.

The *RatioNumerator* must be 1 and the *RatioDenominator* must be 4 for a 1:4 ratio. Alternatively, the *RatioDenominator* may be 1, and the gear ratio can be specified as floating point number 0.25 at *RatioNumerator*. The *RatioNumerator* may be negative.

Inputs/outputs

The axis data structure of type AXIS_REF addresses an axis unambiguously within the system. Among other parameters, it contains the current axis status, including position, velocity or error state.

```
VAR_IN_OUT
    Master : AXIS_REF;
    Slave  : AXIS_REF;
END_VAR
```

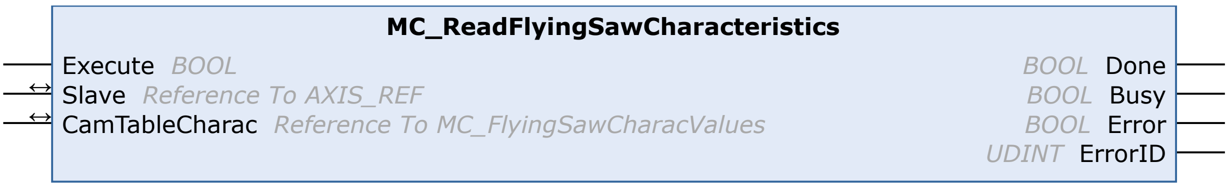
Name	Type	Description
Master	AXIS_REF	Axis data structure of the master
Slave	AXIS_REF	Axis data structure of the slave

Outputs

```
VAR_OUTPUT
    StartSync      : BOOL;
    InSync         : BOOL;
    Busy           : BOOL;
    Active         : BOOL;
    CommandAborted : BOOL;
    Error          : BOOL;
    ErrorID        : UDINT;
END_VAR
```

Name	Type	Description
StartSync	BOOL	Becomes TRUE when the synchronization with the master axis was started.
InSync	BOOL	Becomes TRUE, if the coupling was successfully completed and the slave axis is synchronized with the master axis.
Busy	BOOL	The *Busy* output becomes TRUE when the command is started with *Execute* and remains TRUE for as long as the command is processed. If *Busy* becomes FALSE again, the function block is ready for a new order. At the same time one of the outputs, *InSync*, *CommandAborted* or *Error*, is set.
Active	BOOL	Active indicates that the command is executed. (currently Active=Busy, see BufferMode)
CommandAborted	BOOL	Becomes TRUE, if the command could not be fully executed. The axis may have become decoupled during the coupling process (simultaneous command execution).
Error	BOOL	Becomes TRUE as soon as an error occurs.
ErrorID	UDINT	If the error output is set, this parameter supplies the error number.

4.3 MC_ReadFlyingSawCharacteristics



The function block *MC_ReadFlyingSawCharacteristics* allows the characteristic values for the synchronization phase of the Universal Flying Saw to be read.

Inputs

```
VAR_INPUT
    Execute : BOOL;
END_VAR
```

Name	Type	Description
Execute	BOOL	A rising edge initiates reading the characteristic values from the TwinCAT NC.

The data calculated is not available until the Universal Flying Saw starts.

Inputs/outputs

The axis data structure of type AXIS_REF addresses an axis unambiguously within the system. Among other parameters it contains the current axis status, including position, velocity or error state.

```
VAR_IN_OUT
    Slave          : AXIS_REF;
    CamTableCharac : MC_FlyingSawCharacValues;
END_VAR
```

Name	Type	Description
Slave	AXIS_REF	Axis data structure of the slave
CamTableCharac	MC_FlyingSawCharacValues	Structure containing the characteristic values. [▶ 19]

Outputs

```
VAR_OUTPUT
    Done    : BOOL;
    Busy    : BOOL;
    Error   : BOOL;
    ErrorID : UDINT;
END_VAR
```

Name	Type	Description
Done	BOOL	Is set to TRUE when the data set has been successfully read.
Busy	BOOL	The *Busy* output becomes TRUE when the command is started with *Execute* and remains TRUE as long as the command is processed. If *Busy* becomes FALSE again, the function block is ready for a new order. At the same time one of the outputs, *Done* or *Error*, is set.
Error	BOOL	Becomes TRUE, as soon as an error occurs.
ErrorID	UDINT	If the error output is set, this parameter supplies the error number.

5 Data types

5.1 ST_SyncMode

```
TYPE ST_SyncMode :
STRUCT
    (* mode *)
    GearInSyncMode                          : E_GearInSyncMode;

    (* 32 bit check mask ... *)
    GearInSync_CheckMask_MinPos             : BOOL;
    GearInSync_CheckMask_MaxPos             : BOOL;
    GearInSync_CheckMask_MaxVelo            : BOOL;
    GearInSync_CheckMask_MaxAcc             : BOOL;
    GearInSync_CheckMask_MaxDec             : BOOL;
    GearInSync_CheckMask_MaxJerk            : BOOL;
    GearInSync_CheckMask_OptionalMinPos     : BOOL;
    GearInSync_CheckMask_OptionalMaxPos     : BOOL;
    GearInSync_CheckMask_OvershootPos       : BOOL;
    GearInSync_CheckMask_UndershootPos      : BOOL;
    GearInSync_CheckMask_OvershootVelo      : BOOL;
    GearInSync_CheckMask_UndershootVelo     : BOOL;
    GearInSync_CheckMask_OvershootVeloZero  : BOOL;
    GearInSync_CheckMask_UndershootVeloZero : BOOL;

    (* operation masks ... *)
    GearInSync_OpMask_RollbackLock          : BOOL;
    GearInSync_OpMask_InstantStopOnRollback : BOOL;
    GearInSync_OpMask_PreferConstVelo       : BOOL;
    GearInSync_OpMask_IgnoreMasterAcc       : BOOL;
    GearInSync_OpMask_IgnoreSlaveAcc        : BOOL;
    GearInSync_OpMask_DetailedErrorCodes    : BOOL;
END_STRUCT
END_TYPE
```

Parameterisable boundary conditions, specifying the mode of operation

It is possible to specify a wide variety of boundary conditions for the slave movement in the synchronization phase of the Universal Flying Saw. These boundary conditions make it possible to specify limit values for the slave magnitudes listed in the table below. The *SyncMode* bit mask can be used to check whether the individual limit values are being observed. The boundary conditions specified for the synchronization phase also affect the set value profile for the synchronization. Whether, and in what way, the conditions affect the profile can be seen in the diagram [▶ 36] in the attachment.

Name	Description
GearInSyncMode	See E_GearInSyncMode [▶ 19]

Bit masks for the SyncMode

Name	Value (dec)	Value (hex)	Description	Boundary condition
GearInSync_CheckMask_MinPos	1	0x0000 0001	Checks whether the slave axis has passed below its software minimum end position (machine data). See also ST_GearInPosOptions [▶ 20]	$pos_{Slave} \geq pos_{SlaveMin}$
GearInSync_CheckMask_MaxPos	2	0x0000 0002	Checks whether the software maximum end position (machine data) of the slave axis has been exceeded. See also ST_GearInPosOptions [▶ 20]	$pos_{Slave} \leq pos_{SlaveMax}$

Name	Value (dec)	Value (hex)	Description	Boundary condition
GearInSync_CheckMask_MaxVelo	4	0x0000 0004	Checks whether the maximum permitted slave velocity (machine data) has been exceeded. See also ST_GearInVeloOptions [▸ 20]	$\lvert v_{Slave} \rvert \leq v_{SlaveMax}$
GearInSync_CheckMask_MaxAcc	8	0x0000 0008	Checks whether the maximum slave acceleration (machine data) has been exceeded	$acc_{Slave} \leq acc_{SlaveMax}$
GearInSync_CheckMask_MaxDec	16	0x0000 0010	Checks whether the maximum slave deceleration (machine data) has been exceeded	$dec_{Slave} \leq dec_{SlaveMax}$
GearInSync_CheckMask_MaxJerk	32	0x0000 0020	Checks whether the maximum slave jerk (machine data) has been exceeded.	$j_{Slave} \leq j_{SlaveMax}$
GearInSync_CheckMask_OptionalMinPos			Check is also carried out to see if GearInSync_CheckMask_OptionalMinPos is set to TRUE. See also ST_GearInVeloOptions [▸ 20]	
GearInSync_CheckMask_OptionalMaxPos			Check is also carried out if GearInSync_CheckMask_OptionalMaxPos is set to TRUE. See also ST_GearInVeloOptions [▸ 20]	
GearInSync_CheckMask_OvershootPos	256	0x0000 0100	Checks for overshooting of slave position.	
GearInSync_CheckMask_UndershootPos	512	0x0000 0200	Checks for undershooting of slave position.´	
GearInSync_CheckMask_OvershootVelo	1024	0x0000 0400	Checks for overshooting of slave acceleration.	
GearInSync_CheckMask_UndershootVelo	2048	0x0000 0800	Checks for undershooting of slave acceleration.	
GearInSync_CheckMask_OvershootVeloZero	4096	0x0000 1000	Checks whether the slave velocity has exceeded 0.0.	
GearInSync_CheckMask_UndershootVeloZero	8192	0x0000 2000	Checks whether the slave velocity is below 0.0	
GearInSync_OptMask_DetailedErrorCodes			Enable detailed error codes for the checks.	

