FAST Application Software

Technology module

Winder Dancer-controlled___________________

Reference Manual EN

13531754

Lenze

Contents

1 About this documentation

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 Controller (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« (parameter setting, configuration, diagnostics) • »VisiWinNET® Smart« (visualisation) • »Backup & Restore« (data backup, recovery, update)

More technical documentation for Lenze components

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

	Planning / 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
	Parameter setting / 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 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.

1.1 Document history

Version			Description
4.3	05/2017	TD17	• Content structure has been changed. • General revisions • Figure Signal flow of the technology module (📖 29) corrected. New: • "MaterialCounterAxis" input (AXIS_REF) • Sources for the material length counting (📖 44)
4.2	11/2016	TD17	• General revisions • Access points L_TT1P_scAP_WinderDancerCtrl[Base/State/High] (📖 32) added.
4.1	04/2016	TD17	• General revisions • Figure State machine (📖 27) corrected. • Figure Signal flow of the technology module (📖 29) corrected. • Access points L_TT1P_scAP_WinderDancerCtrl[Base/State/High] (📖 32) added.

4.0	11/2015	TD17	• General revisions
			• New:
			  ▸ System deviation within the reduced sensitivity (📖 65)
			  ▸ Termination of the winding process (📖 66)
			• Content structure has been changed.

3.0	05/2015	TD17	• General revisions • New: Material length counter (📖 43)
2.0	01/2015	TD17	• General editorial revision • Modularisation of the contents for the »PLC Designer« online help
1.0	04/2014	TD00	First edition

1.2 Conventions used

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

Type of information	Highlighting	Examples/notes
Spelling of numbers		
Decimal separator	Point	The decimal point is always used. For example: 1234.56
Text		
Program name	» «	»PLC Designer« ...
Variable names	*italics*	By setting *bEnable* 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;`
		`...`

Icons		
Page reference	(🕮 6)	Reference to further information: Page number in PDF file.

Variable names

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.

1.3 Definition of the notes used

The following signal words and symbols are used in this documentation to indicate dangers and important information:

Safety instructions

Layout of the safety instructions:

> **Pictograph and signal word!**
>
> (characterise the type and severity of danger)
>
> **Note**
>
> (describes the danger and gives information about how to prevent dangerous situations)

Pictograph	Signal word	Meaning
	Danger!	**Danger of personal injury through dangerous electrical voltage** Reference to an imminent danger that may result in death or serious personal injury if the corresponding measures are not taken.
	Danger!	**Danger of personal injury through a general source of danger** Reference to an imminent danger that may result in death or serious personal injury if the corresponding measures are not taken.
STOP	**Stop!**	**Danger of property damage** Reference to a possible danger that may result in property damage if the corresponding measures are not taken.

Application notes

Pictograph	Signal word	Meaning
	Note!	Important note to ensure trouble-free operation
	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.

Stop!

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

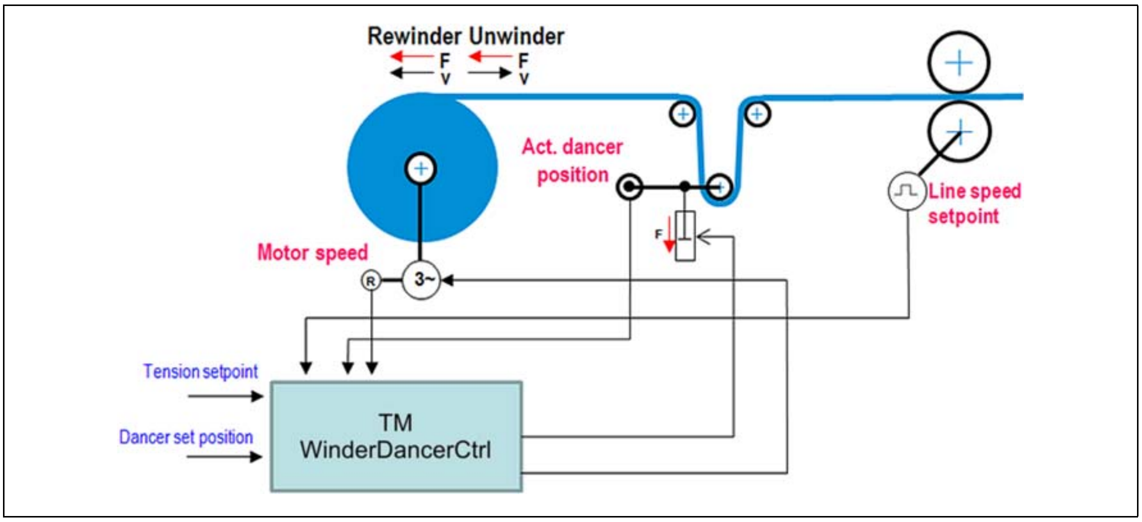
- Always switch off the voltage supply before working on the components of the automation system.
- Do not touch electronic components and contacts unless ESD measures were taken beforehand.
- If required, provide systems with installed inverters with additional monitoring and protective devices according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

3 Functional description of "Winder Dancer-controlled"

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

This technology module can be used to design a dancer-position-controlled winding drive.



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

With dancer position control, the drive is operated in the speed control mode. For feedforward control, the line speed signal multiplied by the reciprocal value of the diameter is used. The dancer position is detected and compared with the set position. If there is a deviation, the dancer position controller corrects the speed setpoint.

- The "Base" version provides dancer position control with diameter calculation. A PI controller can be used for the dancer position control.The dancer limit positions can be set via the L_TT1P_scPar_WinderDancerCtrl[Base/State/High] (📖 21) parameter structure or 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.
- The "State" version provides an extended functional range of the "Base" version. Here, three characteristics are available 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 "High" version 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.

3.1 Overview of the functions

In addition to the basic functions for operating the **L_MC1P_AxisBasicControl** function block, the **Stop function** and the **Holding function**, the technology module offers the following functionalities that are assigned to the "Base", "State", and "High" versions:

Functionality	Versions		
	Base	State	High
Defining the winding direction (winding/unwinding) (📖 34)	●	●	●
Automatic detection of the winding direction (📖 34)	●	●	●
Defining the material feed to the winder (📖 35)	●	●	●
Master value source for diameter calculation (📖 36)	●	●	●
Speed feedforward control (📖 37)	●	●	●
Calculation of the diameter (📖 38)	●	●	●
Holding the diameter (📖 39)	●	●	●
Defining the diameter / signal from the diameter sensor (📖 40)	●	●	●
Diameter calculation with correction of the dancer position (📖 41)	●	●	●
Material length counter (📖 43)	●	●	●
Sources for the material length counting (📖 43)	●	●	●
Manual jog (jogging) (📖 46)	●	●	●
Synchronisation to the line velocity (📖 48)	●	●	●
Trimming (📖 49)	●	●	●
Scaling of the dancer position (📖 50)	●	●	●
Teaching function for dancer limit positions (📖 51)	●	●	●
Monitoring of the dancer position (📖 52)	●	●	●
PI controller for the dancer position control (📖 53)	●	●	●
Tension control via characteristic function (Base version) (📖 54)	●	●	●
Web break monitoring (📖 55)	●	●	●
Persistent variables (📖 56)	●	●	●
Acceleration compensation (📖 58)		●	●
Tension control via characteristic function/winding characteristic (📖 60)		●	●
Identification of the moments of inertia (📖 62)			●
Adaptation of the speed controller gain (📖 64)			●
System deviation within the reduced sensitivity (📖 65)			●
Termination of the winding process (📖 66)			●

»PLC Designer« Online help

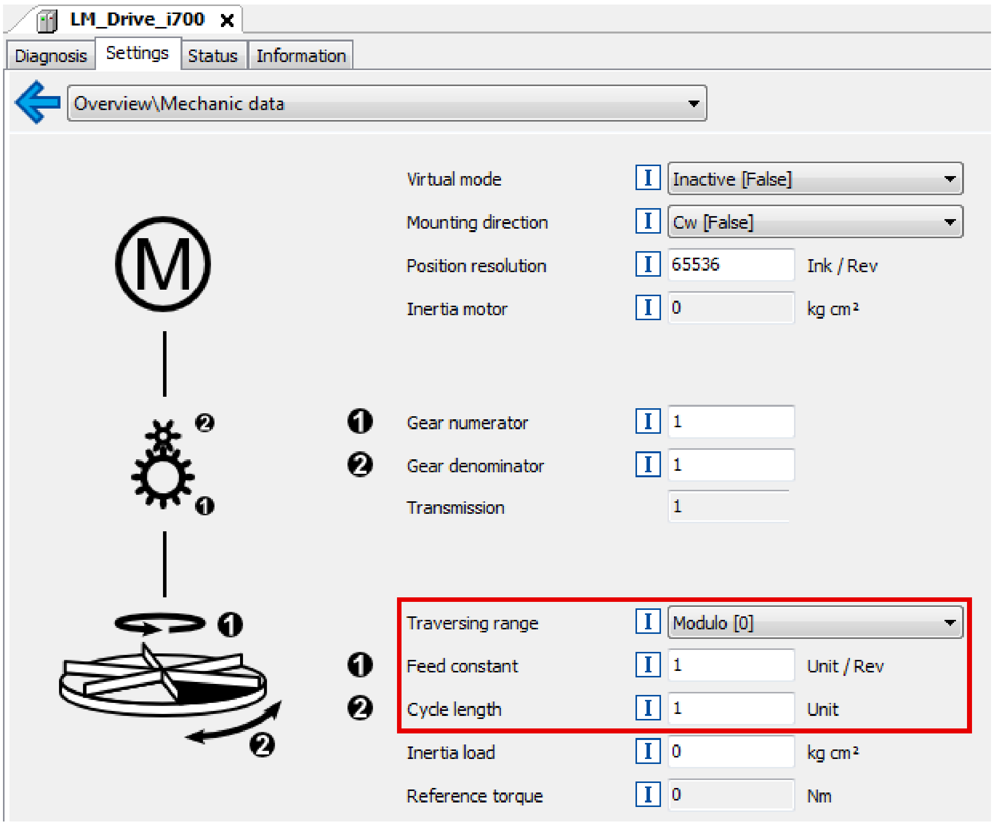
Here you will find detailed information on the **L_MC1P_AxisBasicControl** function block, the **stop function** and the **holding function**.

3.2 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« fand set the following parameters for each axis under the **Settings** tab:



- The feed of the winder axis is parameterised in the unit [revs/s].
- The line velocity is parameterised in the unit [mm/s].

Setting of the operating mode

The operating mode for the winder axis has to be set to "cyclically synchronous position" (csp) because the axis is led via the master position, master velocity and master torque value.

Controlled start of the axes

Motion commands that are set in the inhibited axis state (*xAxisEnabled* = FALSE) after enable (*xRegulatorOn* = TRUE) must be activated again by a FALSE↗TRUE edge.

