

Technology module



Winder Dancer-controlled _____

Reference manual

EN

Contents

1	About this document	4
1.1	Document history	6
1.2	Conventions used	7
1.3	Layout of the safety instructions	8
2	Safety instructions	9
3	Overview of the technology module functions	11
3.1	Functional description	11
3.2	Functional overview of the TM	12
3.3	Important notes on how to operate the technology module	13
4	Application ranges	15
5	L_TW2P_WinderDancerCtrlState function block	16
5.1	Inputs and outputs	18
5.2	Inputs	19
5.3	Outputs	23
5.4	Parameter	26
6	State machine	32
7	Signal flow diagram	33
7.1	Structure of the signal flow	34
7.2	Structure of the access points	36
8	Description of the basic functions	37
8.1	Manual jog (jogging)	38
8.2	Homing	40
8.3	QSP application	43
8.4	Halt	44
8.5	Absolute positioning	45
8.6	Relative positioning	47
8.7	Continuous travel with a defined velocity	50
8.8	Aborting and completing a positioning process	53
8.9	Override for speed	55
9	Description of the technology function	57
9.1	Defining the winding direction (winding/unwinding)	57
9.2	Defining the material feed to the winder	58
9.3	Master value source for the diameter calculation	60
9.4	Speed feedforward control	61
9.5	Diameter calculation	62
9.6	Holding the diameter	64
9.7	Defining the diameter via a sensor	65
9.8	Diameter calculation including consideration of the dancer movement	67
9.9	Identification of the diameter by lifting the dancer	69
9.10	Material length counter	73
9.10.1	Material length counter via 'IrSetLineVel' input	74
9.10.2	Material length counter via 'IrSetLineVelDiamCalc' input	75
9.10.3	Material length counter via 'MaterialCounterAxis' reference axis	76
9.11	Inching mode with 1/d evaluation	77
9.12	Synchronisation to the line velocity	79
9.13	Scaling of the dancer position	80
9.14	Teaching function for dancer limit positions	82

Contents

9.15	Monitoring of the dancer position	84
9.16	PI controller for the dancer position control	85
9.17	Reduced controller dynamics at low system deviation	87
9.18	Tension control via characteristic function (Base version)	88
9.18.1	Linear tensile force characteristic (dwSelectTensCurve = 0)	89
9.18.2	Linear torque characteristic (dwSelectTensCurve = 1)	90
9.18.3	Freely definable characteristics with 65 interpolation points (dwSelectTensCurve = 2) _	91
9.19	Web break monitoring	92
9.20	Persistent variables	94
9.21	Acceleration compensation	96
9.22	Identification of the moments of inertia	98
9.23	Adaptation of the speed controller gain	102
9.24	Limitation of the master line speed	104
9.25	Termination of the winding process	106
	Index	108
	Your opinion is important to us	110

1 About this document

This documentation ...

- contains detailed information on the functionalities of the "Winder Dancer-controlled" technology module;
- is part of the "Controller-based Automation" manual collection. It consists of the following sets of documentation:

Documentation type	Subject
Product catalogue	Controller-based Automation (system overview, sample topologies) Lenze controllers (product information, technical data)
System manuals	Visualisation (system overview/sample topologies)
Communication manuals Online helps	Bus systems • Controller-based Automation EtherCAT® • Controller-based Automation CANopen® • Controller-based Automation PROFIBUS® • Controller-based Automation PROFINET®
Reference manuals Online helps	Lenze controllers: • Controller 3200 C • Controller c300 • Controller p300 • Controller p500
Software manuals Online helps	Lenze engineering tools: • »PLC Designer« (programming) • »Engineer« (parameterisation, configuration, diagnostics) • »VisiWinNET® Smart« (visualisation) • »Backup & Restore« (backup, restore, update)

Further sets of technical documentation with regard to Lenze products

Further information on Lenze products which can be used in conjunction with Controller-based Automation can be found in the following sets of documentation:

Design / configuration / technical data	
☐	Product catalogues • Controller-based Automation • Controllers • Inverter Drives/Servo Drives
Mounting and wiring	
☒	Mounting instructions • Controllers • Communication cards (MC-xxx) • I/O system 1000 (EPM-Sxxx) • Inverter Drives/Servo Drives • Communication modules
☐	Hardware manuals • Inverter Drives/Servo Drives
Parameterisation / configuration / commissioning	
☐	Online help/reference manuals • Controllers • Inverter Drives/Servo Drives • I/O system 1000 (EPM-Sxxx)
☐	Online help/communication manuals • Bus systems • Communication modules
Sample applications and templates	
☐	Online help / software and reference manuals • i700 application sample • Application Samples 8400/9400 • FAST Application Template Lenze/PackML • FAST technology modules

Symbols:

-  Printed documentation
- ☐ PDF file / online help in the Lenze engineering tool



Tip!

Current sets of documentation and software updates with regard to Lenze products can be found in the Download area at:

www.lenze.com

Target group

This documentation is intended for all persons who plan, program and commission a Lenze automation system on the basis of the Lenze FAST Application Software.

1 About this document

1.1 Document history

1.1 Document history

Version			Description
4.0	04/2021	TD06	Error corrections & supplements
3.0	07/2020	TD06	Error corrections & supplements
2.0	12/2019	TD06	Error corrections & supplements
1.0	07/2019	TD06	First edition

1 About this document

1.2 Conventions used

This documentation uses the following conventions to distinguish between different types of information:

Type of information	Writing	Examples/notes
Numeric notation		
Decimal separator	Point	The decimal point is always used. For example: 1234.56
Text		
Program name	» «	»PLC Designer« ...
Variable identifiers	<i>Italics</i>	By setting <i>bEnable</i> to TRUE...
Function blocks	Bold	The L_MC1P_AxisBasicControl function block ...
Function libraries		The L_TT1P_TechnologyModules function library ...
Source code	Font "Courier new"	... dwNumerator := 1; dwDenominator := 1; ...
Symbols		
Page reference	 7)	Reference to further information: Page number in PDF file.

Variable identifiers

The conventions used by Lenze for the variable names of Lenze system blocks, function blocks, and functions are based on the "Hungarian Notation". This notation makes it possible to identify the most important properties (e.g. the data type) of the corresponding variable by means of its name, e.g. xAxisEnabled.

1 About this document

1.3 Layout of the safety instructions

DANGER!

Indicates an extremely hazardous situation. Failure to comply with this instruction will result in severe irreparable injury and even death.

WARNING!

Indicates an extremely hazardous situation. Failure to comply with this instruction may result in severe irreparable injury and even death.

CAUTION!

Indicates a hazardous situation. Failure to comply with this instruction may result in slight to medium injury.

NOTICE

Indicates a material hazard. Failure to comply with this instruction may result in material damage.

Application notes

Pictograph	Signal word	Meaning
	Tip!	Useful tip for easy handling
		Reference to another document

2 Safety instructions

Please observe the safety instructions in this documentation when you want to commission an automation system or a plant with a Lenze Controller.



The device documentation contains safety instructions which must be observed!

Read the documentation supplied with the components of the automation system carefully before you start commissioning the Controller and the connected devices.

DANGER!

High electrical voltage

Injury to persons caused by dangerous electrical voltage

Possible consequences

Death or severe injuries

Protective measures

Switch off the voltage supply before working on the components of the automation system.

After switching off the voltage supply, do not touch live device parts and power terminals immediately because capacitors may be charged.

Observe the corresponding information plates on the device.

DANGER!

Injury to persons

Risk of injury is caused by ...

- unpredictable motor movements (e.g. unintended direction of rotation, too high velocities or jerky movement);
- impermissible operating states during the parameterisation while there is an active online connection to the device.

Possible consequences

Death or severe injuries

Protective measures

- If required, provide systems with installed inverters with additional monitoring and protective devices according to the safety regulations valid in each case (e.g. law on technical equipment, regulations for the prevention of accidents).
- During commissioning, maintain an adequate safety distance to the motor or the machine parts driven by the motor.

NOTICE

Damage or destruction of machine parts

Damage or destruction of machine parts can be caused by ...

- Short circuit or static discharges (ESD);
- unpredictable motor movements (e.g. unintended direction of rotation, too high velocities or jerky movement);
- impermissible operating states during the parameterisation while there is an active online connection to the device.

Protective measures

- Always switch off the voltage supply before working on the components of the automation system.
 - Do not touch electronic components and contacts unless ESD measures were taken beforehand.
 - If required, provide systems with installed inverters with additional monitoring and protective devices according to the safety regulations valid in each case (e.g. law on technical equipment, regulations for the prevention of accidents).
-

3 Overview of the technology module functions

3.1 Functional description

3 Overview of the technology module functions

3.1 Functional description

Variants of winding drives

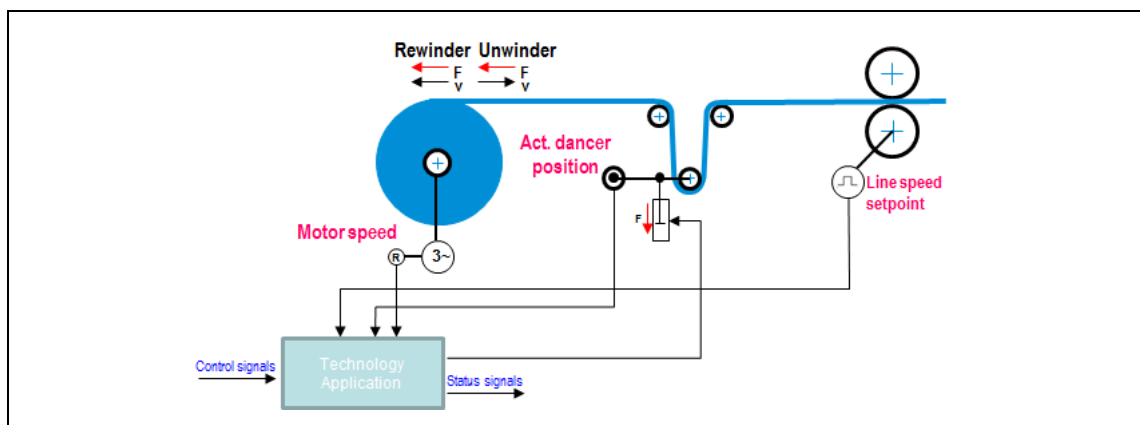
In many technological processes, winding drives are a key component of an entire system. Depending on the material and the winding process, different open and closed-loop control methods are used:

- Dancer position control
- Open-loop tension control
- Closed-loop tension control

Short description

This standardised application can be used to design a dancer position-controlled winding drive.

In the case of the dancer position control, the drive is actuated in closed-loop speed control. The line speed signal multiplied with the reciprocal value of the diameter serves the purpose of feedforward control. The dancer position is recorded and compared with the setpoint position. If a deviation is detected, the dancer position controller provides for a correction of the speed setpoint.



[3-1] Structure of a dancer-controlled winder

- A PI controller can be used for the dancer position control. The dancer limit positions can be set via [L TW2P_scPar_WinderDancerCtrlState](#) (26) identified with the teaching function. Open loop tension control can be set via a linear characteristic function. The tensile force setpoints can be read out directly at the output of the technology module.
- A total of three characteristics are provided for open loop tension control:
 - Characteristic for a linear tensile force profile
 - Characteristic for a linear torque profile
 - User-definable characteristic with 64 grid points

Furthermore, during dancer position control the acceleration compensation can be used for a feedforward control of the torque.

The technology module additionally offers the option to identify the moment of inertia of the winder axis and to integrate this value into the parameterisation of the technology module. Furthermore, the speed controller gain can be adapted during operation as a function of the current moment of inertia.

3.2 Functional overview of the TM

In addition to the basic functions for actuating the **L_MC1P_AxisBasicControlV2** function block, the **Quick stop application function** and the **Holding function**, the technology module offers the following functionalities:

Functionality
Basic Motion Base:
Manual jog (jogging) (📖 38)
Homing (📖 40)
OSP application (📖 43)
Halt (📖 44)
Absolute positioning (📖 45)
Relative positioning (📖 47)
Continuous travel with a defined velocity (📖 50)
Aborting and completing a positioning process (📖 53)
Override for speed (📖 55)
Winder Dancer-controlled:
Defining the winding direction (winding/unwinding) (📖 57)
Automatic detection of the winding direction (📖 57)
Master value source for the diameter calculation (📖 60)
Speed feedforward control (📖 61)
Diameter calculation (📖 62)
Holding the diameter (📖 64)
Defining the diameter via a sensor (📖 65)
Diameter calculation including consideration of the dancer movement (📖 67)
Material length counter (📖 73)
Synchronisation to the line velocity (📖 79)
Scaling of the dancer position (📖 80)
Teaching function for dancer limit positions (📖 82)
Monitoring of the dancer position (📖 84)
PI controller for the dancer position control (📖 85)
Tension control via characteristic function (Base version) (📖 88)
Web break monitoring (📖 92)
Persistent variables (📖 94)
Limitation of the master line speed (📖 104)
Acceleration compensation (📖 96)
Identification of the moments of inertia (📖 98)
Adaptation of the speed controller gain (📖 102)
Identification of the diameter by lifting the dancer (📖 69)
Reduced controller dynamics at low system deviation (📖 87)
Termination of the winding process (📖 106)



»PLC Designer« online help

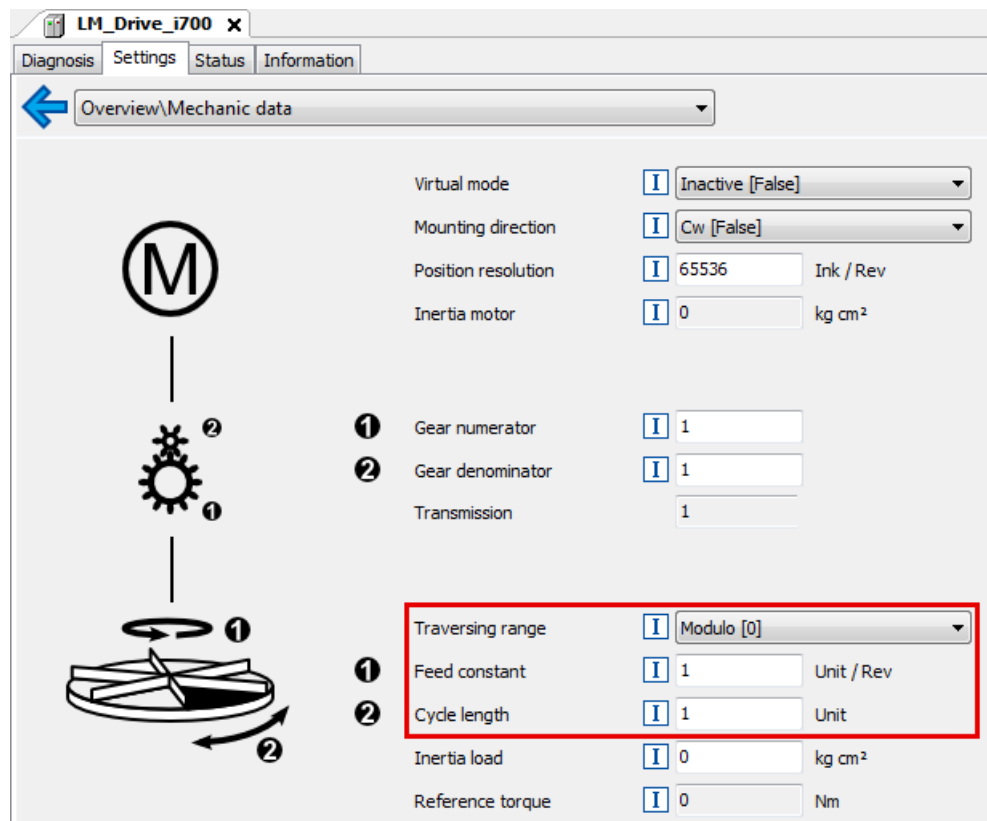
Here you'll find some detailed information with regard to the **L_MC1P_AxisBasicControlV2**.

3.3 Important notes on how to operate the technology module

The technology module ...

- does not support the simulation mode in the »PLC Designer«;
- does not support any virtual axes;
- only supports rotary winder axes.

Go to the »PLC Designer« and set the following parameters for each axis under the **Settings** tab:



Observe that the line speed is specified in the unit [mm/s].

Setting of the operating mode

The operating mode for the winder axis is specified via the technology module. The change-over between "cyclic sync position mode (csp)", "cyclic synchronous speed (csv)" und "cyclic synchronous torque (cst)" in this process internally depends on the active status of the module.

Start of the axes

The method of starting the axes after the controller has been blocked depends on the assigned movement type.

A distinction is made between a movement type whose function is activated by a level and a movement type whose function is activated by an edge.

- The movement type whose function is level-activated is automatically activated after a controller release as long as no higher-priority functions are active. This movement type includes
 - [Manual jog \(jogging\)](#) (438)
 - [QSP application](#) (443)
- The movement type whose function is edge-activated must be activated after a controller release with a FALSE↗TRUE edge. This movement type includes
 - [Homing](#) (440)



Example of a movement type whose function is level-activated: manual jog (jogging)

1. In the inhibited axis state ($xOperationEnabled = FALSE$), $xJogPos$ is set to TRUE.
 - $xEnableOperation = FALSE$ (axis is blocked.)
==> "STANDBY" state ($xOperationEnabled = FALSE$)
 - $xJogPos = TRUE$ (manual jog is to be executed.)
2. Enable axis.
 - $xEnableOperation = TRUE$
==> "JOGGING" state ($xOperationEnabled = TRUE$)



Example of a movement type whose function is edge-activated: reference run (homing)

1. In the inhibited axis state ($xOperationEnabled = FALSE$), $xHomeExecute$ is set to TRUE.
 - $xEnableOperation = FALSE$ (axis is blocked.)
==> "STANDBY" state ($xEnableOperation = FALSE$)
 - $xHomeExecute = TRUE$ (Referencing must be executed.)
2. Enable axis.
 - $xEnableOperation = TRUE$
==> "STANDBY" state (output $xOperationEnabled = TRUE$)
3. Start referencing.
 - $xHomeExecute = FALSE↗TRUE$
==> "HOMING" state

4 Application ranges

When dancer position control is used, the web tension is exclusively generated by a dancer device installed upstream of the winder. The forte of this process is its compensation performance with dynamic disturbances like they can for instance occur during acceleration or in the case of imbalances in the reel. When this happens, the dancer position may vary, however, the tensile force will remain constant. The dancer position control is predominantly used in applications requiring a high absolute tensile force accuracy and consistency.

The limits for the dancer position control are not determined by the torque setting range, but primarily by the mechanical design of the dancer and the dynamics (mass, friction, etc.) and the speed setting range of the drive.

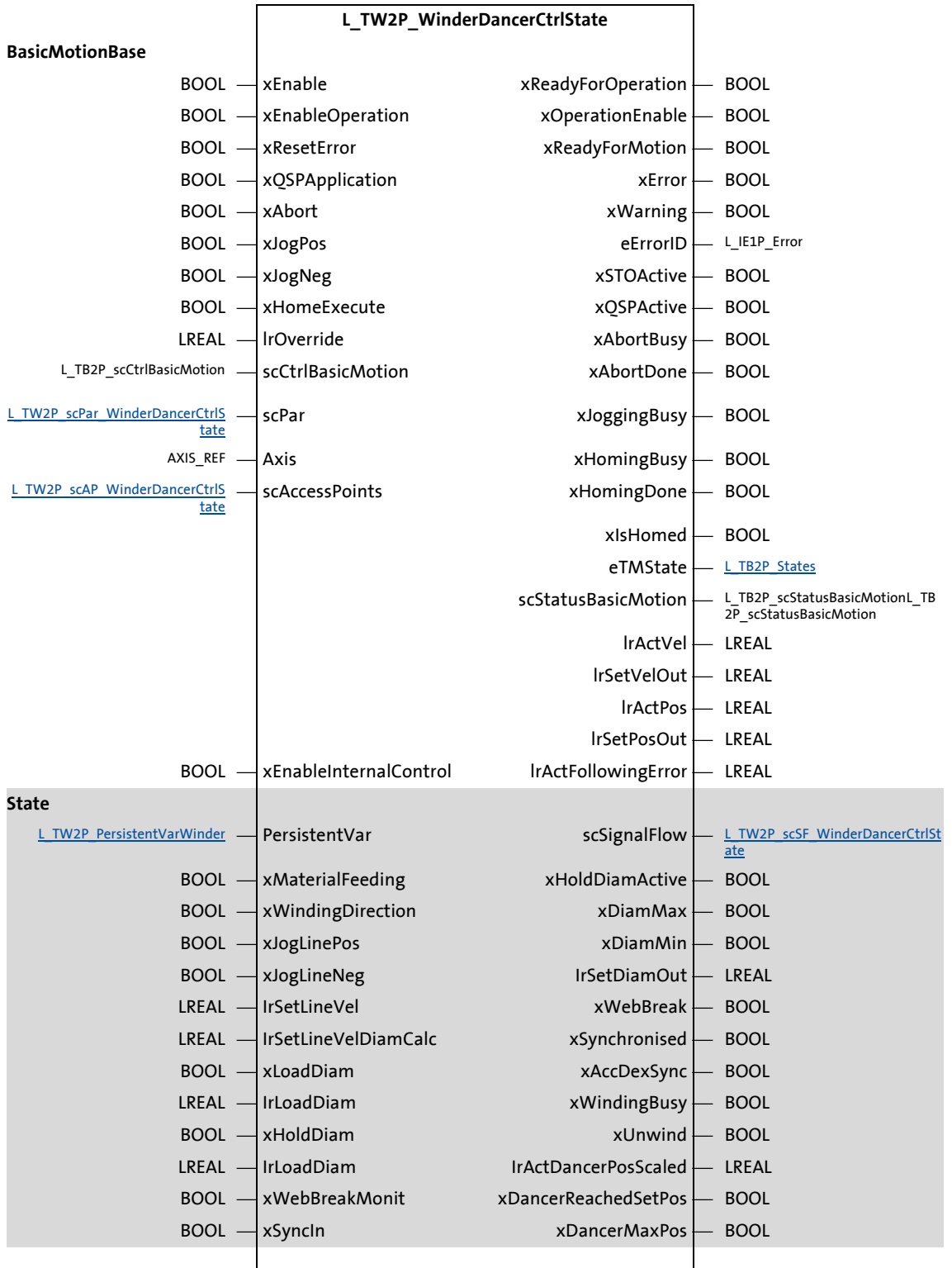
Typical applications with path speeds up to 1000 m / min:

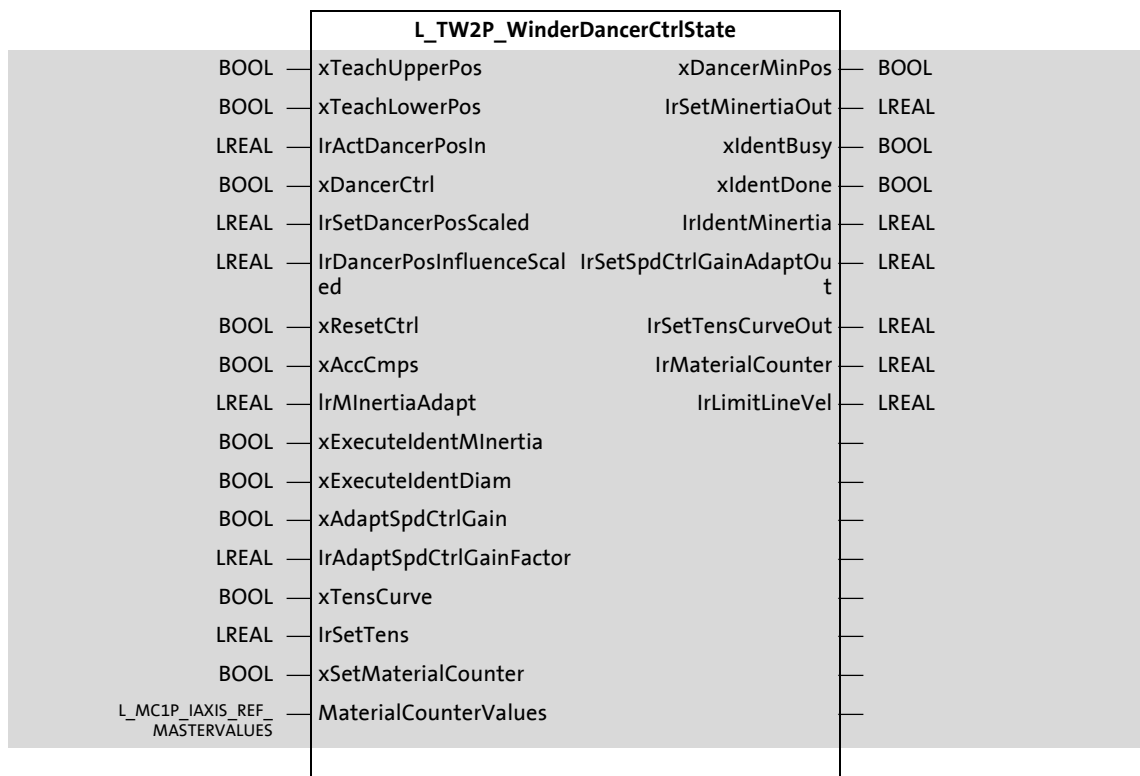
- Winding of round material such as cable or wire
- Textiles, foils, paper

5 L_TW2P_WinderDancerCtrlState function block

The illustration shows that the inputs and outputs are part of the function block.
The additional inputs and outputs of the "State" version are shaded.

► [L_TW2P_scSF_WinderDancerCtrlState](#) (26)





5.1

Inputs and outputs

Identifier	Data type	Description
PersistentVar	L_TW2P_PersistentVarWinder	Reference to persistent variables In the reference, the following data are managed: • Diameter calculated

5.2

Inputs

Identifier	Data type	Description
xEnable	BOOL	Execution of the function block
		TRUE The function block is executed.
		FALSE The function block is not executed.
xEnableOperation	BOOL	TRUE The axis is switched to the "ready for operation" state. Controller enable is issued.
xResetError	BOOL	TRUE Reset axis error or software error.
xQSPApplication	BOOL	TRUE The active movement is cancelled and the axis is brought to a standstill via the QSP application stop ramp. - The state changes to "Stop". - The technology module remains in the "STOP" status as long as <i>xQSPApplication</i> remains TRUE.
xAbort	BOOL	TRUE The active movement is cancelled and the axis is brought to a standstill via the active profile deceleration. The state changes to "Stop". The deceleration ramp can be aborted by commissioning of a new movement (e.g. <i>xMoveVelExecute</i>)
xJogPos	BOOL	TRUE Traverse axis in positive direction (manual jog). If <i>xJogNeg</i> is also TRUE, the <i>eErrorID</i> output displays the InvalidInputJogPosJogNeg error message and the axis stops.
xJogNeg	BOOL	TRUE Traverse axis in negative direction (manual jog). If <i>xJogPos</i> is also TRUE, the <i>eErrorID</i> output displays the InvalidInputJogPosJogNeg error message and the axis stops.
xHomeExecute	BOOL	The input is edge-controlled and evaluates the rising edge.
		FALSE Start homing.
		TRUE The function is aborted via the <i>scCtrlBasicMotion.xHalt</i> structure input.
lrOverride	LREAL	With the <i>lrOverride</i> factor, the defined setpoint speed is multiplied (e.g. an override value of 0.5 generates half the setpoint speed) - Value range: 0 ... 1.0 - Initial value: 1.0 Note: Acceptance of the value is carried out irrespective of the status in <i>xContinuousUpdate</i>
scCtrlBasicMotion L_TB2P_scCtrlBasicMotion		• See description for the L_TB2P_BasicMotionBase technology module
scPar L_TW2P_scPar_WinderDancerCtrlState		The parameter structure contains the parameters of the technology module. The data type depends on the version used (Base/State).
Axis AXIS_REF		Reference to the axis
scAccessPoints L_TW2P_scAP_WinderDancerCtrlState		Structure of the access points Via the access points, the signal flow or the functions of the technology module can be influenced. The data type depends on the version used (Base/State).
PersistentVar L_TW2P_PersistentVarWinder		Content of the persistent variables within the L_TW2P_PersistentVarWinder structure

Identifier	Data type	Description
xMaterialFeeding	BOOL	Material feeding at the reels from the top or from the bottom - Initial value: FALSE ⚠ WARNING! Injury to persons Risk of injury is caused by changing the setting at the <i>xMaterialFeeding</i> input during a movement. This can lead to a speed jump of the winder axis. Possible consequences Death or severe injuries Protective measures Only change the setting of the <i>xMaterialFeeding</i> input if the technology module is not running and the inverter is disabled: <i>xEnable</i> input = FALSE <i>xEnableOperation</i> input = FALSE
		TRUE Material feeding from the top
		FALSE Material feeding from the bottom
xWindingDirection	BOOL	Function of the winder for positive line speed (<i>IrSetLineVel</i> > 0) - Initial value: FALSE ⚠ WARNING! Injury to persons Risk of injury is caused by changing the setting at <i>xWindingDirection</i> input during a movement. This can lead to a speed jump of the winder axis. Possible consequences Death or severe injuries Protective measures Only change the setting of input <i>xWindingDirection</i> if the technology module is not running and the inverter is disabled: <i>xEnable</i> input = FALSE <i>xEnableOperation</i> input = FALSE
		TRUE Unwinder
		FALSE Rewinder
xJogLinePos	BOOL	TRUE Drive axis in positive material flow direction (manual jog). The path parameters (speed, acceleration and deceleration) refer to the circumferential speed of the reel.
xJogLineNeg	BOOL	TRUE Drive axis in negative material flow direction (manual jog). The path parameters (speed, acceleration and deceleration) refer to the circumferential speed of the reel.
IrSetLineVel	LREAL	Current line velocity Unit: mm/s
xLoadDiam	BOOL	TRUE Load the (start) diameter from the <i>IrSetDiam</i> input. Initial value: FALSE
IrLoadDiam	LREAL	Defining a (start) diameter The diameter is loaded cyclically when the <i>xLoadDiam</i> is set to TRUE. - Unit: mm - Initial value: 0
xHoldDiam	BOOL	Hold the current diameter Initial value: FALSE
		TRUE Current diameter is held.
		FALSE The current diameter is not held.
xDiamCalcReduced	BOOL	Mode change-over of the diameter calculation between a long/short distance Initial value: FALSE
		TRUE Diameter is updated after the short distance.
		FALSE Diameter is updated after the long distance.