Bit masks for operation modes

Name	Value (dec)	Value (hex)	Description
GearInSync_OpMask_RollbackLock	65536	0x0001 0000	**Bit = 0: (default)** When the slave has achieved the synchronous phase, synchronous coupling of all the following master movements is maintained until the coupling is removed. This also applies if the master changes direction and moves backwards over the coupling position, see Reversal of the master axis movement / backstop [▸ 33].
			Bit = 1:

Name	Value (dec)	Value (hex)	Description
			Setting this bit activates the backstop, which causes the slave to stop when the master moves backwards beyond the coupling position after a motion reversal, see reversal of the master axis movement / backstop.
GearInSync_OpMask_InstantStopOnRollback	131072	0x0002 0000	**Bit = 0: (default)** On reaching the coupling position, the slave velocity is reduced smoothly following a 5th order polynomial. The polynomial is optimized to halt the slave as quickly as possible, see reversal of the master axis movement / backstop.
			Bit = 1: In terms of the set value, the slave is halted within one NC tick of reaching the coupling position. The slave velocity is set to 0.0 and the position is maintained. **This abrupt stop can trigger the following error monitoring system!** See reversal of the master axis movement / backstop
GearInSync_OpMask_PreferConstVelo	1048576	0x0010 0000	**Bit = 0: (default)** Default setting
			Bit = 1: The system will try to use a phase with constant velocity, instead of just one 5th order polynomial. This can result in a combination of a 5th-order polynomial, a synchronous phase and another 5th-order polynomial (P5-P1-P5), see reversal of the master axis movement / backstop. The maximum given acceleration and deceleration is used. To control and limit the jerk it is recommended to set MAXJERK in the bit mask.
GearInSync_OpMask_IgnoreSlaveAcc	2097152	0x0020 0000	**Bit = 0: (default)** Default setting
			Bit = 1: When calculating the coupling, the acceleration of the master is ignored, i.e. set to zero. This causes the use of internal optimizations. At moderate acceleration, this specification leads to tolerable following errors. After the following error has been reduced, the relative position accuracy is independent of this setting.

The checks listed above apply only to the synchronization phase (GEARINSYNCSTATE_SYNCHRONIZING), not to the phase of synchronized movement. These calculations and checks are also only possible when the assumption is made that the master continues to move with constant velocity after the coupling time, i.e. that it is not subject to acceleration. Making other assumptions for the master makes no sense, since at the time of coupling it is generally not known how the master will move in the future.

See also:

- Error Codes [▶ 24]

5.2 E_GearInSyncMode

```
TYPE E_GearInSyncMode :
(
    GEARINSYNCMODE_POSITIONBASED,
    GEARINSYNCMODE_TIMEBASED
);
END_TYPE
```

Parameter	Description
GEARINSYNCMODE_POSITIONBASED	In this mode of the universal flying saw, a profile dependent on the master position is generated to synchronize the slave axis to the master axis.
GEARINSYNCMODE_TIMEBASED	In this mode of the universal flying saw, a time-dependent motion profile is generated for synchronizing the slave axis to the master axis, which ensures compliance with all dynamic limit values of the slave axis. This mode is currently only available with coupling on velocity.

The time-based motion profile (GEARINSYNCMODE_TIMEBASED) is currently only implemented for the function block MC_GearInVelo.

5.3 MC_FlyingSawCharacValues

```
TYPE MC_FlyingSawCharacValues :
STRUCT
    (* Master Velocity*)
    fMasterVeloNom      : LREAL; (* 1. master nominal velocity (normed:=> 1.0) *)

    (* characteristic slave data *)
    (*===========================*)

    (* Start of cam table *)
    fMasterPosStart     : LREAL; (* 2. master start position*)
    fSlavePosStart      : LREAL; (* 3. slave start position *)
    fSlaveVeloStart     : LREAL; (* 4. slave start velocity *)
    fSlaveAccStart      : LREAL; (* 5. slave start acceleration *)
    fSlaveJerkStart     : LREAL; (* 6. slave start jerk *)


    (* End of cam table*)
    fMasterPosEnd       : LREAL; (* 7. master end position *)
    fSlavePosEnd        : LREAL; (* 8. slave end position *)
    fSlaveVeloEnd       : LREAL; (* 9. slave end velocity *)
    fSlaveAccEnd        : LREAL; (* 10. slave end acceleration *)
    fSlaveJerkEnd       : LREAL; (* 11. slave end jerk *)

    (* minimum slave position *)
    fMPosAtSPosMin      : LREAL; (* 12. master position AT slave minimum position *)
    fSlavePosMin        : LREAL; (* 13. slave minimum position *)

    (* minimum Slave velocity *)
    fMPosAtSVeloMin     : LREAL; (* 14. master position AT slave minimum velocity *)
    fSlaveVeloMin       : LREAL; (* 15. slave minimum velocity *)

    (* minimum slave acceleration *)
    fMPosAtSAccMin      : LREAL; (* 16. master position AT slave minimum acceleration *)
    fSlaveAccMin        : LREAL; (* 17. slave minimum acceleration *)
    fSVeloAtSAccMin     : LREAL; (* 18. slave velocity AT slave minimum acceleration *)

    (* minimum slave jerk and dynamic momentum *)
    fSlaveJerkMin       : LREAL; (* 19. slave minimum jerk *)
    fSlaveDynMomMin     : LREAL; (* 20. slave minimum dynamic momentum (NOT SUPPORTED YET !) *)

    (* maximum slave position *)
    fMPosAtSPosMax      : LREAL; (* 21. master position AT slave maximum position *)
    fSlavePosMax        : LREAL; (* 22. slave maximum position *)

    (* maximum Slave velocity *)
    fMPosAtSVeloMax     : LREAL; (* 23. master position AT slave maximum velocity *)
```

```
    fSlaveVeloMax       : LREAL; (* 24. slave maximum velocity *)

    (* maximum slave acceleration *)
    fMPosAtSAccMax      : LREAL; (* 25. master position AT slave maximum acceleration *)
    fSlaveAccMax        : LREAL; (* 26. slave maximum acceleration *)
    fSVeloAtSAccMax     : LREAL; (* 27. slave velocity AT slave maximum acceleration *)

    (* maximum Slave slave jerk and dynamic momentum *)
    fSlaveJerkMax       : LREAL; (* 28. slave maximum jerk *)
    fSlaveDynMomMax     : LREAL; (* 29. slave maximum dynamic momentum (NOT SUPPORTED YET !) *)

    (* mean and effective values *)
    fSlaveVeloMean      : LREAL; (* 30. slave mean absolute velocity (NOT SUPPORTED YET !) *)
    fSlaveAccEff        : LREAL; (* 31. slave effective acceleration (NOT SUPPORTED YET !) *)

    (* reserved space for future extension *)
    reserved            : ARRAY[32..47] OF LREAL;

    (* organization structure of the cam table *)
    CamTableID          : UDINT;
    NumberOfRows        : UDINT; (* number of cam table entries, e.g. number of points *)
    NumberOfColumns     : UDINT; (* number of table columns, typically 1 or 2 *)
    TableType           : UINT;  (* MC_TableType *)
    Periodic            : BOOL;

    reserved2           : ARRAY[1..121] OF BYTE;
END_STRUCT
END_TYPE
```

Type definition for the characteristic parameters of a flying saw synchronization.

5.4 ST_GearInPosOptions

```
TYPE ST_GearInPosOptions :
STRUCT
    PositionLimitMin:             LREAL ;
    PositionLimitMax:             LREAL ;
END_STRUCT
END_TYPE
```

Parameter	Description
PositionLimitMin	Check is also carried out to see if GearInSync_CheckMask_OptionalMinPos is set to TRUE.
PositionLimitMax	Check is also carried out if GearInSync_CheckMask_OptionalMaxPos is set to TRUE.

See also:

- ST_SyncMode [▶ 16]

5.5 ST_GearInVeloOptions

```
TYPE ST_GearInVeloOptions :
STRUCT
    PositionLimitMin:             LREAL ;
    PositionLimitMax:             LREAL ;
END_STRUCT
END_TYPE
```

Parameter	Description
PositionLimitMin	Check is also carried out to see if GearInSync_CheckMask_OptionalMinPos is set to TRUE.
PositionLimitMax	Check is also carried out if GearInSync_CheckMask_OptionalMaxPos is set to TRUE.

See also:

- ST_SyncMode [▶ 16]

6 Sample program

Documents about this

- Sample Flying Saw.zip (Resources/zip/9007201828535435.zip)

7 Support and Service

Beckhoff and their partners around the world offer comprehensive support and service, making available fast and competent assistance with all questions related to Beckhoff products and system solutions.

Download finder

Our download finder contains all the files that we offer you for downloading. You will find application reports, technical documentation, technical drawings, configuration files and much more.

The downloads are available in various formats.

Beckhoff's branch offices and representatives

Please contact your Beckhoff branch office or representative for local support and service on Beckhoff products!