In this way it is prevented that the drive starts in an uncontrolled manner after controller enable.

Example Manual jog (jogging) (46)**:**

1. In the inhibited axis state (*xAxisEnabled* = FALSE), *xJogPos* is set to TRUE.
   - *xRegulatorOn* = FALSE (axis is inhibited.)
     ==> "READY" state (*xAxisEnabled* = FALSE)
   - *xJogPos* = TRUE (manual jog is to be executed.)
2. Enable axis.
   - *xRegulatorOn* = TRUE
     ==> "READY" state (*xAxisEnabled* = TRUE)
3. Execute manual jog.
   - *xJogPos* = FALSE↗TRUE
     ==> "JOGPOS" state

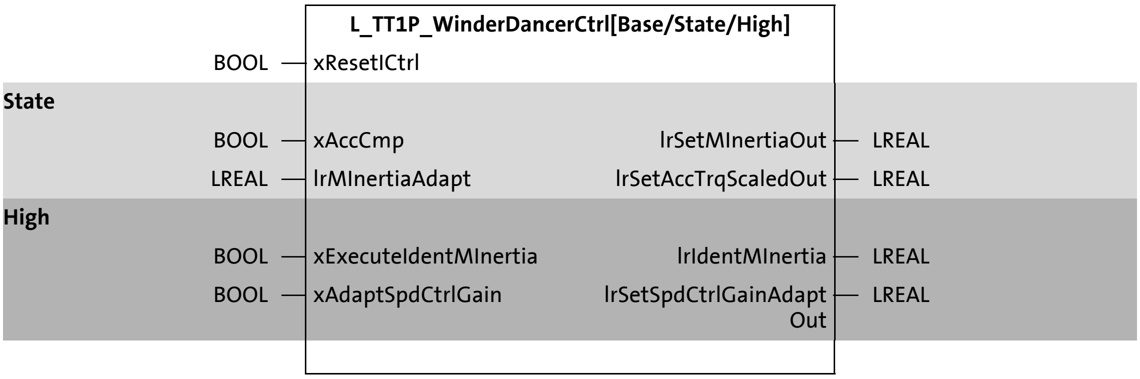
3.3 Function block L_TT1P_WinderDancerCtrl[Base/State/High]

The figure shows the relation of the inputs and outputs to the "Base", "State" and "High" versions. The additional inputs and outputs of the "State" and "High" versions are shaded.

L_TT1P_WinderDancerCtrl[Base/State/High]

Base

Input type	Input	Output	Output type
BOOL	xEnableInternalControl	xInternalControlActive	BOOL
BOOL	xEnable	eTMState	L_TT1P_States
scCtrl_ABC	scCtrlABC	scStatusABC	scStatus_ABC
BOOL	xResetError	xError	BOOL
BOOL	xRegulatorOn	xWarning	BOOL
BOOL	xStop	eErrorID	L_IE1P_Error
BOOL	xHalt	scErrorInfo	L_TT1P_scErrorInfo
L_TT1P_scPar_WinderDancerCtrl[Base/State/High]	scPar	xAxisEnabled	BOOL
AXIS_REF	Axis	scSignalFlow	L_TT1P_scSF_WinderDancerCtrl[Base/State/High]
L_TT1P_PersistentVarWinder	PersistentVar	xDone	BOOL
L_TT1P_scAP_WinderDancerCtrl[Base/State/High]	scAccessPoints	xBusy	BOOL
AXIS_REF	MaterialCounterAxis	xSynchronised	BOOL
BOOL	xMaterialFeeding	xAccDecSync	BOOL
BOOL	xWindingDirection	xUnwind	BOOL
BOOL	xLoadDiam	xWebBreak	BOOL
LREAL	lrSetDiam	xHoldDiamActive	BOOL
BOOL	xHoldDiam	xDiamMax	BOOL
BOOL	xDiamCalcReduced	xDiamMin	BOOL
BOOL	xSetMaterialCounter	lrSetDiamOut	LREAL
BOOL	xWebBreakMonit	lrSetDiamScaledOut	LREAL
BOOL	xJogLinePos	lrMaterialCounter	LREAL
BOOL	xJogLineNeg	lrSetLineVelScaledOut	LREAL
BOOL	xTrimLinePos	lrActTotalTrqScaled	LREAL
BOOL	xTrimLineNeg	lrWndSpdRef	LREAL
BOOL	xSyncLineVel	xDancerReachedSetPos	BOOL
BOOL	xTensCurve	xDancerMaxPos	BOOL
LREAL	lrSetTens	xDancerMinPos	BOOL
LREAL	lrSetLineVel	lrActDancerPosScaled	LREAL
LREAL	lrSetLineVelDiamCalc	lrSetDancerPosCtrlScaledOut	LREAL
BOOL	xTeachUpperPos	lrSetTensCurveOut	LREAL
BOOL	xTeachLowerPos		
BOOL	xDancerCtrl		
LREAL	lrActDancerPosIn		
LREAL	lrSetDancerPosScaled		
LREAL	lrDancerPosInfluenceScaled		



3.3.1 Inputs and outputs

Designator / **Data type**	**Description**	**Available in version**		
		Base	**State**	**High**
Axis AXIS_REF	Reference to the axis	●	●	●
PersistentVar L_TT1P_PersistentVar Winder	Reference to persistent variables In the reference, the following data are managed: • Diameter calculated • Dancer limit positions "learned" (see Teaching function for dancer limit positions (📖 51))	●	●	●

3.3.2 Inputs

Designator / Data type	Description		Available in version: Base	State	High
xEnableInternalControl BOOL	TRUE	In the visualisation, the internal control of the axis can be selected via the "Internal Control" axis.	●	●	●
xEnable BOOL	Execution of the function block TRUE: The function block is executed. FALSE: The function block is not executed.		●	●	●
scCtrlABC scCtrl_ABC	Input structure for the **L_MC1P_AxisBasicControl** function block • scCtrlABC can be used in "Ready" state. • If there is a request, the state changes to "Service". • The state change from "Service" back to "Ready" takes place if there are no more requests.		●	●	●
xResetError BOOL	TRUE	Reset axis error or software error.	●	●	●
xRegulatorOn BOOL	TRUE	Activate controller enable of the axis (via the **MC_Power** function block).	●	●	●
xStop BOOL	TRUE	Cancel the active movement and brake the axis to a standstill with the deceleration defined via the lrStopDec parameter. • The state changes to "Stop". • The technology module remains in the "Stop" state as long as xStop is set to TRUE (or xHalt = TRUE). • The input is also active with "Internal Control".	●	●	●
xHalt BOOL	TRUE	Cancel the active movement and brake the axis to a standstill with the deceleration defined via the lrHaltDec parameter. • The state changes to "Stop". • The technology module remains in the "Stop" state as long as xStop is set to TRUE (or xHalt = TRUE).	●	●	●
scPar L_TT1P_scPar_WinderDancerCtrl[Base/State/High]	The parameter structure contains the parameters of the technology module. The data type depends on the version used (Base/State/High).		●	●	●
scAccessPoints L_TT1P_scAP_WinderDancerCtrl[Base/State/High]	Structure of the access points The data type depends on the version used (Base/State/High).		●	●	●
MaterialCounterAxis AXIS_REF	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 increased 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 (lrSetLineVel or lrSetLineVelDiamCalc input). ▶ Material length counter (📖 43) ▶ Sources for the material length counting (📖 44)		●	●	●
xMaterialFeeding BOOL	Material feeding at the reels from the top or from the bottom TRUE: Material feeding from the top FALSE: Material feeding from the bottom		●	●	●
xWindingDirection BOOL	Winder function at positive line velocity (lrSetLineVel input > 0) TRUE: Unwinder FALSE: Rewinder		●	●	●

Designator / Data type	Description		Available in version: Base	State	High
xLoadDiam BOOL	TRUE	Load the (start) diameter from the lrSetDiam input. • Unit: mm	●	●	●
lrSetDiam LREAL	Defining a (start) diameter The diameter is loaded cyclically when the xLoadDiam is set to TRUE. • Unit: mm		●	●	●
xHoldDiam BOOL	TRUE	Current diameter is held.	●	●	●
xDiamCalcReduced BOOL	Change-over of the diameter calculation mode between long/ short distance		●	●	●
	TRUE	Diameter is updated after the short distance.			
	FALSE	Diameter is updated after the long distance.			
xSetMaterialCounter BOOL	The input is edge-controlled and evaluates the FALSE↗TRUE edge.		●	●	●
	TRUE	Sets the material length counter (lrMaterialCounter output) to the value set under the lrSetMaterialPos parameter.			
xWebBreakMonit BOOL	TRUE	Activate web break monitoring.	●	●	●
xJogLinePos BOOL	TRUE	Drive axis in positive material flow direction (manual jog). If xJogLineNeg is also TRUE, the traversing direction selected first remains set.	●	●	●
xJogLineNeg BOOL	TRUE	Drive axis in negative material flow direction (manual jog). If xJogLinePos is also TRUE, the traversing direction selected first remains set.	●	●	●
xTrimLinePos BOOL	TRUE	Enable the velocity offset in positive material flow direction when the winder axis is synchronised with the line (xSyncLineVel = TRUE)	●	●	●
xTrimLineNeg BOOL	TRUE	Enable the velocity offset in negative material flow direction when the winder axis is synchronised with the line (xSyncLineVel = TRUE)	●	●	●
xSyncLineVel BOOL	TRUE	Synchronise winder axis with the line.	●	●	●
xTensCurve BOOL	TRUE	Enable tensile force characteristic.	●	●	●
lrSetTens LREAL	Tensile force setpoint • Unit: N		●	●	●
lrSetLineVel LREAL	Current line velocity • Unit: mm/s		●	●	●
lrSetLineVelDiamCalc LREAL	Current line velocity for diameter calculation • Unit: mm/s		●	●	●
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.	●	●	●
xDancerCtrl BOOL	TRUE	Activate dancer position control.	●	●	●

Designator / Data type	Description		Available in version: Base	State	High
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). • Unit: s		●	●	●
lrSetDancerPosScaled LREAL	Scaled setpoint for the dancer position • Unit: x 100 % • Value range: -1 ... 1 (-100 ... 100 %)		●	●	●
lrDancerPosInfluenceScaled LREAL	Influence range of the dancer position controller • Unit: x 100 % • Value range: 0 ... 1 (0 ... 100 %)		●	●	●
xResetICtrl BOOL	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 to '0' via the ramp function. The correcting variable from the P component is not affected.	●	●	●
xAccCmps BOOL	TRUE	Activate acceleration compensation during the dancer position control.		●	●
lrMInertiaAdapt LREAL	Multiplier for the current moment of inertia			●	●
xExecuteIdentMInertia BOOL	The input is edge-controlled and evaluates the rising edge.				●
	FALSE↗TRUE	The moment of inertia at the winder shaft is detected. The lrIdentMInertia output displays the detected moment of inertia in $kgcm^2$.			
xAdaptSpdCtrlGain BOOL	TRUE	Switch on adaptation of the speed controller gain.			●