Identifier	Data type	Description
xWebBreakMonit	BOOL	Activate/Deactivate web break monitoring • Initial value: FALSE
		TRUE Activate web break monitoring.
		FALSE Deactivate web break monitoring.
xSyncln	BOOL	Enable of the axis via <i>xEnableOperation</i> required.
		TRUE Synchronise winder axis with the line
		FALSE Synchronise winder axis with the line
xTeachUpperPos	BOOL	TRUE The current dancer position is stored as the upper limit value.
xTeachLowerPos	BOOL	TRUE The current dancer position is stored as the lower limit value.
lrActDancerPosIn	LREAL	Current dancer position The actual position of the dancer is led back to the Controller in the form of an analog signal (0 ... 10V).
xDancerCtrl	BOOL	TRUE Activate dancer position control.
		TRUE↘ Dancer position control is deactivated.
		FALSE With the <i>scPar.eDancerCtrlStopMode</i> defined, the winding process is stopped.
lrSetDancerPosScaled	LREAL	Scaled setpoint for the dancer position • Unit: x 100 % • Initial value: 0 • Value range: -1 ... 1 (-100 ... 100 %)
lrDancerPosInfluenceScaled	LREAL	Influence range of the dancer position controller • Unit: x 100 % • Initial value: 0 • Value range: 0 ... 1 (0 ... 100 %)
xResetICtrl	BOOL	TRUE The I component of the PI controller can be switched off and the correcting variable (output of the controller) from the I component can be led to the value 0 via the ramp function. The correcting variable from the P component is not affected.
xAccCmps	BOOL	Activate/Deactivate acceleration compensation during the dancer position control • Initial value: FALSE
lrMinertiaAdapt	LREAL	Multiplier for the current moment of inertia • Initial value: 0
xExecutIdentMinertia	BOOL	The input is edge-controlled and evaluates the rising edge.
		FALSE↗ The moment of inertia at the winder shaft is detected.
		TRUE The lrIdentMinertia output displays the detected moment of inertia in kgcm ² .
xExecutIdentDiam	BOOL	The input is edge-controlled and evaluates the rising edge.
		FALSE↗ Identification of the diameter is started.
		TRUE The dancer is lifted while the line speed is zero. By means of the dancer lift, the reel diameter is calculated from the data determined, which refer to the material length covered and the angle of rotation.
xAdaptSpdCtrlGain	BOOL	Switch adaptation of the speed controller gain on/off. • Initial value: FALSE
		TRUE Enable adaptation of the speed controller gain.
		FALSE Adaptation of the speed controller gain inactive.
lrAdaptSpdCtrlGainFactor	LREAL	The resulting value of the adaptation of the speed controller gain can be influenced multiplicatively via this input. • Value range: 0 ... 1 • Initial value: 1

Identifier	Data type	Description
xTensCurve	BOOL	Enable tensile force characteristic Initial value: FALSE
		TRUE Enable tensile force characteristic.
		FALSE Lock tensile force characteristic.
IrSetTens	LREAL	Tensile force setpoint • Unit: N • Initial value: 0
xSetMaterialCounter	BOOL	The input is edge-controlled and evaluates the FALSE→TRUE edge.
		TRUE Sets the material length counter (<i>IrMaterialCounter</i> output) to the value set under the <i>IrSetMaterialPos</i> parameter.
MaterialCounterValues L_MC1P_IAXIS_REF_MASTERVALUES		Here a modulo axis of a measuring wheel can be connected to the material. If an axis is connected to the input, the material length is determined on the basis of the data from the reference axis. This procedure is also suitable for noisy signals. If no axis is connected here, the material length is determined from the integration of the material speed (<i>IrSetLineVel</i> or <i>IrSetLineVelDiamCalc</i> input). ► Material length counter (📖 73) ► Sources for the material length counting (📖 73) Supported axes: • Encoder • 9400 HighLine servo inverter • i700 servo inverter • i950 servo inverter

5.3

Outputs

Identifier	Data type	Description
xReadyForOperation	BOOL	The axis is faultless. The DC bus voltage is available. STO is not active.
xOperationEnabled	BOOL	The axis is enabled. The brake is released.
xReadyForMotion	BOOL	The axis is enabled and faultless. The brake is released.
xError	BOOL	TRUE There is an error in the technology module.
xWarning	BOOL	TRUE There is a warning in the technology module.
eErrorID	L_IE1P_Error	ID of the error or warning message if <i>xError</i> = TRUE or <i>xWarning</i> = TRUE. "FAST technology modules" reference manual: Here you can find information on error or warning messages.
xSTOActive	BOOL	TRUE A Safe Torque Off (STO) has been activated and is active. The axis is torqueless, the controller is inhibited.
xQSPActive	BOOL	TRUE The <i>xQSPApplication</i> input has been set. The technology module is in the "STOP" state, the axis is brought to a standstill. The output is reset as soon as the axis is at a standstill.
xAbortBusy	BOOL	TRUE The function activated is aborted by the <i>xAbsort</i> input.
xAbortDone	BOOL	TRUE The function activated has been aborted by the <i>xAbsort</i> input.
xJoggingBusy	BOOL	TRUE The manual jog function is active. The technology module is in the "JOGGING" state.
xHomingBusy	BOOL	TRUE Homing is carried out. The technology module is in the "HOMING" state.
xHomingDone	BOOL	TRUE Homing has been completed successfully.
xIsHomed	BOOL	TRUE The axis has been referenced ("Reference known").
eTMState		Current state of the technology module ► State machine (□ 32)
scStatusBasicMotion	L_TB2P_scStatusBasicMotion	Structure of the status data of the function block • See description for the L_TB2P_BasicMotionBase technology module
IrActVel	LREAL	Current velocity • Unit: unit / s
IrSetVelOut	LREAL	Setpoint speed • Unit: unit / s
IrActPos	LREAL	Actual position • Unit: unit
IrSetPosOut	LREAL	Setpoint position • Unit: unit
IrActFollowingError	LREAL	Current following error • Unit: unit

Identifier	Data type	Description
xHoldDiamActive	BOOL	Status of the diameter calculation. Note: When this output is toggled, we recommend that the hysteresis function is used (<i>scPar.IrHystlineStandstill</i> parameter). ► Signal flow diagram (📖 33)
		TRUE Current diameter is held.
		FALSE The current diameter is updated.
xDiamMax	BOOL	TRUE The maximum diameter has been reached.
xDiamMin	BOOL	TRUE The minimum diameter has been reached.
IrSetDiamOut	LREAL	Current diameter calculated • Unit: mm
xWebBreak	BOOL	TRUE A web break has occurred.
xSynchronised	BOOL	TRUE The winder is synchronised with the line speed.
xAccDecSync	BOOL	TRUE The synchronisation function is active. Synchronisation of the winder is carried out or cancelled.
xWindingBusy	BOOL	TRUE Winding operation has been activated.
xUnwind	BOOL	Status bit for unwinder and rewinder Note: When this output is toggled, we recommend that the hysteresis function is used (<i>scPar.IrHystlineStandstill</i> parameter). ► Signal flow diagram (📖 33)
		TRUE Unwinder
		FALSE Rewinder
IrActDancerPosScaled	LREAL	Current scaled dancer position • Unit: x 100 % • Value range: -1 ... 1 (-100 ... 100 %)
xDancerReachedSetPos	BOOL	TRUE The dancer has reached the setpoint position and is within the <i>scPar.IrDancerInPosWindowScaled</i> tolerance window.
xDancerMaxPos	BOOL	TRUE The dancer has reached the upper limit position.
xDancerMinPos	BOOL	TRUE The dancer has reached the lower limit position.
IrSetMIInertiaOut	LREAL	Current moment of inertia at the winder shaft • Unit: kgcm ²
xDentBusy	BOOL	TRUE The identification of the mass inertia or the friction is determined.
xDentDone	BOOL	TRUE The identification of the mass inertia or the friction has been completed.
IrIdentMIInertia	LREAL	Identified moment of inertia at the winder shaft • Unit: kgcm ²
IrSetSpdCtrlGainAdaptOut	LREAL	Adaptation of the speed controller gain • Unit: x 100 % (1 = 100 %)
IrSetTensCurveOut	LREAL	Current tensile force setpoint from the curve function

Identifier	Data type	Description
IrMaterialCounter	LREAL	Display of the material length counter content on the winder Depending on the definition of the winding direction, the material length counter is incremented or decremented • Unit: mm ► Defining the winding direction (winding/unwinding) (📖 57).
IrLimitLineVel	LREAL	The maximum permitted line speed is determined from the calculated diameter and the <code>scPar.IrMaxWndSpd</code> parameter. • Unit [mm/s]

5.4

Parameter

L_TW2P_scPar_WinderDancerCtrlState

Identifier	Data type	Description
IrVel	LREAL	Speed for movements that are commissioned via the scCtrlBasicMotion structure. • Unit: unit/s • Initial value: 3600
IrAcc	LREAL	Acceleration for movements that are commissioned via the scCtrlBasicMotion structure. Selection of the value for the maximum acceleration speed variation. • Unit: unit/s ² • Initial value: 36000
IrDec	LREAL	Deceleration for movements that are commissioned via the scCtrlBasicMotion structure. Selection of the value for the maximum deceleration speed variation until standstill. • Unit: unit/s ² • Initial value: 100
IrJerk	LREAL	Jerk for movements that are commissioned via the scCtrlBasicMotion structure. • Einheit: Unit/s ³ • Initial value: 0
eDirection	MC_DIRECTION	Traversing direction: • <i>mcCurrentDirection</i> : Retain current direction (only in case of modulo axis) • <i>mcPositiveDirection</i> : Positive direction • <i>mcNegativeDirection</i> : Negative direction • <i>mcShortestWay</i> : Shortest distance (only in case of modulo axis) • Initial value: <i>mcPositiveDirection</i>
scHomeExtTP	MC_TRIGGER_REF	Transfer of an external touch probe event: • Only relevant for "external encoder" touch probe configuration. • For describing the MC_TRIGGER_REF structure, see the MC_TouchProbe function block.
IrLineJerk	LREAL	Jerk for inching mode with the 1 / d evaluation and for the purpose of compensation with regard to the clutch function • Unit: rev / s ³ • Initial value: 100000
IrJogLineAcc	LREAL	Acceleration for the inching mode with the 1 / d evaluation. Selection of the value for the maximum acceleration speed variation. • Unit: mm/s ² • Initial value: 100
IrJogLineDec	LREAL	Deceleration for the inching mode with the 1 / d evaluation. Selection of the value for the maximum deceleration speed variation. • Unit: mm/s ² • Initial value: 10
IrJogLineVel	LREAL	Speed for the inching mode with the 1 / d evaluation. Selection of the maximum speed used for traversing. • Unit: mm/s • Initial value: 10
IrSyncLineAcc	LREAL	Acceleration for synchronising to line velocity • Unit: mm/s ² • Initial value: 100
IrSyncLineDec	LREAL	Deceleration for synchronising to line velocity • Unit: mm/s ² • Initial value: 100

Identifier	Data type	Description
IrWebBreakWindow	LREAL	Web break window The current diameter is compared to the previous diameter across the web break window from the last calculation cycle. • Unit: x 100 % (1.0 = 100 %) • Initial value: 0.1 (10 %)
IrMaxDiam	LREAL	Maximum diameter • Unit: mm • Initial value: 180
IrMinDiam	LREAL	Minimum diameter • Unit: mm • Initial value: 50
rFiltTimeDiam	REAL	PT1 filter time for the current diameter (<i>IrSetDiamOut</i>) • Unit: s • Initial value: 0.05
IrDiamCalcRegularDist	LREAL	Regular calculation distance for the diameter • Unit: rev • Initial value: 1
IrDiamCalcReducedDist	LREAL	Reduced calculation distance for the diameter • Unit: rev • Initial value: 0.1
alrAdaptDiamX ARRAY [1...9] OF LREAL		Grid points of the curve function for diameter loading • Values that may be applied to the analog input <i>IrSetDiam</i> . • Unit: mm • Initial values: 0, 100, 200, 300, 400, 500, 600, 700, 800
alrAdaptDiamY ARRAY [1...9] OF LREAL		Grid points of the curve function for diameter loading • Function values for the diameter • Unit: mm • Initial values: 0, 100, 200, 300, 400, 500, 600, 700, 800
IrTensCurveCtrlScaled	LREAL	Gradient of the characteristic for the tension control open loop • Unit: x 100 % (1 = 100 %) • Initial value: 0 • The value '1' causes a constant tensile force profile and thus a proportionally increasing setpoint for the diameter.
IrTensCurveStartDiamScaled	LREAL	Initial point of the characteristic for the tension control open loop • Unit: x 100 % • 1 = 100 % = <i>IrMaxDiam</i> parameter • Initial value: 0 (0 %)
IrLineVelRef	LREAL	Maximum line velocity • Unit: mm/s • Initial value: 1000
IrMinLineVel	LREAL	Minimum line velocity Up to this velocity, the diameter is held. • Unit: mm/s • Initial value: 1
rFiltTimeMaterialCounter	REAL	Filter time constant for the material length counter (<i>IrMaterialCounter</i> output) • Initial value: 0 (filter is deactivated.)
IrSetMaterialPos	LREAL	Position of the material length counter With a FALSE→TRUE edge at the <i>xSetMaterialCounter</i> input, the material length counter (<i>IrMaterialCounter</i> output) is set to the value in <i>IrSetMaterialPos</i> . • Unit: mm • Initial value: 0
IrMaxWndSpd	LREAL	Definition of the maximum winder shaft speed (at the output end). Unit: rev / s Initial value: 100

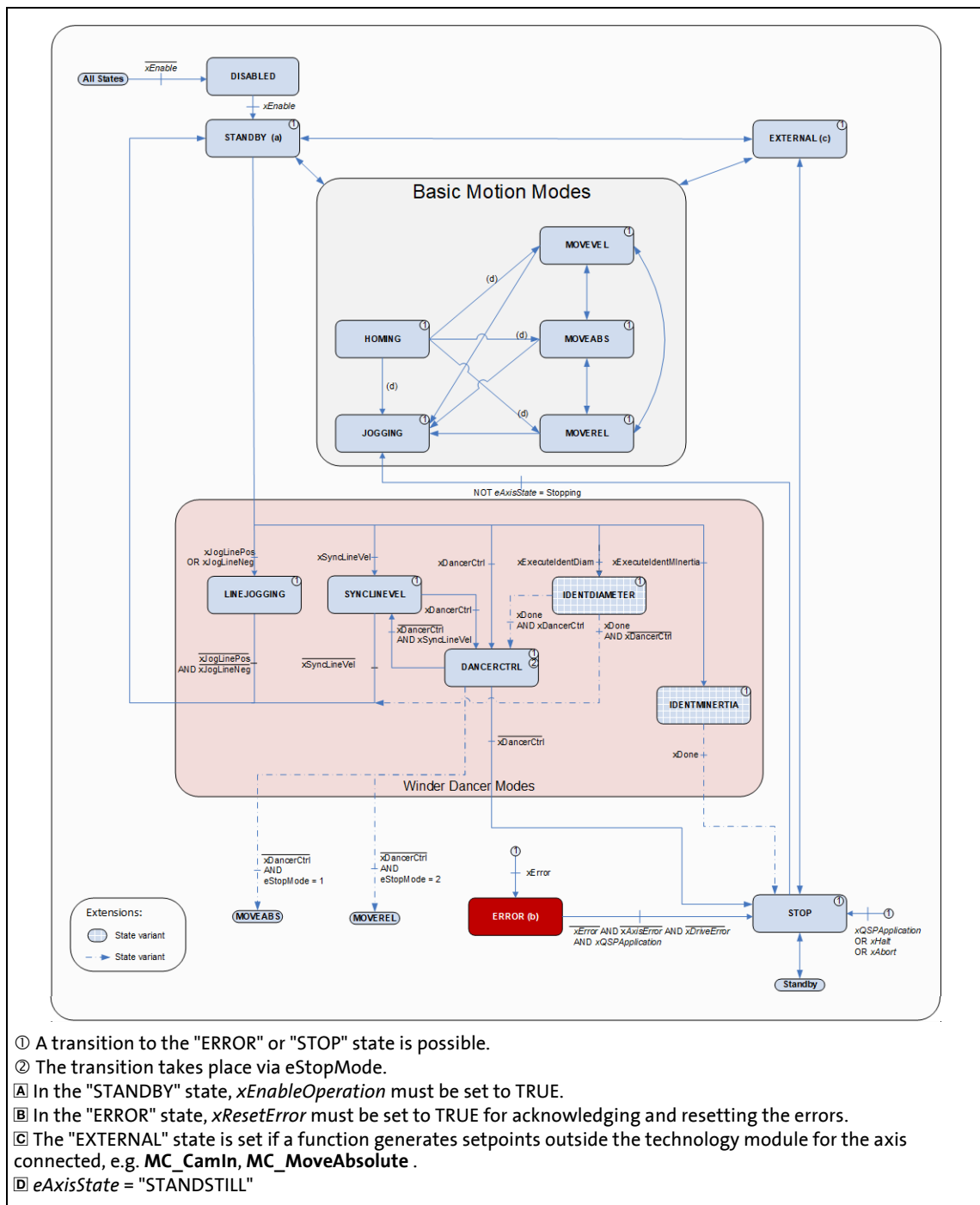
Identifier	Data type	Description
xLineVelDiamCalc	BOOL	Diameter calculation Initial value: FALSE
		TRUE For calculating the diameter, the velocity from the <i>IrSetLineVelDiamCalc</i> input is used.
		FALSE For calculating the diameter, the velocity from the <i>IrSetLineVel</i> input is used.
IrDancerPosRamp	LREAL	Acceleration ramp for dancer position setpoints - Unit: 1 / s - Initial value: 1
IrDancerPosCtrlGain	LREAL	Controller gain Initial value: 1
IrDancerPosCtrlResetTime	LREAL	Controller reset time - Unit: s - Initial value: 0 controller reset time is deactivated.)
eResetCtrlMode L_TW2P_ResetCtrlMode		Mode for setting the function for resetting the correcting variable from the I component via the <i>xResetCtrl</i> input = TRUE Initial value: 1
	0	Instant With a rising edge <i>xResetCtrl</i> = TRUE, the correcting variable from the I component is set to 0 immediately (abruptly).
	1	Ramp With a rising edge <i>xResetCtrl</i> = TRUE, the correcting variable from the I component is ramped to zero via the <i>scPar.IrDancerPosCtrlResetTime</i> parameter.
IrDancerPosCtrlLimPos	LREAL	Limitation of the dancer position controller correcting variable (output of the controller) in positive direction Initial value: 1
IrDancerPosCtrlLimNeg	LREAL	Limitation of the dancer position controller correcting variable (output of the controller) in negative direction Initial value: -1
IrDancerMaxPosScaled	LREAL	Maximum dancer position for the <i>xDancerMaxPos</i> status bit - Unit: x 100 % (1 = 100 %) - Initial value: 0.95 (95 %)
IrDancerMinPosScaled	LREAL	Minimum dancer position for the <i>xDancerMaxPos</i> status bit - Unit: x 100 % (1 = 100 %) - Initial value: -0.95 (-95 %)
xTeachDancerLimits	BOOL	Source of the dancer position limitations Initial value: FALSE
		TRUE The limit positions <i>IrDancerLowerLimit</i> and <i>IrDancerUpperLimit</i> are used until the teaching function has been executed. After the teaching function has been executed manually, the saved limit positions from the teaching function are always used.
		FALSE The limit positions <i>IrDancerLowerLimit</i> and <i>IrDancerUpperLimit</i> are used.
xUpperPosReached	BOOL	Referenced upper dancer position
		TRUE The upper dancer limit position already referenced
xLowerPosReached	BOOL	Referenced lower dancer position
		TRUE The lower dancer limit position already referenced
IrDancerUpperLimit	LREAL	Analog value for the upper dancer position limit Initial value: 10 000 000
IrDancerLowerLimit	LREAL	Analog value for the lower dancer position limit Initial value: 0

Identifier	Data type	Description						
IrDancerMaterialLength	LREAL	Length of the material in the dancer - Unit: mm - Initial value: 0 - The value 0 is used to deactivate examination of the dancer movement.						
IrDancerInPosWindowScaled	LREAL	Window for the set position of the dancer to control the <i>xDancerReachedSetPos</i> status bit. - Unit: x 100 % (1.0 = 100 %) - Initial value: 0.2 (20 %)						
rFiltTimeActDancerPosIn	REAL	PT1 filter time for the <i>IrActDancerPosIn</i> input - Unit: s - Initial value: 0.005						
rFiltTimeActDancerVelComp	REAL	Filter time for the speed compensation - Unit: s - Initial value: 0						
wWebBreakMode	WORD	Mode for web break monitoring Initial value: 1 <table><tr><td>0</td><td>Web break monitoring via diameter calculation and dancer position</td></tr><tr><td>1</td><td>Web break monitoring only via the dancer position</td></tr><tr><td>2</td><td>Web break monitoring only via diameter calculation</td></tr></table>	0	Web break monitoring via diameter calculation and dancer position	1	Web break monitoring only via the dancer position	2	Web break monitoring only via diameter calculation
0	Web break monitoring via diameter calculation and dancer position							
1	Web break monitoring only via the dancer position							
2	Web break monitoring only via diameter calculation							
dwSelectTensCurve	DWORD	Selection of the characteristic for tension control Initial value: 0 <table><tr><td>0</td><td>Linear tensile force profile</td></tr><tr><td>1</td><td>Linear torque profile</td></tr><tr><td>2</td><td>Tensile force profile on the basis of a specified characteristic</td></tr></table>	0	Linear tensile force profile	1	Linear torque profile	2	Tensile force profile on the basis of a specified characteristic
0	Linear tensile force profile							
1	Linear torque profile							
2	Tensile force profile on the basis of a specified characteristic							
alrTensCurve	ARRAY [1...65] OF LREAL	Characteristic for the tension control open loop (consisting of 65 values)						
rFiltTimeAccSpd	REAL	PT1 filter time for the winder shaft speed for acceleration compensation - Unit: s - Initial value: 0.005						
IrAccCmpsDeadBandTrq Scaled	LREAL	Lagging range (Dead band) for the current acceleration torque - Unit: x 100 % (1.00 = 100 %); reference variable: rated motor torque [Nm] - Initial value: 0.1 (10 %)						
IrAccCmpsGainAcc	LREAL	Gain factor for the acceleration torque in positive direction - Unit: x 100 % (1.00 = 100 %) - Value range: 0 ... 2 (0 ... 200%) - Initial value: 1.05 (105 %)						
IrAccCmpsGainDec	LREAL	Gain factor for the acceleration torque in negative direction - Unit: x 100 % (1.00 = 100 %) - Value range: 0 ... 2 (0 ... 200%) - Initial value: 0.95 (95 %)						
IrConstMIInertia	LREAL	Constant moment of inertia at the winder shaft - Unit: kgcm² - Initial value: 1						
IrMaxMIInertia	LREAL	Maximally permissible moment of inertia at the winder shaft - Unit: kgcm² - Initial value: 10						
rFiltTimeIdentMIInertiaSpd	REAL	PT1 filter time for the speed at the winder shaft during the identification of the moment of inertia - Unit: s - Initial value: 0.01						

Identifier	Data type	Description						
rFiltTimeIdentMinertiaTrq	REAL	PT1 filter time for the torque at the winder shaft during the identification of the moment of inertia - Unit: s - Initial value: 0.005						
IrIdentMinertiaMaxSpd Scaled	LREAL	Maximum speed of the winder shaft during the identification of the moment of inertia - Unit: x 100 % (1.0 = 100 % = <i>IrWndSpdRef</i>) - Value range: 0 ... 1 (0 ... 100 %) - Initial value: 0.2 (20 %)						
IrIdentMinertiaMaxTrq Scaled	LREAL	Maximum torque of the winder shaft during the identification of the moment of inertia - Unit: x 100 % (1.0 = 100 %) reference variable: rated motor torque - Value range: 0 ... 1 (0 ... 100 %) - Initial value: 0.2 (20 %)						
eAdaptSpdCtrlGainMode	ENUM	Mode selection for adaptation of the speed controller gain. Initial value: 2 <table><tr><td>0</td><td>DiamToSquare; $VP = f(d^2)$</td></tr><tr><td>1</td><td>Diam; $VP = f(d)$</td></tr><tr><td>2</td><td>Inertia; $VP = f(J)$</td></tr></table>	0	DiamToSquare; $VP = f(d^2)$	1	Diam; $VP = f(d)$	2	Inertia; $VP = f(J)$
0	DiamToSquare; $VP = f(d^2)$							
1	Diam; $VP = f(d)$							
2	Inertia; $VP = f(J)$							
IrAdaptSpdCtrlLowLimit	LREAL	Lower limitation of speed controller gain in the drive. The <i>IrSetSpdCtrlGainAdaptOut</i> adaptation value must not be smaller than the <i>scPar</i> value. <i>IrAdaptSpdCtrlLowLimit</i> . - Value range: 0 to 1 - Initial value: 1.0 (100 %)						
alrSpdCtrlGainAdaptX ARRAY [1...9] OF LREAL		Characteristic function for the speed control gain The X axis corresponds to the adaption factor currently calculated for the adaptation of the speed controller gain. - Unit: x 100 % (1 = 100 % = <i>IrMaxMinertia</i> parameter) - Initial values: [0.0, 0.2, 0.4, 0.6, 0.8, 1.0, 1.2, 1.4, 1.6]						
alrSpdCtrlGainAdaptY ARRAY [1...9] OF LREAL		Characteristic function for the speed control gain The Y axis corresponds to the gain factor of the speed controller. - Unit: x 100 % (1 = 100 %) - Initial values: - Initial values: [0.0, 0.2, 0.4, 0.6, 0.8, 1.0, 1.2, 1.4, 1.6]						
IrReducedGainWindow	LREAL	Area of the system deviation with a reduced gain / sensitivity. With <i>IrReducedGainWindow</i> , the tolerance zone of the system deviation is defined, in which <i>IrReducedGain</i> takes effect. Initial value: 0.0						
IrReducedGain	LREAL	Gain of system deviation within the reduced sensitivity Initial value: 0.0						
eDancerCtrlStopMode L_TW2P_DancerCtrlStopMode		Mode in which the winding process ("DANCERCTRL" state) is stopped. Initial value: 0 <table><tr><td>0</td><td>Holding The axis is stopped via the deceleration (<i>IrHaltDec</i>) and the jerk (<i>IrJerk</i>).</td></tr><tr><td>1</td><td>Move ABS Absolute travel/positioning: The axis travels to the target position (<i>IrPos_Dist</i>) with the velocity (<i>IrVel</i>), acceleration (<i>IrAcc</i>), deceleration (<i>IrDec</i>) and the jerk (<i>IrJerk</i>).</td></tr><tr><td>2</td><td>Move Rel Relative travel/positioning: The axis is brought to a standstill with the velocity (<i>IrVel</i>), acceleration (<i>IrAcc</i>), deceleration (<i>IrDec</i>) and the jerk (<i>IrJerk</i>) after the distance travelled (<i>IrPos_Dist</i>).</td></tr></table>	0	Holding The axis is stopped via the deceleration (<i>IrHaltDec</i>) and the jerk (<i>IrJerk</i>).	1	Move ABS Absolute travel/positioning: The axis travels to the target position (<i>IrPos_Dist</i>) with the velocity (<i>IrVel</i>), acceleration (<i>IrAcc</i>), deceleration (<i>IrDec</i>) and the jerk (<i>IrJerk</i>).	2	Move Rel Relative travel/positioning: The axis is brought to a standstill with the velocity (<i>IrVel</i>), acceleration (<i>IrAcc</i>), deceleration (<i>IrDec</i>) and the jerk (<i>IrJerk</i>) after the distance travelled (<i>IrPos_Dist</i>).
0	Holding The axis is stopped via the deceleration (<i>IrHaltDec</i>) and the jerk (<i>IrJerk</i>).							
1	Move ABS Absolute travel/positioning: The axis travels to the target position (<i>IrPos_Dist</i>) with the velocity (<i>IrVel</i>), acceleration (<i>IrAcc</i>), deceleration (<i>IrDec</i>) and the jerk (<i>IrJerk</i>).							
2	Move Rel Relative travel/positioning: The axis is brought to a standstill with the velocity (<i>IrVel</i>), acceleration (<i>IrAcc</i>), deceleration (<i>IrDec</i>) and the jerk (<i>IrJerk</i>) after the distance travelled (<i>IrPos_Dist</i>).							