The addresses of Beckhoff's branch offices and representatives round the world can be found on our internet page: www.beckhoff.com

You will also find further documentation for Beckhoff components there.

Beckhoff Support

Support offers you comprehensive technical assistance, helping you not only with the application of individual Beckhoff products, but also with other, wide-ranging services:

- support
- design, programming and commissioning of complex automation systems
- and extensive training program for Beckhoff system components

Hotline:	+49 5246 963-157
e-mail:	support@beckhoff.com

Beckhoff Service

The Beckhoff Service Center supports you in all matters of after-sales service:

- on-site service
- repair service
- spare parts service
- hotline service

Hotline:	+49 5246 963-460
e-mail:	service@beckhoff.com

Beckhoff Headquarters

Beckhoff Automation GmbH & Co. KG

Huelshorstweg 20
33415 Verl
Germany

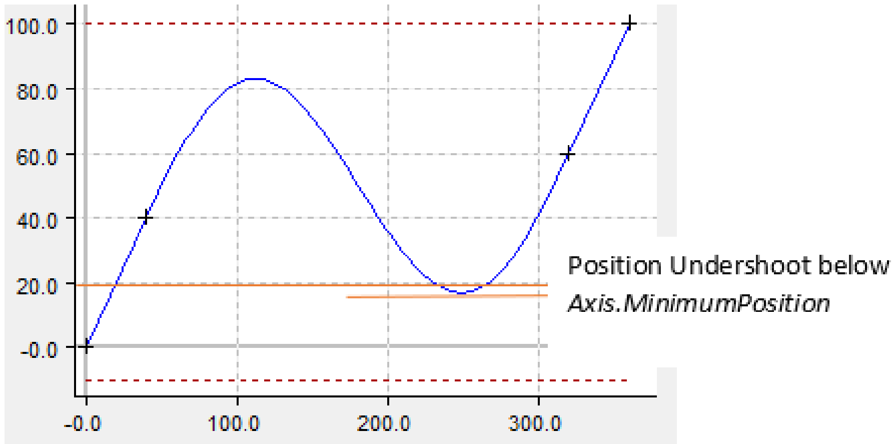
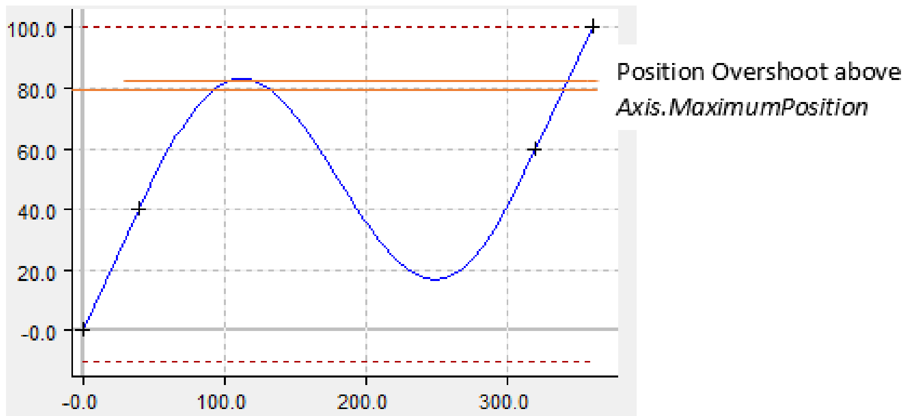
Phone:	+49 5246 963-0
e-mail:	info@beckhoff.com
web:	www.beckhoff.com

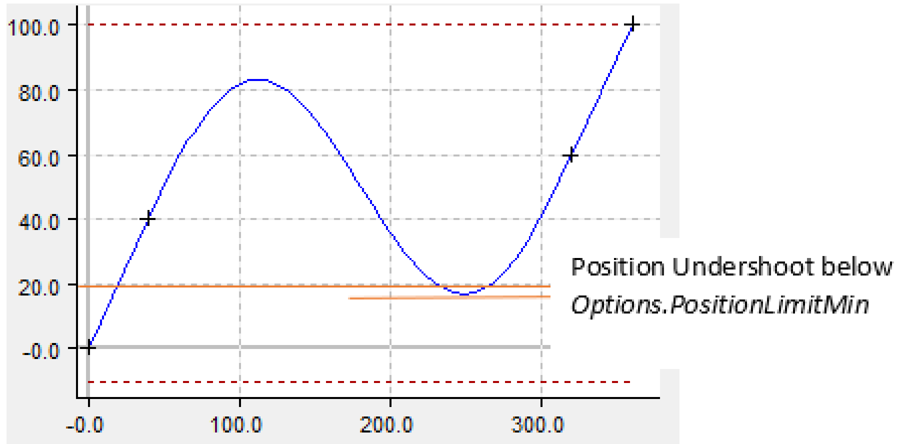
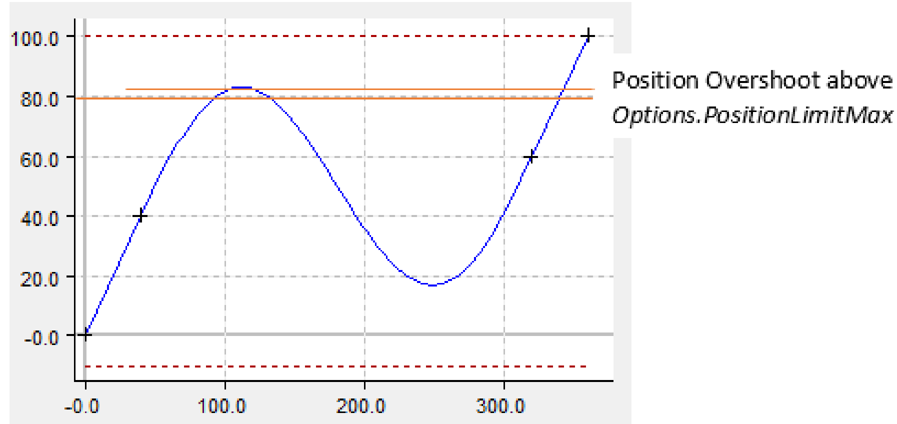
8 Appendix

8.1 Error Codes

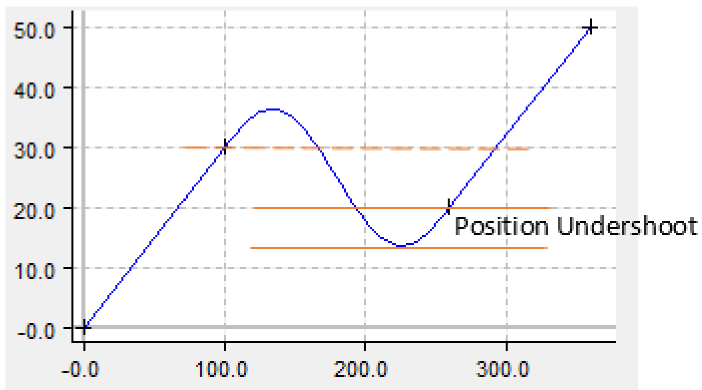
If you have set GearInPos.SyncMode.GearInSync_OpMask_DetailedErrorCodes := TRUE, the flying saw checks return the detailed messages described below. Otherwise the error number 42DF is output in all checking cases.

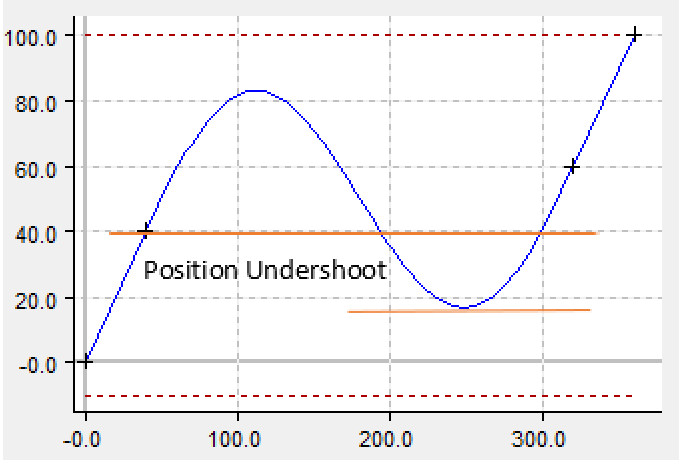
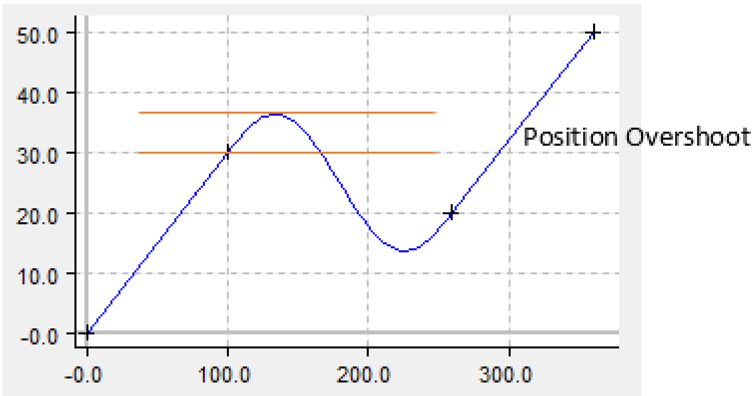
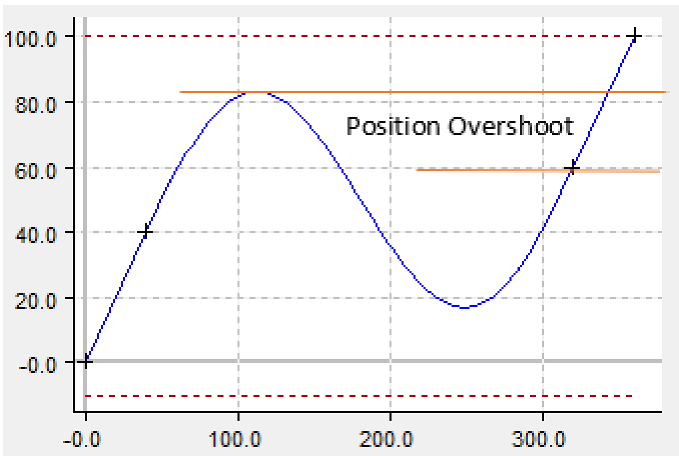
Checks for position limits