3.3.3 Outputs

Designator / Data type	Description		Available in version: Base	State	High
xInternalControlActive BOOL	The internal control of the axis is activated via the visualisation. (xEnableInternalControl input = TRUE)		●	●	●
eTMState L_TT1P_States	Current state of the technology module ▶ State machine (📖 27)		●	●	●
scStatusABC scStatus_ABC	Structure of the status data of the **L_MC1P_AxisBasicControl** function block		●	●	●
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 xError = TRUE or xWarning = TRUE. **"FAST technology modules" reference manual:** Here you can find information on error or warning messages.		●	●	●
scErrorInfo L_TT1P_scErrorInfo	Error information structure for a more detailed analysis of the error cause		●	●	●
xAxisEnabled BOOL	TRUE	The axis is enabled.	●	●	●
scSignalFlow L_TT1P_scSF_WinderDancerCtrl[Base/State/High]	Structure of the signal flow The data type depends on the version used (Base/State/High). ▶ Signal flow diagrams (📖 28)		●	●	●
xDone BOOL	TRUE	The request/action has been completed successfully.	●	●	●
xBusy BOOL	TRUE	The request/action is currently being executed.	●	●	●
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.	●	●	●
xUnwind BOOL	Status bit for unwinder and rewinder		●	●	●
	TRUE	Unwinder			
	FALSE	Rewinder			
xWebBreak BOOL	TRUE	A web break has occurred.	●	●	●
xHoldDiamActive BOOL	TRUE	Current diameter is held.	●	●	●
xDiamMax BOOL	TRUE	The maximum diameter has been reached.	●	●	●
xDiamMin BOOL	TRUE	The minimum diameter has been reached.	●	●	●
lrSetDiamOut LREAL	Current diameter calculated • Unit: mm		●	●	●
lrSetDiamScaledOut LREAL	Current diameter calculated and scaled • Unit: x 100 % • 1 = 100 % = parameter lrMaxDiam		●	●	●

Designator / Data type	Description		Available in version: Base	State	High
lrMaterialCounter LREAL	Display of the material length counter content on the winder Depending on the Defining the winding direction (winding/unwinding) (📖 34), the material length counter is incremented or decremented. • Unit: mm		●	●	●
lrSetLineVelScaledOut LREAL	Current line velocity scaled • Unit: x 100 % • 1 = 100 % = parameter lrLineVelRef		●	●	●
lrActTotalTrqScaled LREAL	Current scaled torque of the winder shaft • Reference variable: Rated/reference torque of the motor. • Unit: x 100 % (1 = 100 %)		●	●	●
lrWndSpdRef LREAL	Reference of the winder speed at minimum diameter and maximum line velocity.		●	●	●
xDancerReachedSetPos BOOL	TRUE	The dance has reached the set position.	●	●	●
xDancerMaxPos BOOL	TRUE	The dancer has reached the upper limit position.	●	●	●
xDancerMinPos BOOL	TRUE	The dancer has reached the lower limit position.	●	●	●
lrActDancerPosScaled LREAL	Current scaled dancer position • Unit: x 100 % • Value range: -1 ... 1 (-100 ... 100 %)		●	●	●
lrSetDancerCtrlScaledOut LREAL	Actual correcting variable of the dancer position controller • 1 = 100 % = parameter lrLineVelRef		●	●	●
lrSetTensCurveOut LREAL	Current tensile force setpoint from the curve function		●	●	●
lrSetMInertiaOut LREAL	Current moment of inertia at the winder shaft • Unit: $kgcm^2$			●	●
lrSetAccTrqScaledOut LREAL	Torque component of acceleration compensation scaled to the rated/reference torque of the motor			●	●
lrIdentMInertia LREAL	Identified moment of inertia at the winder shaft • Unit: $kgcm^2$				●
lrSetSpdCtrlGainAdaptOut LREAL	Adaptation of the speed controller gain • Unit: x 100 % (1 = 100 %)				●

3.3.4 Parameters

L_TT1P_scPar_WinderDancerCtrl[Base/State/High]

The **L_TT1P_scPar_WinderDancerCtrl[Base/State/High]** structure contains the parameters of the technology module.

Designator / Data type	Description	Available in version		
		Base	State	High
lrStopDec LREAL	Deceleration for the stop function and when hardware/ software limit switches and the following error monitoring function are triggered • Unit: revs/s • Initial value: 10000	●	●	●
lrStopJerk LREAL	Jerk for the stop function and for the triggering of the hardware limit switches, software limit positions, and the following error monitoring function • Unit: $revs/s^3$ • Initial value: 100000	●	●	●
lrHaltDec LREAL	Deceleration for the holding function Specification of the maximum speed variation which is to be used for deceleration to standstill. • Unit: $revs/s^2$ • Initial value: 3600 • Only positive values are permissible.	●	●	●
lrJerk LREAL	Jerk for compensation of a holding function • Unit: $revs/s^3$ • Initial value: 100000	●	●	●
lrLineJerk LREAL	Jerk for manual jog and compensation of a trimming or clutch function • Unit: mm/s^3 • Initial value: 10000	●	●	●
lrJogLineAcc LREAL	Acceleration for manual jog Specification of the maximum speed variation which is to be used for acceleration. • Unit: mm/s^2 • Initial value: 100	●	●	●
lrJogLineDec LREAL	Deceleration for manual jog Specification of the maximum speed variation which is to be used for deceleration to standstill. • Unit: mm/s^2 • Initial value: 100	●	●	●
lrJogLineVel LREAL	Maximum speed to be used for manual jog. • Unit: mm/s • Initial value: 10	●	●	●
lrTrimLineAcc LREAL	Acceleration for trimming Selection of the speed change relative to the line speed to be used for accelerating. The acceleration acting on the drive is the sum of line and trimming acceleration. • Unit: mm/s^2 • Initial value: 100	●	●	●
lrTrimLineDec LREAL	Deceleration for trimming Selection of the speed change relative to the line speed to be used for decelerating. The deceleration acting on the drive is the sum of line and trimming acceleration. • Unit: mm/s^2 • Initial value: 100	●	●	●

Designator Data type	Description	Available in version		
		Base	State	High
lrTrimLineVel LREAL	Velocity for trimming Selection of the velocity used for trimming. • Unit: mm/s • Initial value: 10	●	●	●
lrSyncLineAcc LREAL	Acceleration for synchronising to line velocity • Unit: mm/s^2 • Initial value: 100	●	●	●
lrSyncLineDec LREAL	Deceleration for synchronising to line velocity • Unit: mm/s^2 • Initial value: 100	●	●	●
lrWebBreakWindow LREAL	Web break window The current diameter is compared with the previous diameter across the web break window. • Unit: x 100 % (1.0 = 100 %) • Initial value: 0.1 (10 %)	●	●	●
lrMaxDiam LREAL	Maximum diameter • Unit: mm • Initial value: 180	●	●	●
lrMinDiam LREAL	Minimum diameter • Unit: mm • Initial value: 50	●	●	●
rFiltTimeDiam REAL	PT1 filter time for the current diameter (lrSetDiamOut) • Unit: s • Initial value: 0.05	●	●	●
lrDiamCalcRegularDist LREAL	Regular calculation distance for the diameter • Unit: rev • Initial value: 1	●	●	●
lrDiamCalcReducedDist 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 lrSetDiam. • 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	●	●	●
lrTensCurveCtrlScaled 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.	●	●	●
lrTensCurveStartDiamScaled LREAL	Initial point of the characteristic for the tension control open loop • Unit: x 100 % • 1 = 100 % = parameter lrMaxDiam • Initial value: 0 (0 %)	●	●	●
lrLineVelRef LREAL	Maximum line velocity • Unit: mm/s • Initial value: 1000	●	●	●

Designator Data type	Description	Available in version		
		Base	State	High
lrMinLineVel 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 (lrMaterialCounter output) • Initial value: 0 (filter is deactivated.)	●	●	●
lrSetMaterialPos LREAL	Position of the material length counter With a FALSE↗TRUE edge at the xSetMaterialCounter input, the material length counter (lrMaterialCounter output) is set to the value in lrSetMaterialPos. • Unit: mm	●	●	●
xLineVelDiamCalc BOOL	Diameter calculation • Initial value: FALSE TRUE: For calculating the diameter, the velocity from the lrSetLineVelDiamCalc input is used. FALSE: For calculating the diameter, the velocity from the lrSetLineVel input is used.	●	●	●
lrDancerPosRamp LREAL	Acceleration ramp for dancer position setpoints • Unit: 1/s • Initial value: 1	●	●	●
lrDancerPosCtrlGain LREAL	Controller gain • Initial value: 1	●	●	●
lrDancerPosCtrlResetTime LREAL	Controller reset time • Unit: s • Initial value: 0 controller reset time is deactivated.)	●	●	●
lrDancerPosCtrlLimPos LREAL	Limitation of the dancer position controller correcting variable (output of the controller) in positive direction • Initial value: 1	●	●	●
lrDancerPosCtrlLimNeg LREAL	Limitation of the dancer position controller correcting variable (output of the controller) in negative direction • Initial value: -1	●	●	●
lrDancerMaxPosScaled LREAL	Maximum dancer position for the xDancerMaxPos status bit • Unit: x 100 % (1 = 100 %) • Initial value: 0.95 (95 %)	●	●	●
lrDancerMinPosScaled LREAL	Minimum dancer position for the xDancerMaxPos 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 **lrDancerLowerLimit** and **lrDancerUpperLimit** 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 **lrDancerLowerLimit** and **lrDancerUpperLimit** are used.	●	●	●
lrDancerUpperLimit LREAL	Analog value for the upper dancer position limit • Initial value: 10000000	●	●	●
lrDancerLowerLimit LREAL	Analog value for the lower dancer position limit • Initial value: 0	●	●	●