Identifier	Data type	Description
IrIdentDiamVel	LREAL	Circumferential speed of the winding shaft for identification of the diameter. • Unit: mm/s • Initial value: 10
IrIdentDiamAcc	LREAL	Circumferential acceleration of the winding shaft for identification of the diameter. • Unit: mm/s ² • Initial value: 100
IrIdentDiamDec	LREAL	Circumferential deceleration of the winding shaft for identification of the diameter. • Unit: mm/s ² • Initial value: 100
IrIdentDiamJerk	LREAL	Circumferential jerk of the winding shaft for identification of the diameter. • Unit: mm/s ³ • Initial value: 0
wIdentDiamCalcCycles	WORD	Number of calculation cycles for successful identification • Initial value: 2
IrIdentDiamMaxDancerPos	LREAL	Scaled end position of the dancer during identification. The identification run is aborted if the end position is exceeded. Setting range: -1 ... 1 Initial value: 0.5
IrIdentDiamSpdCtrlGain	LREAL	The speed controller gain is set when the speed controller gain <i>xAdaptSpdCtrlGain</i> = TRUE during diameter identification. Setting range: 0 ... 1 Initial value: 0.5
IrHystlineStandstill	LREAL	Hysteresis width for the line speed This stabilises the noisy line speed and the direction recognition. After detecting the maximum noise amplitude of the input signal, set this value. Unit: mm/s Initial value: 0

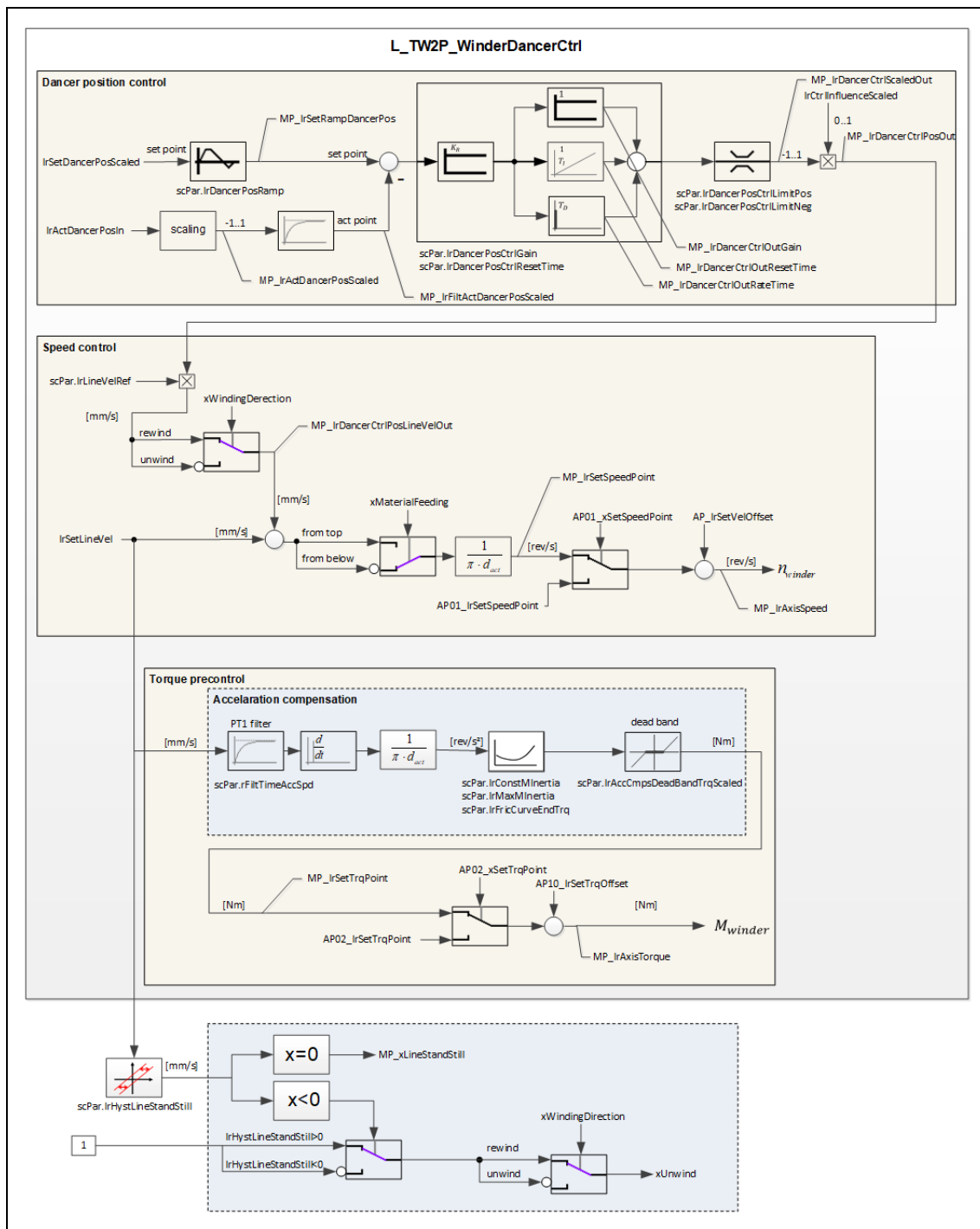
6



[6-1] State machine of the technology module

7 Signal flow diagram

In the following illustration, the main signal flow of the function is represented:



[7-1] Signal flow of the technology module

7.1 Structure of the signal flow

L_TW2P_scSF_WinderDancerCtrlState

The content of the structure is read-only and offers a practical diagnostics option within the signal flow.

► [Signal flow diagram](#) (33)

Identifier	Data type	Description
MP_lrWndVelRef	LREAL	Rated speed of the reel (output end) • Unit: units/s
MP_lrWndSpeedRef	LREAL	Rated speed of the reel (output end) • Unit: rev/s
MP_lirRevCounter	LREAL	Counter content of the material length counter (whole numbers of complete reel revolutions)
MP_lirRevCounterResidual	LREAL	Counter content of the material length counter (residual value after one complete reel revolution) • Unit: 1 x 100 % • 100 % = One reel revolution
MP_lrAxisSpeed	LREAL	Current speed of the reel (output end) • Unit: rev/s
MP_xDiamCalculated	LREAL	FALSE→TRUE edge: The diameter has been calculated. The measuring point is only set to TRUE for one PLC cycle.
MP_lrSetDiamUnfiltered	LREAL	Current diameter calculated • Unit: mm
MP_lrSetDiamAdapted	LREAL	Diameter resulting from the <i>lrSetDiam</i> input. • Unit: mm
MP_lrSetDiamScaledOut	LREAL	Scaled diameter resulting from the <i>lrSetDiam</i> input. • Unit: 1x100% • 100% <i>lrMaxDiam</i>
MP_lrSetLineVelScaledOut	LREAL	Scaled line speed • Unit 1 x 100% • 100% <i>lrLineVelRef</i>
MP_lrActTotalTrqScaled	LREAL	Torque calculated from the measured current (scaled) • Unit x100% • 100% nominal torque
MP_lrSetSpeedPoint	LREAL	Resulting speed setpoint • Unit: rev/s
MP_lrDancerPosScaled0To1	LREAL	Scaled dancer position • Unit: 1 x 100 % • Value range: -1 ... 1 (-100 ... 100 %) • -100 % = lower dancer end position • 100 % = upper dancer end position
MP_lrDancerCompVelOut	LREAL	Resulting speed of the dancer • Unit: 1 x 100 %/s
MP_lrFiltDancerCompVelOut	LREAL	Filtered resulting speed of the dancer • Unit: 1 x 100 %/s
MP_lrFiltActDancerPos Scaled	LREAL	Actual scaled filtered dancer position • Unit: 1 x 100 % • Value range: -1 ... 1 (-100 ... 100 %)
MP_lrSetRampDancerPos	LREAL	Target position of the dancer after the ramp generator This position is the reference variable of the winding drive. • Unit: 1 x 100 % • Value range: -1 ... 1 (-100 ... 100 %)

Identifier	Data type	Description
MP_lrdancerCtrlPosOut	LREAL	Correcting variable of the dancer position controller - Unit: 1 x 100 % - Value range: -1 ... 1 (-100 ... 100 %)
MP_lrdancerCtrlLineVelOut	LREAL	Correcting variable of the dancer position controller converted to the line speed Unit: mm/s
MP_lrdancerCtrlOutGain	LREAL	Scaled correcting variable from the proportional component (P-component) of the dancer position controller
MP_lrdancerCtrlOutResetTime	LREAL	Scaled correcting variable from the integrating component (I-component) of the dancer position controller
MP_lrdancerCtrlOutRateTime	LREAL	Scaled correcting variable from the differentiating portion (D portion) of the dancer position controller
MP_lrdancerCtrlScaledOut	LREAL	Scaled correcting variable of the dancer position controller - Unit: 1 x 100 % - Value range: -1 ... 1 (-100 ... 100 %)
MP_lrdancerTorque	LREAL	Torque of the reel (output end) Unit: Nm
MP_lrdancerSetTrqPoint	LREAL	Torque pilot control value of the reel (output end) Unit: Nm
MP_lrdancerAccCompTrq	LREAL	Torque resulting from acceleration compensation Unit: Nm
MP_lrdancerAccTrqScaledOut	LREAL	Resulting torque from the friction compensation Unit: Nm
MP_xLineStandStill	BOOL	TRUE Standstill of the line has been detected.

7.2

Structure of the access points

L_TW2P_scAP_WinderDancerCtrlState

The access points (AP) can be used to influence signals. In the initial state, the access points do not have any effect.

Each access point acts as an alternative branch and is activated via an OR operation or a switch.

Identifier	Data type	Description
AP_xSetVelOffset	BOOL	Release of the access point <i>AP_IrVelOffset</i>
	TRUE	The access point overwrites the values at the access point in the signal flow.
AP_IrVelOffset	LREAL	Cyclic specification of the offset for the velocity of the winder axis with regard to the winding shaft (gearbox output side) • Unit: units/s The offset value is set immediately and abruptly without ramp generator!
AP_xSetTrqOffset	BOOL	Enable of the AP10_IrSetTrqOffset access point
	TRUE	The access point overwrites the values at the access point in the signal flow.
AP_IrSetTrqOffset	LREAL	Cyclic specification of the offset for the torque of the winder axis with regard to the winding shaft (gearbox output side) • Unit: Nm The offset value is set immediately and abruptly without ramp generator!
AP_xSetVelOffsetDiamCalc	BOOL	Enable of the access point AP11_IrSetVelOffsetDiamCalc:
	TRUE	The access point overwrites the values at the access point in the signal flow
AP_IrVelOffsetDiamCalc	LREAL	Cyclic presetting of the offset for the speed of the winder axis in relation to the winding shaft (gear output side) for the calculation of the diameter. • Unit: units/s The offset value is set immediately and abruptly without a ramp generator!
AP_xSetSpeedPointDiamCalc	BOOL	Release of the access point <i>AP_IrSetSpeedPointDiamCalc</i>
	TRUE	The access point overwrites the values at the access point in the signal flow.
AP_IrSetSpeedPointDiamCalc	LREAL	Cyclic selection of the speed of the winder axis used for calculating the diameter. This access point has no effect on the setpoint of the winder axis but is included in the calculation of the diameter.
AP_xSetSpeedPoint	BOOL	Enable of the AP01_IrSetSpeedPoint access point
	TRUE	The access point overwrites the values at the access point in the signal flow.
AP_IrSetSpeedPoint	LREAL	Speed setpoint for the winding drive • Unit: revs/s
AP_xSetTrqPoint	BOOL	Enable of the AP02_IrSetTrqPoint access point
	TRUE	The access point overwrites the values at the access point in the signal flow.
AP_IrSetTrqPoint	LREAL	Torque setpoint for the feedforward control • Unit: Nm

8 Description of the basic functions

Overview of the basic functions

- ▶ [Manual jog \(jogging\)](#) (📖 38)
- ▶ [Homing](#) (📖 40)
- ▶ [QSP application](#) (📖 43)
- ▶ [Halt](#) (📖 44)
- ▶ [Absolute positioning](#) (📖 45)
- ▶ [Relative positioning](#) (📖 47)
- ▶ [Continuous travel with a defined velocity](#) (📖 50)
- ▶ [Aborting and completing a positioning process](#) (📖 53)
- ▶ [Override for speed](#) (📖 55)

8 Description of the basic functions

8.1 Manual jog (jogging)

8.1 Manual jog (jogging)

Prerequisite

- The axis is enabled (*xReadyForMotion* = TRUE).

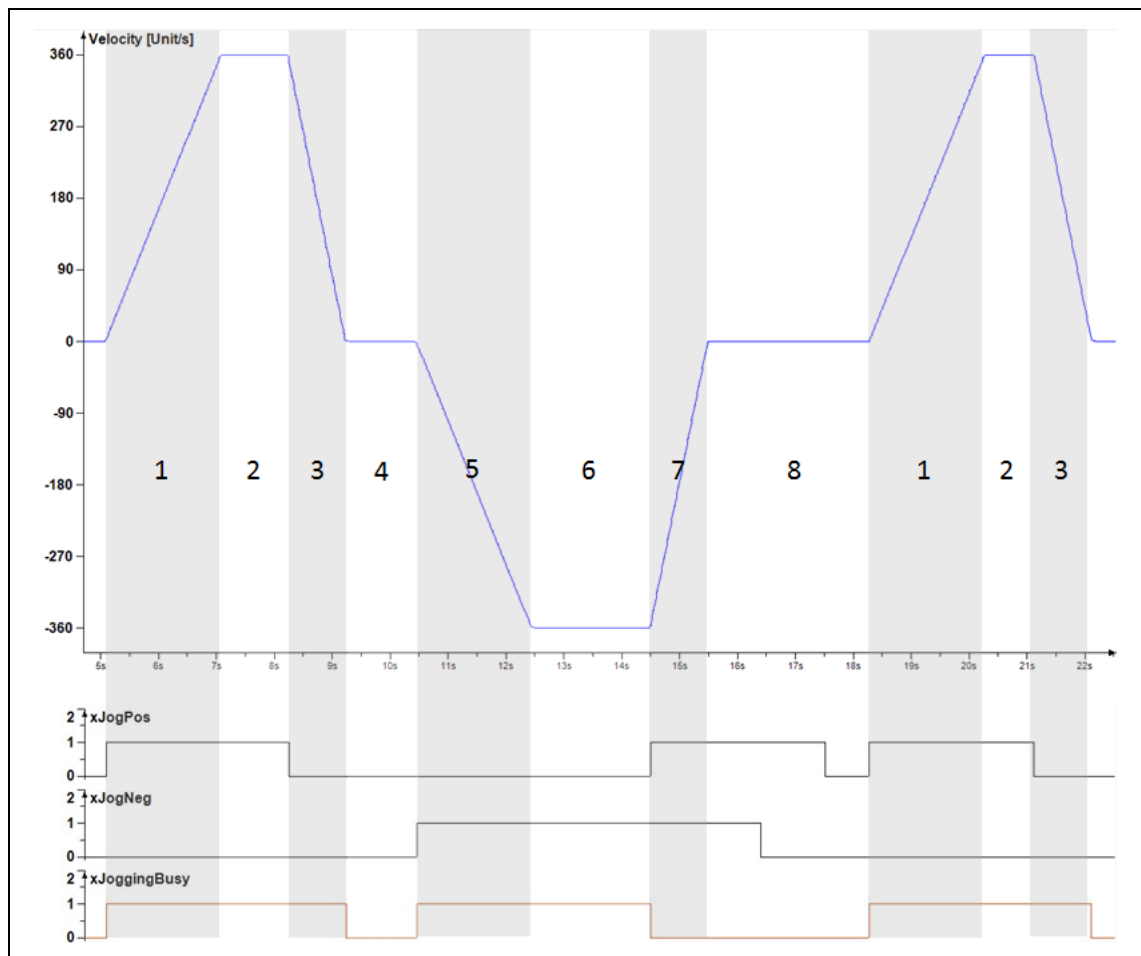
Execution

With the *xJogPos* input = TRUE, the axis is traversed in the positive direction and with the *xJogNeg* input = TRUE, it is traversed in the negative direction. The axis is traversed as long as *xJogPos* or *xJogNeg* remain set to TRUE.

The current travel command cannot be replaced by the other jog command. If *xJogPos* = TRUE and *xJogNegPos* = TRUE, the axis brakes into standstill and the *eTMState* status output changes to the "ERROR" state.

Only if both inputs have been reset, the technology module changes back to the "STANDBY" state.

Travel diagram



No	xJogPos	xJogNeg	xJogging Busy	eTMState	Info
1	FALSE → TRUE	FALSE	FALSE → TRUE	STANDBY → JOGGING	The axis accelerates to the positive manual speed 360 units/s with 180 unit/s².
2	TRUE	FALSE	TRUE	JOGGING	The axis travels constantly with 360 units/s.
3	FALSE → TRUE	FALSE	TRUE	JOGGING	The axis decelerates to a standstill with 360 unit/s².
4	FALSE	FALSE	TRUE → FALSE	JOGGING → STANDBY	The axis has reached a standstill.
5	FALSE	FALSE → TRUE	FALSE → TRUE	STANDBY → JOGGING	The axis accelerates to the negative manual speed -360 units/s with 180 unit/s².
6	FALSE	TRUE	TRUE	JOGGING	The axis travels constantly with -360 units/s.
7	TRUE	TRUE	TRUE → FALSE	ERROR	The axis decelerates to a standstill with 360 unit/s².
8	FALSE	FALSE	FALSE	ERROR → STANDBY	The axis is at a standstill and changes to "STANDBY" if both inputs xJogPos and xJogNeg have the value FALSE.

Parameters to be set

The parameters for manual jog can be found in the AXIS_REF in subindex 181 ... 184.

The parameter values can be changed during operation. They are accepted and become effective immediately when setting the inputs *xJogPos* = TRUE or *xJogNeg* = TRUE.

The manual jog parameters from the PLC program can also be written via the **L_MC1P_SetBasicParameter** block.

8.2**Homing****Prerequisites**

- The technology module is in "STANDBY" mode.
- The axis is enabled (*xReadyForMotion* = TRUE).

Execution

With a rising edge (FALSE→TRUE) at the *xHomeExecute* input, homing is started (output *xHomingBusy* = TRUE). The axis travels until the home position is reached.

When the reference switch has been reached, *xisHomed* switches to TRUE.

When the homing process has been completed successfully, *xHomingDone* changes to TRUE.

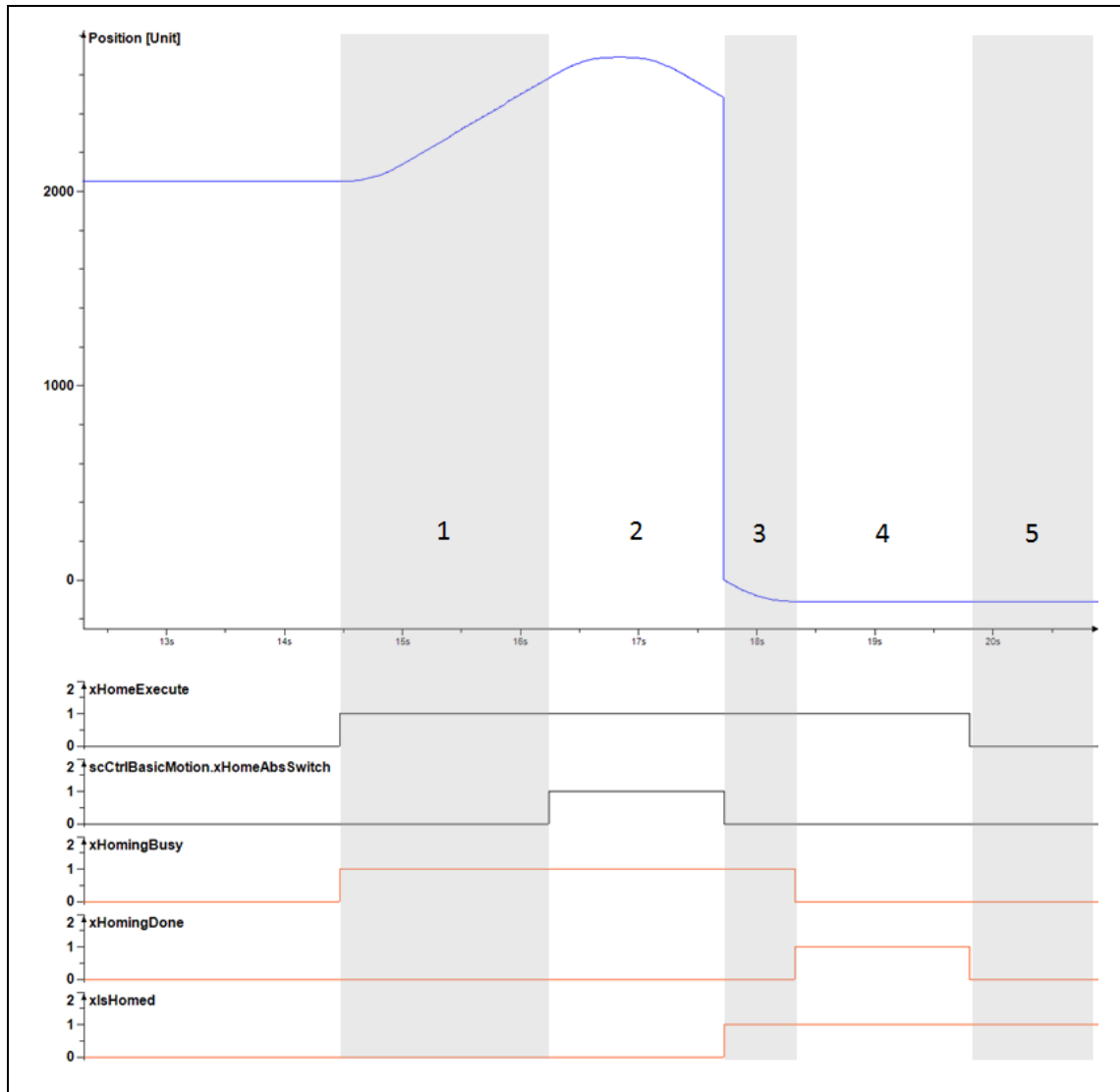
If *xHomeExecute* = FALSE, the state is changed to "STANDBY" and *xHomingDone* is set = FALSE.

Abort of the homing process

The function is aborted via the *xHalt* input.

Homing is not interrupted if *xHomeExecute* is set to = FALSE early.

Travel diagram



[8-1] Homing in the CwRpCwRn module with the home position "0"

No	xHome Execute	xHome AbsSwitch	xHoming Busy	xHoming Done	xIsHomed	eTMState	Info
1	FALSE → TRUE	FALSE	FALSE → TRUE	FALSE	FALSE	STANDBY → HOMING	The axis starts homing in positive direction.
2	TRUE	TRUE → FALSE	TRUE	FALSE	FALSE	HOMING	The axis changes the direction of rotation due to the positive edge of xHomeAbsSwitch.
3	TRUE	FALSE → TRUE	TRUE	FALSE	FALSE → TRUE	HOMING	The negative edge of xHomeAbsSwitch sets the position of the axis to the home position 0 units and the axis is brought to a standstill.
4	TRUE	FALSE	TRUE → FALSE	FALSE → TRUE	TRUE	HOMING	As soon as the axis has reached the standstill, xHomingBusy is set = FALSE and xHomingDone is set = TRUE. xHomingDone remains TRUE for as long as xHomeExecute is set = TRUE, but always at least for one clock cycle.
5	TRUE → FALSE	FALSE	FALSE	TRUE → FALSE	TRUE	HOMING → STANDBY	The axis is ready for operation and can be positioned in an absolute fashion.

Parameters to be set

The parameters for homing can be found in the `AXIS_REF` in subindex 70 ... 83.

The manual jog parameters from the PLC program can also be written via the `L_MC1P_SetBasicParameter` block.

8.3 QSP application

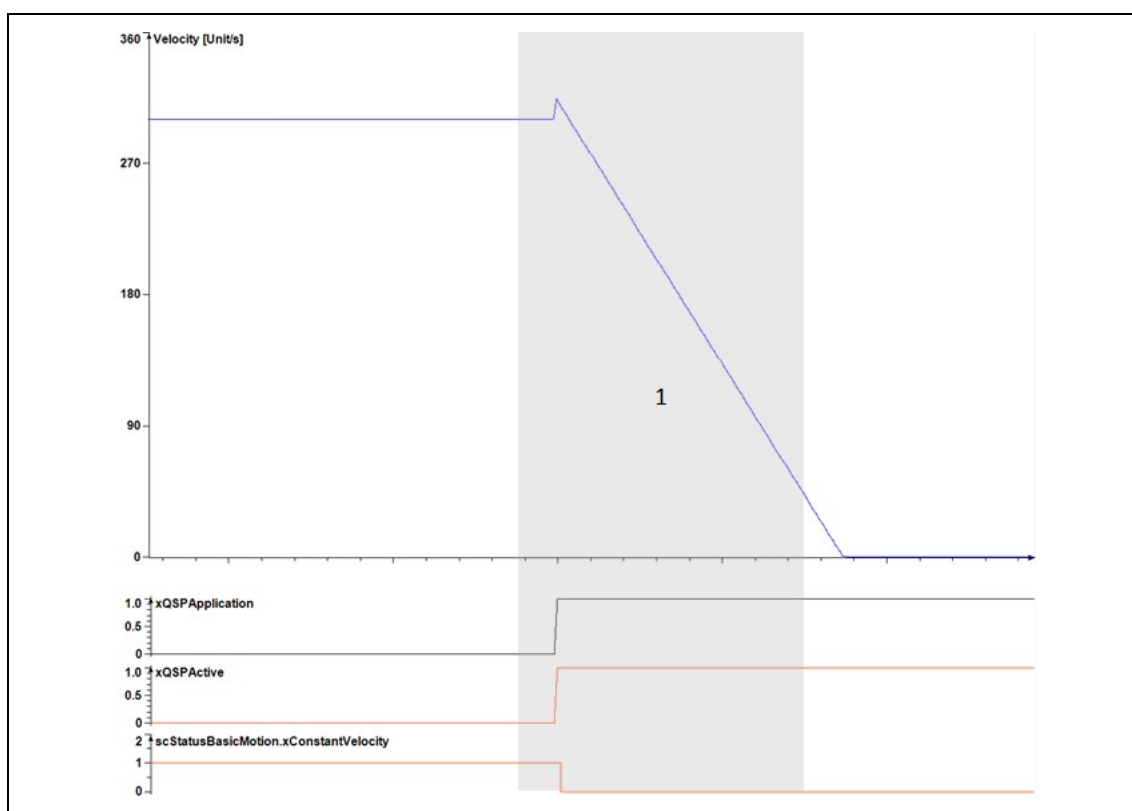
Prerequisite

- The axis is enabled ($xReadyForMotion = TRUE$).

Execution

With a rising edge ($FALSE \rightarrow TRUE$) at the $xQSPApplication$ input, the current movement of the axis is interrupted and the axis is brought to a standstill on the basis of its last current speed value on the QSP application deceleration ramp. The parameter for the QSP application deceleration ramp is set in the AXIS_REF.

Travel diagram



No	xQSP Application	xQSPActive	xConstant Velocity	eTMState	Info
1	$FALSE \rightarrow TRUE$	$FALSE \rightarrow TRUE$	$TRUE \rightarrow FALSE$	MOVEVEL → STOP	The speed setpoint of the axis is set to the last current speed and brought to a standstill via the QSP deceleration ramp.

Parameters to be set

The parameters for "QSP Application" can be found in the AXIS_REF in subindex 48 and 49.

The manual jog parameters from the PLC program can also be written via the **L_MC1P_SetBasicParameter** block.

8.4

Halt

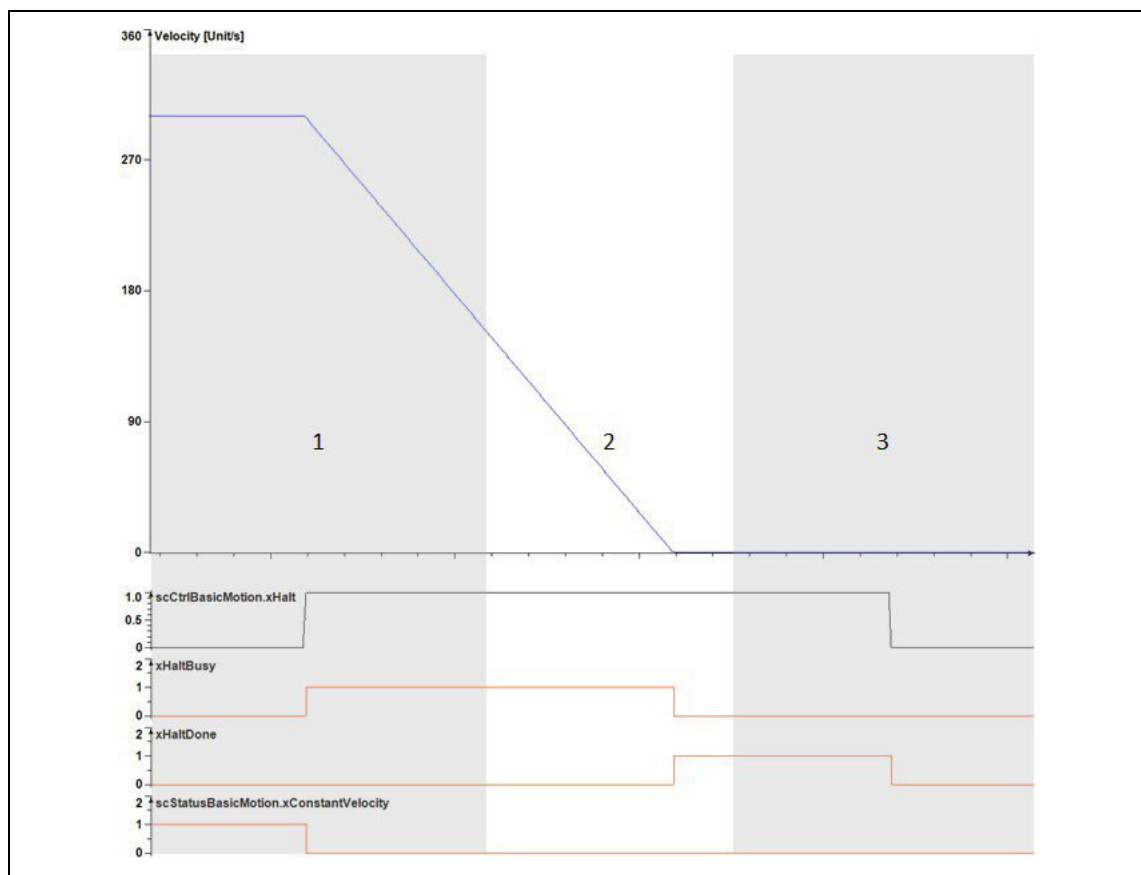
Prerequisite

- The axis is enabled ($xReadyForMotion = TRUE$).