Error (hex)	Error (Dec)	Description
4372	17266	The synchronization profile would violate the lower end position of the slave.  Check when GearInSync_CheckMask_MinPos is active.
4373	17267	The synchronization profile would violate the upper end position of the slave.  Check when GearInSync_CheckMask_MaxPos is active.
4374	17268	The synchronization profile would violate the user limit position Options.PositionLimitMin.

Error (hex)	Error (Dec)	Description
		 Check when GearInSync_CheckMask_OptionalMinPos is active.
4375	17269	The synchronization profile would violate the user limit position Options.PositionLimitMax.  Check when GearInSync_CheckMask_OptionalMaxPos is active.

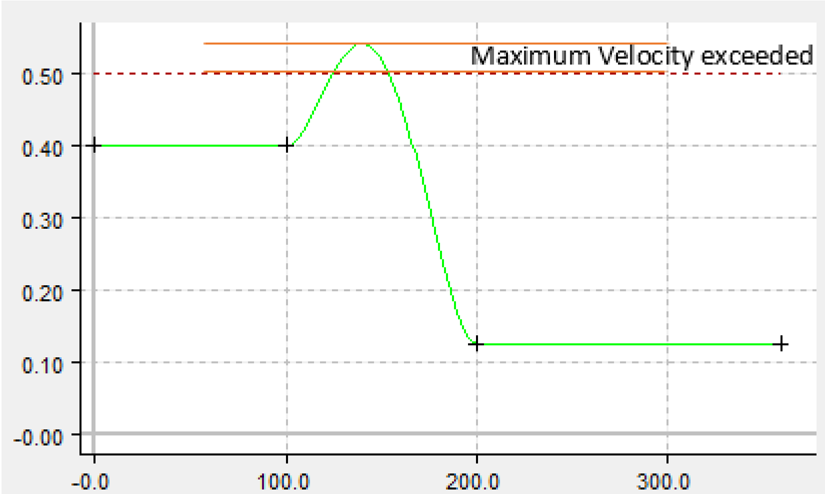
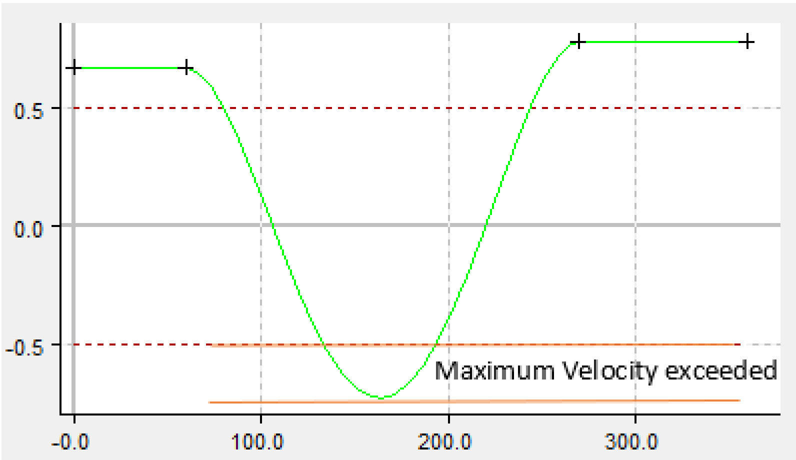
Checks for position overshoots/undershoots

Error (hex)	Error (Dec)	Description
4376	17270	The synchronization point lies under the starting point. As a result, the profile swings under both the start position and the synchronous position.  Check when GearInSync_CheckMask_UndershootPos is active.
4377	17271	The synchronization profile would swing back under the slave start position of the flying saw.

Error (hex)	Error (Dec)	Description
		 Check when GearInSync_CheckMask_UndershootPos is active.
4378	17272	The synchronization point lies under the starting point. The synchronization profile would swing beyond the slave start position of the flying saw.  Check when GearInSync_CheckMask_OvershootPos is active.
4379	17273	The synchronization profile would swing beyond the slave synchronous position of the flying saw.  Check when GearInSync_CheckMask_OvershootPos is active.

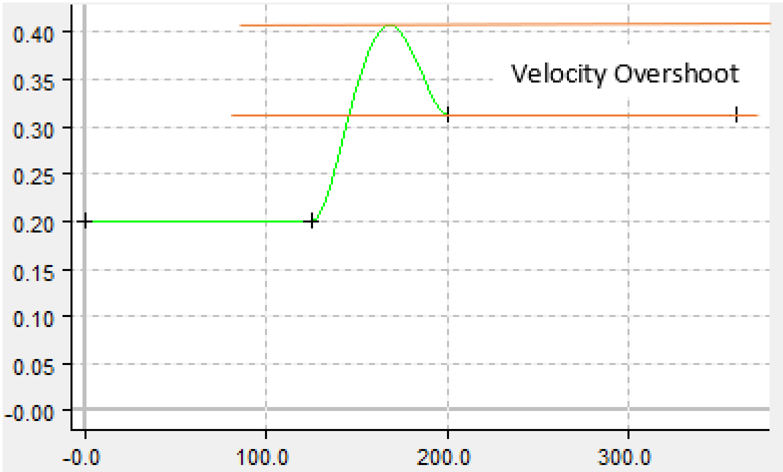
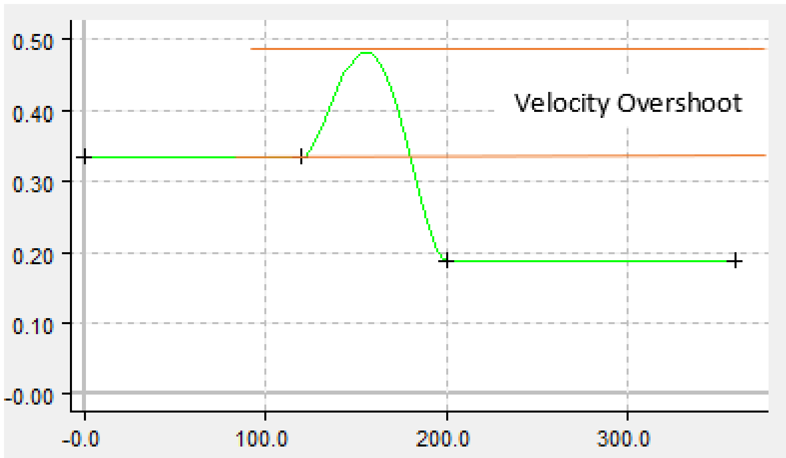
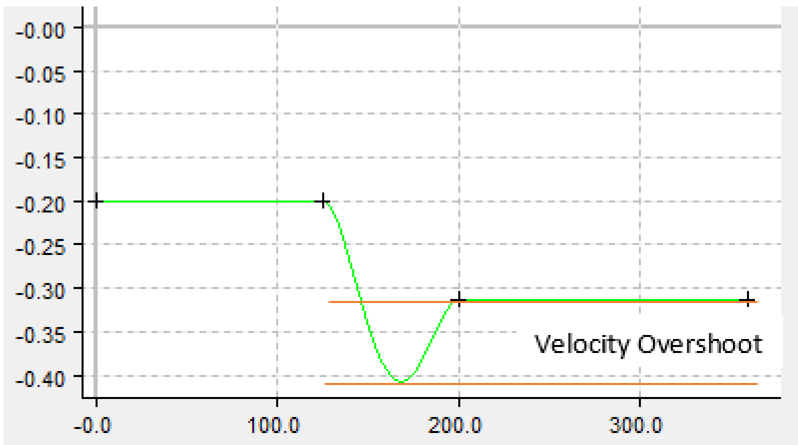
Checks for maximum velocity

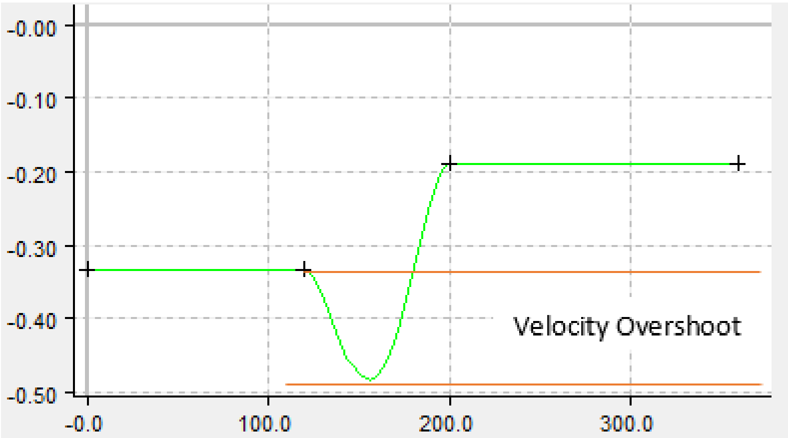
Error (hex)	Error (Dec)	Description
437A	17274	The maximum velocity of the synchronization profile is higher than the maximum velocity of the slave axis.