<table>
<tr><th rowspan="2">Designator
Data type</th><th rowspan="2" colspan="2">Description</th><th colspan="3">Available in version</th></tr>
<tr><th>Base</th><th>State</th><th>High</th></tr>
<tr><td>lrDancerMaterialLength
LREAL</td><td colspan="2">Length of the material in the dancer
• Unit: mm
• Initial value: 0
• The value '0' is used for deactivating monitoring of the dancer movement.</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>lrDancerInPosWindowScaled
LREAL</td><td colspan="2">Window for the set position of the dancer to control the xDancerReachedSetPos status bit.
• Unit: x 100 % (1.0 = 100 %)
• Initial value: 0.2 (20 %)</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>rFiltTimeActDancerPosIn
REAL</td><td colspan="2">PT1 filter time for the lrActDancerPosIn input
• Unit: s
• Initial value: 0.005</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>rFiltTimeActDancerVelComp
REAL</td><td colspan="2">Filter time for the speed compensation
• Unit: s
• Initial value: 0</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td rowspan="4">wWebBreakMode
WORD</td><td colspan="2">Mode for web break monitoring
• Initial value: 1</td><td rowspan="4">●</td><td rowspan="4">●</td><td rowspan="4">●</td></tr>
<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 position of the dancer</td></tr>
<tr><td>2</td><td>Web break monitoring only via diameter calculation</td></tr>
<tr><td rowspan="4">dwSelectTensCurve
DWORD</td><td colspan="2">Selection of the characteristic for tension control
• Initial value: 0</td><td rowspan="4"></td><td rowspan="4">●</td><td rowspan="4">●</td></tr>
<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 according to predefined characteristic</td></tr>
<tr><td>alrTensCurve
ARRAY [1...65] OF LREAL</td><td colspan="2">Characteristic for tension control open loop consisting of 65 values.</td><td></td><td>●</td><td>●</td></tr>
<tr><td>rFiltTimeAccSpd
REAL</td><td colspan="2">PT1 filter time for the winder shaft speed for acceleration compensation
• Unit: s
• Initial value: 0.005</td><td></td><td>●</td><td>●</td></tr>
<tr><td>lrAccCmpsDeadBandTrqScaled
LREAL</td><td colspan="2">Lagging range (dead band) for the current acceleration torque
• Unit: Nm
• Initial value: 0.1</td><td></td><td>●</td><td>●</td></tr>
<tr><td>lrAccCmpsGainAcc
LREAL</td><td colspan="2">Gain factor for the acceleration torque in positive direction
• Unit: x 100 % (1.00 = 100 %)
• Initial value: 1.05 (105 %)</td><td></td><td>●</td><td>●</td></tr>
<tr><td>lrAccCmpsGainDec
LREAL</td><td colspan="2">Gain factor for the acceleration torque in negative direction
• Unit: x 100 % (1.00 = 100 %)
• Initial value: 0.95 (95 %)</td><td></td><td>●</td><td>●</td></tr>
<tr><td>lrConstMInertia
LREAL</td><td colspan="2">Constant moment of inertia at the winder shaft
• Unit: $kgcm^2$
• Initial value: 9</td><td></td><td>●</td><td>●</td></tr>
<tr><td>lrMaxMInertia
LREAL</td><td colspan="2">Maximally permissible moment of inertia at the winder shaft
• Unit: $kgcm^2$
• Initial value: 50</td><td></td><td>●</td><td>●</td></tr>
</table>

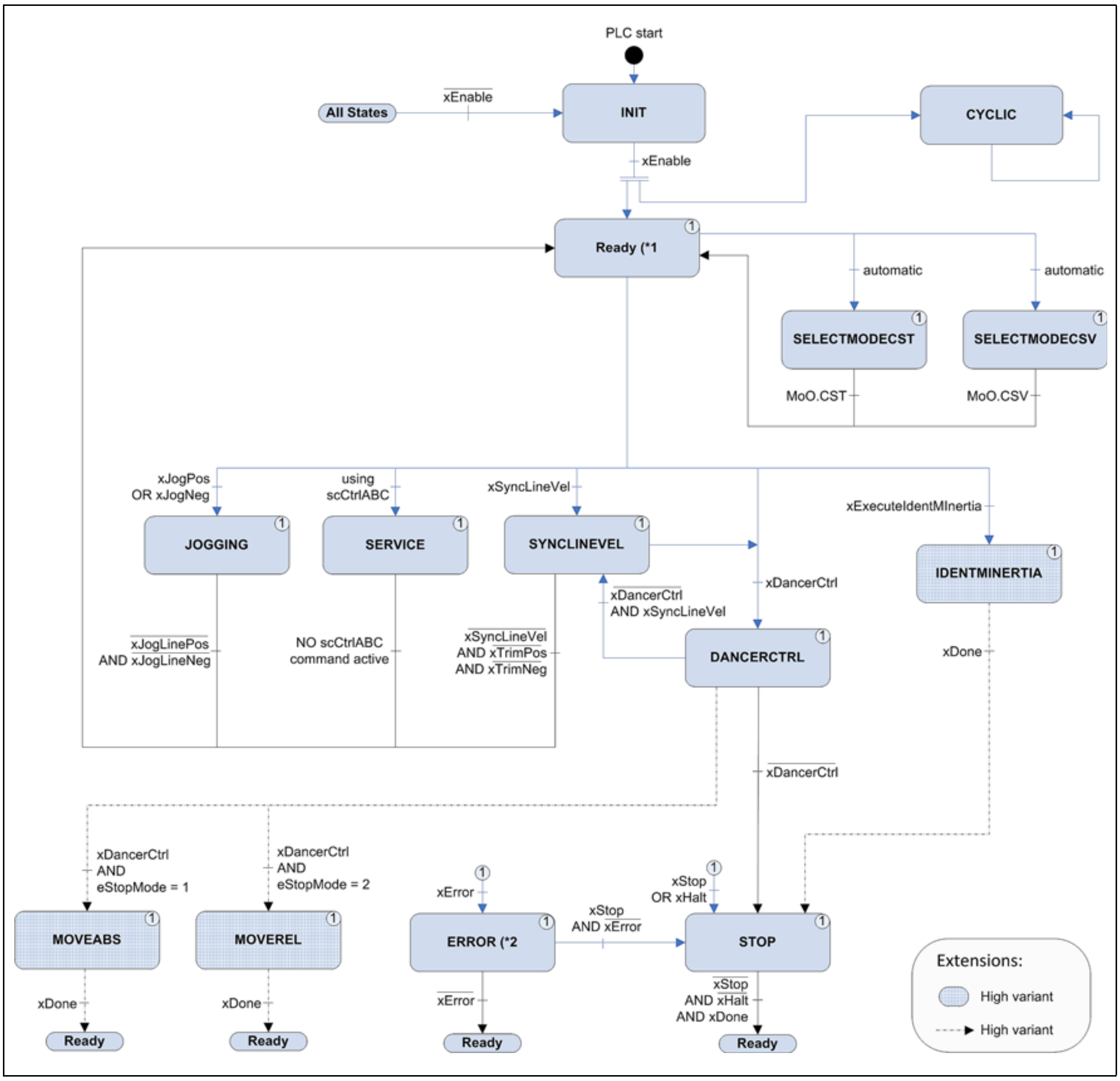
Designator Data type	Description	Available in version: Base	State	High
rFiltTimeIdentMInertiaSpd REAL	PT1 filter time for the speed at the winder shaft during the identification of the moment of inertia • Unit: s • Initial value: 0.01			●
rFiltTimeIdentMInertiaTrq REAL	PT1 filter time for the torque at the winder shaft during the identification of the moment of inertia • Unit: s • Initial value: 0.005			●
lrIdentMInertiaMaxSpd Scaled LREAL	Maximum speed of the winder shaft during the identification of the moment of inertia • Unit: x 100 % (1.0 = 100 % = lrWndSpdRef) • Initial value: 0.2 (20 %)			●
lrIdentMInertiaMaxTrq Scaled LREAL	Maximum torque of the winder shaft during the identification of the moment of inertia • Unit: x 100 % (1.0 = 100 %) • Initial value: 0.2 (20 %)			●
alrSpdCtrlGainAdaptX ARRAY [1...9] OF LREAL	Characteristic function for the speed control gain The X axis corresponds to the scaled moment of inertia. • Unit: x 100 % (1 = 100 % = parameter lrMaxMInertia) • 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:
	  • [0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 0.98, 0.95, 0.95]
	  • Lower limitation: 0.5 = 50 %
	  • Upper limitation: 1.0 = 100 %
	Linear increase of the gain up to 100 % of the moment of inertia
				●

lrReducedGainWindow LREAL	Range of system deviation with reduced gain/sensitivity • Initial value: 0.0			●
lrReducedGain LREAL	Gain of system deviation within the reduced sensitivity • Initial value: 0.0			●
eDancerCtrlStopMode L_TT1P_DancerCtrlStop Mode	Mode in which the winding process ("DANCERCTRL" state) is stopped. • Initial value: 0			●
	0 — Halt, The axis is stopped via the deceleration (lrHaltDec) and the jerk (lrJerk).			
	1 — Move ABS Absolute travel/positioning: The axis travels to the target position (lrPos_Dist) with the velocity (lrVel), acceleration (lrAcc), deceleration (lrDec) and the jerk (lrJerk).			
	2 — Move Rel Relative travel/positioning: The axis is brought to a standstill with the velocity (lrVel), acceleration (lrAcc), deceleration (lrDec) and the jerk (lrJerk) after the distance travelled (lrPos_Dist).			

Designator Data type	Description	Available in version		
		Base	State	High
lrPos_Dist LREAL	Relevant for the mode: • eDancerCtrlStopMode = 1: Absolute target position in [units] (reference to the absolute position is the zero position) • eDancerCtrlStopMode = 2: Distance to be covered in [units] (reference to the setpoint position at the starting time of the command.)			●
lrVel LREAL	Velocity Only relevant for the eDancerCtrlStopMode modes = 1 and 2 (Move ABS, Move Rel). Specification of the maximum velocity which is to be used for travel/positioning. • Unit: units(winder)/s, in standard case rev/s • Initial value: 50			●
lrAcc LREAL	Acceleration Only relevant for the eDancerCtrlStopMode modes = 1 and 2 (Move ABS, Move Rel). Specification of the maximum speed variation which is to be used for acceleration. • Unit: units(winder)/s^2, in standard case rev/s^2 • Initial value: 100			●
lrDec LREAL	Deceleration Only relevant for the eDancerCtrlStopMode modes = 1 and 2 (Move ABS, Move Rel). Selection of the maximum speed variation which is to be used for deceleration to standstill. • Unit: units(winder)/s^2, in standard case rev/s^2 • Initial value: 100			●