Execution

With a rising edge ($FALSE \rightarrow TRUE$) at the *scCtrlBasicMotion.xHalt* structure input, the current movement of the axis is interrupted and the axis is brought to a standstill with a halt deceleration ramp.

Travel diagram



No	xHalt	xHaltBusy	xHaltDone	xConstant Velocity	eTMState	Info
1	$FALSE \rightarrow TRUE$	$FALSE \rightarrow TRUE$	$FALSE$	$TRUE \rightarrow FALSE$	$MOVEVEL \rightarrow STOP$	The axis decelerates with the halt deceleration ramp: $xHaltBusy = TRUE$ $xConstantVelocity = FALSE$.
2	$TRUE$	$TRUE \rightarrow FALSE$	$FALSE \rightarrow TRUE$	$FALSE$	$STOP$	The axis is brought to a standstill: $xHaltBusy = FALSE$ $xHaltDone = TRUE$
3	$TRUE \rightarrow FALSE$	$FALSE \rightarrow TRUE$	$TRUE \rightarrow FALSE$	$FALSE$	$STOP \rightarrow STANDBY$	$xHalt = FALSE$ $xHaltDone = TRUE \rightarrow FALSE$.

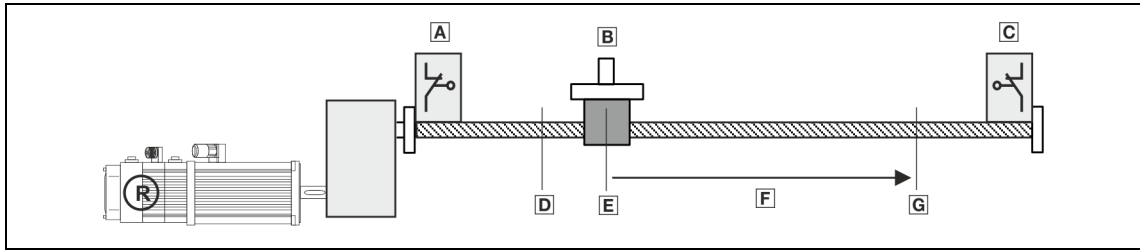
Parameters to be set

The parameter for the halt can be found in the *AXIS_REF* in subindex 186 and 187.

8 Description of the basic functions

8.5 Absolute positioning

8.5 Absolute positioning



[8-2] Example: Absolute positioning of a linear axis

Pos.	Description
A	Negative hardware limit switch
B	Load (e.g. slide)
C	Positive hardware limit switch
D	Zero position of the measuring system
E	Current actual position
F	Distance to be travelled
G	Target position (<i>lrPos_Dist</i> input)

With absolute positioning, the target position in the *lrPos_Dist* input (G) is defined as a unique absolute position, relating to the zero position of the measuring system (D). The distance to be travelled (F) results from the distance between the current actual position (E) and the absolute target position (G).

Prerequisites

- The axis is enabled (*xReadyForMotion* = TRUE).
- The axis is referenced (*xisHomed* = TRUE).

Execution

With a rising edge (FALSE→TRUE) at the *xMoveAbsExecute* input, a travel to the absolute target position in the *lrSetPos_Dist* input is started on the basis of the parameters for the motion profile (see below). The technology module changes to the "MOVEABS" state.

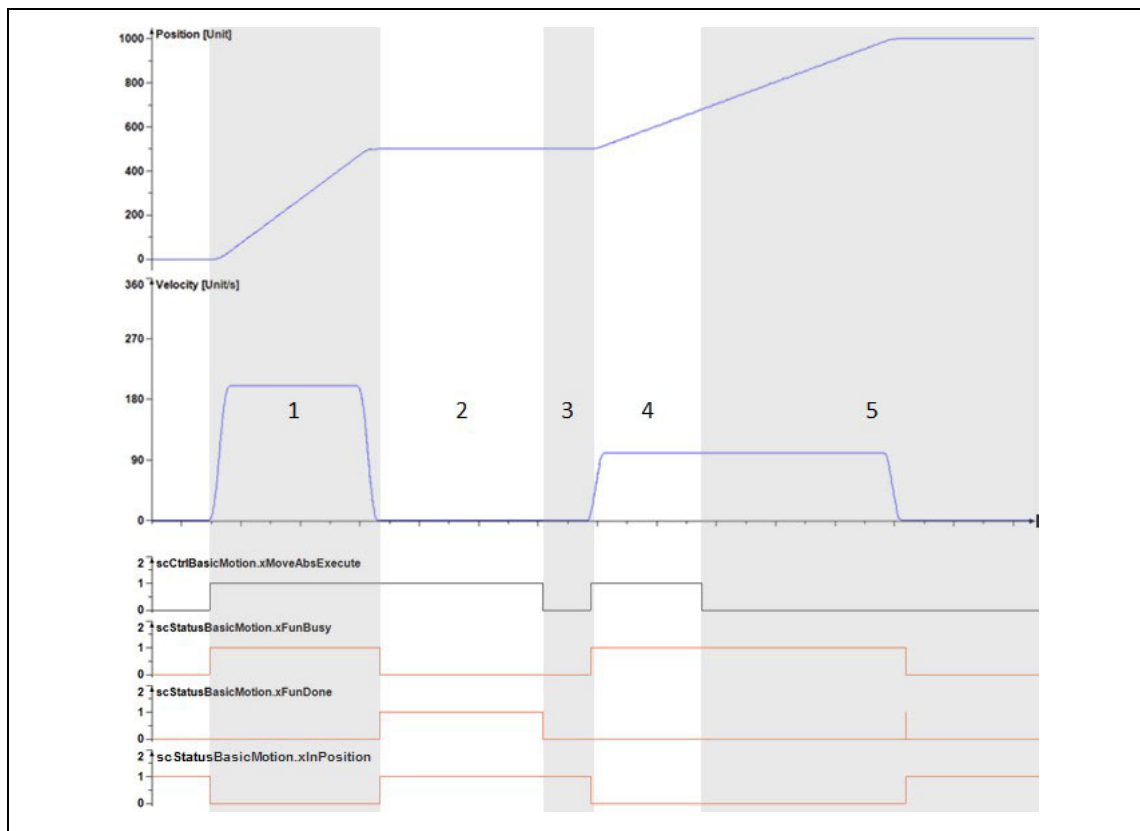
In order to reach the target position, the axis is brought to a standstill via the *lrDec* parameter. The axis reaches a standstill exactly in the target position.

After reaching the target position, the state changes back to STANDBY (*xMoveAbsExecute* = FALSE).

During the travel process, the absolute target position in the *lrPos_Dist* input can be changed, which requires triggering the Execute input for positioning again.

Positioning can be interrupted via *xAbort*, *xHalt* or *xQSPApplication*. Via *xFinishPositioning*, positioning can be continued again and completed.

Travel diagram



No	xMoveAbsExecute	xFuncBusy	xFunDone	xInPosition	eTMState	Info
1	FALSE ↗ TRUE	FALSE ↗ TRUE	FALSE	TRUE ↘ FALSE	STANDBY → MOVEABS	The axis starts the positioning run to the absolute target position 500 units.
2	TRUE	TRUE ↘ FALSE	FALSE ↗ TRUE	FALSE ↗ TRUE	MOVEABS → STANDBY	When the target position has been reached, <code>xInPosition</code> is set to TRUE.
3	TRUE ↘ FALSE	FALSE	TRUE ↘ FALSE	TRUE	STANDBY	When <code>xMoveAbsExecute</code> is set to FALSE, the <code>xFunDone</code> status signal changes from TRUE ↘ FALSE.
4	FALSE ↗ TRUE	FALSE ↗ TRUE	FALSE	TRUE ↘ FALSE	STANDBY → MOVEABS	The axis starts the positioning run to the absolute target position 1000 units.
5	TRUE ↘ FALSE	TRUE ↘ FALSE	FALSE ↗ TRUE ↘ FALSE	FALSE ↗ TRUE	MOVEABS → STANDBY	After the target position has been reached, <code>xFunDone</code> remains TRUE for exactly one cycle, because <code>xMoveAbsExecute</code> = FALSE has been requested during the positioning process.

Parameters to be set

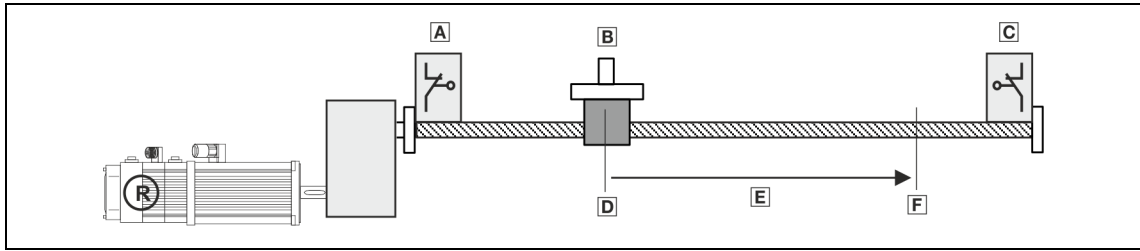
- *IrVel*, initial value: 3600 [units/s]
- *IrAcc*, initial value: 36000 [units/s²]
- *IrDcc*, initial value: 36000 [units/s²]
- *IrJerk*, initial value: 0 [units/s³]

Further parameters for absolute positioning can be found in the `AXIS_REF` in subindex 130 ... 137.

8 Description of the basic functions

8.6 Relative positioning

8.6 Relative positioning



[8-3] Example: Relative positioning of a linear axis

Pos.	Description
A	Negative hardware limit switch
B	Load (e.g. slide)
C	Positive hardware limit switch
D	Current actual position
E	Distance to be travelled (= <i>lrPos_Dist</i> input)
F	Target position (input current actual position + <i>lrPos_Dist</i>)

With relative positioning, the target position defined in the *lrPos_Dist* input corresponds to the distance to be travelled ($F = E$), meaning that the drive exactly travels the distance defined from the current actual position (D).

Prerequisite

- The axis is enabled (*xReadyForMotion* = TRUE).

Execution

With a rising edge (FALSE→TRUE) at the *xMoveRelExecute* input, a travel to the relative target position (*lrPos_Dist* input) is started on the basis of the parameters for the motion profile (see below). The technology module changes to the "MOVEREL" state.

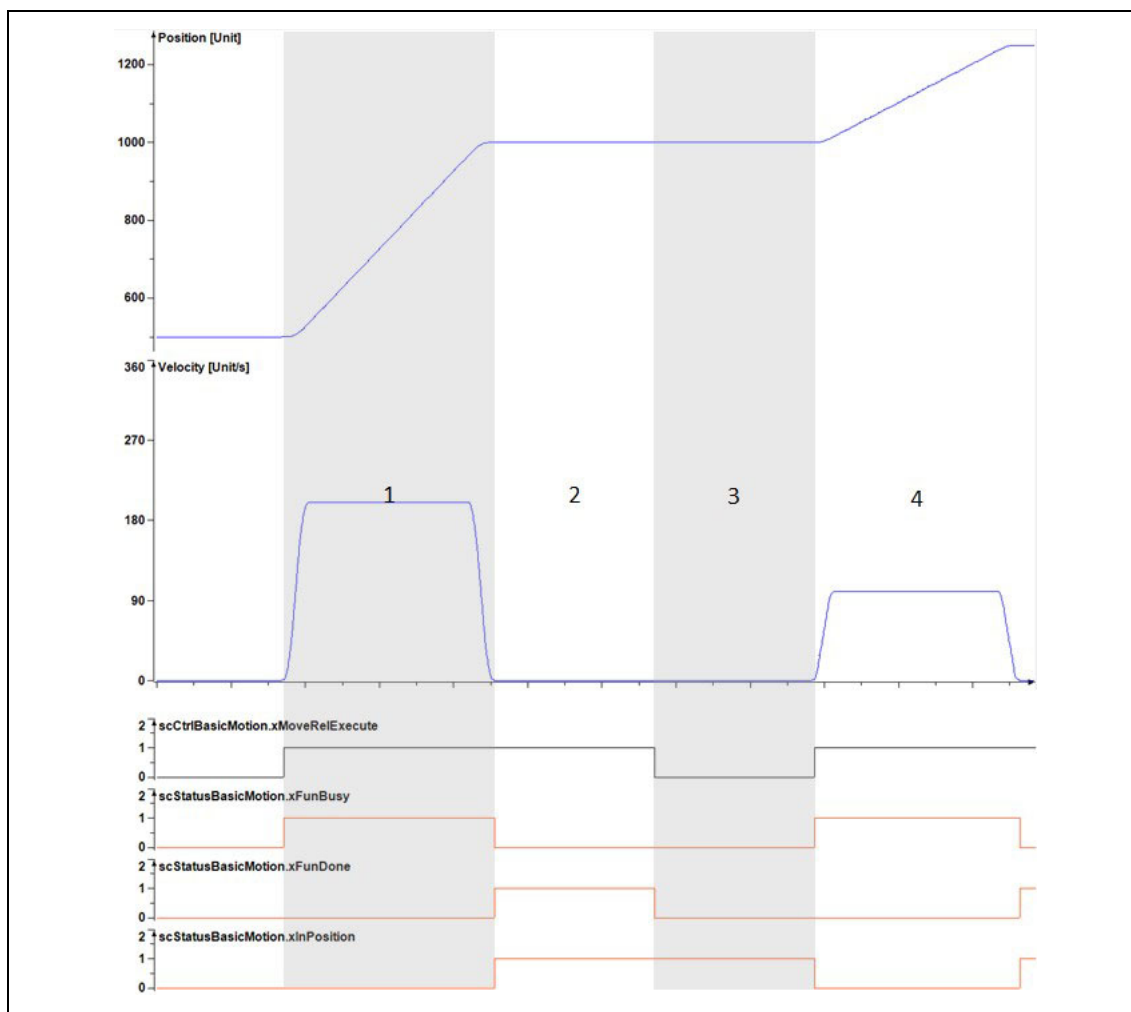
In order to reach the target position, the axis is brought to a standstill via the *lrDec* parameter. The axis reaches a standstill exactly in the target position.

After reaching the target position, the state changes back to STANDBY (*xMoveRelExecute* = FALSE).

During the travel process, the absolute target position in the *lrPos_Dist* input can be changed, which requires triggering the Execute input for positioning again.

Positioning can be interrupted via *xAbort*, *xHalt* or *xQSPApplication*. Via *xFinishPositioning*, positioning can be continued again and completed.

Travel diagram



No	xMoveRelExecute	xFuncBusy	xFuncDone	xInPosition	eTMState	Info
1	FALSE ↗ TRUE	FALSE ↗ TRUE	FALSE	FALSE	STANDBY → MOVEREL	The axis starts the positioning run to the relative target position 500 units.
2	TRUE	TRUE ↘ FALSE	FALSE ↗ TRUE	FALSE ↗ TRUE	MOVEREL → STANDBY	As soon as the relative target position has been reached, <i>xInPosition</i> and <i>xFuncDone</i> are set to TRUE.
3	FALSE ↗ TRUE	FALSE-	TRUE ↘ FALSE	TRUE	STANDBY	When <i>xMoveRelExecute</i> is set = FALSE, <i>xFuncDone</i> changes from TRUE ↘ FALSE.
4	FALSE ↗ TRUE	FALSE ↗ TRUE ↘ FALSE	TRUE ↘ FALSE ↗ TRUE	TRUE ↘ FALSE ↗ TRUE	STANDBY → MOVEREL → STANDBY	The axis starts the positioning run to the relative target position 250 units. After reaching the relative target position, <i>xInPosition</i> and <i>xFuncDone</i> are set to TRUE.

Parameters to be set

- *IrVel*, initial value: 3600 [units/s]
- *IrAcc*, initial value: 36000 [units/s²]
- *IrDcc*, initial value: 36000 [units/s²]
- *IrJerk*, initial value: 0 [units/s³]

Further parameters for relative positioning can be found in the AXIS_REF in subindex 130 ... 137.

8.7 Continuous travel with a defined velocity

Prerequisite

- The axis is enabled (*xReadyForMotion* = TRUE).

Execution

With a rising edge (FALSE → TRUE) at the *xMoveVelExecute* input, the continuous travel is started on the basis of the parameters for the motion profile (see below). The technology module changes to the "MOVEVEL" state.

The axis travels until a software limit position has been reached or a hardware limit switch has been approached:

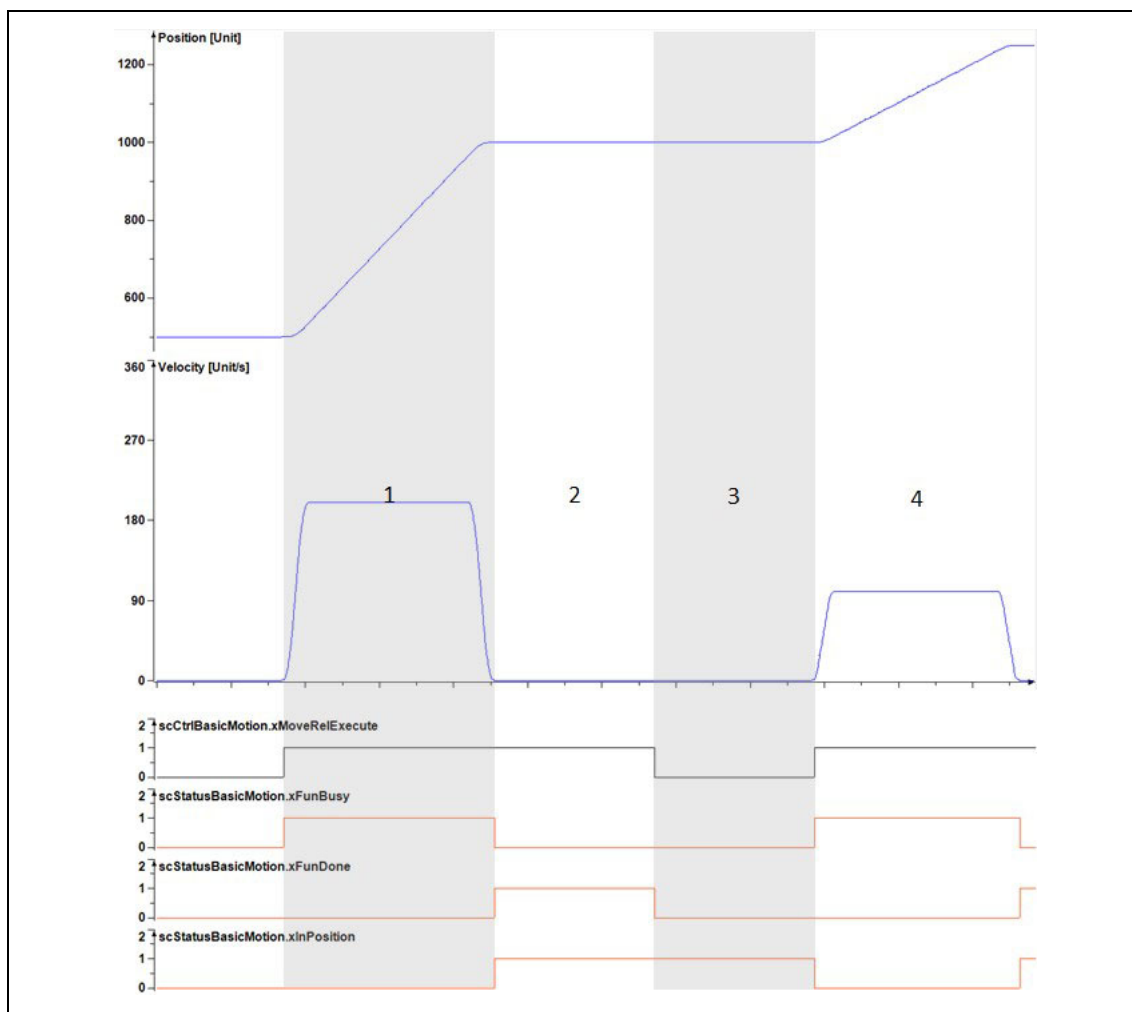
- Input *xDisableSWLimit* = FALSE
- Output *xSwLimitEnabled* = TRUE
- Output *xSwLimitSwitchPos/Neg* = TRUE
- Output *xHwLimitSwitchPos/Neg* = TRUE

When a software limit position has been reached, the axis is brought to a standstill via the *IrDec* parameter.

When a hardware limit switch has been reached, the axis is brought to a standstill via the QSP application deceleration ramp.

Then the state is changed back to "STANDBY".

Travel diagram



No	<code>xMoveVelExecute</code>	<code>xAbort</code>	<code>xFuncBusy</code>	<code>xFuncDone</code>	<code>xConstantVelocity</code>	<code>eTMState</code>	Info
1	FALSE ↗ TRUE ↘ FALSE	FALSE	FALSE ↗ TRUE	FALSE	FALSE	STANDBY → MOVEVEL	The axis accelerates to the target speed of 200 units/s.
2	FALSE	FALSE	TRUE	FALSE ↗ TRUE	FALSE ↗ TRUE	MOVEVEL	After reaching the target speed, <code>xFuncDone</code> changes from FALSE ↗ TRUE and <code>xConstantVelocity</code> changes from FALSE ↗ TRUE.
3	FALSE	FALSE ↗ TRUE	TRUE ↘ FALSE	TRUE ↘ FALSE ↗ TRUE	TRUE ↘ FALSE	MOVEVEL → STANDBY	The speed run is terminated by <code>xAbort = TRUE</code> and brought to a standstill with the profile deceleration. When standstill has been reached, <code>xFuncBusy</code> is set = FALSE.

Parameters to be set

- *IrVel*, initial value: 3600 [units/s]
- *IrAcc*, initial value: 36000 [units/s²]
- *IrDcc*, initial value: 36000 [units/s²]
- *IrJerk*, initial value: 0 [units/s³]

Further parameters for continuous travel can be found in the AXIS_REF in subindex 50 ... 53 and subindex 104 ... 105.

8.8 Aborting and completing a positioning process

Prerequisite

- The axis is enabled ($xReadyForMotion = \text{TRUE}$).

Execution

An active positioning process is aborted with a rising edge ($\text{FALSE} \rightarrow \text{TRUE}$) at the $xAbort$ input.

The axis travels to a standstill with the deceleration ramp $scPar.IrDec$.

- Exception: **TM SpeedControl** and **TM TablePositioning**

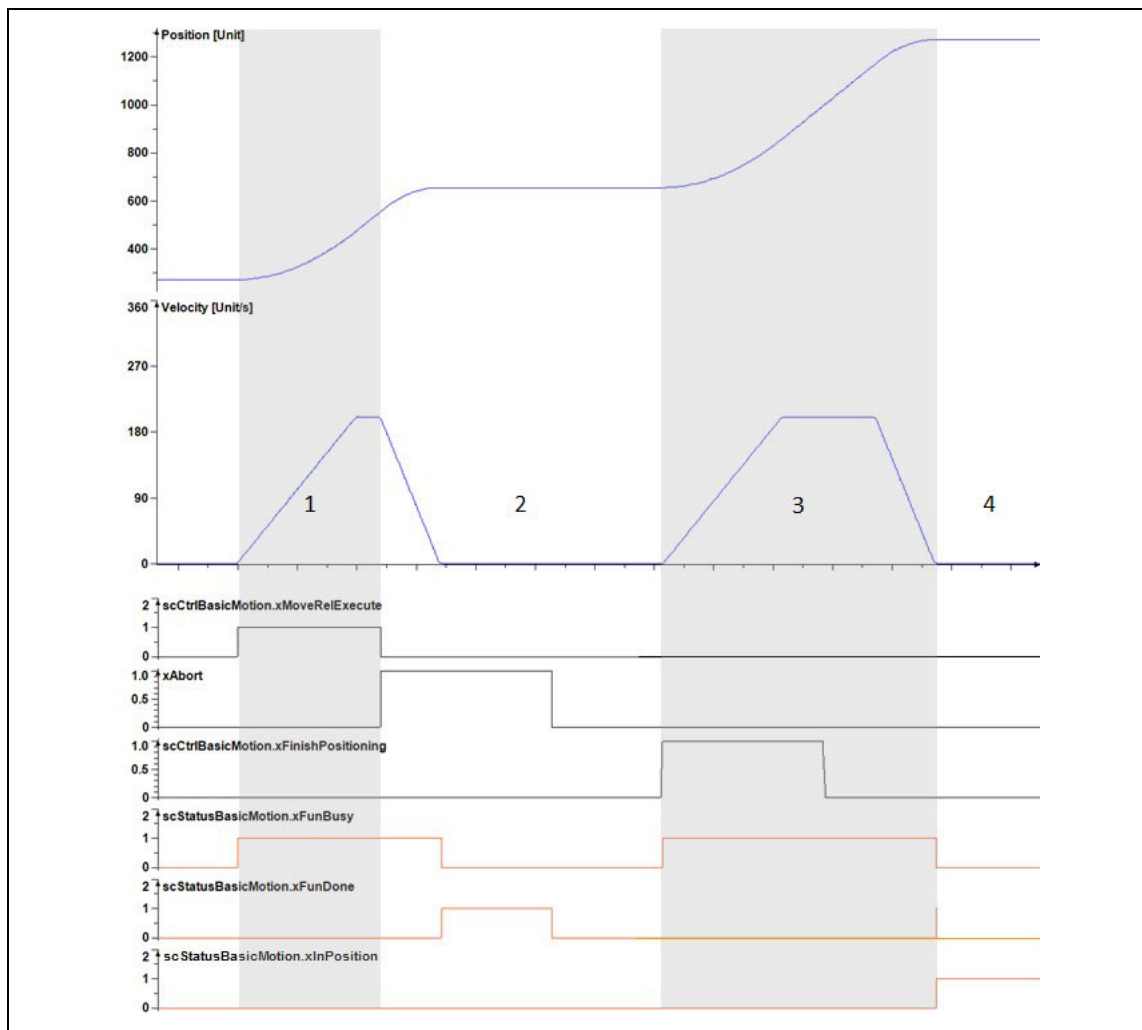
Both technology modules provide several profile data sets, one of which can be active.

If one of these profile data sets is active, the axis moves to standstill with the deceleration ramp of this active profile data set.

If no profile data set is active, the default deceleration ramp $scPar.IrDec$ is used.

The positioning process is completed with a rising edge ($\text{FALSE} \rightarrow \text{TRUE}$) at the $xFinishPositioning$ input.

Travel diagram



No	xMoveRel Execute	xAbort	xFinish Positioning	xInPosition	eTMState	Info
1	FALSE ↗ TRUE	FALSE	FALSE	FALSE	STANDBY→ MOVEREL	The axis starts the positioning process with the relative target position 1000 units.
2	TRUE↘ FALSE	TRUE↘ FALSE	FALSE	FALSE	MOVEREL→ STOP→ STANDBY	The positioning process is aborted with <i>xAbsort</i> = TRUE. The axis is brought to a standstill with the profile deceleration.
3	FALSE	FALSE ↗ TRUE	FALSE↗ TRUE↘ FALSE	FALSE	STANDBY→ MOVEREL	With <i>xFinishPositioning</i> = TRUE, the positioning process is continued.
4	FALSE	FALSE	FALSE	FALSE ↗ TRUE	MOVEREL→ STANDBY	As soon as the relative target position has been reached, <i>xInPosition</i> and <i>xFunDone</i> are set to TRUE.
	FALSE ↗ TRUE	FALSE↗ TRUE↘ FALSE	TRUE↘ FALSE↗ TRUE	TRUE↘ FALSE↗ TRUE	STANDBY→ MOVEREL→ STANDBY	The axis starts the positioning process with the relative target position 250 units. After reaching the relative target position, <i>sxInPosition</i> and <i>xFunDone</i> are set to TRUE.

8.9

Override for speed

Prerequisite

- The axis is enabled ($xReadyForMotion = \text{TRUE}$).