Error (hex)	Error (Dec)	Description
		 Check when GearInSync_CheckMask_MaxVelo is active.
437B	17275	The maximum velocity of the synchronization profile is higher than the maximum velocity of the slave axis.  Check when GearInSync_CheckMask_MaxVelo is active.

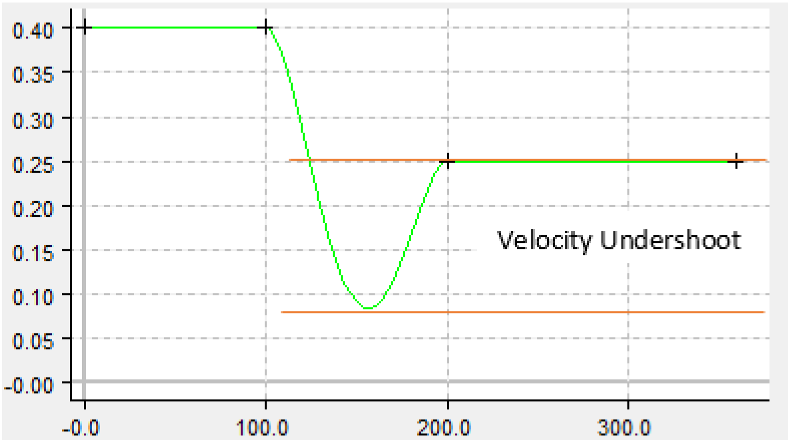
Checks for velocity overshoots

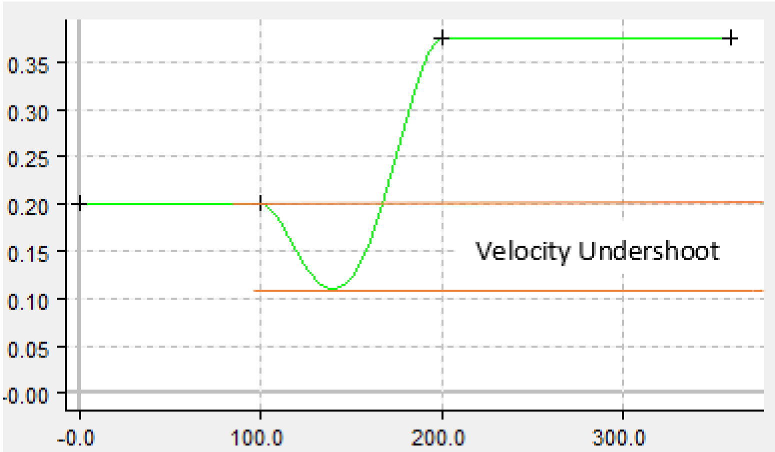
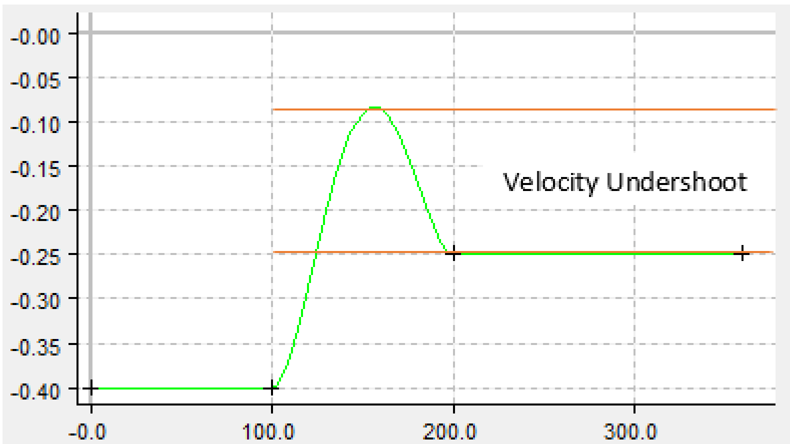
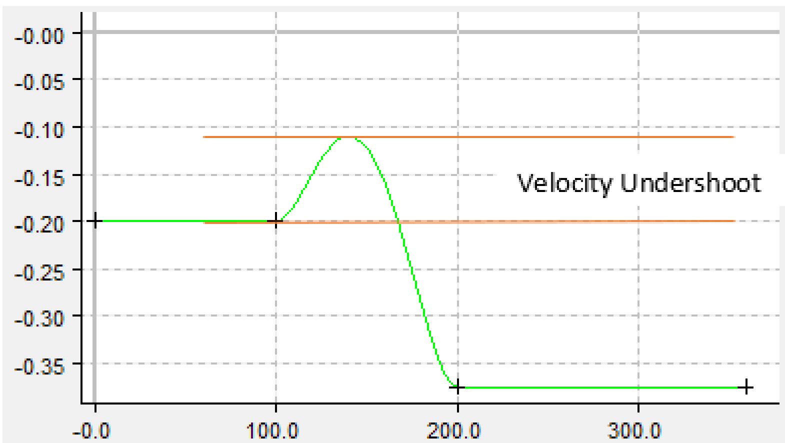
Error (hex)	Error (Dec)	Description
437C	17276	The maximum velocity of the synchronization profile would be higher than the synchronous velocity. (Positive direction of travel and starting velocity lower than synchronous velocity)

Error (hex)	Error (Dec)	Description
		 Check when GearInSync_CheckMask_OvershootVelo is active.
437D	17277	The maximum velocity of the synchronization profile would be higher than the starting velocity and synchronous velocity. (Positive direction of travel and starting velocity higher than synchronous velocity)  Check when GearInSync_CheckMask_OvershootVelo is active.
437E	17278	The maximum velocity of the synchronization profile would be higher than the synchronous velocity. (Negative direction of travel and starting velocity lower than synchronous velocity) 

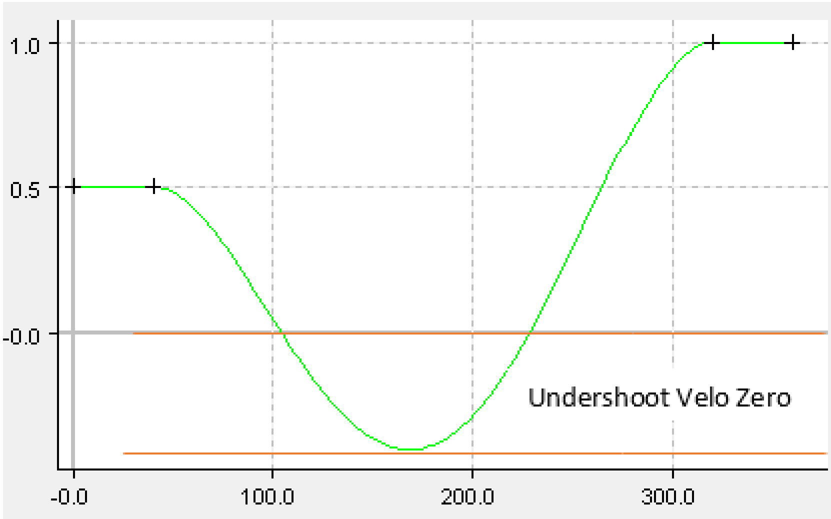
Error (hex)	Error (Dec)	Description
		Check when GearInSync_CheckMask_OvershootVelo is active
437F	17279	The maximum velocity of the synchronization profile would be higher than the starting velocity and synchronous velocity. (Negative direction of travel and starting velocity higher than synchronous velocity)  Check when GearInSync_CheckMask_OvershootVelo is active.