3.4 State machine



[3-2] State machine of the technology module

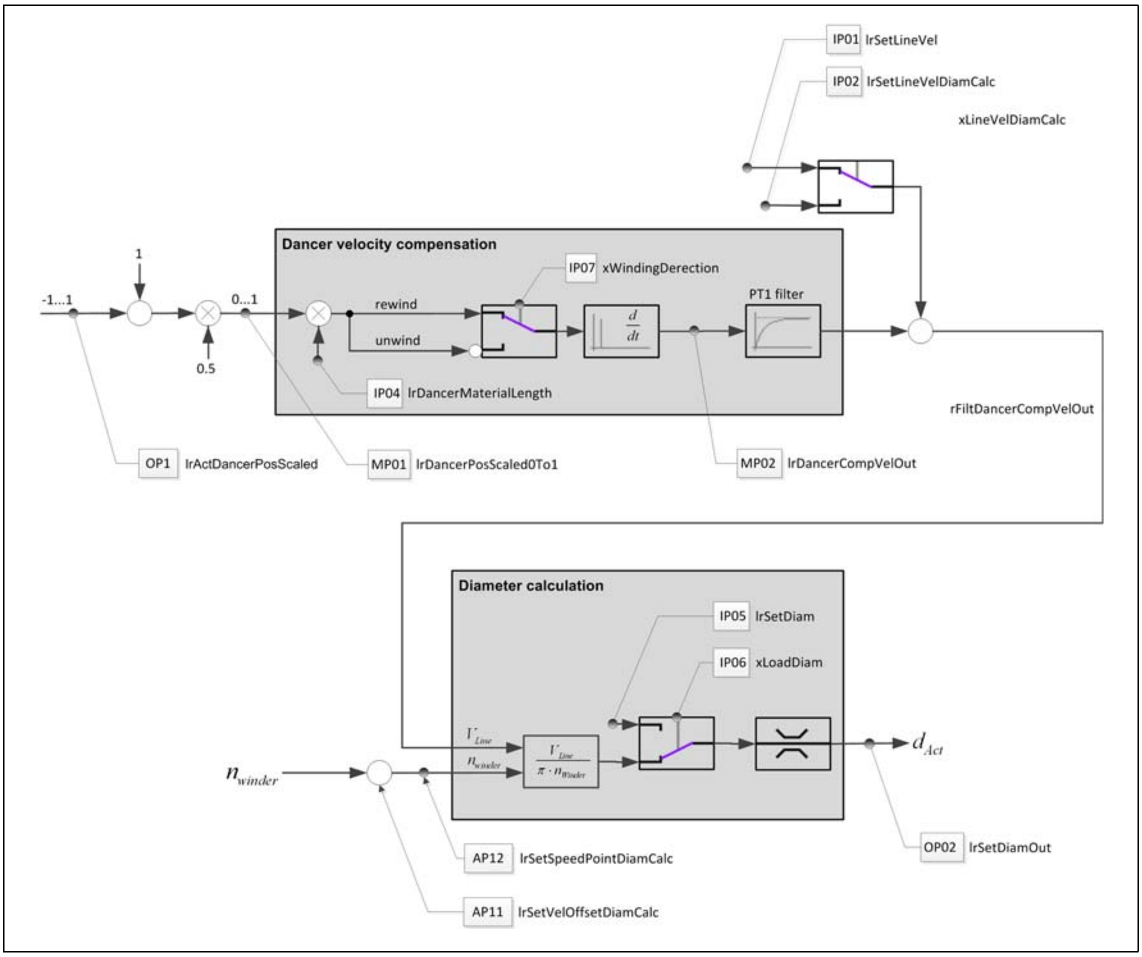
(*1 In the "Ready" state, xRegulatorOn has to be set to TRUE.

(*2 In the "ERROR" state, xResetError has to be set to TRUE in order to acknowledge and reset the errors.

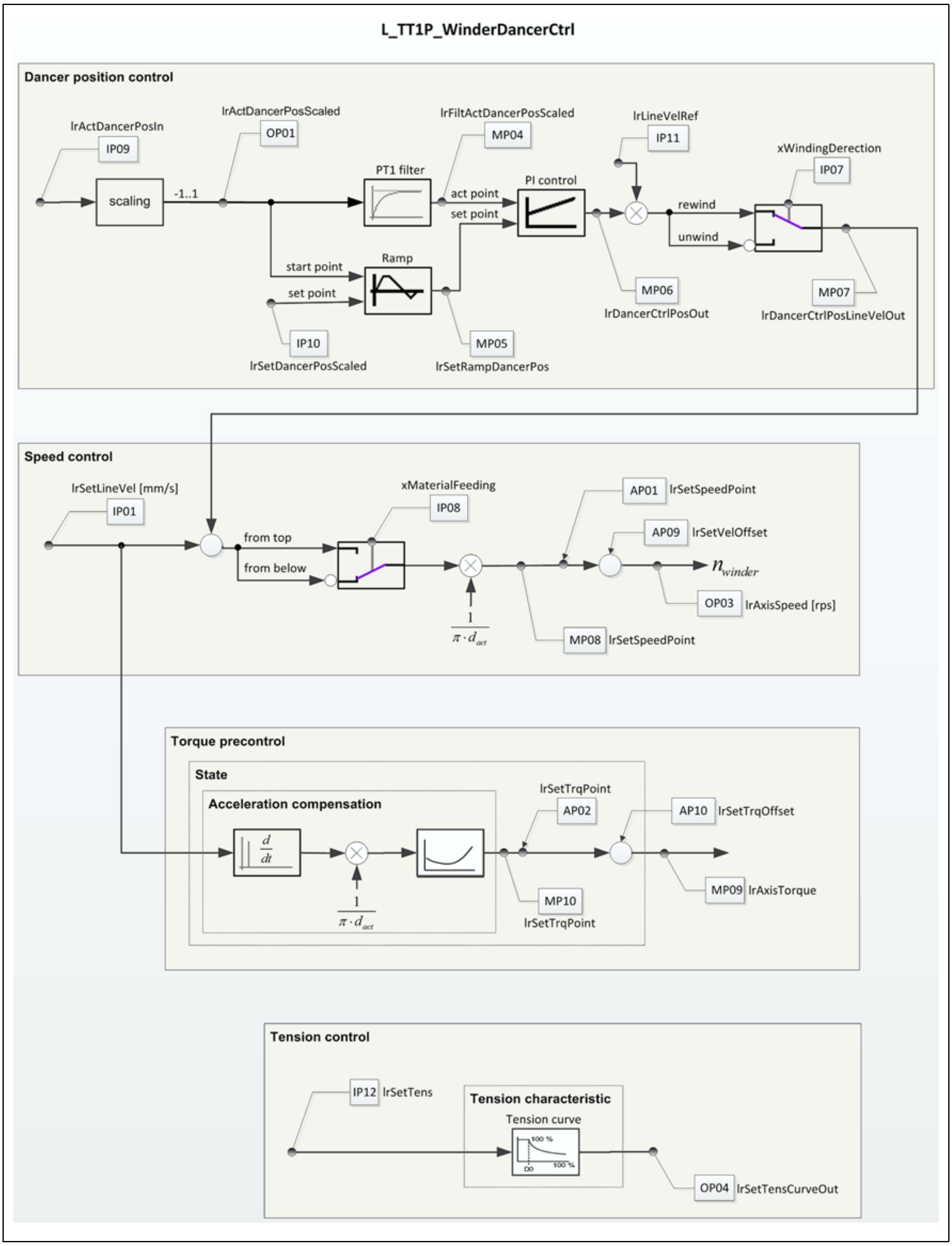
3.5 Signal flow diagrams

The illustrations [3-3] and [3-4] show the main signal flow of the implemented functions.
The signal flow of the additional functions such as "manual jog" is not displayed here.

Diameter calculation



[3-3] Signal flow for calculating the diameter



[3-4] Signal flow of the technology module

3.5.1 Structure of the signal flow

L_TT1P_scSF_WinderDancerCtrl[Base/State/High]

The contents of the **L_TT1P_scSF_WinderDancerCtrl[Base/State/High]** structure are read-only and offer a practical diagnostics option within the signal flow (Signal flow diagrams (📖 28)).

Designator Data type	Description	Available in version		
		Base	State	High
IP01_lrSetLineVel LREAL	Current line velocity • Unit: mm/s	●	●	●
IP02_lrSetLineVelDiamCalc LREAL	Current line velocity for diameter calculation • Unit: mm/s	●	●	●
IP03_xLineVelDiamCalc BOOL	Source of the line velocity for diameter calculation • Initial value: FALSE TRUE: For calculating the diameter, the velocity from the lrSetLineVelDiamCalc input is used. FALSE: For calculating the diameter, the velocity from the lrSetLineVel input is used.	●	●	●
IP04_lrDancerMaterial Length LREAL	Length of the material in the dancer • Unit: mm • Initial value: 1 • The value '0' is used for deactivating monitoring of the dancer movement.	●	●	●
IP05_lrSetDiam LREAL	Defining a (start) diameter The diameter is loaded cyclically when the xLoadDiam is set to TRUE. • Unit: mm	●	●	●
IP06_xLoadDiam BOOL	TRUE: The diameter is loaded from the lrSetDiam signal.	●	●	●
IP07_xWindingDirection BOOL	Winder function at positive line velocity (lrSetLineVel input > 0) TRUE: Unwinder FALSE: Rewinder	●	●	●
IP08_xMaterialFeeding BOOL	Material feeding at the reels from the top or from the bottom TRUE: Material feeding from the top FALSE: Material feeding from the bottom	●	●	●
IP09_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). • Unit: s	●	●	●
IP10_lrSetDancerPosScaled LREAL	Scaled setpoint for the dancer position • Unit: x 100 % • Value range: -1 ... 1 (-100 ... 100 %)	●	●	●
IP11_lrLineVelRef LREAL	Maximum line velocity • Unit: mm/s • Initial value: 1000	●	●	●
IP12_lrSetTens LREAL	Tensile force setpoint • Unit: N	●	●	●
MP01_lrDancerPosScaled 0To1 LREAL	Scaled dancer position • Unit: x 100 % • Value range: 0 ... 1 (0 ... 100 %)	●	●	●
MP02_lrDancerCompVelOut LREAL	Resulting velocity of the dancer • Unit: mm/s	●	●	●

Designator Data type	Description	Available in version		
		Base	State	High
MP03_rFiltDancerCompVel Out REAL	Filtered resulting velocity of the dancer • Unit: mm/s	●	●	●
MP04_rFiltActDancerPos Scaled REAL	Current scaled filtered dancer position • Unit: x 100 % • Value range: -1 ... 1 (-100 ... 100 %)	●	●	●
MP05_lrSetRampDancerPos LREAL	Set position of the dancer after the ramp generator This position is the reference value of the winding drive. • Unit: x 100 % • Value range: -1 ... 1 (-100 ... 100 %)	●	●	●
MP06_lrDancerCtrlPosOut LREAL	Correcting variable of the dancer position controller • Unit: x 100 % • Value range: -1 ... 1 (-100 ... 100 %)	●	●	●
MP07_lrDancerCtrlLineVel Out LREAL	Correcting variable of the dancer position controller converted into line velocity • Unit: mm/s	●	●	●
MP08_lrSetSpeedPoint LREAL	Resulting speed setpoint • Unit: revs/s	●	●	●
MP09_lrAxisTorque LREAL	Torque for feedforward control • Unit: Nm		●	●
MP10_lrSetTrqPoint LREAL	Resulting torque setpoint for the feedforward control • Unit: Nm		●	●
MP11_lrDancerCtrlOutGain LREAL	The correcting variable from the proportional component (P component) of the dancer position controller (scaled)	●	●	●
MP12_lrDancerCtrlOutReset Time LREAL	The correcting variable from the integrating component (I component) of the dancer position controller (scaled)	●	●	●
MP13_lrDancerCtrlOutRate Time LREAL	The correcting variable from the differential component (D component) of the dancer position controller (scaled)	●	●	●
OP01_lrActDancerPosScaled LREAL	Current scaled dancer position • Unit: x 100 % • Value range: -1 ... 1 (-100 ... 100 %)	●	●	●
OP02_lrSetDiamOut LREAL	Current diameter calculated • Unit: mm	●	●	●
OP03_lrAxisSpeed LREAL	Current speed of the winding drive • Unit: revs/s	●	●	●
OP04_lrSetTensCurveOut LREAL	Current tensile force from the curve function • Unit: N	●	●	●

3.5.2 Structure of the access points

L_TT1P_scAP_WinderDancerCtrl[Base/State/High]

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.