Execution

In order to be able to set overrides for the active profile data, the following inputs are available at the technology module:

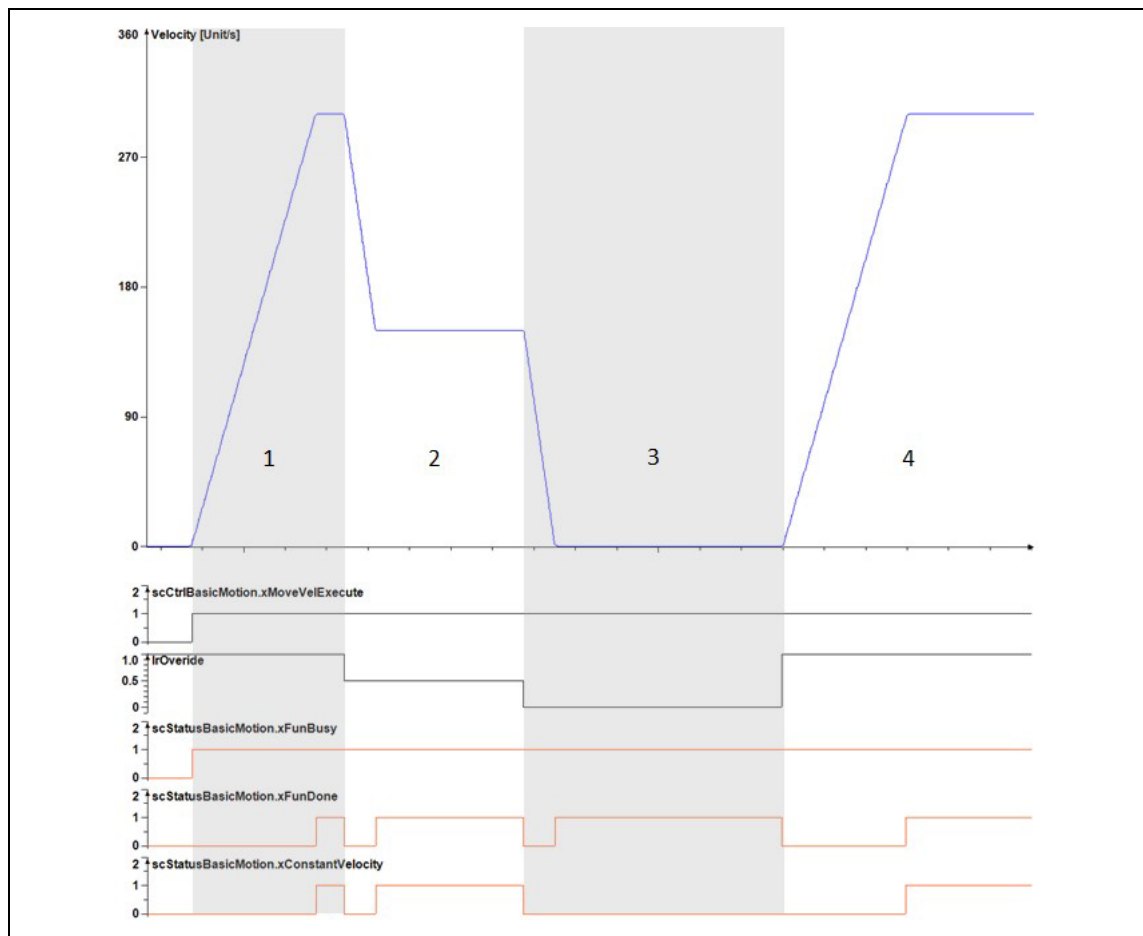
- *IrOverride*, initial value: 1.0
- *scCtrlBasicMotion.IrOverrideAcc*, initial value: 1.0 (100 %)
- *scCtrlBasicMotion.IrOverrideJerk*, initial value: 1.0 (100 %)

For profile-generating functions, separate overrides for speed, acceleration, deceleration and jerk can be set.

The values set act as a factor and are multiplied with the value active in each case.

Example

In this example, the use of the *IrOverride* input is shown.



8 Description of the basic functions

8.9 Override for speed

No	xMoveVel Execute	IrOverride	xFuncBusy	xFunDone	xConstant Velocity	eTMState	Info
1	FALSE ↗ TRUE	1	FALSE ↗ TRUE	FALSE ↗ TRUE	FALSE ↗ TRUE	STANDBY→ MOVEVEL	The axis accelerates to the target speed of 300 units/s.
2	TRUE	0.5	TRUE	TRUE↘ FALSE↗ TRUE	TRUE↘ FALSE↗ TRUE	MOVEVEL	The override value is set to 0.5 and the axis decelerates to 150 unit/s with the profile deceleration
3	TRUE	0	TRUE	TRUE↘ FALSE↗ TRUE	TRUE↘ FALSE	MOVEVEL	The override value is set to 0 and the axis is brought to a standstill to 0 unit/s with the profile deceleration
4	TRUE	1	TRUE	TRUE↘ FALSE↗ TRUE	FALSE ↗ TRUE	MOVEVEL	The override value is set to 1 and the axis accelerates to 300 unit/s with the profile deceleration

9 Description of the technology function

9.1 Defining the winding direction (winding/unwinding)

In order that the feedforward control values, the disturbance compensation and the correcting signal of the dancer position controller always act in the required direction, the "normal winding direction" must be defined once.

The *xWindingDirection* input serves to set whether the winding drive is to operate as unwinder or rewinder with regard to the normal material flow direction with positive line velocity.

- *xWindingDirection* = TRUE: Unwinder (material is unwound.)
- *xWindingDirection* = FALSE: Rewinder (material is rewound.)

⚠ WARNING!

Injury to persons

Risk of injury is caused by changing the setting at input *xWindingDirection* during a movement. This can lead to a speed jump of the winder axle.

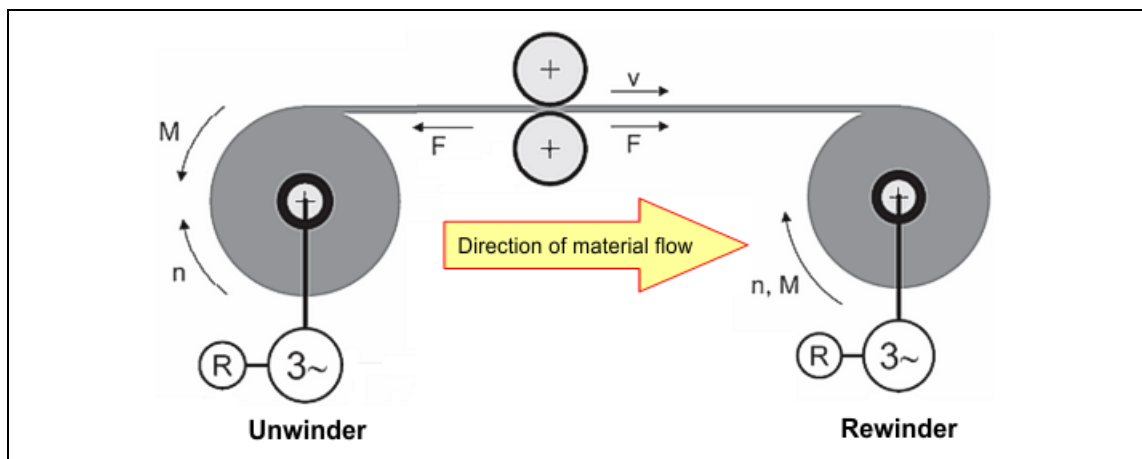
Possible consequences

Death or severe injuries

Protective measures

Only change the setting of input *xWindingDirection* if the technology module is not running and the inverter is inhibited:

- Input *xEnable* = FALSE
- Input *xEnableOperation* = FALSE



[9-1] Effective direction of speed and torque as a function of the material flow

Automatic detection of the winding direction

After defining the winding direction, the winding drives can be actuated with a negative line speed even in the opposite direction. A modification of the signal flow is not required if the direction of the material flow is inverted. The current winding direction is shown at the *xUnwind* output.

9.2 Defining the material feed to the winder

The *xMaterialFeeding* input serves to define whether the material is fed to the winder from the top or the bottom.

► [L_TW2P_WinderDancerCtrlState function block](#) (16)

⚠ WARNING!

Injury to persons

Risk of injury is caused by changing the setting at input *xMaterialFeeding* during a movement. This can lead to a speed jump of the winder axle.

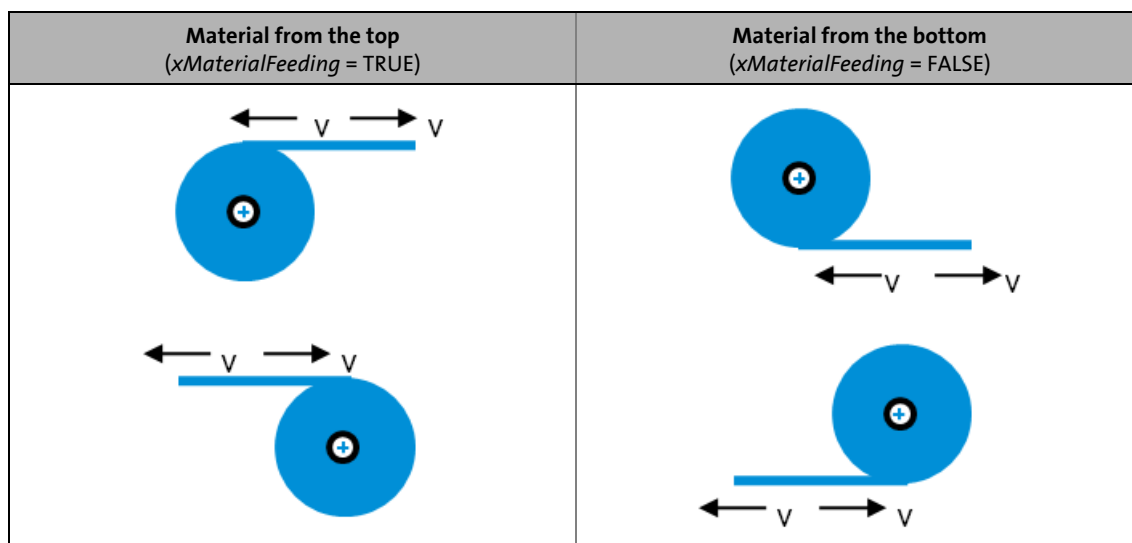
Possible consequences

Death or severe injuries

Protective measures

Only change the setting of input *xMaterialFeeding* if the technology module is not running and the inverter is inhibited:

- Input *xEnable* = FALSE
- Input *xEnableOperation* = FALSE



In principle, the rotation direction of the winder shaft is adapted to the material flow through the mounting direction of the motor.

Parameters to be set

The direction of rotation of the winder shaft is generally adapted to the material flow through the motor mounting direction:

Index: Subindex	Parameter name	Display
0x5050:010	Motor mounting direction	x

The parameters for the direction of rotation of the winder shaft are located in the parameter structure

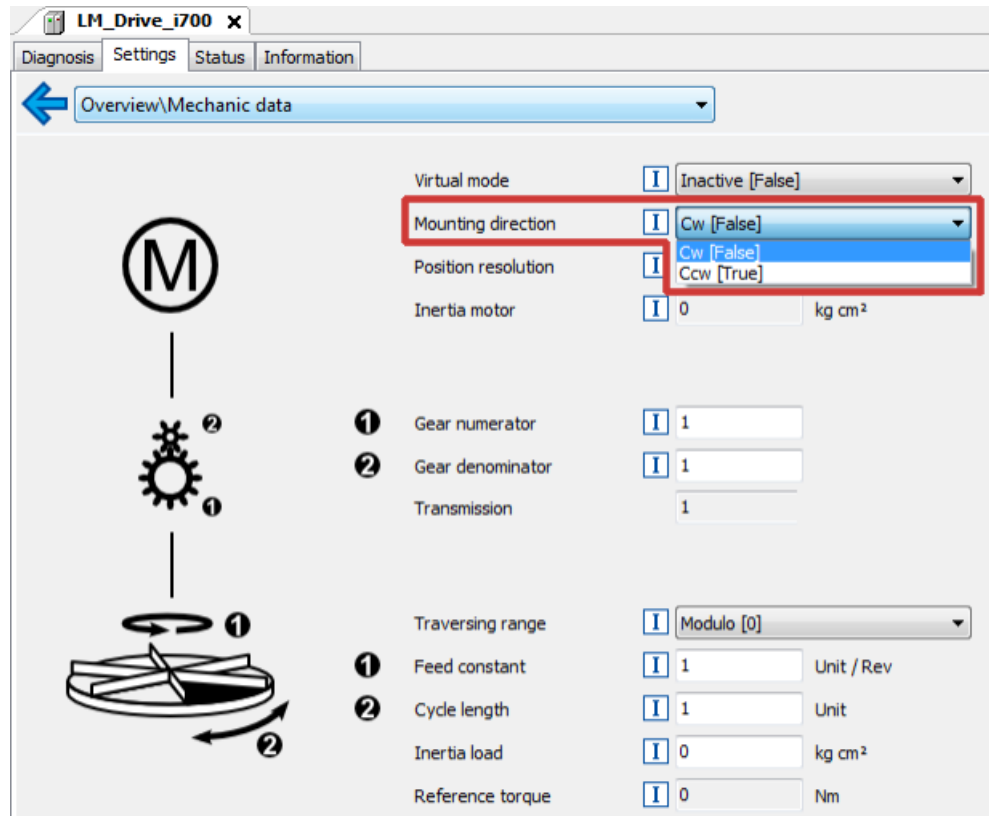
► [L_TW2P_scPar_WinderDancerCtrlState](#) (26)

9 Description of the technology function

9.2 Defining the material feed to the winder

Example

The direction of rotation of the axis can be set in the »PLC Designer« under the **Settings Configuration** tab:



9.3 Master value source for the diameter calculation

The technology module always operates with the line speed at the *IrSetLineVel* input.

For calculating the diameter, a separate encoder can be used for measuring the line speed between the dancer and the winding axis. In this case, the line speed for the diameter calculation from the encoder has to be interconnected at the *IrSetLineVelDiamCalc* input and the *xLineVelDiamCalc* parameter must be set to TRUE. In this way, taking the dancer movement into consideration is dispensable - the *IrDancerMaterialLength* parameter has to be set to 0.

If no encoder is used, the *xLineVelDiamCalc* parameter must be set to FALSE. The line speed at *IrSetLineVel* is used to calculate the diameter. In order to optimise the diameter calculation, the dancer movement can be taken into consideration.

► [Diameter calculation including consideration of the dancer movement](#) (📖 67)

Parameters to be set

The parameters for the diameter calculation with or without encoder are located in the parameter structure, see extract:

Identifier	Data type	Description
xLineVelDiamCalc	BOOL	Diameter calculation Initial value: FALSE
		TRUE For calculating the diameter, the velocity from the <i>IrSetLineVelDiamCalc</i> input is used.
		FALSE For calculating the diameter, the velocity from the <i>IrSetLineVel</i> input is used.
IrDancerMaterialLength	LREAL	Length of the material in the dancer - Unit: mm - Initial value: 0 - The value 0 is used to deactivate examination of the dancer movement.

► [L_TW2P_scPar_WinderDancerCtrlState](#) (📖 26)

9.4 Speed feedforward control

The setpoint speed for the speed feedforward control is calculated by dividing the line velocity at the *IrSetLineVel* input by the current diameter and the figure π :

Calculation of the setpoint speed for speed feedforward control		
$n_{\text{Set}} = \frac{v_{\text{Line}}}{d_{\text{act}} \times \pi}$		
Symbol	Description	Dimension unit
n_{Set}	Setpoint speed for speed feedforward control	rev/s
v_{Line}	Line velocity at the <i>IrSetLineVel</i> input	mm/s
d_{act}	Current diameter	mm

In order that the winder speed complies with the motor setpoint speed and the line velocity signal, the suitable setting for the motor reference speed is mandatory. For this purpose, the calculation and parameterisation are carried out automatically and not by the user.

The scaled winder setpoint speed at the *IrWndSpdRef* output refers to the motor speed required at minimum diameter d_{min} for reaching the reference line speed at the circumference of the reel.



How to check the speed feedforward control:

- Load the diameter calculator with the minimum diameter (d_{min}):
 - IrSetDiam* input = 0 (or $\leq d_{\text{min}}$)
 - Input *xLoadDiam* = TRUE
- With synchronisation to the line speed if the *xSyncLineVel* input = TRUE, the winder axis follows the line speed setpoint in a purely speed-controlled fashion without correcting the dancer position.
 - [Synchronisation to the line velocity](#) (□ 79)
- Start the line velocity master and increase the speed up to e.g. 50%. Now the winder should rotate with a speed corresponding to half the reference speed calculated at the *IrWndSpdRef* output.
- Now, the circumferential speed of the winder has to correspond to half the *IrLineVelRef* reference. The current line velocity signal is displayed at the output of the technology module *IrSetLineVelScaledOut* = 0.5 [$\times 100\%$] = 50 %.

If the speed or the direction of rotation is not correct, check the definition of the system data described above.

9.5

Diameter calculation

The current diameter is calculated by dividing the line velocity by the winder speed and the figure π :

Calculation of the current diameter		
$d_{act} = \frac{v_{Line}}{n_{Winder} \times \pi}$		
Symbol	Description	Dimension unit
d_{act}	Current diameter	mm
v_{Line}	Line velocity	mm/s
n_{Winder}	Winder speed	rev/s

In fact, however, for the calculation integrated speed values are used instead of instantaneous values. This causes a average determination. The number of revolutions causing a recalculation of the diameter is specified via the parameter *IrDiamCalcRegularDist*. The initial value of this parameter is set to 1 winding shaft revolution.

For quick diameter changes of *IrDiamCalcRegularDist*, it can be switched to the fast calculation mode by setting the *xDiamCalcReduced* input = TRUE. The lower calculation distance is set with the *IrDiamCalcReducedDist* parameter. Here, the initial value 1/10 is defined for the winder shaft revolution.

The smaller calculation distance is also activated automatically when a starting diameter is loaded. This state remains until a new diameter has been calculated. The function is required if the real diameter of the reel may deviate significantly from the loaded diameter. Thus, the winder shaft only rotates for a short distance with the "wrong" diameter. After the diameter has been calculated, a suitable value is available again.

As basic parameters, the minimum diameter (*IrMinDiam*) and maximum diameter (*IrMaxDiam*) are to be defined. On the one hand, these values act as a limitation, and secondly, all percentage or scaled diameter values refer to the maximum diameter.

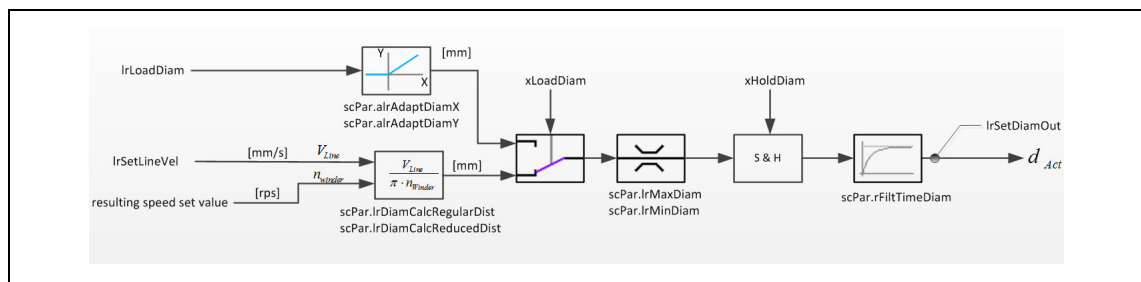
Parameters to be set

The parameters for the diameter calculation are located in the parameter structure, see extract:

Identifier	Data type	Description
IrMaxDiam	LREAL	Maximum diameter - Unit: mm - Initial value: 180
IrMinDiam	LREAL	Minimum diameter - Unit: mm - Initial value: 50
IrDiamCalcRegularDist	LREAL	Regular calculation distance for the diameter - Unit: rev - Initial value: 1
IrDiamCalcReducedDist	LREAL	Reduced calculation distance for the diameter - Unit: rev - Initial value: 0.1

► [L_TW2P_scPar_WinderDancerCtrlState](#) (26)

Signal flow for calculating the diameter



9.6 Holding the diameter

For some operating states of the winder, in which the line velocity does not correspond to the circumferential velocity of the reel, the current diameter cannot be calculated from the line velocity and the motor speed. In this case, the calculation of new values must be prevented and the diameter must be held at the old value.

If the diameter is held, the *xHoldDiamActive* output is = TRUE

This is done automatically under the following conditions:

- Line speed < minimum line speed
- In the "STOP", "ERROR", "READY", "JOGGING" and "SYNCLINEVEL" states.

For the user holding of the diameter, set the *xHoldDiam* input to TRUE.

If the actual line speed is noisy, the "Hold diameter" function can toggle.

Recommendation: Use the hysteresis function via the parameter *IrHystlineStandstill*.

The parameter *IrHystlineStandstill* defines the hysteresis width. The current line speed is evaluated via the hysteresis. The actual speed resulting from the hysteresis function is compared with the minimum line speed.

Parameters to be set

Identifier	Data type	Description
IrMinLineVel	LREAL	Minimum line velocity Up to this velocity, the diameter is held. • Unit: mm/s • Initial value: 1
xHoldDiam	BOOL	Hold the current diameter Initial value: FALSE
		TRUE Current diameter is held.
		FALSE The current diameter is not held.
IrHystlineStandstill	LREAL	Hysteresis width for the line speed This stabilises the noisy line speed and the direction recognition. After detecting the maximum noise amplitude of the input signal, set this value. Unit: mm/s Initial value: 0

► [L_TW2P_scPar_WinderDancerCtrlState](#) (26)

9.7 Defining the diameter via a sensor

At the start of a winding process, it may be required to define a start diameter or use the signal of a diameter sensor to be able to properly start the setpoint generation for the winding drive and also the interference compensation.

The *lrLoadDiam* input serves to define a start diameter that is accepted with the highest priority and loaded cyclically when *xLoadDiam* = TRUE.

An external diameter value, e.g. of an ultrasonic sensor, can also be connected to the *lrSetDiam* input. This analog value can be adapted via a curve function $Y = f(x)$. The curve function is set with nine grid points via the parameters *alrAdaptDiamX*[1...9] and *alrAdaptDiamY*[1...9]. In order that the analog value is used as start diameter, the adapted curve progression is initialised with *alrAdaptDiamY* = *alrAdaptDiamX*. The sensor signal can also be loaded permanently.

In practice, ultrasonic sensors with an analog output variable (0 ... 10 V) are often used for recording a diameter. A curve function $Y = f(x)$ serves to compensate for potential nonlinearities of such a sensor. The function acts on the value which can be diagnosed via the 0x5051:075 parameter. The curve function is set with nine interpolation points via parameters *Setdiameteradaption:scPar.alrAdaptDiamX* and *Setdiameteradaption:scPar.alrAdaptDiamY*. With the Lenze setting of the curve function, the value is not adapted, the start diameter is passed through 1:1.

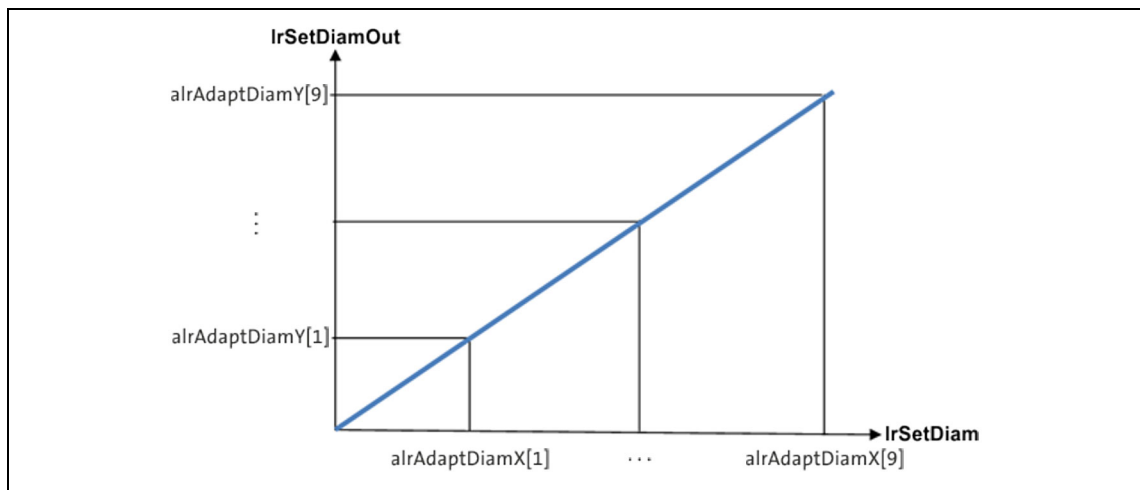
Parameters to be set

Identifier	Data type	Description
<i>lrLoadDiam</i>	LREAL	Defining a (start) diameter The diameter is loaded cyclically when the <i>xLoadDiam</i> is set to TRUE. - Unit: mm - Initial value: 0
<i>alrAdaptDiamX</i> ARRAY [1...9] OF LREAL		Grid points of the curve function for diameter loading - Values that may be applied to the analog input <i>lrSetDiam</i>. - Unit: mm - Initial values: 0, 100, 200, 300, 400, 500, 600, 700, 800
<i>alrAdaptDiamY</i> ARRAY [1...9] OF LREAL		Grid points of the curve function for diameter loading - Function values for the diameter - Unit: mm - Initial values: 0, 100, 200, 300, 400, 500, 600, 700, 800

► [L_TW2P_scPar_WinderDancerCtrlState](#) (26)

Example: Loading of a diameter via a curve function**Tip!**

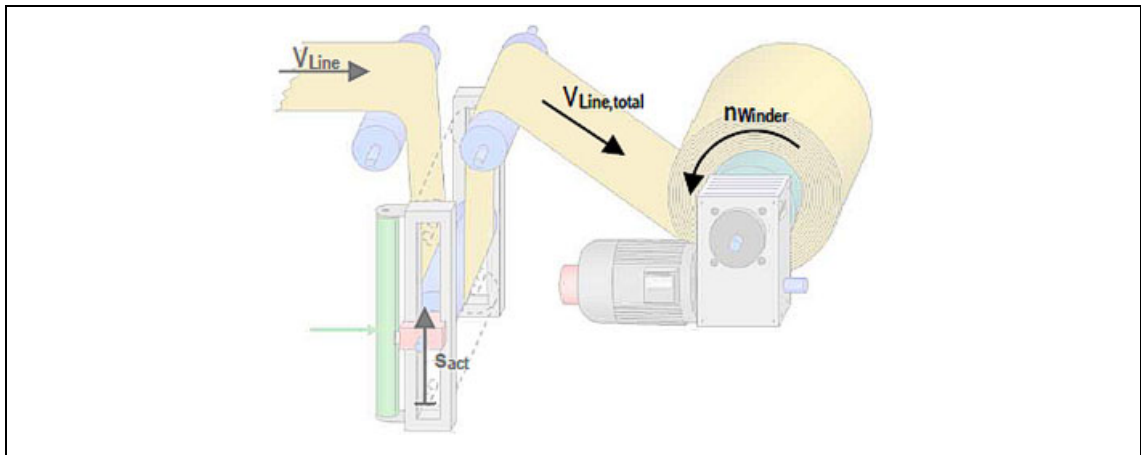
If the real diameter for loading is not known (no sensor, no operator input possible), a mean diameter should be loaded.



9.8 Diameter calculation including consideration of the dancer movement

In order to be able to correct the dancer position, the circumferential speed of the winder must be faster or slower than the line. If the circumferential speed upstream of the dancer is significantly reduced in comparison to the line speed in this process, the resulting circumferential speed $V_{\text{Line;total}}$ for the diameter calculation should be used.

This is usually the case in applications storing larger amounts of material in the dancer mechanism.



[9-2] Resulting circumferential velocity when the dancer position changes

The speed resulting from the movement of the dancer can be determined from the differentiation of the dancer position. The maximum material length stored corresponds to a change of the dancer position of 200%.

Storage volume in the dancer

The storage volume is defined via the *IrDancerMaterialLength* parameter and results from twice the path between the two limit positions multiplied by the number of material loops.

The dancer position is filtered via a PT1 filter with the *scPar.rFiltTimeActDancerVelComp* time constant.

This position is differentiated. Then the additional speed from the dancer movement is combined with the actual line speed in order to calculate the diameter *IrSetLineVel*.

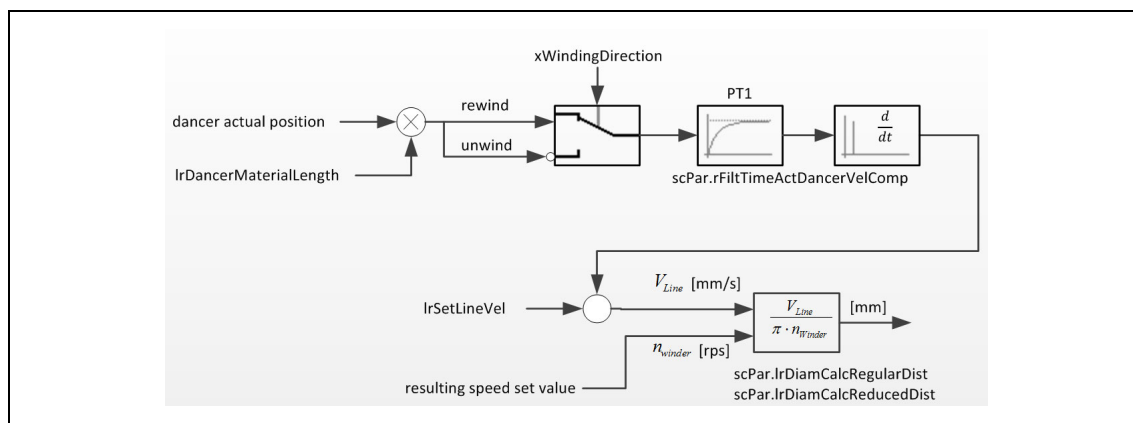
In order that the compensation can take effect, a suitable material length must be set in *scPar.IrDancerMaterialLength*. With the value 0, no additional speed is generated. Therefore, no compensation takes place.

Parameters to be set

Identifier	Data type	Description
IrDancerMaterialLength	LREAL	Length of the material in the dancer - Unit: mm - Initial value: 0 - The value 0 is used to deactivate examination of the dancer movement.
rFiltTimeActDancerVelComp	REAL	Filter time for the speed compensation - Unit: s - Initial value: 0

► [L_TW2P_scPar_WinderDancerCtrlState](#) (26)

Signal flow with dancer compensation



9.9 Identification of the diameter by lifting the dancer

If the winding process is started with an incorrect diameter, this may cause very extensive instabilities, since neither the speed feedforward control variable nor the adaption of the speed controller gain is correct.

In systems in which reels with unknown diameters are changed, the diameters should be identified in advance. Otherwise high levels of instability in the winding process can result because neither the speed feedforward control value nor the speed controller gain is suitable.

Lifting the dancer at line speed = zero makes it possible to identify a reel diameter by taking into consideration the dancer movement in the line speed signal.

In the course of the diameter identification and with an active adaptation of the speed controller gain (*xAdaptSpdCtrlGain* = TRUE), the speed controller gain is set to the defined value *scPar.lrlDiamSpCtrlGain*.