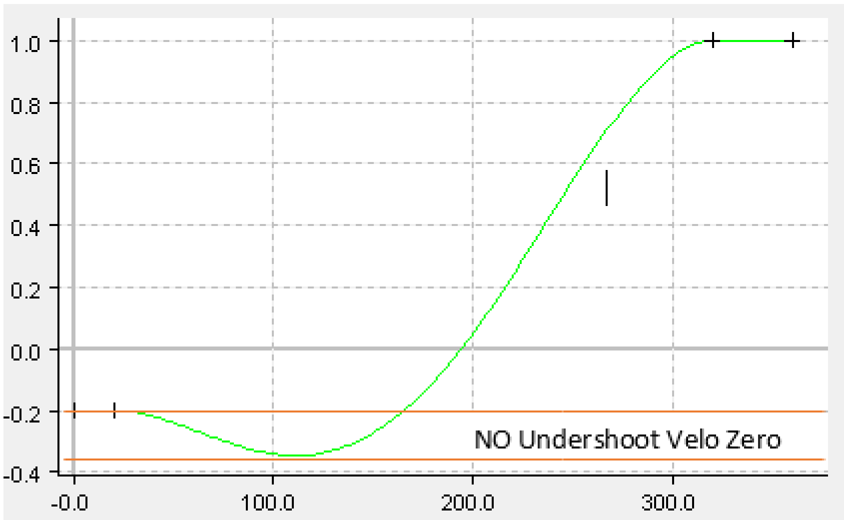
Checks for velocity undershoots

Error (hex)	Error (Dec)	Description
4380	17280	The minimum velocity of the synchronization profile lies below the synchronous velocity. (Positive direction of travel and starting velocity higher than synchronous velocity)  Check when GearInSync_CheckMask_UndershootVelo is active.
4381	17281	The minimum velocity of the synchronization profile lies below the starting velocity. (Positive direction of travel and starting velocity lower than synchronous velocity)

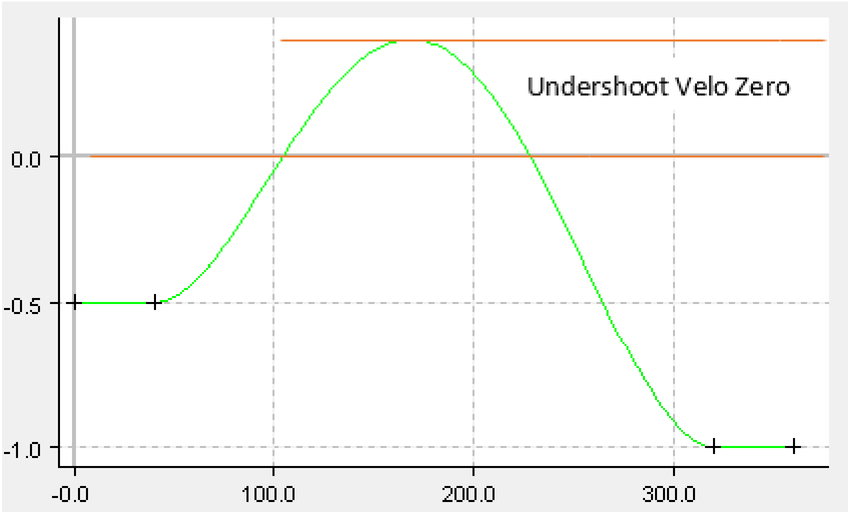
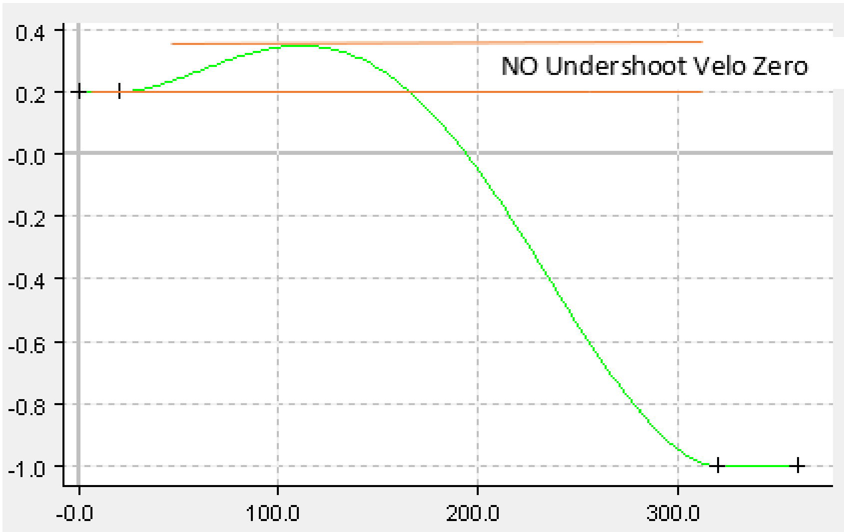
Error (hex)	Error (Dec)	Description
		 Check when GearInSync_CheckMask_UndershootVelo is active.
4382	17282	The minimum velocity of the synchronization profile lies below the synchronous velocity. (Negative direction of travel and starting velocity higher than synchronous velocity)  Check when GearInSync_CheckMask_UndershootVelo is active.
4383	17283	The minimum velocity of the synchronization profile lies below the starting velocity. (Negative direction of travel and starting velocity lower than synchronous velocity)  Check when GearInSync_CheckMask_UndershootVelo is active.

Checks for reversal of the direction of travel

<table>
<tr><th>Error (hex)</th><th>Error (Dec)</th><th>Description</th></tr>
<tr><td>4386</td><td>17286</td><td>The velocity of the flying saw swings below zero; the motion is reversed.
If the slave is already moving in the opposite direction at the beginning, it is not regarded as UndershootVeloZero.
(Master moves in the positive direction)




Check when GearInSync_CheckMask_UndershootVeloZero is active.</td></tr>
<tr><td>4387</td><td>17287</td><td>The velocity of the flying saw swings below zero; the motion is reversed.
If the slave is already moving in the opposite direction at the beginning, it is not regarded as UndershootVeloZero.
(Master moves in the negative direction)</td></tr>
</table>

Error (hex)	Error (Dec)	Description
		  Check when GearInSync_CheckMask_UndershootVeloZero is active.

Checks for maximum acceleration

Error (hex)	Error (Dec)	Description
4388	17288	The maximum acceleration of the synchronization profile would be higher than the maximum acceleration of the slave axis. Check when GearInSync_CheckMask_MaxAcc is active.
4389	17289	The maximum deceleration of the synchronization profile would be higher than the maximum deceleration of the slave axis. Check when GearInSync_CheckMask_MaxDec is active.

Checks for the maximum jerk

Error (hex)	Error (Dec)	Description
438A	17290	The maximum jerk of the synchronization profile would be higher than the maximum jerk of the slave axis. Check when GearInSync_CheckMask_MaxJerk is active (check the SlaveJerkMax).
438B	17291	The maximum jerk of the synchronization profile would be higher than the maximum jerk of the slave axis.

Error (hex)	Error (Dec)	Description
		Check when GearInSync_CheckMask_MaxJerk is active (check the SlaveJerkMin).

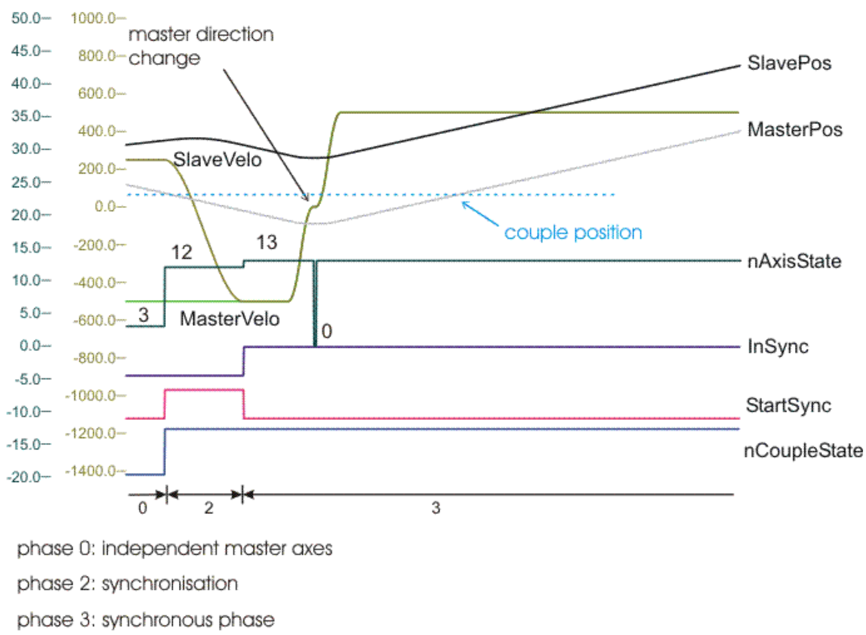
8.2 Reversal of the master axis movement / backstop

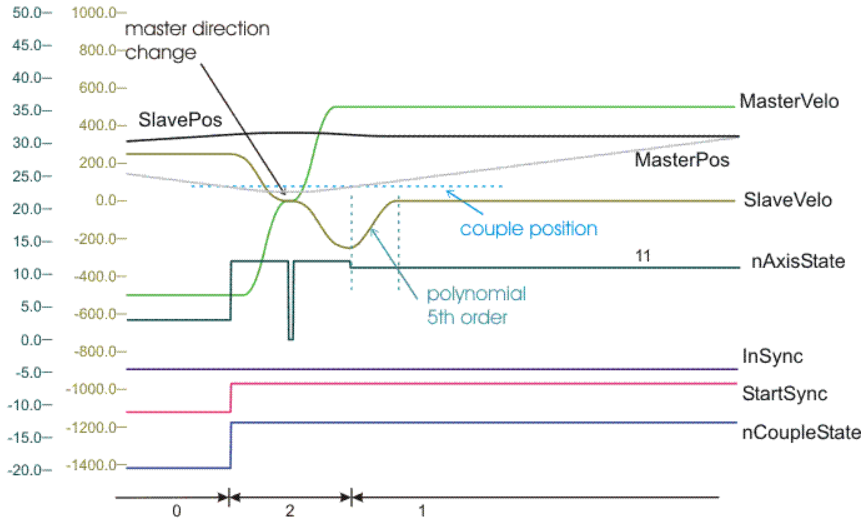
The behavior of the Universal Flying Saw in the event of a motion reversal of the master can be defined via 2 bits of the SyncMode. The *GEARINSYNC_OPMASK_ROLLBACKLOCK* bit activates the backstop, which immobilizes the slave if the master moves backwards beyond the coupling position (the position where the Universal Flying Saw was started) after a motion reversal. The second bit, *GEARINSYNC_OPMASK_INSTANTSTOPONROLLBACK*, governs dynamic aspects of how the slave comes to a halt.