<table>
<tr><th rowspan="2">Designator
Data type</th><th rowspan="2" colspan="2">Description</th><th colspan="3">Available in version</th></tr>
<tr><th>Base</th><th>State</th><th>High</th></tr>
<tr><td rowspan="2">AP01_xSetSpeedPoint
BOOL</td><td colspan="2">Enable of the AP01_lrSetSpeedPoint access point</td><td rowspan="3">●</td><td rowspan="3">●</td><td rowspan="3">●</td></tr>
<tr><td>TRUE</td><td>The access point overwrites the values at the access point in the signal flow.</td></tr>
<tr><td>AP01_lrSetSpeedPoint
LREAL</td><td colspan="2">Speed setpoint for the winding drive
• Unit: revs/s</td></tr>
<tr><td rowspan="2">AP02_xSetTrqPoint
BOOL</td><td colspan="2">Enable of the AP02_lrSetTrqPoint access point</td><td rowspan="3"></td><td rowspan="3">●</td><td rowspan="3">●</td></tr>
<tr><td>TRUE</td><td>The access point overwrites the values at the access point in the signal flow.</td></tr>
<tr><td>AP02_lrSetTrqPoint
LREAL</td><td colspan="2">Torque setpoint for the feedforward control
• Unit: Nm</td></tr>
<tr><td rowspan="2">AP03_xSetDancerCtrlOut Gain
BOOL</td><td colspan="2">Enable of the AP03_lrSetDancerCtrlOutGain access point</td><td rowspan="3"></td><td rowspan="3">●</td><td rowspan="3">●</td></tr>
<tr><td>TRUE</td><td>The access point overwrites the values at the access point in the signal flow.</td></tr>
<tr><td>AP03_lrSetDancerCtrlOut Gain
LREAL</td><td colspan="2">Cyclic loading of the correcting variable of the proportional component (P component) of the dancer position controller (scaled)</td></tr>
<tr><td rowspan="2">AP04_xSetDancerCtrlOut ResetTime
BOOL</td><td colspan="2">Enable of the AP04_lrSetDancerCtrlOutResetTime access point</td><td rowspan="3"></td><td rowspan="3">●</td><td rowspan="3">●</td></tr>
<tr><td>TRUE</td><td>The access point overwrites the values at the access point in the signal flow.</td></tr>
<tr><td>AP04_lrSetDancerCtrlOut ResetTime
LREAL</td><td colspan="2">Cyclic loading of the correcting variable of the integral-action component (I component) of the dancer position controller (scaled)</td></tr>
<tr><td rowspan="2">AP05_xSetDancerCtrlOut RateTime
BOOL</td><td colspan="2">Enable of the AP05_lrSetDancerCtrlOutRateTime access point</td><td rowspan="3"></td><td rowspan="3">●</td><td rowspan="3">●</td></tr>
<tr><td>TRUE</td><td>The access point overwrites the values at the access point in the signal flow.</td></tr>
<tr><td>AP05_lrSetDancerCtrlOut RateTime
LREAL</td><td colspan="2">Cyclic loading of the correcting variable of the differential component (D component) of the dancer position controller (scaled)</td></tr>
<tr><td rowspan="2">AP09_xSetVelOffset
BOOL</td><td colspan="2">Enable of the AP09_lrSetVelOffset access point</td><td rowspan="3">●</td><td rowspan="3">●</td><td rowspan="3">●</td></tr>
<tr><td>TRUE</td><td>The access point overwrites the values at the access point in the signal flow.</td></tr>
<tr><td>AP09_lrSetVelOffset
LREAL</td><td colspan="2">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!</td></tr>
<tr><td rowspan="2">AP10_xSetTrqOffset
BOOL</td><td colspan="2">Enable of the AP10_lrSetTrqOffset access point</td><td rowspan="3">●</td><td rowspan="3">●</td><td rowspan="3">●</td></tr>
<tr><td>TRUE</td><td>The access point overwrites the values at the access point in the signal flow.</td></tr>
<tr><td>AP10_lrSetTrqOffset
LREAL</td><td colspan="2">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!</td></tr>
</table>

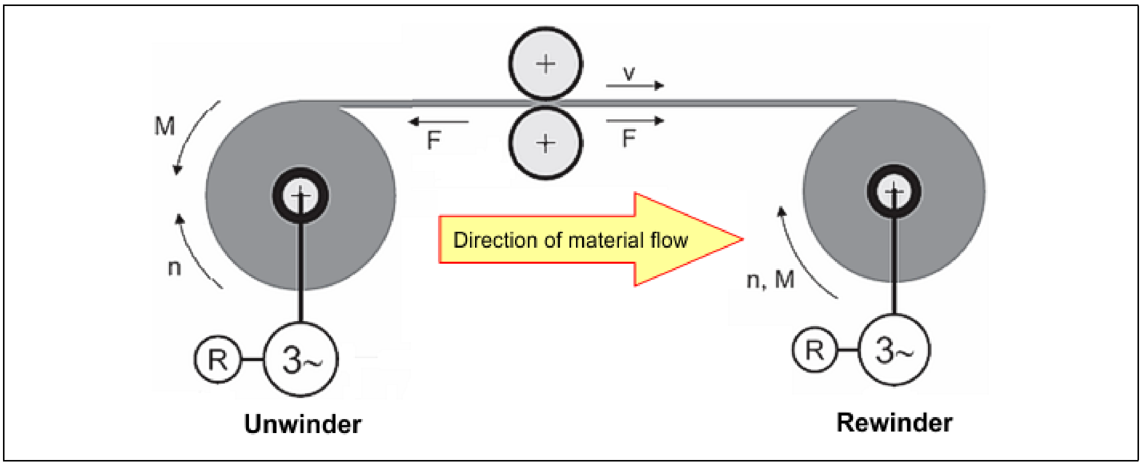
<table>
<tr><th rowspan="2">Designator
Data type</th><th rowspan="2" colspan="2">Description</th><th colspan="3">Available in version</th></tr>
<tr><th>Base</th><th>State</th><th>High</th></tr>
<tr><td rowspan="2">AP11_xSetVelOffsetDiamCalc
BOOL</td><td colspan="2">Enable of the access point AP11_lrSetVelOffsetDiamCalc:</td><td rowspan="3">●</td><td rowspan="3">●</td><td rowspan="3">●</td></tr>
<tr><td>TRUE</td><td>The access point overwrites the values at the access point in the signal flow</td></tr>
<tr><td>AP11_lrSetVelOffsetDiamCalc
LREAL</td><td colspan="2">Cyclic selection of the offset for the velocity 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.</td></tr>
<tr><td rowspan="2">AP12_xSetSpeedPointDiamCalc
BOOL</td><td colspan="2">Enable of the access point AP12_lrSetSpeedPointDiamCalc:</td><td rowspan="3">●</td><td rowspan="3">●</td><td rowspan="3">●</td></tr>
<tr><td>TRUE</td><td>The access point overwrites the values at the access point in the signal flow.</td></tr>
<tr><td>AP12_lrSetSpeedPointDiamCalc
LREAL</td><td colspan="2">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.</td></tr>
</table>

3.6 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.)



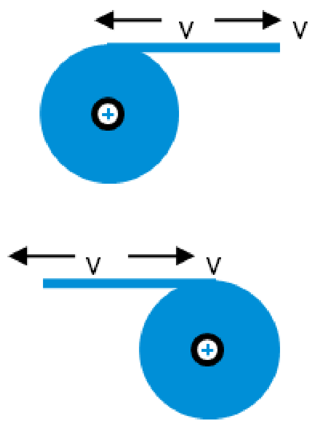
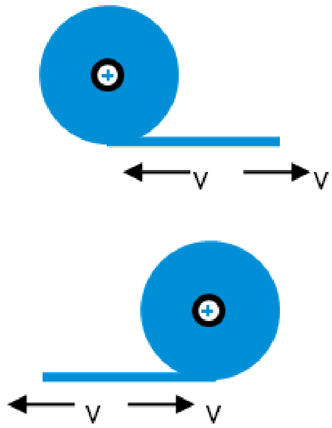
[3-5] Effective direction of speed and torque as a function of the material flow

3.7 Automatic detection of the winding direction

After Defining the winding direction (winding/unwinding) (📖 34), the winding drives can also be operated in the opposite direction with a negative line velocity. No intervention in the signal flow is necessary when the material flow direction is reversed. The current winding direction is provided at the *xUnwind* output.