Condition

- The state machine of the technology module is in **Standby**.
- The axis is enabled (input *xRegulatorOn* = TRUE)
- The speed master of the system is at a standstill: line speed = 0
- The material length stored in the dancer, with the dancer lift available for the identification, must suffice to be able to carry out a winder shaft movement of at least twice the diameter calculation cycle *scPar.lrlDiamCalcReducedDist* even with a maximum diameter.
- At the start of the identification run, the dancer is located in a position from which a distance can be covered with a length sufficient to pass through at least 1 - 2 calculation cycles of the diameter calculation, e.g. in the lower limit position.
- The maximum material length stored in the dancer is set in [mm] in *scPar.lrlDancerMaterialLength*.
- The dancer limit positions have been suitably parameterised or taught, so that the position at the *lrlActDancerPosScaled* output are displayed scaled in a range from -1.0 to 1.0 (-100 % to 100 %).
- The motion parameters for the movement during the identification process are set appropriately to match the mechanical requirements.

Starting identification

Identification starts with a FALSE→TRUE edge at the *xExecuteldentDiam* input. The technology module then switches to the "IDENTDIAMETER" state.

Function sequence

The diameter is set to a start value of 50% of maximum diameter *scPar.IrMaxDiam*. Internal holding of the diameter is deactivated because the minimum line speed (*scPar.IrMinLineVel*) is not achieved.

If the speed controller adaption is activated by setting the *xAdaptSpdCtrlGain* input = TRUE, the speed controller adaption is set to the value from the *scPar.IrIdentDiamSpCtrlGain* parameter. The reduced calculation distance, *scPar.IrDiamCalcReducedDist*, is activated internally.

The movement of the winder axis starts with the specified profile parameters (speed, acceleration, deceleration and jerk).



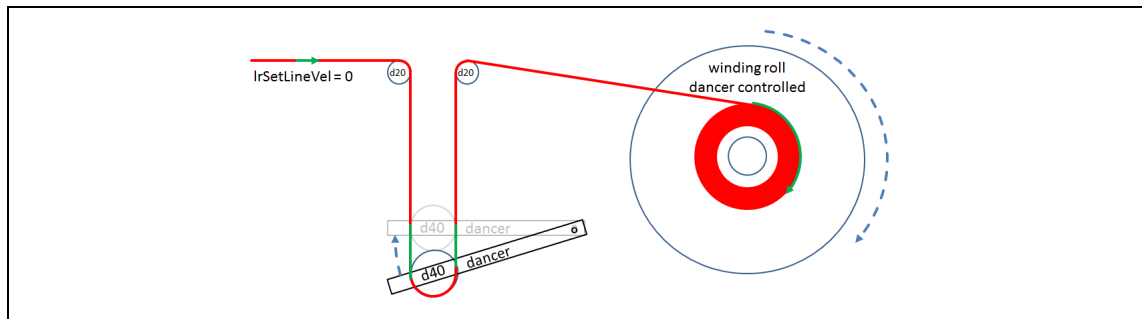
Tip!

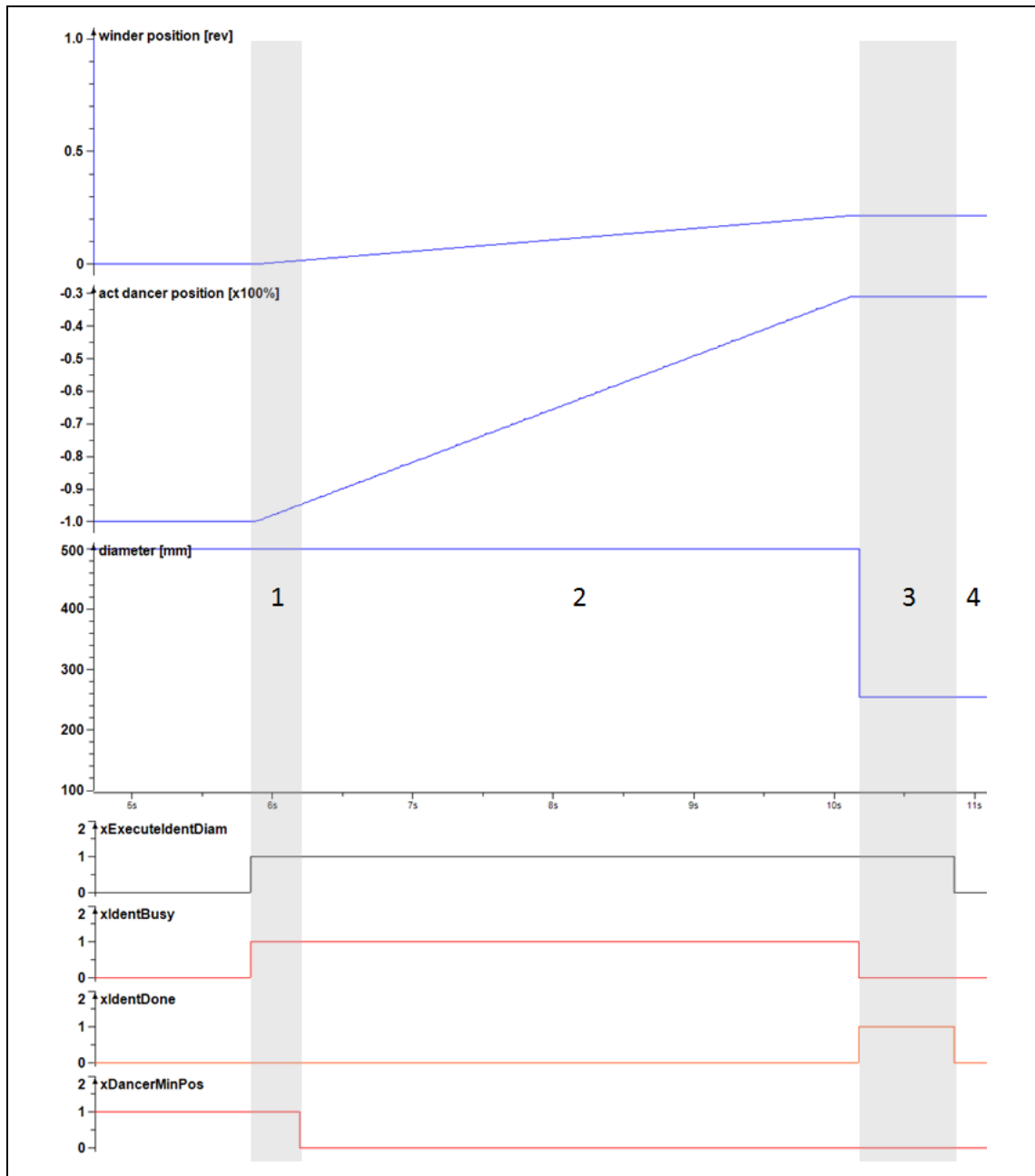
The profile parameters relate to the circumferential speed of the winder.

- Speed *scPar.IrIdentDiamVel* in [mm/s]
- Acceleration (*scPar.IrIdentDiamAcc*) in [mm/s²]
- Deceleration (*scPar.IrIdentDiamDec*) in [mm/s²]
- Jerk *scPar.IrIdentDiamJerk* in [mm/s³]

The active identification is shown at the IDENTBUSY status = TRUE. During the identification process, the dancer is lifted by rewinding of the reel. Identification has been successful if the diameter calculation cycle specified could be passed through with the expected frequency from *DiamIdent:scPar.IrDiamCalcReducedDist* at the winder shaft. The IDENTBUSY status is then reset to FALSE.

After successful completion of the identification run, the dancer can be closed-loop-controlled to the setpoint position immediately if *xDancerCtrl* = TRUE.





No	xExecutIdentDiam	xIdentBusy	xIdentDone	xDancerMinPos	eTMState	INFO
1	FALSE ↗ TRUE	FALSE ↗ TRUE	FALSE	TRUE	STANDBY ↗ IDENTDIAMETER	Identification of the diameter is commissioned. The rewinder (from the top) rewinds the material, lifting the dancer. The lower dancer limit position is deactivated, therefore rendering the calculation of the diameter inactive.
2	TRUE	TRUE	FALSE	TRUE ↘ FALSE	IDENTDIAMETER	The lower dancer limit position becomes inactive, which activates the diameter calculation over the reduced calculation distance.
3	TRUE	TRUE ↘ FALSE	FALSE ↗ TRUE	FALSE	IDENTDIAMETER ↘ STANDBY	2x reduced calculation distances are executed by the winder axis. This completes the identification process. The dancer lift and the selection of the material stored in the dancer determine the length of the material rewind. The length of the material rewind and the path of the winder axis determines the diameter. The "Done" status is set.
4	FALSE ↗ TRUE	FALSE	TRUE ↘ FALSE	FALSE	STANDBY	By resetting the job for identifying the diameter, the "Done" status for the identification is also reset.

Identification abort

If, during the identification process, the maximum dancer position, *scPar.lrlIdentDiamMaxDancerPos*, or the upper dancer limit position is exceeded (*scPar.lrlDancerMaxPosScaled*) without having been able to complete the identification process successfully, the movement is stopped with the parameters of FAST Stop, the ERROR status is reported: *xError* = TRUE *eErrorID* = 17165 "ActDancerPosGTMaxDancerPos".

The pending error can be acknowledged via the *xResetError* input. The technology module changes to the READY state if no error is pending.

9.10 Material length counter

Enable of the material length counter

The material length counter is activated with the *xEnable* input = TRUE.

The material length is calculated by integrating the line speed at the *IrSetLineVel* input and is shown at the *IrMaterialCounter* output (in millimetres). Depending on the winding direction, the material length is incremented or decremented.

► [Defining the winding direction \(winding/unwinding\)](#) (📖 57)

For the analog signal of the line speed, the current value of the material length can be filtered with a PT1 characteristic. The filter time is set with the *rFiltTimeMaterialCounter* parameter.

The current value of the material length is stored in the persistent data.

► [L_TW2P_PersistentVarWinder](#) (📖 95)

Preloading the material length counter

For the initialisation of the material length, an initial material length can be set via the *IrSetMaterialPos* parameter. With a FALSE→TRUE edge at the *xSetMaterialCounter* input, the initial material length is accepted with the highest priority.

Parameters to be set

Identifier	Data type	Description
IrSetMaterialPos	LREAL	Position of the material length counter With a FALSE→TRUE edge at the <i>xSetMaterialCounter</i> input, the material length counter (<i>IrMaterialCounter</i> output) is set to the value in <i>IrSetMaterialPos</i> . • Unit: mm • Initial value: 0
rFiltTimeMaterialCounter	REAL	Filter time constant for the material length counter (<i>IrMaterialCounter</i> output) • Initial value: 0 (filter is deactivated.)

► [L_TW2P_scPar_WinderDancerCtrlState](#) (📖 26)

Sources for the material length counting

The material length can be counted from one of three different sources, and the counting procedure can be carried out in two different ways.

Source	Selection of the source		INFO
	Connection to <i>MaterialCounterAxis</i>	<i>xLineVelDiamCalc</i>	
IrSetLineVel	no	FALSE	Material length counter via 'IrSetLineVel' input
IrSetLineVelDiamCalc	no	TRUE	Material length counter via 'IrSetLineVelDiamCalc' input
MaterialCounterAxis	Yes	is not evaluated	Material length counter via 'MaterialCounterAxis' reference axis

9.10.1 Material length counter via 'IrSetLineVel' input**Conditions**

- No reference axis is connected to the *MaterialCounterAxis* input.
- *xLineVelDiamCalc* parameter = FALSE

Operating mode

In addition to the position (*IrSetMaterialPos*), the line speed at the *IrSetLineVel* input is integrated into the material length count.

The resulting value is shown as material length at the *IrMaterialCounter* output and stored persistently.

With a FALSE→TRUE edge at the *xSetMaterialCounter* input, the material length from the *IrSetMaterialPos* parameter is loaded, setting the material length at the *IrMaterialCounter* output directly to the value of *IrSetMaterialPos*. The further count is added to the value of the material length set at the output.

In the case of a noisy signal, the material counting process is falsified by integration of the line speed. In this process, the material length counter shows a drift, even if the line is stable.

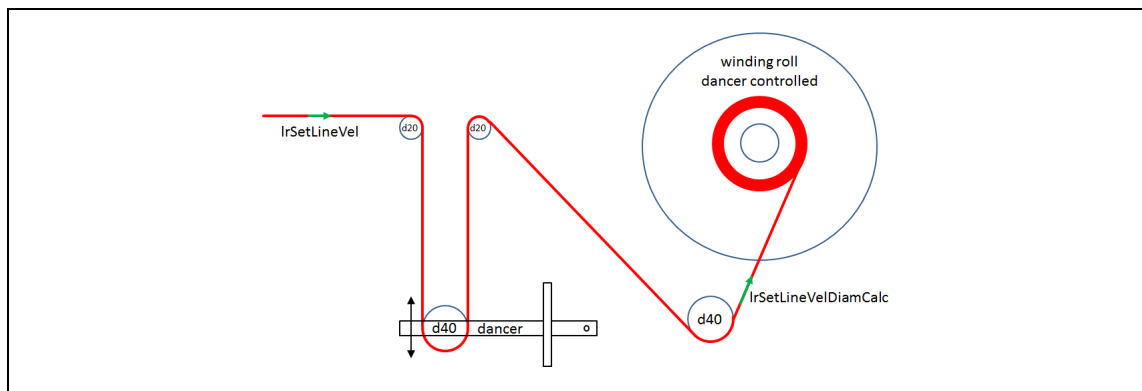
9.10.2 Material length counter via 'IrSetLineVelDiamCalc' input

Conditions

- No reference axis is connected to the *MaterialCounterAxis* input.
- *xLineVelDiamCalc* parameter = FALSE

Operating mode

In addition to the position (*IrSetMaterialPos*), the line speed at the *IrSetLineVelDiamCalc* input is integrated into the material length count. With the input, the speed should be measured directly upstream of the reel and downstream from the dancer, thus strongly reducing the impact of the dancer movement for the material length count.



The resulting value is shown as material length at the *IrMaterialCounter* output and is stored persistently.

With a FALSE→TRUE edge at the *xSetMaterialCounter* input, the material length from the *IrSetMaterialPos* parameter is loaded, setting the material length at the *IrMaterialCounter* output directly to the value of *IrSetMaterialPos*. The further count is added to the value of the material length set at the output.

In the case of a noisy signal, the material counting process is falsified by integration of the line speed. In this process, the material length counter shows a drift, even if the line is stable.

9.10.3 Material length counter via 'MaterialCounterAxis' reference axis

Conditions

A reference axis (modulo axis) is connected to the *MaterialCounterAxis* input.

Operating mode

The loss-free number of revolutions determined is used as a basis for the material length counting. This procedure is suitable for noisy signals!

In this process, whole revolutions and the fractional part of a revolution are recorded. Via the feed constant of the reference axis (modulo axis), the material length is shown at the *lrMaterialCounter* output.

The number of revolutions counted can be read out via the *MP20_liRevCounter* measuring point. The fractional amount of a revolution is shown via the *MP21_lrRevCounterResidual* measuring point. The values of these measuring points are stored persistently.

With a FALSE↗TRUE edge at the *xSetMaterialCounter* input, the material length is loaded from the *lrSetMaterialPos* parameter. In this process, the material length is converted to the number of revolutions and stored via the feed constant of the axis.

The material length from the *lrSetMaterialPos* parameter is shown at the *lrMaterialCounter* output.

An exact material length counting process can only be carried out with a slip-free measuring wheel. A measuring wheel that is prone to slip on the material causes errors in the material length counting.

9.11 Inching mode with 1/d evaluation



Tipp!

The "Safe speed" limitation function (SLS1, SLS2 ...) can limit the speed from the "Manual jog (jogging)" function (*xLineJogPos*, *xLineJogNeg*), if the parameter "160" (Follower SLS Reaction) of the winder axis is set to TRUE.

With this mode, manual inching of the line is executed. The function differs with regard to the **Manual jog** BaseMotion function. The feed constant in [mm/rev] of the winder is not constant, since the circumference of the winder changes with the current diameter. The actual speed of the winding shaft therefore not only depends on the manual speed parameterised, but also on the current diameter. This means that the current diameter, *IrSetDiamOut* output, should correspond to the real diameter if the inching mode is requested, otherwise the winding shaft may be moved at a speed that is not expected.

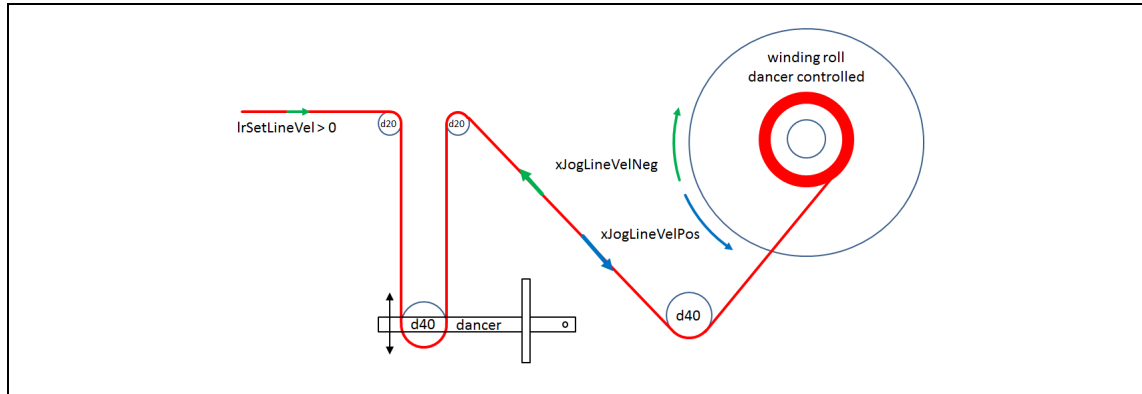
'Positive inching' means that the material is moved in the direction of the material flow.

'Negative inching' means that the material is moved opposite to the direction of the material flow.

The direction of rotation of the winding shaft is therefore also influenced by the material feeding which is defined via the *xMaterialFeeding* input.

When the *xJogLinePos* = TRUE, the line is driven in positive direction and when the *xJogLineNeg* = TRUE, it is driven in negative direction. The line is driven as long as the input remains set to TRUE. The running travel command cannot be replaced by the other jog command.

The parameterisable setpoints *IrJogLineVel*, *IrJogLineAcc* and *IrJogLineDec* for manual jog refer to the circumferential velocity or line velocity and not to the motor speed.



Parameters to be set

Identifier	Data type	Description
IrJogLineAcc	LREAL	Acceleration for the inching mode with the 1 / d evaluation. Selection of the value for the maximum acceleration speed variation. • Unit: mm/s ² • Initial value: 100
IrJogLineDec	LREAL	Deceleration for the inching mode with the 1 / d evaluation. Selection of the value for the maximum deceleration speed variation. • Unit: mm/s ² • Initial value: 10
IrJogLineVel	LREAL	Speed for the inching mode with the 1 / d evaluation. Selection of the maximum speed used for traversing. • Unit: mm/s • Initial value: 10

► [L_TW2P_scPar_WinderDancerCtrlState](#) (26)

9.12 Synchronisation to the line velocity

In the course of commissioning, still without material, a simple speed follower function can be used. This process serves to check whether the axis generally follows the master value with the speed expected, even if the diameter varies. In practice, a manual tachometer is contacted at the circumference of the winder as test equipment, displaying the actual circumferential speed. This can be compared to the *IrSetLineVel* shown. The current diameter *IrSetDiamOut* should correspond to the real diameter if the speed follower function is requested, otherwise the winding shaft may be moved at a speed that is not expected.

The winder axis is synchronised to the line velocity when *xSyncLineVel* is set to TRUE.

The parameters *IrSyncLineAcc* and *IrSyncLineDec* refer to the circumferential velocity or line velocity and not to the motor speed.

Parameters to be set

Identifier	Data type	Description
IrSyncLineAcc	LREAL	Acceleration for synchronising to line velocity • Unit: mm/s² • Initial value: 100
IrSyncLineDec	LREAL	Deceleration for synchronising to line velocity • Unit: mm/s² • Initial value: 100

► [L_TW2P_scPar_WinderDancerCtrlState](#) (26)

9.13 Scaling of the dancer position

The actual position of the dancer is led back to the Controller in the form of an analog signal (0 ... 10 V). The analog signal has to be applied to the *IrActDancerPosIn* input.

The dancer position is filtered via a PT1 filter with the *rFiltTimeActDancerPosIn* time constant. The filtered position is scaled from -1 to 1 and provided at the *IrActDancerPosScaled* output:

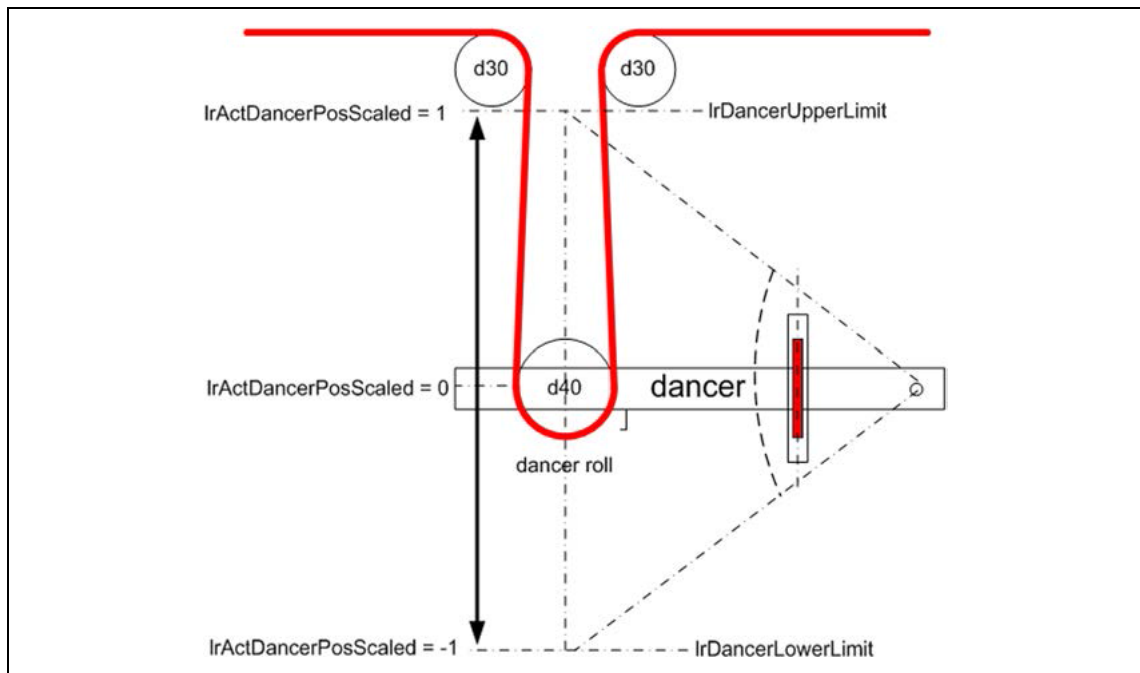
- Lower dancer limit position (in the dancer, the maximum material length is stored) corresponds to *IrActDancerPosScaled* = -1
- Upper dancer limit position (in the dancer, the minimum material length is stored) corresponds to *IrActDancerPosScaled* = 1.

This facilitates the selection of the setpoint and the monitoring of the dancer position.

Sources for determining the dancer limit positions

The winder technology has a total of two sources for determining the dancer limit positions.

- Source 1: Selection of the dancer limit positions via *scPar*
In order to select the dancer limit positions via the *scPar* parameter structure, the *scPar.xTechDancerLimits* parameter is to be set to FALSE. With this setting, the limit values for the upper and lower dancer position (dancer limit positions) are specified by the *IrDancerUpperLimit* and *IrDancerLowerLimit* parameters.
- Source 2: The dancer limit positions are learned via the teach function.
For the teach function of the dancer limit positions, the *scPar.xTechDancerLimits* parameter should be set to TRUE. In this setting, the dancer limit positions obtained from the teach function are used when the values for the upper and lower dancer limit position from the teach function are provided. If no values are provided by the teach function, the values for the dancer limit positions from the parameter structure will always be specified.

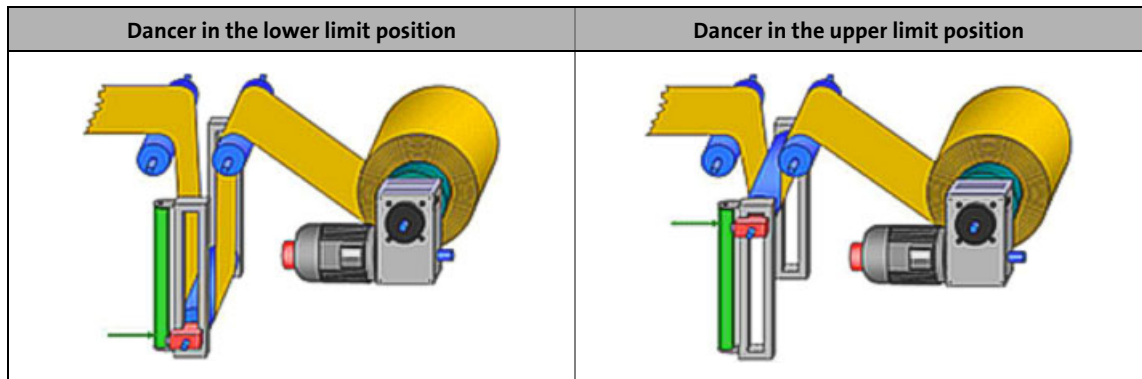


Parameters to be set

Identifier	Data type	Description
xTeachDancerLimits	BOOL	Source of the dancer position limitations • Initial value: FALSE
		TRUE The limit positions <i>IrDancerLowerLimit</i> and <i>IrDancerUpperLimit</i> are used until the teaching function has been executed. After the teaching function has been executed manually, the saved limit positions from the teaching function are always used.
		FALSE The limit positions <i>IrDancerLowerLimit</i> and <i>IrDancerUpperLimit</i> are used.
IrDancerUpperLimit	LREAL	Analog value for the upper dancer position limit • Initial value: 10 000 000
IrDancerLowerLimit	LREAL	Analog value for the lower dancer position limit • Initial value: 0
rFiltTimeActDancerPosIn	REAL	PT1 filter time for the <i>IrActDancerPosIn</i> input • Unit: s • Initial value: 0.005

► [L_TW2P_scPar_WinderDancerCtrlState](#) (26)

9.14 Teaching function for dancer limit positions



How to execute the teaching manually:

Note: Limit positions already learned are overwritten when executing the teaching function.

1. Set `xTeachDancerLimits` parameter = TRUE
2. Move the dancer manually to the lower limit position so that a maximum of material is available in the dancer.
3. Set the `xTeachLowerPos` position = TRUE.

At rising edge (FALSE→TRUE) the current input value `IrActDancerPosIn` is stored. If the input value is connected via the `PersistentVar` input, it is written to the Retain/Persistent memory in the `IrDancerLowerLimit` variable.

4. Move the dancer manually to the upper limit position so that a minimum of material is saved in the dancer.
5. Set the `xTeachUpperPos` input = TRUE.

At rising edge (FALSE→TRUE) the current input value `IrActDancerPosIn` is stored. If the input value is connected via the `PersistentVar` input, it is written to the Retain/Persistent memory in the variable `IrDancerUpperLimit`.

6. After completing the teaching function, check the plausibility of the persistent variables.

▶ [L_TW2P_scPar_WinderDancerCtrlState](#) (26)

▶ [L_TW2P_PersistentVarWinder](#) (95)



To check the learned limit positions via the dancer position control:

1. Set the controller enable with *xEnableOperation* input = TRUE.
2. Activate the dancer position control with *xDancerCtrl* input = TRUE.
Note: The material must be at a standstill.
3. Set the scaled setpoint for the dancer position at *IrSetDancerPosScaled* input = 1.
 - Check whether the dancer has reached the upper limit position.
 - If the upper limit position has not been reached, the teaching function must be executed again.
4. Set the scaled setpoint for the dancer position at *IrSetDancerPosScaled* input = -1.
 - Check whether the dancer has reached the lower limit position.
 - If the lower limit position has not been reached, the teaching function must be executed again.

As an alternative to teaching, you can enter the respective input values manually into the parameters *IrDancerUpperLimit* and *IrDancerLowerLimit*.

Parameters to be set

Identifier	Data type	Description
xTeachDancerLimits	BOOL	Source of the dancer position limitations Initial value: FALSE
		TRUE The limit positions <i>IrDancerLowerLimit</i> and <i>IrDancerUpperLimit</i> are used until the teaching function has been executed. After the teaching function has been executed manually, the saved limit positions from the teaching function are always used.
		FALSE The limit positions <i>IrDancerLowerLimit</i> and <i>IrDancerUpperLimit</i> are used.
IrDancerUpperLimit	LREAL	Analog value for the upper dancer position limit Initial value: 10 000 000
IrDancerLowerLimit	LREAL	Analog value for the lower dancer position limit Initial value: 0
xUpperPosReached	BOOL	Referenced upper dancer position
		TRUE The upper dancer limit position already referenced
xLowerPosReached	BOOL	Referenced lower dancer position
		TRUE The lower dancer limit position already referenced

► [L_TW2P_scPar_WinderDancerCtrlState](#) (26)

9.15 Monitoring of the dancer position

For operating the winder, monitoring of the dancer position is of importance in the following cases:

- When the dancer position controller has been enabled, the machine should only be started if the dancer is in the set position: Output *xDancerReachedSetPos* = TRUE.
- If the dancer reaches the area of the limit positions during operation, a web break may occur. This is probably the case if the dancer is at the lower limit position for a longer period of time.
 - *xDancerMinPos* output = TRUE: The lower limit position has been reached.
 - *xDancerMaxPos* output = TRUE: The upper limit position has been reached.