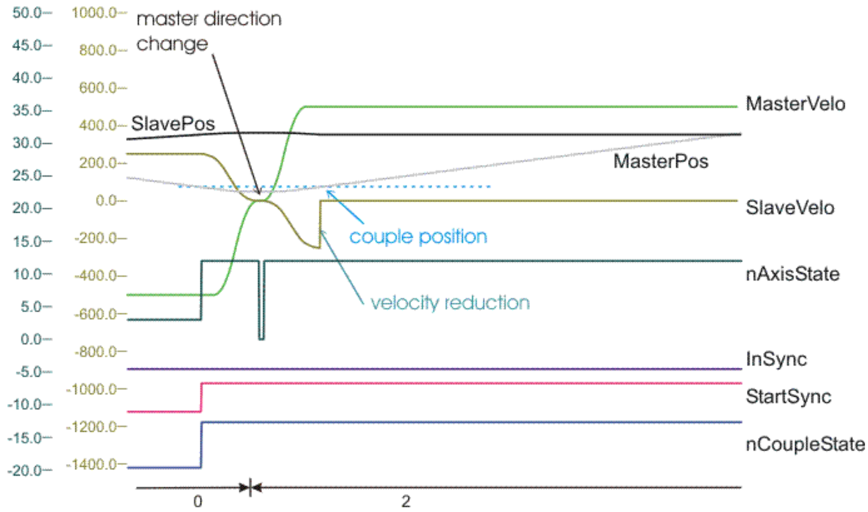
NOTICE

The effect of these two bits must be differentiated according to whether the synchronous phase is reached before the motion reversal, or whether the motion reversal already occurs in the synchronization phase.

The following overview explains the effect of the *GEARINSYNC_OPMASK_ROLLBACKLOCK* and *GEARINSYNC_OPMASK_INSTANTSTOPONROLLBACK* bits in detail.

ROLL BACK LOCK	*INSTANT STOP ON ROLLBACK*	Synchro-nous phase reached?	Description
0	0	yes	**Case 1:** With the bit combination specified on the left, synchronous coupling is maintained for all master movements once the Universal Flying Saw is in the synchronous phase. In the figure below, a motion reversal of the master axis occurs in the synchronous phase, so that it moves backwards beyond the coupling position. The synchronous coupling is maintained here, so that the slave also moves backwards beyond the coupling position. 
0	0	no	**Case 2:** Before reaching the synchronous phase, the motion of the master axis is reversed in the synchronization phase so that it moves backwards beyond the coupling position. Since the synchronous phase has not been

<table>
<tr><th>ROLL BACK LOCK</th><th>INSTANT STOP ON ROLLBACK</th><th>Synchro-nous phase reached?</th><th>Description</th></tr>
<tr><td></td><td></td><td></td><td>reached in this case, the velocity of the slave axis is reduced with a 5th order polynomial when the coupling position is reached and the axis is brought to a standstill.


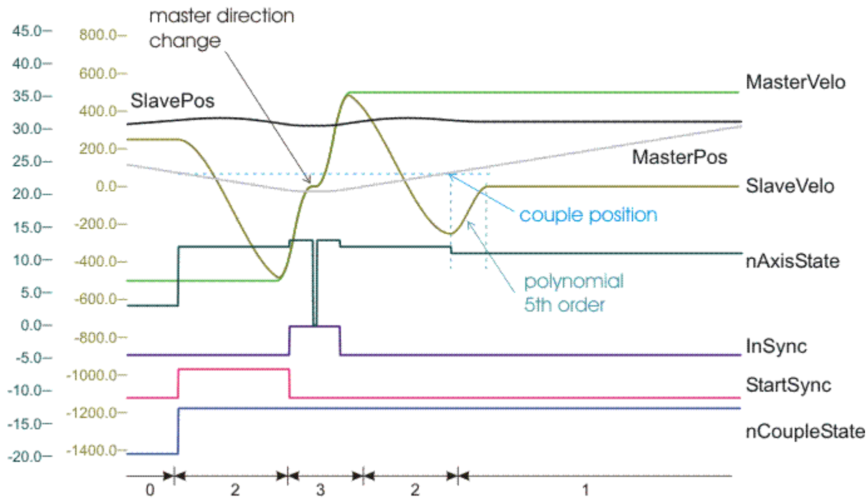
phase 0: independent master axes
Phase 1: pre phase
Phase 2: synchronisation</td></tr>
<tr><td>0</td><td>1</td><td>no</td><td>Case 3:
As in case 2, except that the velocity of the slave axis is reduced to zero within one tick after reaching the coupling position.
The velocity reduction in one tick can trigger the following error monitoring of the axis.


phase 0: independent master axes
phase 2: synchronisation</td></tr>
<tr><td>0</td><td>1</td><td>yes</td><td>Case 4:
Behavior is identical to case 1.</td></tr>
<tr><td>1</td><td>0</td><td>no</td><td>Case 5:</td></tr>
</table>

<table>
<tr><th>ROLL BACK LOCK</th><th>INSTANT STOP ON ROLLBACK</th><th>Synchro-nous phase reached?</th><th>Description</th></tr>
<tr><td></td><td></td><td></td><td>Behavior is identical to case 2.</td></tr>
<tr><td>1</td><td>0</td><td>yes</td><td>

Case 6:

The bit combination indicated on the left is used to activate the backstop of the Universal Flying Saw. In the figure below, a motion reversal of the master axis occurs in the synchronous phase, so that it moves backwards beyond the coupling position. With this backward movement of the master axis, the velocity of the slave axis is reduced to zero with a 5th order polynomial as soon as the coupling position is reached. Reverse movement of the slave axis is therefore prevented in that the slave velocity is continuously reduced as soon as the coupling position is reached.



phase 0: independent master axes

phase 1: pre phase

phase 2: synchronisation

phase 3: synchronous phase

</td></tr>
<tr><td>1</td><td>1</td><td>no</td><td>

Case 7:

Behavior is identical to case 3.

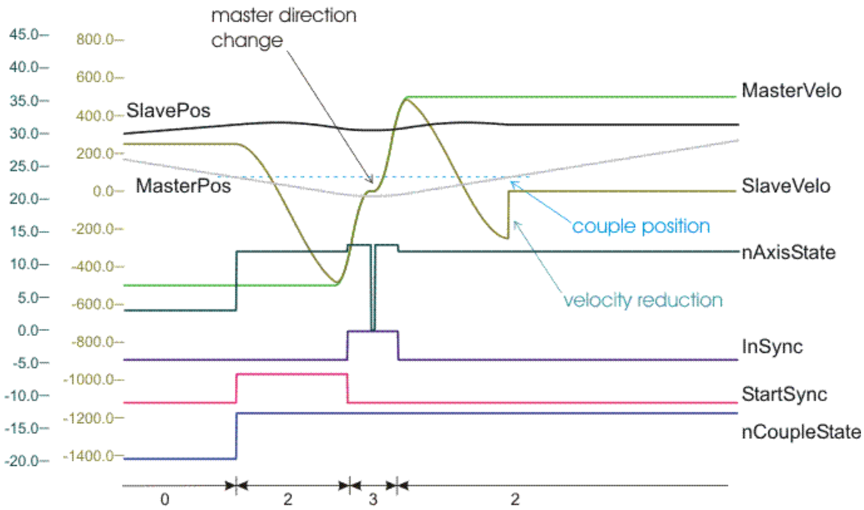
</td></tr>
<tr><td>1</td><td>1</td><td>yes</td><td>

Case 8:

The bit combination indicated on the left is used to activate the backstop of the Universal Flying Saw. In the figure below, a motion reversal of the master axis occurs in the synchronous phase, so that it moves backwards beyond the coupling position. During this backward movement of the master axis, the velocity of the slave axis is reduced to zero in one tick as soon as the coupling position is reached. Reverse movement of the slave axis is therefore prevented in that the slave velocity is reduced as soon as the coupling position is reached.