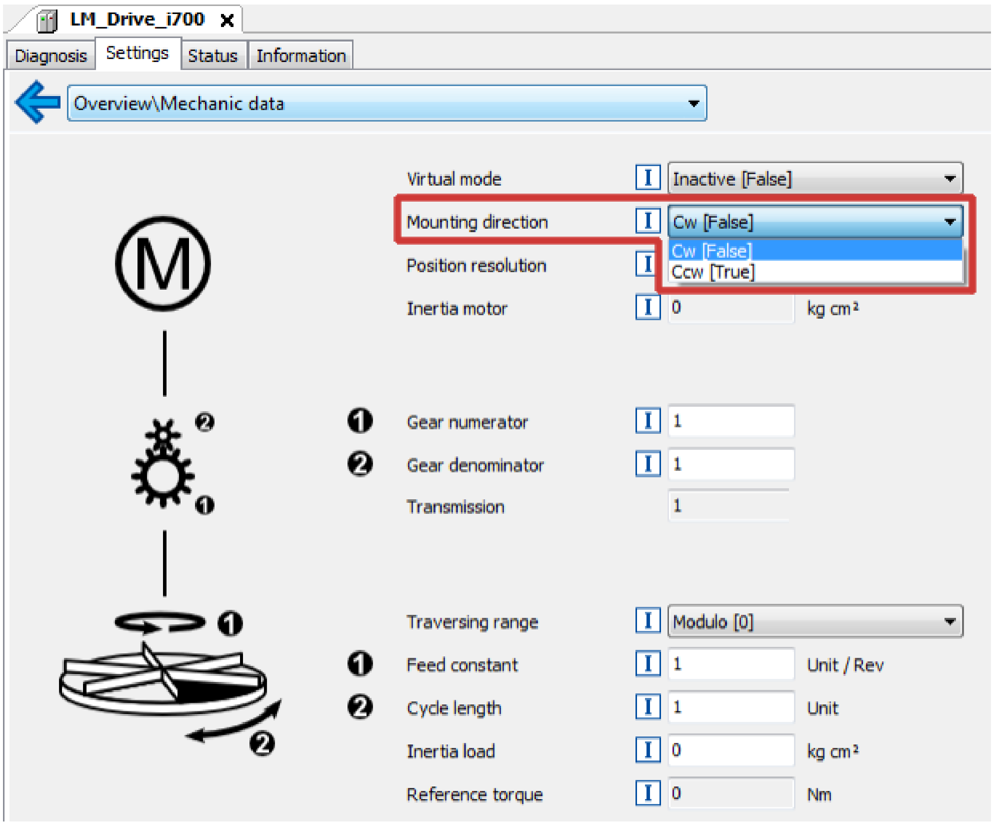
3.8 Defining the material feed to the winder

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

Material from the top (xMaterialFeeding = TRUE)	Material from the bottom (xMaterialFeeding = FALSE)
	

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

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



3.9 Master value source for diameter calculation

The technology module always operates with the line speed at the *lrSetLineVel* 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 *lrSetLineVelDiamCalc* input and the *xLineVelDiamCalc* parameter must be set to TRUE. In this way, taking the dancer movement into consideration is dispensable - the *lrDancerMaterialLength* parameter has to be set to 0.

If no encoder is used, the *xLineVelDiamCalc* parameter has to be set to FALSE. The line speed at *lrSetLineVel* is used to calculate the diameter. In order to optimise the diameter calculation, the dancer movement can be taken into consideration (see Diameter calculation with correction of the dancer position (📖 41)).

Parameters to be set

The parameters for the diameter calculation with or without encoders are located in the L_TT1P_scPar_WinderDancerCtrl[Base/State/High] (📖 21) parameter structure.

```
xLineVelDiamCalc : BOOL := FALSE;
lrDancerMaterialLength : LREAL := 0;
```

3.10 Speed feedforward control

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

Calculation of the setpoint speed for speed feedforward control		
$$n_{Set} = \frac{v_{Line}}{d_{act} \cdot \pi}$$		
Symbol	Description	Dimension unit
n_{Set}	Setpoint speed for speed feedforward control	revs/s
v_{Line}	Line velocity at the lrSetLineVel 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 *lrWndSpdRef* output refers to the motor speed required at minimum diameter (d_{min}) in order to reach the reference line velocity at the circumference of the reel.

Check speed feedforward control

- Load the diameter calculator with the minimum diameter (d_{min}):
  Input *lrSetDiam* = 0 (or ≤ d_{min})
  Input *xLoadDiam* = TRUE
- In case of Synchronisation to the line velocity (📖 48) with the *xSyncLineVel* input = TRUE, the winder axis follows the line velocity setpoint in a speed-controlled manner without correcting the dancer position.
  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 *lrWndSpdRef* output.
- Now, the circumferential speed of the winder has to correspond to half the *lrLineVelRef* reference. The current line velocity signal is displayed at the output of the technology module *lrSetLineVelScaledOut* = 0.5 [x 100 %] = 50 %.

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

3.11 Calculation of the diameter

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

Calculation of the current diameter		
$$d_{act} = \frac{v_{Line}}{n_{Winder} \cdot \pi}$$		
Symbol	Description	Dimension unit
d_{act}	Current diameter	mm
v_{Line}	Line velocity	mm/s
n_{Winder}	Winder speed	revs/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 *lrDiamCalcRegularDist*. The initial value of this parameter is set to 1 winding shaft revolution.

For quick diameter changes of *lrDiamCalcRegularDist*, it can be switched to the fast calculation mode by setting the *xDiamCalcReduced* input = TRUE. The lower calculation distance is set with the *lrDiamCalcReducedDist* 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.

Parameters to be set

The parameters for diameter calculation are located in the L_TT1P_scPar_WinderDancerCtrl[Base/State/High] (📖 21) parameter structure.

```
lrDiamCalcRegularDist : LREAL := 1;
lrDiamCalcReducedDist : LREAL := 0.1;
```

3.12 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 velocity < minimum line velocity
  (*lrMinLineVel* [mm/s] from the L_TT1P_scPar_WinderDancerCtrl[Base/State/High] (📖 21)) parameter structure;
- Winder speed < *lrMinLineVel* [mm/s] / (π x d [mm]);
- In the states STOP, ERROR, READY, JOGGING and SYNCLINEVEL.

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

3.13 Defining the diameter / signal from the diameter 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.

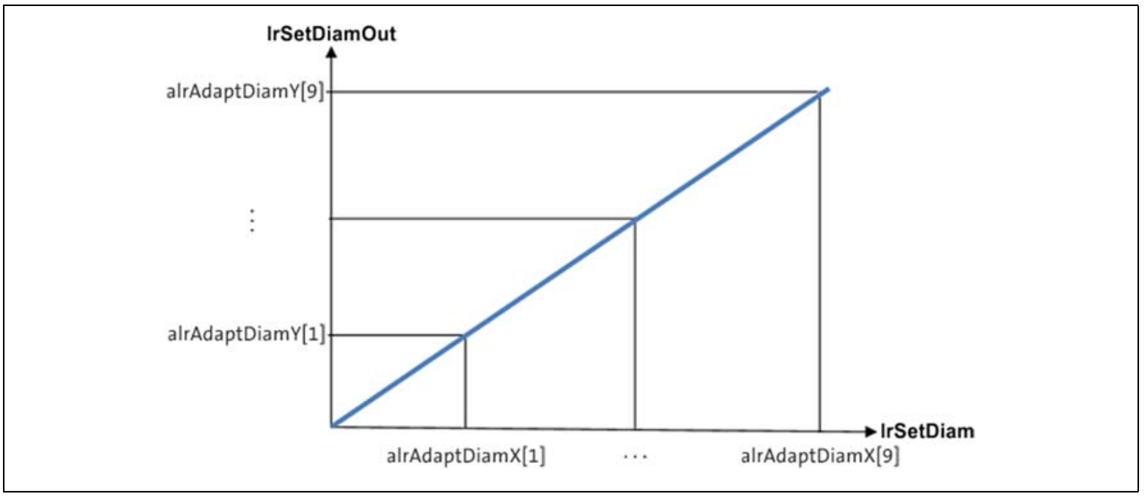
The *lrSetDiam* input serves to define a start diameter that is accepted with the highest priority and loaded cyclically when *xLoadDiam* = TRUE.

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.

Parameters to be set

The parameters for the curve function are located in the L_TT1P_scPar_WinderDancerCtrl[Base/State/High] (📖 21) parameter structure.

```
alrAdaptDiamX : ARRAY[1...9] OF LREAL := [0,100,200,300,400,500,600,700,800];
alrAdaptDiamY : ARRAY[1...9] OF LREAL := [0,100,200,300,400,500,600,700,800]
```

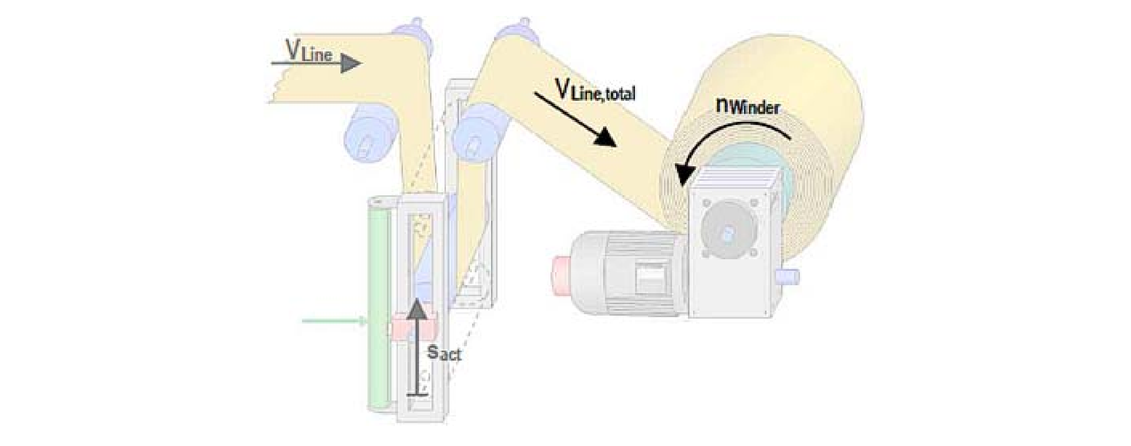


[3-6] Loading of a diameter via a curve function

3.14 Diameter calculation with correction of the dancer position

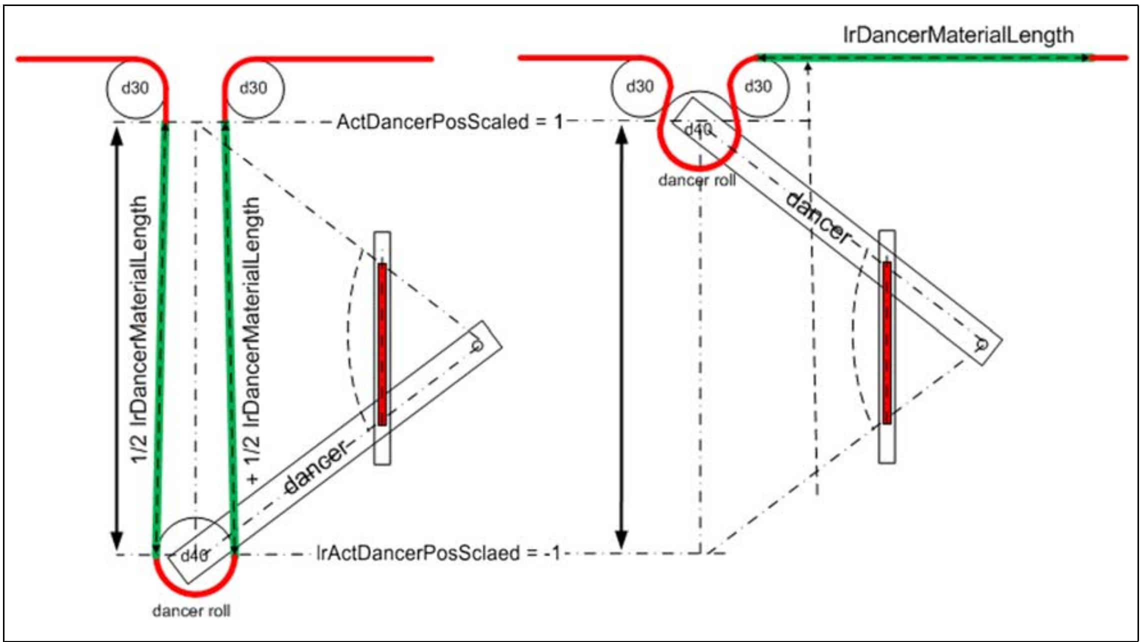
If the circumferential velocity of the winder is faster or slower than the line, the dancer position has to be corrected. If, in this connection, the circumferential velocity significantly increases or decreases compared to the line velocity before the dancer, the resulting circumferential velocity ($V_{Line;\ total}$) has to be used for diameter calculation.