Parameters to be set

Identifier	Data type	Description
IrDancerMaxPosScaled	LREAL	Maximum dancer position for the <i>xDancerMaxPos</i> status bit • Unit: x 100 % (1 = 100 %) • Initial value: 0.95 (95 %)
IrDancerMinPosScaled	LREAL	Minimum dancer position for the <i>xDancerMaxPos</i> status bit • Unit: x 100 % (1 = 100 %) • Initial value: -0.95 (-95 %)
IrDancerInPosWindowScaled	LREAL	Window for the set position of the dancer to control the <i>xDancerReachedSetPos</i> status bit. • Unit: x 100 % (1.0 = 100 %) • Initial value: 0.2 (20 %)

► [L_TW2P_scPar_WinderDancerCtrlState](#) (26)

9.16 PI controller for the dancer position control

With dancer position control, the drive is actuated in closed-loop speed control. Activating the dancer position control with the *xDancerCtrl* input = TRUE initiates closed-loop control to an actual dancer position value.

The line speed signal multiplied with the reciprocal value of the diameter serves the purpose of feedforward control. The dancer position is recorded and compared with the setpoint position. If a deviation is detected, the dancer position controller provides for a correction of the speed setpoint.

The dancer controller works according to a typical PI structure. The control is enabled with *xDancerCtrl* = TRUE. As long as the *xDancerCtrl* input remains FALSE, the system deviation remains zero and the ramp generator for the dancer position setpoint is loaded with the actual value of the dancer limit position.

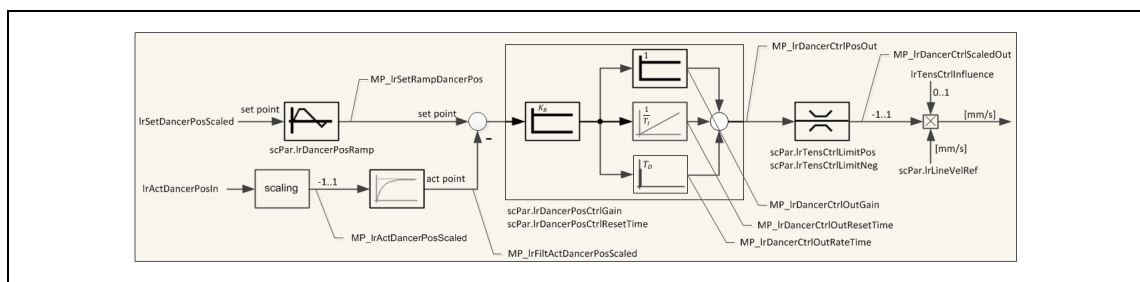
With a actual value signal of the dancer of the *lrActDancerPosIn* input, which is resolved to a limited extent, a PT1 filtering can have a positive impact on the stability of the control loop. On the other hand, dead times in the control loop should be avoided if a high dynamic performance is required. The filter time can be set with the *rFiltTimeActDancerPosIn* parameter (standard setting: 5 ms).

The correcting variable of the PI controller is led via a limiter, allowing for a limitation via the *scPar.lrDancerPosCtrlLimitPos* and *scPar.lrDancerPosCtrlLimitNeg* parameters.

The correcting variable has an additive effect in the main speed setpoint path. The impact via the *lrDancerPosInfluenceScaled* input acts multiplicatively on the correcting variable of the PI controller. In this way, the impact of the correction value is defined.

The controller gain is set via *lrDancerPosCtrlGain*. The I component of the PI controller can be set with the *lrDancerPosCtrlResetTime* parameter (controller reset time). In the standard setting, *lrDancerPosCtrlResetTime* is set = 0 (deactivated).

When the dancer position control has been activated, the dancer must be brought to the setpoint position first. In order to ensure that the dancer is lifted in a controlled fashion, the current actual position value for the position setpoint is loaded to the ramp generator first. The ramp is set via the *scPar.lrDancerPosRamp* parameter (standard setting: 1 = 100 %/s). This makes it unnecessary to display the impact of the dancer controller. The setting *scPar.lrDancerPosRamp* = 1 is set as a default in the technology and means that the setpoint is led from 0 to 1 within one second. In the winding mode, any change of the setpoint at the *lrSetDancerPosScaled* input is led via ramp generator.



[9-3] Signal flow of PI dancer position controller

When the input *xResetCtrl* = TRUE, the I component of the PI controller is switched off and the correcting variable (output of the controller) from the I component is led via the ramp function to '0'. The correcting variable from the P component is not affected.

Parameters to be set

Identifier	Data type	Description
IrDancerPosRamp	LREAL	Acceleration ramp for dancer position setpoints • Unit: 1 / s • Initial value: 1
IrDancerPosCtrlGain	LREAL	Controller gain • Initial value: 1
IrDancerPosCtrlResetTime	LREAL	Controller reset time • Unit: s • Initial value: 0 controller reset time is deactivated.)

► [L_TW2P_scPar_WinderDancerCtrlState](#) (26)

9.17 Reduced controller dynamics at low system deviation

A reduced dynamic performance of the controller with minor system deviations in most cases has a beneficial influence on the control loop damping behaviour.

The system deviation results from the difference of the setpoint value and the actual value, i.e. the difference from the *IrSetDancerPosScaled* input and the *IrActDancerPosScaled* output.

Parameters to be set

Identifier	Data type	Description
IrReducedGainWindow	LREAL	Area of the system deviation with a reduced gain / sensitivity. With <i>IrReducedGainWindow</i> , the tolerance zone of the system deviation is defined, in which <i>IrReducedGain</i> takes effect. • Initial value: 0.0
IrReducedGain	LREAL	Gain of system deviation within the reduced sensitivity • Initial value: 0.0

► [L_TW2P_scPar_WinderDancerCtrlState](#) (26)

9.18 Tension control via characteristic function (Base version)

For many rewinders, the tensile force must be reduced as the diameter increases to prevent the reel from being displaced, depending on the surface and type of winding material. This is referred to as the winding characteristic or tensile force characteristic.

It is common to adjust the tensile force in the winder control unit so that any adapted setpoints can be reproduced on a pneumatic actuator, for example.

The actual tensile force setpoint from the *IrSetTens* input is evaluated using a linear characteristic function on the basis of the diameter to ensure that the material-dependent characteristic is achieved.

The adaptation can be done according to different principles:

- ▶ [Linear tensile force characteristic \(dwSelectTensCurve = 0\)](#) (26 89)
- ▶ [Linear torque characteristic \(dwSelectTensCurve = 1\)](#) (26 90)
- ▶ [Freely definable characteristics with 65 interpolation points \(dwSelectTensCurve = 2\)](#) (26 91)

Parameters to be set

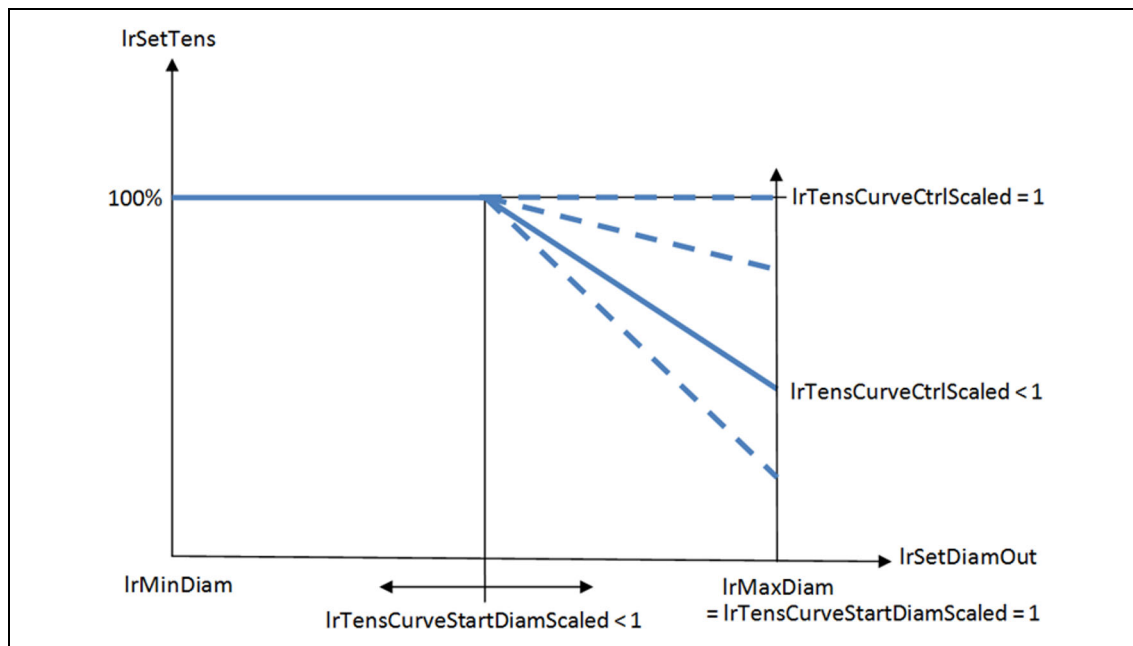
Identifier	Data type	Description
dwSelectTensCurve	DWORD	Selection of the characteristic for tension control Initial value: 0
		0 Linear tensile force profile
		1 Linear torque profile
		2 Tensile force profile on the basis of a specified characteristic
alrTensCurve	ARRAY[1..65] OF LREAL	Characteristic for tension control open loop consisting of 65 values.
IrTensCurveStartDiamScaled	LREAL	Initial point of the characteristic for the tension control open loop - Unit: x 100 % 1 = 100 % = <i>IrMaxDiam</i> parameter - Initial value: 0 (0 %)
IrTensCurveCtrlScaled	LREAL	Gradient of the characteristic for the tension control open loop - Unit: x 100 % (1 = 100 %) - Initial value: 0 The value '1' causes a constant tensile force profile and thus a proportionally increasing setpoint for the diameter.
IrMaxDiam	LREAL	Maximum diameter - Unit: mm - Initial value: 180
IrMinDiam	LREAL	Minimum diameter - Unit: mm - Initial value: 50

- ▶ [L_TW2P_scPar_WinderDancerCtrlState](#) (26 26)

9.18.1 Linear tensile force characteristic ($dwSelectTensCurve = 0$)

The characteristic is characterised by an initial range with constant evaluation (100 %) and a second range in which the tensile force is adapted to the diameter.

The parameter *IrTensCurveStartDiamScaled* defines the diameter at which the reduction in tensile force should start. With the parameter *IrTensCurveCtrlScaled* the tensile force at the maximum diameter is evaluated.

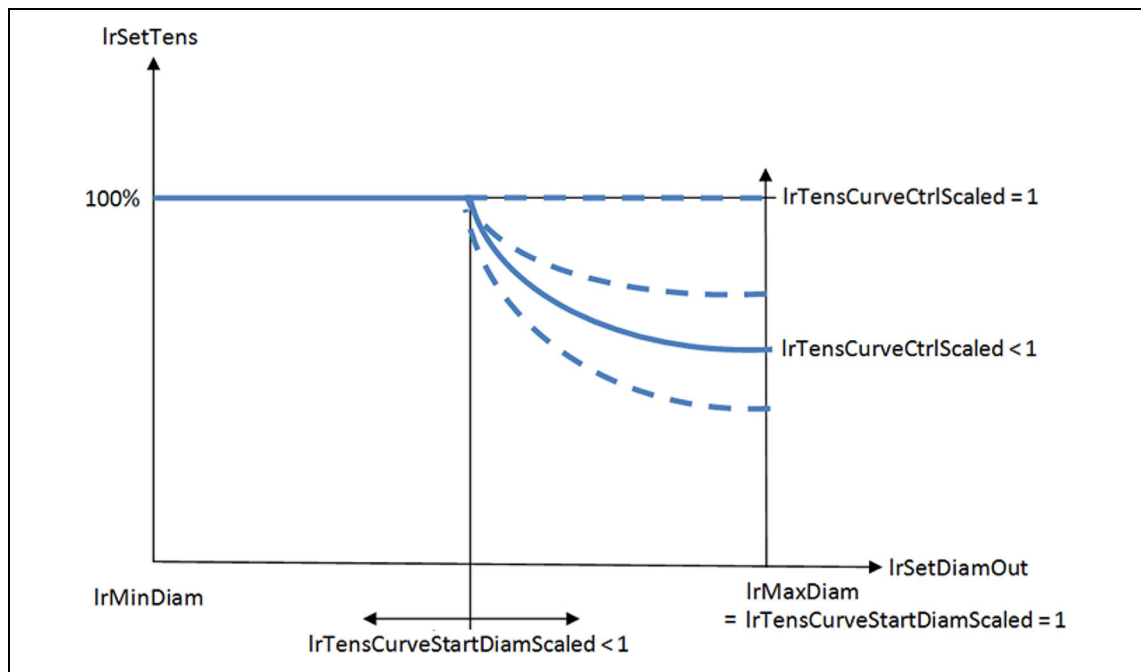


[9-4] Linear tensile force characteristic

9.18.2 Linear torque characteristic (dwSelectTensCurve = 1)

The characteristic is characterised by an initial range with constant evaluation (100 %) and a second range in which the tensile force is adapted linearly over the torque curve. Consequently, the characteristic is adapted via non-linear functions over the diameter.

The parameter *IrTensCurveStartDiamScaled* defines the diameter at which the reduction in tensile force starts. The parameter *IrTensCurveCtrlScaled* defines the percentage of the tensile force at the maximum diameter.

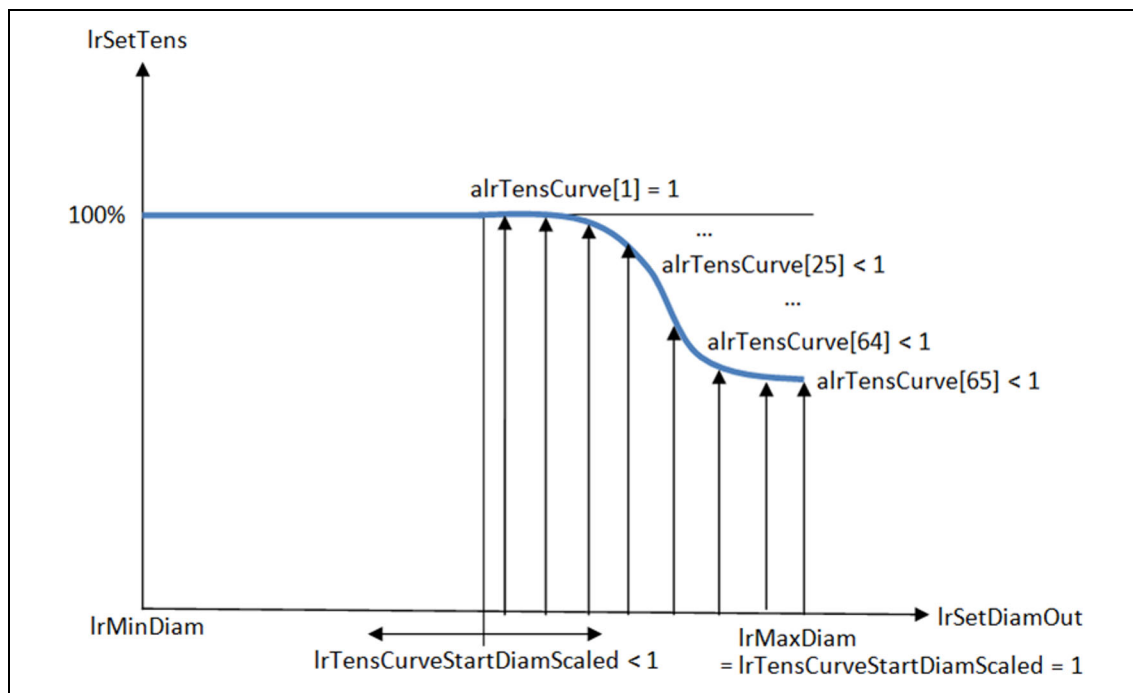


[9-5] Linear torque characteristic

9.18.3 Freely definable characteristics with 65 interpolation points ($dwSelectTensCurve = 2$)

The characteristic can be parameterized via the *alrTensCurve* array. It comprises 65 values, evenly distributed over the diameter range of the parameters *IrMinDiam* and *IrMaxDiam* (minimum and maximum diameter).

All 65 values must be specified for the characteristic.



[9-6] Freely definable interpolation points

9.19 Web break monitoring

The technology module provides two options for web break monitoring.

Via the *scPar.wWebBreakMode* parameter, three options for web break monitoring can be selected:

- ▶ [Web break monitoring via the evaluation of the dancer position](#)
- ▶ [Web break monitoring via the change in diameter](#)
 - Web break monitoring via the evaluation of the dancer position and change in diameter

Web break monitoring via the evaluation of the dancer position

As an alternative to the web break monitoring on the basis of the direction of change in the diameter, a web break can be assumed with the dancer position-controlled winder if the dancer reaches the lower limit position during operation. The monitoring function is activated with *xWebBreakMonit* = TRUE and *wWebBreakMode* parameter = 1.

Responses to a web break:

- Holding the current diameter.
- Setting the *xWebBreak* output = TRUE.

Web break monitoring via the change in diameter

In the case of a web break, the diameter calculated develops in the opposite direction of the winding direction (unwinding or rewinding), since the actual circumferential speed no longer corresponds to the line speed signal.

The monitoring function is activated with the *xWebBreakMonit* input = TRUE, making a diameter change opposed to the winding direction only permissible within the window set in the *lrWebBreakWindow* parameter.

Rewinding or unwinding is automatically detected via the sign of the line velocity and the winding direction set via the *xWindingDirection* input.

Web break monitoring may only be activated when the calculated diameter corresponds to the real diameter.

If web break monitoring is activated (*xWebBreakMonit* = TRUE), a diameter change contrary to the winding direction specified via the *xUnwind* output.

After loading a starting diameter which deviates considerably from the real diameter contrary to the winding direction, this may lead to an unintentional triggering of the monitoring function. If, for instance, a starting diameter of 50% is loaded for a rewinder and the real diameter only is 45%, then a change of the diameter value to the real 45% is prevented when web break monitoring is activated.

Parameters to be set

Identifier	Data type	Description
wWebBreakMode	WORD	Mode for web break monitoring Initial value: 1
	0	Web break monitoring via diameter calculation and dancer position
	1	Web break monitoring only via the dancer position
	2	Web break monitoring only via diameter calculation
lrWebBreakWindow	LREAL	Web break window The current diameter is compared to the previous diameter across the web break window from the last calculation cycle. - Unit: x 100 % (1.0 = 100 %) - Initial value: 0.1 (10 %)

► [L_TW2P_scPar_WinderDancerCtrlState](#) (26)

9.20 Persistent variables

The technology module provides the possibility of persistently saving the parameters determined such as the winding diameter. For this purpose, the following setting has to be made in the »PLC Designer« for the technology module.

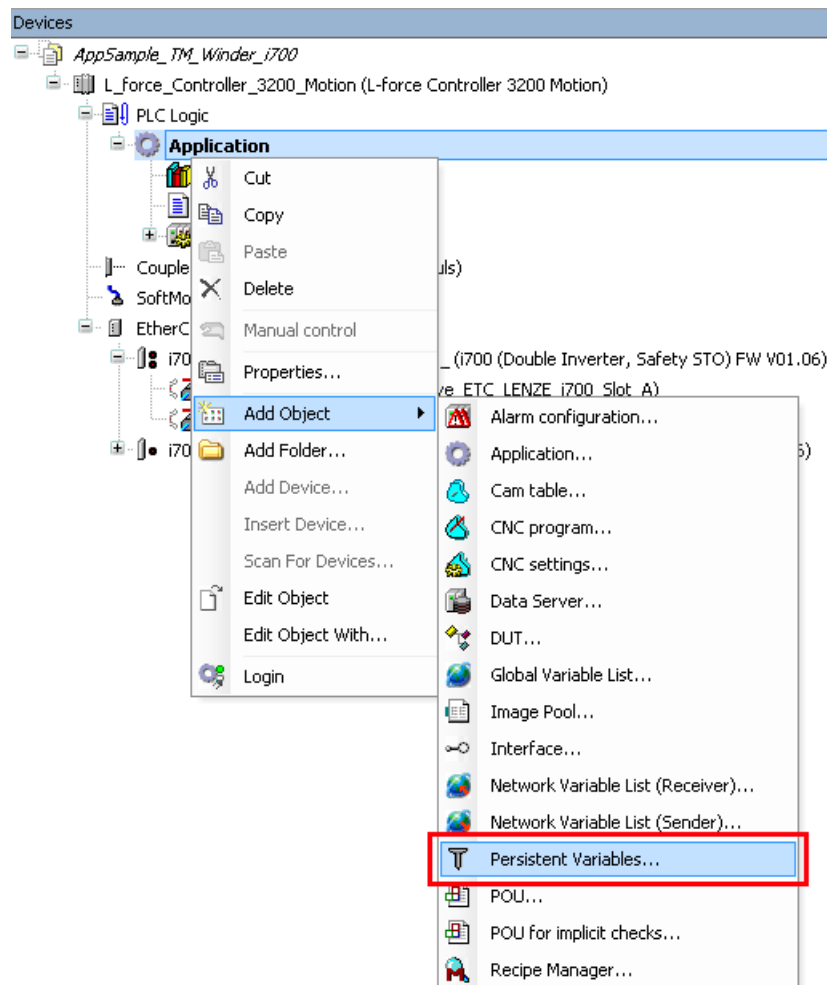


How to create persistent variables in the »PLC Designer«:

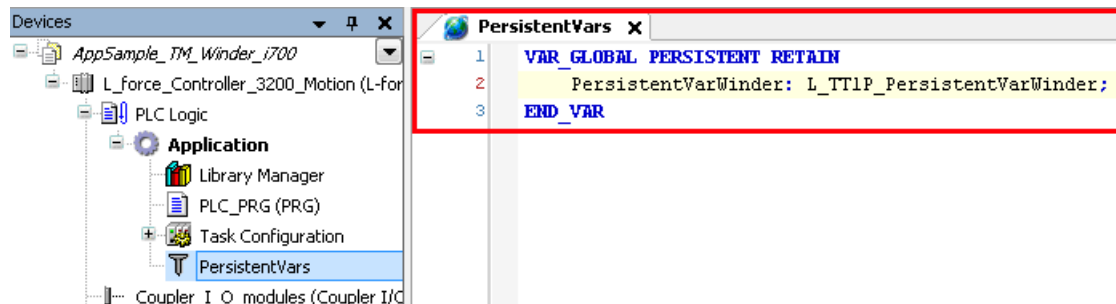
Note!

This procedure does not apply to the Application Template, since it already provides structures for persistent data of the machine modules.

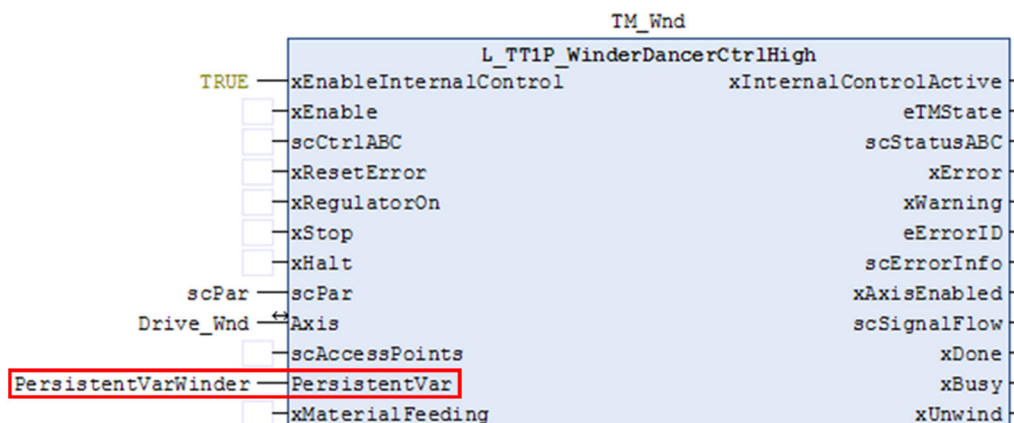
1. Add the global list for managing persistent variables in the context menu to **Application** with the **Add object → Persistent variables...** command.



2. Instance the reference of the persistent variables "L_TT1P_PersistentVarWinder" in the global structure of the persistent variables.



3. Interconnect the instance of the persistent variables with the *PersistentVar* input.



L_TW2P_PersistentVarWinder

Content of the persistent variables within the L_TW2P_PersistentVarWinder structure:

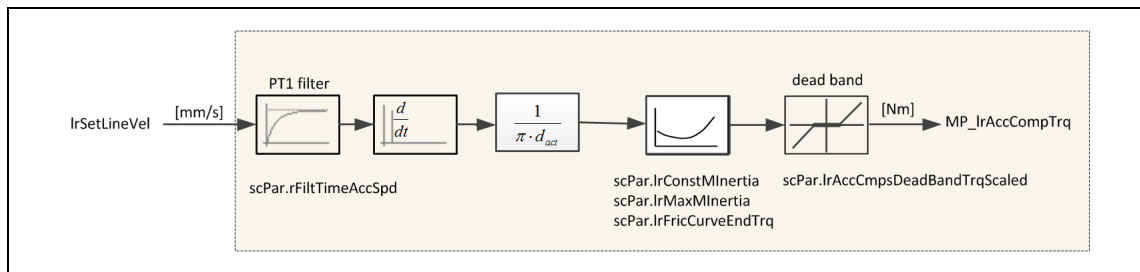
Identifier	Data type	Description
IrDiam	LREAL	The diameter calculated • Unit: mm
IrMaterialCounter	LREAL	Counter content of the material • Unit: mm
liRevCounter	LINT	Number of revolutions counted for the material length counter • Unit: rev
IrRevCounterResidual	LREAL	The fractional part of an incomplete revolution for the material length counter • Unit: rev
xUpperPosTeached	BOOL	Referenced upper dancer limit position TRUE The upper dancer limit position has already been referenced.
xLowerPosTeached	BOOL	Referenced upper dancer limit position TRUE The lower dancer limit position has already been referenced.
xDancerUpperLimit	LREAL	The analog sensor value from the <i>IrActDancerPosIn</i> input for the upper dancer limit position.
xDancerLowerLimit	LREAL	The analog sensor value from the <i>IrActDancerPosIn</i> input for the lower dancer limit position.

9.21 Acceleration compensation

The acceleration in the line velocity setpoint is a disturbance in the winding process. The torque consumed for acceleration is lacking for the tensile force.

Therefore, the acceleration torque has to be calculated and pilot-controlled as an additional torque.

Calculation of the acceleration torque		
$M = 2 \cdot \pi \cdot \left(\frac{\partial n}{\partial t} \right) \cdot (J_{const} + J_{var})$ with $J_{var} = (J_{max} - J_{const}) \cdot \left(\frac{d_{act}^4 - d_{min}^4}{d_{max}^4 - d_{min}^4} \right) \cdot B$		
Symbol	Description	Dimension unit
M	Acceleration torque	Nm
∂n	(Delta) motor speed	revs/s
∂t	(Delta) time	s
J_{const}	Constant moment of inertia	kgm ²
J_{var}	Variable (diameter-dependent) moment of inertia	
J_{max}	Maximum moment of inertia	
d_{act}	Current diameter	mm
d_{min}	Minimum diameter (sleeve diameter)	
d_{max}	Maximum diameter	
B	Material width	mm



The line speed signal is differentiated for generating an acceleration.

In practice, a line speed signal that is not ideal with a constant characteristic can be expected. Depending on the resolution and stability of this signal, the signal may have to be smoothed via the PT1 filter. The filter time is defined via the *rFiltTimeAccSpd* parameter.

The dead band function is applied for the purpose of noise suppression in the compensation torque calculated. In this way it is avoided that small acceleration changes only resulting from the quantisation or the noise of the line speed signal lead to a compensation torque. The *lrAccCmpsDeadBandTrqScaled* parameter defines up to which value the dead band takes effect.

The acceleration compensation is enabled when the *xAccCmp* input = TRUE.

Different material widths or material densities can be taken into consideration as a percentage via the *lrMInertiaAdapt* input.

The *lrAccCmpsGainAcc* and *lrAccCmpsGainDec* parameters evaluate the compensation torque calculated as a function of the effective direction. In order to avoid slack material, compensation is to be stronger during acceleration processes than during deceleration processes.

Different material widths or material densities can be taken into consideration as a percentage via the *lrMInertiaAdapt* input.

Parameters to be set

Identifier	Data type	Description
rFiltTimeAccSpd	REAL	PT1 filter time for the winder shaft speed for acceleration compensation - Unit: s - Initial value: 0.005
lrAccCmpsDeadBandTrqScaled	LREAL	Lagging range (Dead band) for the current acceleration torque - Unit: x 100 % (1.00 = 100 %); reference variable : rated motor torque [Nm] - Initial value: 0.1 (10 %)
lrAccCmpsGainAcc	LREAL	Gain factor for the acceleration torque in positive direction - Unit: x 100 % (1.00 = 100 %) - Value range: 0 ... 2 (0 ... 200%) - Initial value: 1.05 (105 %)
lrAccCmpsGainDec	LREAL	Gain factor for the acceleration torque in negative direction - Unit: x 100 % (1.00 = 100 %) - Value range: 0 ... 2 (0 ... 200%) - Initial value: 0.95 (95 %)
lrConstMIInertia	LREAL	Constant moment of inertia at the winder shaft - Unit: kgcm² - Initial value: 1
lrMaxMIInertia	LREAL	Maximally permissible moment of inertia at the winder shaft - Unit: kgcm² - Initial value: 10

► [L_TW2P_scPar_WinderDancerCtrlState](#) (26)

9.22 Identification of the moments of inertia

A suitable moment of inertia is required if the acceleration compensation or the adaption of the speed controller gain is to be used.

For this purpose, the constant *scPar.IrConstMInertia* and the maximum moment of inertia *scPar.MaxMInertia* must be appropriately parameterised. During operation, the current moment of inertia is calculated as a function of the current diameter is automatically calculated and output at *IrSetMInertiaOut*.