The velocity reduction in one tick can trigger the following error monitoring of the axis.

</td></tr>
</table>

ROLL BACK LOCK	*INSTANT STOP ON ROLLBACK*	**Synchro-nous phase reached?**	**Description**
			 phase 0: independent master axes phase 2: synchronisation phase 3: synchronous phase

8.3 Calculating the synchronisation phase

An attempt is made when calculating the synchronisation phase to find an optimum solution while observing the boundary conditions specified by the user. If it is not possible to observe the specified boundary conditions, the coupling is declined and an appropriate error message is issued.

Optimizations

As can be seen in the flow chart below, the individual bit masks partly influence the internal optimization steps of the profile calculation, since an optimum is searched for depending on predefined rules (see Parameterizable boundary conditions). Essentially, a 5th order polynomial or a combination of a 5th order polynomial with a 1st order polynomial is used. A 5th order polynomial is generally not free from overshoot, but the accelerations are more moderate than when combining a 5th order polynomial with a 1st order polynomial. The combination of the 1st and 5th order polynomials is calculated in such a way that it is always free from overshoot. However, higher accelerations and decelerations occur with it. If, for example, the actual velocity matches the synchronous velocity, but a certain position difference must be compensated. Then the optimum velocity is calculated internally as a function of the maximum acceleration. Result is a 5th power polynomial a 1st order polynomial with the calculated velocity and a 5th power polynomial. At least one of the two 5th power polynomials exploits the maximum acceleration. To avoid extreme jerk values, the jerk check should be switched on.

The optimizations shown can only be carried out if both the master and slave axes are free of acceleration at the time of coupling. For accelerated axes, a 5th order polynomial is used for synchronization, which is checked for compliance with the specified boundary conditions, but cannot be optimized.

NOTICE

If the master axis is an encoder axis (an "external encoder system"), something which as a rule is never mathematically free from acceleration, particular care must be taken to filter the actual acceleration. Alternatively the determination of the actual acceleration can be deselected in the encoder, i.e. set to zero. The NC also has an internal algorithm for this combination (master encoder axis with the Universal Flying Saw as a slave). This algorithm sets master accelerations whose magnitudes are less than (2.0 • scaling factor / cycle time^2) to zero at the coupling time.

Optimisation step 1:

Aim: "Velocity profile free from undershoot or overshoot"

An attempt is first made to calculate a profile that synchronises the velocity without overshoot or undershoot (a combination of a first order polynomial and a 5th order polynomial or vice versa: in abbreviated form, polynomial1+polynomial5 or polynomial5+polynomial1). If the acceleration check is active at this stage, and if one or more of these limit values (acc, dec) is exceeded, then another profile, which in general is not free from overshoot (polynomial5) is calculated. If one of the active limit values (acc, dec) is still exceeded with his profile, then the synchronisation command is finally declined with an error code.

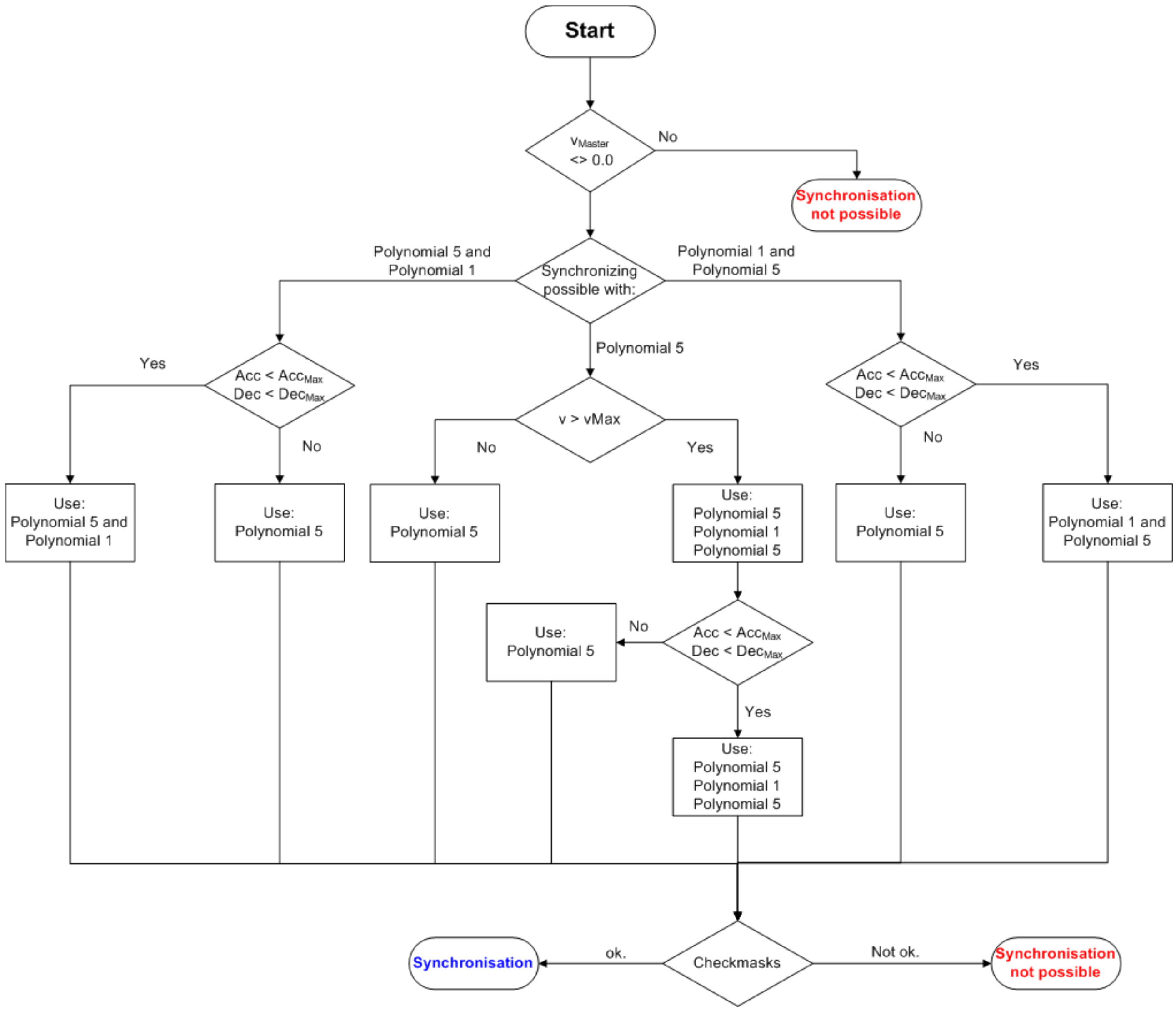
Optimisation step 2:

Aim: "Limitation to maximum permitted velocity"

If the first optimization step is not possible, the second optimization step checks whether the maximum permitted velocity for the slave axis is exceeded by a general standard profile (polynomial5). If this is the case, an attempt is made to generate a profile in which the maximum profile velocity is precisely the maximum velocity permitted to the slave axis (machine data) (polynomial5+polynomial1+polynomial5). It should be noted that this optimization attempt usually results in larger values of acceleration or deceleration. If the acceleration check is active at this stage, and if one or more of these limit values is exceeded, then this second optimization step is rejected, and finally a profile, which is not in general free from overshoot (polynomial5), is calculated. If one of the active limit values (acc, dec) is still exceeded with his profile, then the synchronization command is finally declined with an error code.

Optimisation flow diagram:

The optimizations carried out internally are illustrated in the following flow diagram. Essentially, the slave set value profile is calculated as a 5th order polynomial. This 5th order polynomial can be combined with a first order polynomial in order to maintain the parameterized boundary conditions. The way in which the individual boundary conditions influence the selection of the polynomials, and therefore the form of the set value profile, can be seen from the flow diagram illustrated below. The label "polynomial n and polynomial m" expresses the fact that polynomial n is first used in the synchronization phase, followed by polynomial m.


Start
vMaster
<> 0.0
No
Synchronisation
not possible
Polynomial 5 and
Polynomial 1
Synchronizing
possible with:
Polynomial 1 and
Polynomial 5
Polynomial 5
Yes
Acc < AccMax
Dec < DecMax
No
v > vMax
No
Yes
Acc < AccMax
Dec < DecMax
Yes
No
Use:
Polynomial 5 and
Polynomial 1
Use:
Polynomial 5
Use:
Polynomial 5
Use:
Polynomial 5
Polynomial 1
Polynomial 5
Use:
Polynomial 5
Use:
Polynomial 1 and
Polynomial 5
Use:
Polynomial 5
No
Acc < AccMax
Dec < DecMax
Yes
Use:
Polynomial 5
Polynomial 1
Polynomial 5
Synchronisation
ok.
Checkmasks
Not ok.
Synchronisation
not possible

More Information:
www.beckhoff.com/tf5055

Beckhoff Automation GmbH & Co. KG
Hülshorstweg 20
33415 Verl
Germany
Phone: +49 5246 9630
info@beckhoff.com
www.beckhoff.com