The constant and maximum moment of inertia can be determined via manual calculation or by means of identification.

The maximum moment of inertia corresponds to the reel with the maximum moment of inertia, for which the winder is designed. In order to adapt the reel actually used, which can differ with regard to the material density, the maximum diameter and / or the material width, evaluation factors are provided for the acceleration compensation and speed controller gain in each case.

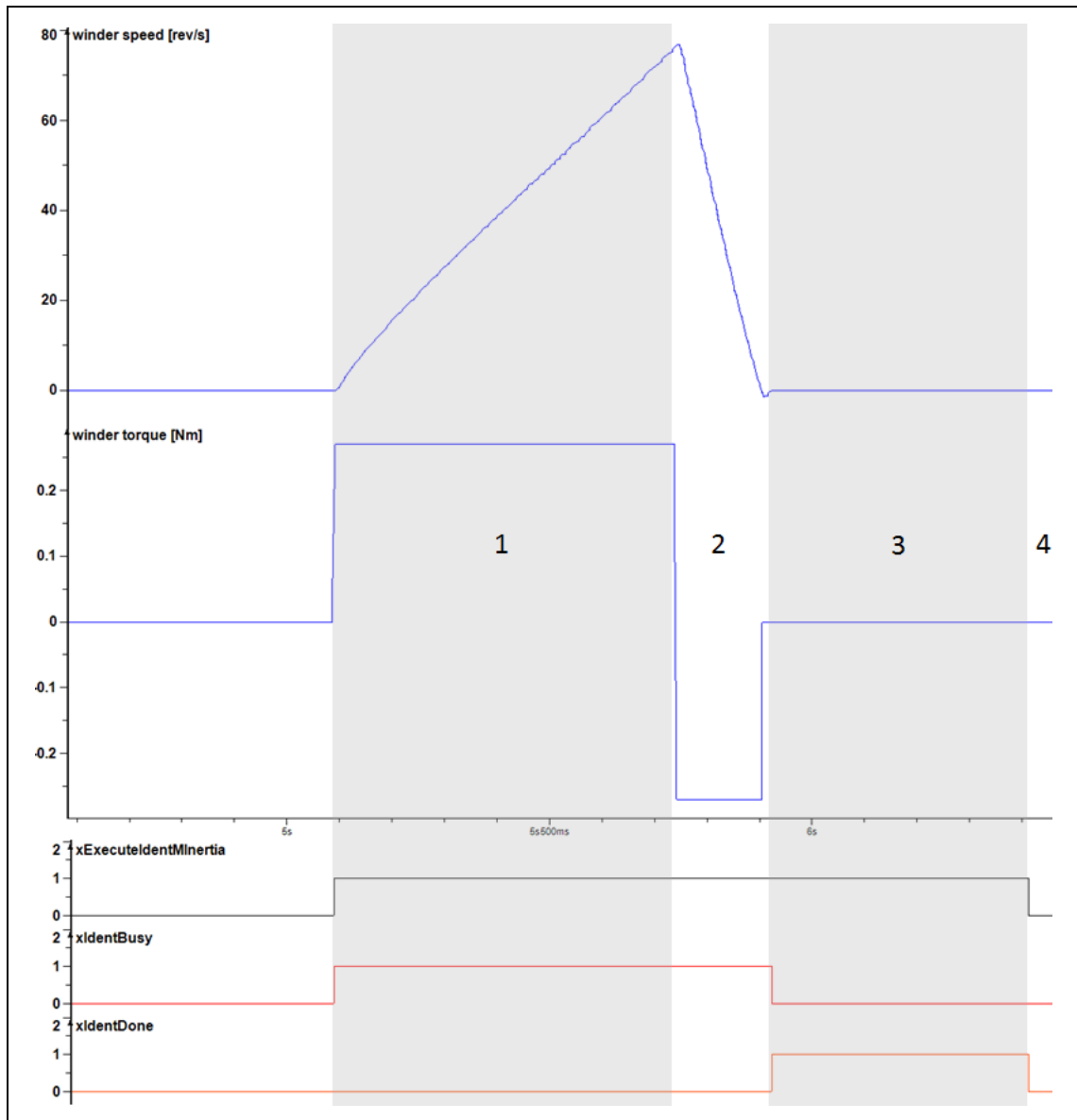
Identification of the constant moment of inertia

The winder shaft is empty (no material).

The *IrIdentMInertiaMaxSpdScaled* parameter serves to define the maximum motor speed in [x 100 %] regarding the maximally reachable winder speed *IrWndSpdRef*. Typically, speeds between 50 ... 60 % are sufficient.

The *IrIdentMInertiaMaxTrqScaled* parameter serves to define the acceleration torque. This value always has to be higher than the maximally occurring friction – recommendation: 25 %.

The mass moment of inertia is detected with a rising edge (FALSE→TRUE) at the *xExecutIdentMInertia* input. The *IrIdentMInertia* output displays the detected moment of inertia.



No	xExecutIdentMI inertia	xIdentBusy	xIdentDone	eTMState	INFO
1	FALSE $\nearrow$ TRUE	FALSE $\nearrow$ TRUE	FALSE	STANDBY $\hookrightarrow$ IDENTMINERTIA	The axis is controlled in the CST torque mode. With the positive torque, which is defined by the parameter setting for the identification process, the axis is accelerated up to the identified speed defined.
2	TRUE	TRUE	FALSE	IDENTMINERTIA	The speed that is determined via the parameter setting for the identification process has been reached. The torque is commissioned in negative direction. The axis is now decelerated.
3	TRUE $\searrow$ FALSE	TRUE $\searrow$ FALSE	FALSE $\nearrow$ TRUE	IDENTMINERTIA $\hookrightarrow$ STANDBY	The axis reaches standstill and the torque is switched off. The histories recorded are evaluated and the mass inertia is determined.
4	TRUE $\searrow$ FALSE	FALSE	TRUE $\searrow$ FALSE	STANDBY	If the assignment for identification is reset, the "DONE" status is also reset.

The control stability of the drive and the actual value resolution of speed and torque are directly included in the calculation result of the identification process. Filtering of the current values often improves the result of the identification process:

- Filter time for the current torque: $rFiltTimeIdentMI inertiaTrq$
- Filter time for the current winder speed: $rFiltTimeIdentMI inertiaSp$

Identification of the maximum moment of inertia

The winder is loaded with the maximum possible reel (maximum diameter and maximum width).

The maximum motor speed *IrIdentMInertiaMaxSpdScaled* has to be parameterised so that the maximally permissible circumferential velocity of the winder will not be exceeded (e.g. *IrIdentMInertiaMaxSpdScaled* = 10 % at $d_{\max}/d_{\min} = 10$) – recommended: 25 %.

The mass moment of inertia is detected with a rising edge (FALSE→TRUE) at the *xExecutIdentMInertia* input.

Termination of the identification

The identification is completed as soon as the motor has reached standstill again, no errors have been reported and *xDone* has been set to TRUE.

The detected inertia of the winder shaft (J_{Winder}) is displayed at the *IrIdentMInertia* output and has to be checked for plausibility.

The identification of both types of inertia should be executed several times. In doing so, the filter time for the *rFiltTimeIdentMInertiaSpd* speed can be altered.

A significant non-linear friction in the system adversely affects the calculation of the moment of inertia in the technology module.

Parameters to be set

Identifier	Data type	Description
<i>rFiltTimeIdentMInertiaSpd</i>	REAL	PT1 filter time for the speed at the winder shaft during the identification of the moment of inertia - Unit: s - Initial value: 0.01
<i>rFiltTimeIdentMInertiaTrq</i>	REAL	PT1 filter time for the torque at the winder shaft during the identification of the moment of inertia - Unit: s - Initial value: 0.005
<i>IrIdentMInertiaMaxSpdScaled</i>	LREAL	Maximum speed of the winder shaft during the identification of the moment of inertia - Unit: x 100 % (1.0 = 100 % = <i>IrWndSpdRef</i>) - Value range: 0 ... 1 (0 ... 100 %) - Initial value: 0.2 (20 %)
<i>IrIdentMInertiaMaxTrqScaled</i>	LREAL	Maximum torque of the winder shaft during the identification of the moment of inertia - Unit: x 100 % (1.0 = 100 %) - reference variable: rated motor torque - Value range: 0 ... 1 (0 ... 100 %) - Initial value: 0.2 (20 %)

► [L_TW2P_scPar_WinderDancerCtrlState](#) (26)

Calculation of the maximum moment of inertia

If no reel is available, the maximum moment of inertia can be calculated as follows:

Calculation of the maximum moment of inertia		
The <u>density</u> of the winding material is known:		
$J_{\text{MaxWinder}} = i^2 \times J_{\text{Motor}} + (\pi / 32 \times 10^8) \times B \times \rho (d_{\text{max}}^4 - d_{\text{min}}^4)$		
The <u>mass</u> of the winding material is known:		
$J_{\text{MaxWinder}} = i^2 \times J_{\text{Motor}} + (m \times d_{\text{max}}^4 / 800)$		
Symbol	Description	Dimension unit
$J_{\text{MaxWinder}}$	Maximum moment of inertia of the winder shaft	kgcm ²
J_{Motor}	Moment of inertia of the motor shaft	kgcm ²
i	Gearbox factor	
B	Material width	mm
ρ	Material density	kg/dm ³
d_{max}	Maximum diameter	mm
d_{min}	Minimum diameter (sleeve diameter)	mm
m	Mass	kg

The constant moment of inertia is basically determined by the motor. Its moment of inertia is specified in the catalogue.

The values calculated or identified have to be transferred to the respective parameter:

- *scPar.IrConstMI*
- *scPar.IrMaxMI*

In order to store the values safe against mains failure in the device, the parameter set must be saved!

9.23 Adaptation of the speed controller gain

Conditions

The winder axis must be enabled (input *xRegulatorOn* = TRUE).

Operating principle

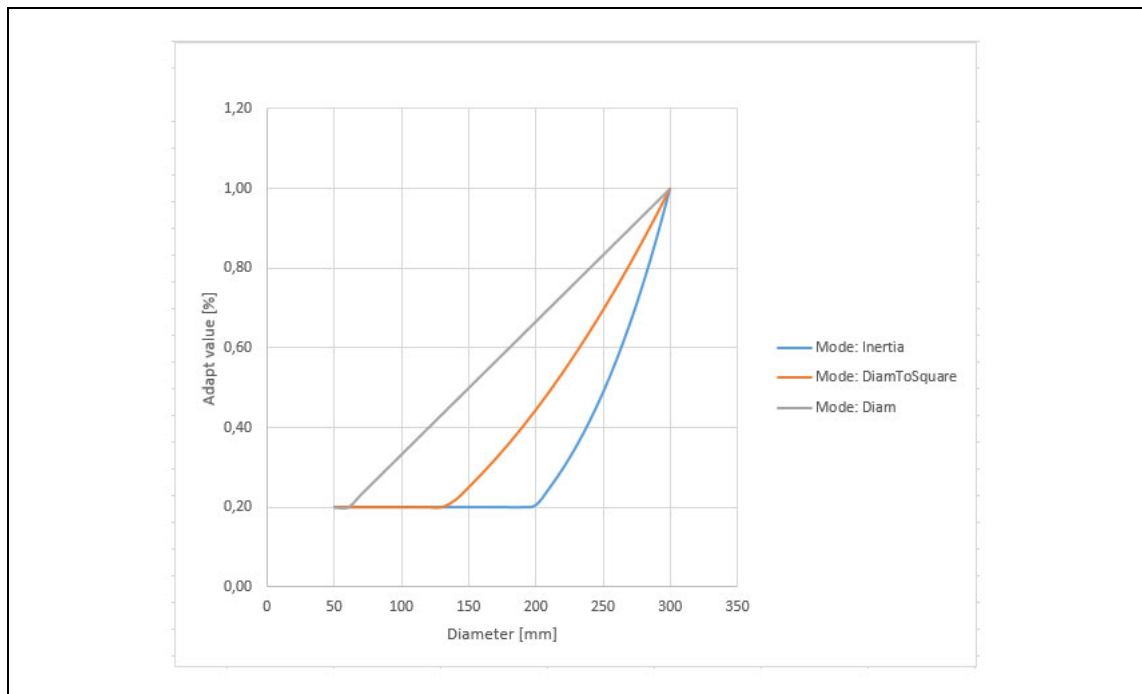
Looking at the motor and the reel as a rigid one-mass system, the optimum gain of the speed controller is directly proportional to the moment of inertia.

Since the moment of inertia in most cases significantly changes during the winding process, it may be necessary to control the gain of the speed controller along with the moment of inertia in order to provide for a stable control response.

- According to the mode selected in *scPar.eAdaptSpdCtrlGainMode*, an adaptation factor is generated in the TM - either as a direct function of the current diameter or of the moment of inertia.
- The gain set for the speed controller is multiplied with the factor *scPar.lfAdaptSpdCtrlGainFactor* (*scPar.lfAdaptSpdCtrlLowLimit* 0 ... 1). Therefore, the gain set only takes effect with the maximum diameter or the maximum moment of inertia.

The *eAdaptSpdCtrlGainMode* parameter provides the following methods for generating the adaptation factor:

- *DiamToSquare*: The adaptation factor corresponds to d_{act}^2 / d_{max}^2
- *Diam*: The adaptation factor corresponds to d_{act} / d_{max}
- *Inertia*: The adaptation factor corresponds to J_{act} / J_{max}



[9-7] Adaptation of the speed controller gain with an exemplary lower limitation to *scPar.lfAdaptSpdCtrlLowLimit* = 0.2

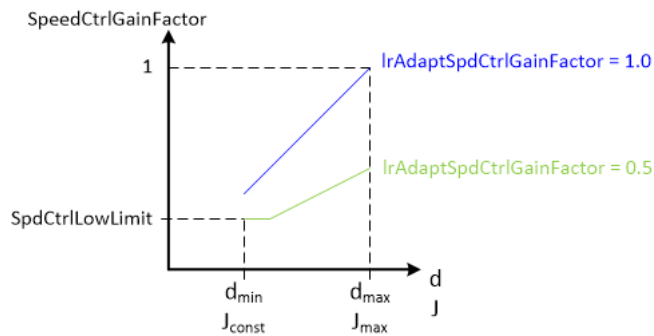
In the case of very rigid mechanics and winding materials as well as minimised dead times in the control loop, the adaptation method *scPar.eAdaptSpdCtrlGainMode* = 2 (*Inertia*) can be applied.

In this case, the moments of inertia in the parameters *scPar.lfConstMinertia* and *scPar.lfMaxMinertia* must be entered in a plausible fashion.

In many applications it suffices to use the default adaptation method *scPar.eAdaptSpdCtrlGainMode* = 0 (*DiamToSquare*). In this case, the moment of inertia has no impact on the adaptation.

Activating/Deactivating adaption of the speed controller gain

The adaptation of the speed controller gain is active in all TM states if the *scPar.xAdaptSpdCtrlGain* parameter is set to TRUE.



Parameters to be set

Identifier	Data type	Description
eAdaptSpdCtrlGainMode	ENUM	Mode selection for adaptation of the speed controller gain. • Initial value: 2
	0	DiamToSquare; $VP = f(d^2)$
	1	Diam; $VP = f(d)$
	2	Inertia; $VP = f(J)$
IrAdaptSpdCtrlLowLimit	LREAL	Lower limitation of speed controller gain in the drive. The <i>IrSetSpdCtrlGainAdaptOut</i> adaptation value must not be smaller than the <i>IrAdaptSpdCtrlLowLimit</i> value. • Value range: 0 to 1 • Initial value: 0
alrSpdCtrlGainAdaptX	ARRAY [1...9] OF LREAL	Characteristic function for the speed control gain The X axis corresponds to the adaptation factor currently calculated for the adaptation of the speed controller gain. • Unit: x 100 % (1 = 100 % = <i>IrMaxMinertia</i> parameter) • Initial values: [0.0, 0.2, 0.4, 0.6, 0.8, 1.0, 1.2, 1.4, 1.6]
alrSpdCtrlGainAdaptY	ARRAY [1...9] OF LREAL	Characteristic function for the speed control gain The Y axis corresponds to the gain factor of the speed controller. • Unit: x 100 % (1 = 100 %) • Initial values: [0.0, 0.2, 0.4, 0.6, 0.8, 1.0, 1.2, 1.4, 1.6]
xAdaptSpdCtrlGain	BOOL	Switch adaptation of the speed controller gain on/off. • Initial value: FALSE
	TRUE	Enable adaptation of the speed controller gain.
	FALSE	Adaptation of the speed controller gain inactive.

► [L_TW2P_scPar_WinderDancerCtrlState](#) (26)

9.24 Limitation of the master line speed

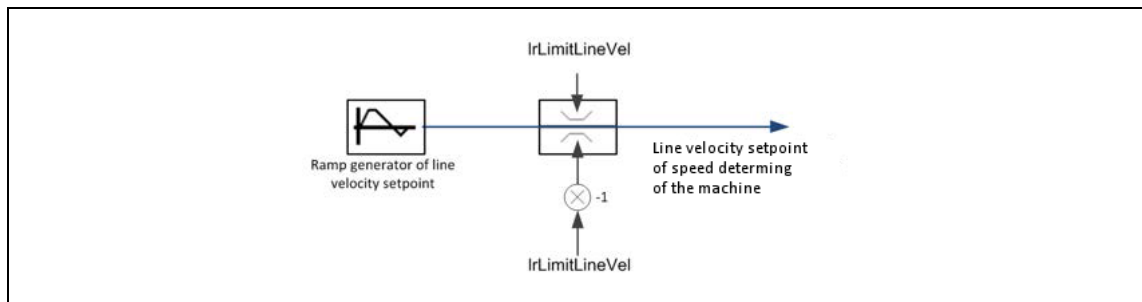
In order to reduce the drive performance for small winder diameters or not to exceed permissible gearbox input speeds, it may be necessary to limit the line speed of the system. The limitation is calculated in the winder TM.

The maximum speed of the winder shaft (drive-side) is defined via parameter *scPar.IrMaxWndSpd*. The limitation function is enabled directly when this parameter is entered.

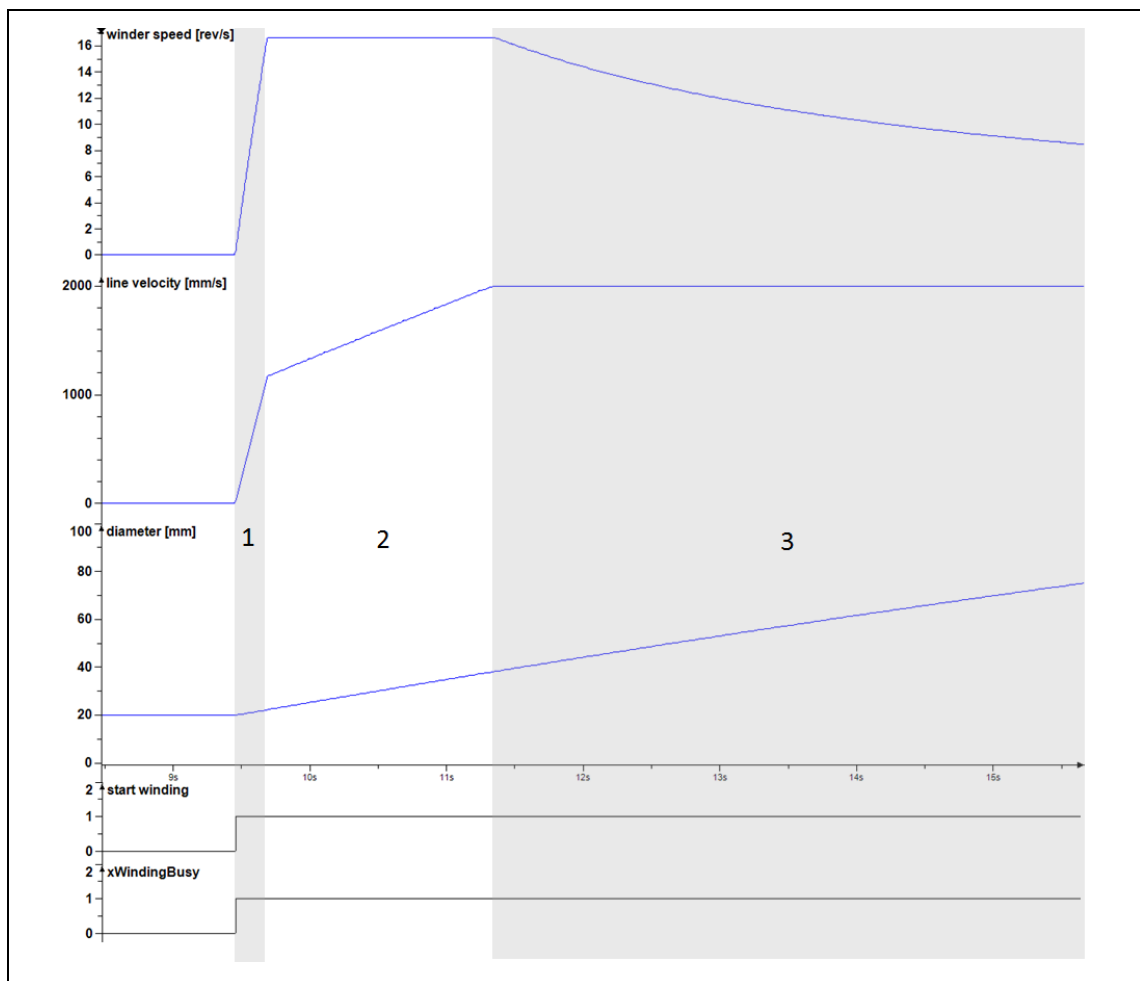
At the *IrLimitLineVel* output, the maximum permissible line speed is output in [mm/s]. Exceeding the *IrLimitLineVel* line speed inevitably means maximum winder shaft speed *scPar.IrMaxWndSpd* will also be exceeded.

Example

The following graphic illustrates the necessary limitation of the line speed between the reference line speed (V_{ref}) to the maximum line speed (V_{max}) for a rewinder which starts at the minimum diameter (D_{min}) so as not to exceed the permissible speed of the *scPar.IrMaxWndSpd* winder shaft.



[9-8] Signal flow for calculation of the line speed setpoint limitation



[9-9] Example for the limitation of the line speed

No	xWindingBusy	Limitation is active	Line speed has reached the maximum value	INFO
1	TRUE	FALSE	FALSE	The TM is in the winding mode. The line speed is increased.
2	TRUE	TRUE	FALSE	The winding drive reaches the maximum permissible speed. The line limitation from the winder limits the line speed.
3	TRUE	FALSE	TRUE	The line has reached the maximum speed and is traversed constantly. The diameter increases further. The speed of the winder is reduced.

Parameters to be set

Identifier	Data type	Description
lrMaxWndSpd	LREAL	Definition of the maximum winder shaft speed (at the output end). Unit: rev / s Initial value: 100

► [L_TW2P_scPar_WinderDancerCtrlState](#) (26)

9.25 Termination of the winding process

For terminating the winding process ("DANCERCTRL" state), there are two options:

- A. Set *xDancerCtrl*input = FALSE and *xSyncLineVel* input = TRUE.

The technology module changes from the winding process to the synchronous travel. Here, the winder circumferential speed is synchronised to the line velocity (*lrSetLineVel* input).

State change: "DANCERCTRL" → "SYNCLINEVEL"

- B. Set *xDancerCtrl* input = FALSE and *xSyncLineVel* input = FALSE.

The winding process is terminated as a function of the setting of the *eDancerCtrlStopMode* parameter.

- *eDancerCtrlStopMode* = 0: Hold
The axis is brought to a standstill via the deceleration (*lrHaltDec*) and the jerk (*lrJerk*).
- *eDancerCtrlStopMode* = 1: Move ABS
With the speed (*lrVel*), acceleration (*lrAcc*), deceleration (*lrDec*) and the jerk (*lrJerk*), the axis is traversed to the target position (*lrPos_Dist*).
The target position is specified at the input of the technology module via the *scCtrlBasicMotion.lrPos_Dist* structure.
- *eDancerCtrlStopMode* = 2: Move Rel
With the speed (*lrVel*), acceleration (*lrAcc*), deceleration (*lrDec*), and the jerk (*lrJerk*), the axis is brought to a standstill after the distance travelled (*lrPos_Dist*).
The path is specified via the *scCtrlBasicMotion.lrPos_Dist* structure at the input of the technology module.

State change: "DANCERCTRL" → "STOP"

Parameters to be set

Identifier	Data type	Description
eDancerCtrlStopMode L_TW2P_DancerCtrlStopMode		Mode in which the winding process ("DANCERCTRL" state) is stopped. • Initial value: 0
	0	Holding The axis is stopped via the deceleration (<i>IrHaltDec</i>) and the jerk (<i>IrJerk</i>).
	1	Move ABS Absolute travel/positioning: The axis travels to the target position (<i>IrPos_Dist</i>) with the velocity (<i>IrVel</i>), acceleration (<i>IrAcc</i>), deceleration (<i>IrDec</i>) and the jerk (<i>IrJerk</i>).
	2	Move Rel Relative travel/positioning: The axis is brought to a standstill with the velocity (<i>IrVel</i>), acceleration (<i>IrAcc</i>), deceleration (<i>IrDec</i>) and the jerk (<i>IrJerk</i>) after the distance travelled (<i>IrPos_Dist</i>).
IrVel	LREAL	Speed for movements that are commissioned via the scCtrlBasicMotion structure. • Unit: unit/s • Initial value: 3600
IrAcc	LREAL	Acceleration for movements that are commissioned via the scCtrlBasicMotion structure. Selection of the value for the maximum acceleration speed variation. • Unit: unit/s ² • Initial value: 36000
IrDec	LREAL	Deceleration for movements that are commissioned via the scCtrlBasicMotion structure. Selection of the value for the maximum deceleration speed variation until standstill. • Unit: unit/s ² • Initial value: 100
IrJerk	LREAL	Jerk for movements that are commissioned via the scCtrlBasicMotion structure. • Einheit: Unit/s ³ • Initial value: 0
IrPos_Dist	LREAL	Absolute positioning Absolute target position relating to the zero position of the axis. • Unit: units • Initial value: 0
		Relative positioning Distance to be traversed relating to the setpoint position at the start time of the command. • Unit: units • Initial value: 0

► [L_TW2P_scPar_WinderDancerCtrlState](#) (26)

A

Aborting a positioning process [53](#)
Absolute positioning [45](#)
Acceleration compensation [96](#)
Access points [36](#)
Adaptation of the speed controller gain [102](#)
Application notes [8](#)

C

Calculate max. moment of inertia [101](#)
Competing a positioning process [53](#)
Continuous travel with a defined velocity [50](#)
Controlled start of the axes [14](#)
Conventions used [7](#)
Correcting the dancer position [67](#)
Correction of the dancer position [67](#)

D

Dancer position (scaling) [80](#)
Dancer position monitoring [84](#)
Defining a start diameter [65](#)
Defining the diameter [65](#)
Defining the winding direction (rewinding/unwinding) [57](#)
Deviation within the reduced sensitivity [87](#)
Diameter calculation [62](#)
Diameter calculation with correction of the dancer position [67](#)
Diameter sensor signal [65](#)
Document history [6](#)

E

E-mail to Lenze [110](#)

F

Feedback to Lenze [110](#)
Functional description of "Winder Dancer-controlled (V2)" [11](#)

H

Halt, [44](#)
Holding the diameter [64](#)
Homing [40](#)

I

Identification of the diameter [69](#)
Identification of the moments of inertia [98](#)
inputs [19](#)
Inputs and outputs [18](#)

L

L_TW2P_scPar_WinderDancerCtrlState [26](#)
L_TW2P_scPar_WinderDancerCtrlState parameter structure [26](#)
L_TW2P_WinderDancerCtrlState [16](#)
Layout of the safety instructions [8](#)
Limitation of the master line speed [104](#)

Limiting master line speed [104](#)

M

Manual jog (jogging) [38](#)
Master value source for diameter calculation [60](#)
Material feed to the winder [58](#)
Material length counter [73](#)
Material length counting (sources) [73](#)
Monitoring of the dancer position [84](#)

N

Notes on how to operate the technology module [13](#)

O

Operating mode [13](#)
outputs [23](#)
Override for speed [55](#)

P

Persistent variables [94](#)
PI controller for the dancer position control [85](#)

Q

QSP application [43](#)

R

Relative positioning [47](#)

S

Safety instructions [9](#)
Scaling of the dancer position [80](#)
Signal flow diagram [33](#)
Signal flow of the "winder dancer-controlled" technology module [33](#)
Signal from the diameter sensor [65](#)
Sources for the material length counting [73](#)
Speed controller gain (adaptation) [102](#)
Speed feedforward control [61](#)
Speed override [55](#)
Start of the axes [14](#)
State machine [32](#)
States [32](#)
Structure of the access points
L_TW2P_scAP_WinderDancerCtrlState [36](#)
Structure of the signal flow
L_TW2P_scSF_WinderDancerCtrlState [34](#)
Synchronisation to the line velocity [79](#)
System deviation within the range of reduced sensitivity [87](#)

T

Target group [5](#)
Teaching function for dancer limit positions [82](#)
Technology module functions (overview) [12](#)
Tension control via characteristic function (Base version) [88](#)

Index

Terminating the winding process [106](#)

Termination of the winding process [106](#)

V

Variable identifiers [7](#)

W

Web break monitoring [92](#)

Winding direction (automatic detection) [57](#)



Your opinion is important to us

We have compiled these instructions to the best of our knowledge with the objective of supporting you optimally when you handle our product.

Perhaps we have not succeeded in achieving this objective in every respect. If you have suggestions for improvement, please e-mail us to:

feedback-docu@lenze.com

Thank you very much for your support.

Your Lenze documentation team