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



[3-7] 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%.



[3-8] Example: The stored material length corresponds to twice the dancer path

The storage volume is defined via the *lrDancerMaterialLength* 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 *rFiltTimeActDancerPosIn* time constant. The filtered position is scaled from -1 to 1 and provided at the *lrActDancerPosScaled* output.

The scaled dancer position is converted to a material length in millimeters via the *lrDancerMaterialLength* parameter using the following formula:

Equation for converting the scaled dancer position to a material length

$$\mathrm{DancerPos} = \left(\frac{lrActDancerPosScaled + 1}{2}\right) \cdot lrDancerMaterialLength$$

Symbol	Description	Dimension unit
DancerPos	Dancer position	
lrActDancerPosScaled	Current scaled dancer position • Value range: -1 ... 1 (-100 ... 100 %)	%
lrDancerMaterialLength	Length of the material in the dancer	mm

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

Additional speed for diameter calculation

When the following settings are made, each dancer movement is taken into account in the diameter calculation:

- Parameterisation of the maximum material length that can be stored in the dancer mechanism.
  lrDancerMaterialLength = maximum material length [mm]
- With the default setting *lrDancerMaterialLength* = 0 [mm], the additional speed from the dancer movement is not taken into account in the calculation of the diameter.

Parameters to be set

The parameters for diameter calculation are located in the L_TT1P_scPar_WinderDancerCtrl[Base/State/High] (📖 21) parameter structure.

```
lrDancerMaterialLength : LREAL := 0;
rFiltTimeActDancerPosIn : REAL := 0.005;
```

3.15 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 *lrSetLineVel* input and is shown at the *lrMaterialCounter* output (in millimetres). Depending on the Defining the winding direction (winding/unwinding) (📖 34), the material length is incremented or decremented.

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 default setting is '0 ms').

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

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

Parameters to be set

The parameters for the material length counter are located in the L_TT1P_scPar_WinderDancerCtrl[Base/State/High] (📖 21) parameter structure.

```
rFiltTimeMaterialCounter : LREAL := 0;
lrSetMaterialPos : REAL := 0;
```

3.16 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.

3.16.1 Source: "lrSetLineVel" input

Conditions

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

Operating principle

In addition to the position (*lrSetMaterialPos* parameter), the line speed is integrated into the material length counting process at the *lrSetLineVel* input. The resulting value is shown as material length at the *lrMaterialCounter* output and is stored persistently.

With a FALSE↗TRUE edge at the *xSetMaterialCounter* input, the material length is loaded from the *lrSetMaterialPos* parameter, directly setting the material length to the value of *lrSetMaterialPos* at the *lrMaterialCounter* output. The values from the continuing counting process are added to the value of the material length set at the output.

Note!

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.

3.16.2 Source: "lrSetLineVelDiamCalc" input

Conditions

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

Operating principle

In addition to the position (*lrSetMaterialPos* parameter), the line speed for the Calculation of the diameter (📖 38) is integrated into the material length counting process at the *lrSetLineVelDiamCalc* input. The resulting value is shown as material length at the *lrMaterialCounter* output and is stored persistently.

With a FALSE↗TRUE edge at the *xSetMaterialCounter* input, the material length is loaded from the *lrSetMaterialPos* parameter, directly setting the material length to the value of *lrSetMaterialPos* at the *lrMaterialCounter* output. The values from the continuing counting process are added to the value of the material length set at the output.

Note!

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.

3.16.3 Source: "MaterialCounterAxis" input (reference axis)

Conditions

- A reference axis (modulo axis) is connected to the *MaterialCounterAxis* input.
- The loss-free number of revolutions determined is used as a basis for the material length counting. – This procedure is suitable for noisy signals!

Operating principle

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.

Note!

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.

3.17 Manual jog (jogging)

For the manual jog of the winder, the *lrJogLineVel* manual jog velocity is used.

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 *lrJogLineVel, lrJogLineAcc* and *lrJogLineDec* for manual jog refer to the circumferential velocity or line velocity and not to the motor speed.

Parameters to be set

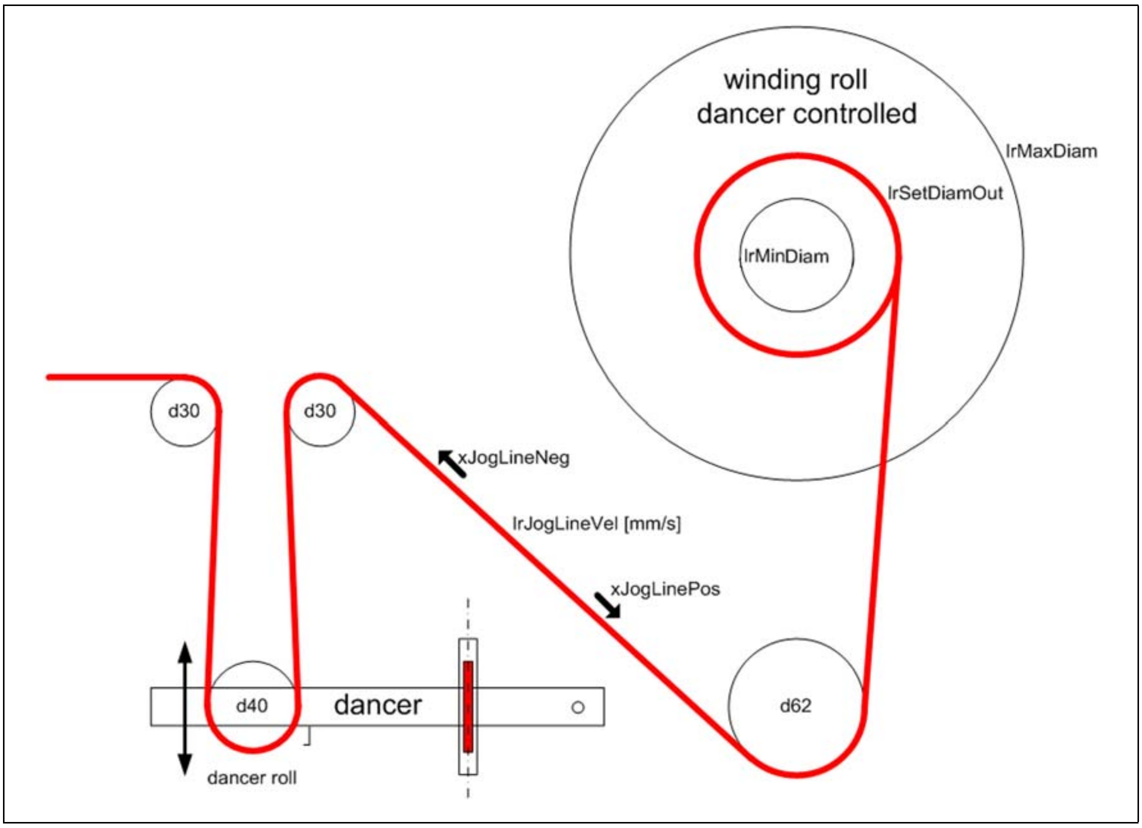
The parameters for the manual jog are located in the L_TT1P_scPar_WinderDancerCtrl[Base/State/High] (📖 21) parameter structure.

```
lrLineJerk   : LREAL := 10000; // Jerk [mm/s^3]
lrJogLineVel : LREAL := 100;   // Velocity [mm/s]
lrJogLineAcc : LREAL := 100;   // Acceleration [mm/s^2]
lrJogLineDec : LREAL := 10;    // Deceleration [mm/s^2]
```

The parameter values can be changed during operation. They are accepted when resetting the inputs *xJogLinePos* = TRUE or *xJogLineNeg* = TRUE.

Example

- *xWindingDirection* = FALSE: Rewinder (material is rewound.)
- *xMaterialFeeding* = FALSE: The material is fed from below.



[3-9] Manual jog of the line

Note!

During manual jog, the diameter calculator is stopped and the diameter is held (*xHoldDiamActive* = TRUE).

3.18 Synchronisation to the line velocity

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

The parameters *lrSyncLineAcc* and *lrSyncLineDec* refer to the circumferential velocity or line velocity and not to the motor speed.

Parameters to be set

The parameters for synchronising to the line velocity are located in the L_TT1P_scPar_WinderDancerCtrl[Base/State/High] (📖 21) parameter structure.

```
lrLineJerk    : LREAL := 10000; // Jerk [mm/s^3]
lrSyncLineAcc : LREAL := 100;   // Acceleration [mm/s^2]
lrSyncLineDec : LREAL := 100;   // Deceleration [mm/s^2]
```

Note!

During manual jog, the diameter calculator is stopped and the diameter is held (*xHoldDiamActive* = TRUE).

3.19 Trimming

Note!

Trimming can only be used if the winder axis is synchronised to the line velocity.

▶ Synchronisation to the line velocity (📖 48)

When the *xTrimLinePos* input = TRUE, the line is trimmed in positive direction and when the *xTrimLineNeg* input = TRUE, the line is trimmed in negative direction.

For trimming, the *lrTrimLineVel* trimming velocity is added to the *lrSetLineVel* line velocity. In the trimming process, the overall setpoint can exceed the trimming setpoint maximally by the value of the minimum line velocity.

The parameterisable setpoints *lrTrimLineVel, lrTrimLineAcc* and *lrTrimLineDec* for the positive and negative trimming operation refer to the circumferential velocity or line velocity and not to the motor speed.

Parameters to be set

The parameters for trimming are located in the L_TT1P_scPar_WinderDancerCtrl[Base/State/High] (📖 21) parameter structure.

```
lrLineJerk    : LREAL := 10000; // Jerk [mm/s^3]
lrTrimLineVel : LREAL := 100;   // Velocity [mm/s]
lrTrimLineAcc : LREAL := 100;   // Acceleration [mm/s^2]
lrTrimLineDec : LREAL := 10;    // Deceleration [mm/s^2]
```

3.20 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 *lrActDancerPosIn* input.

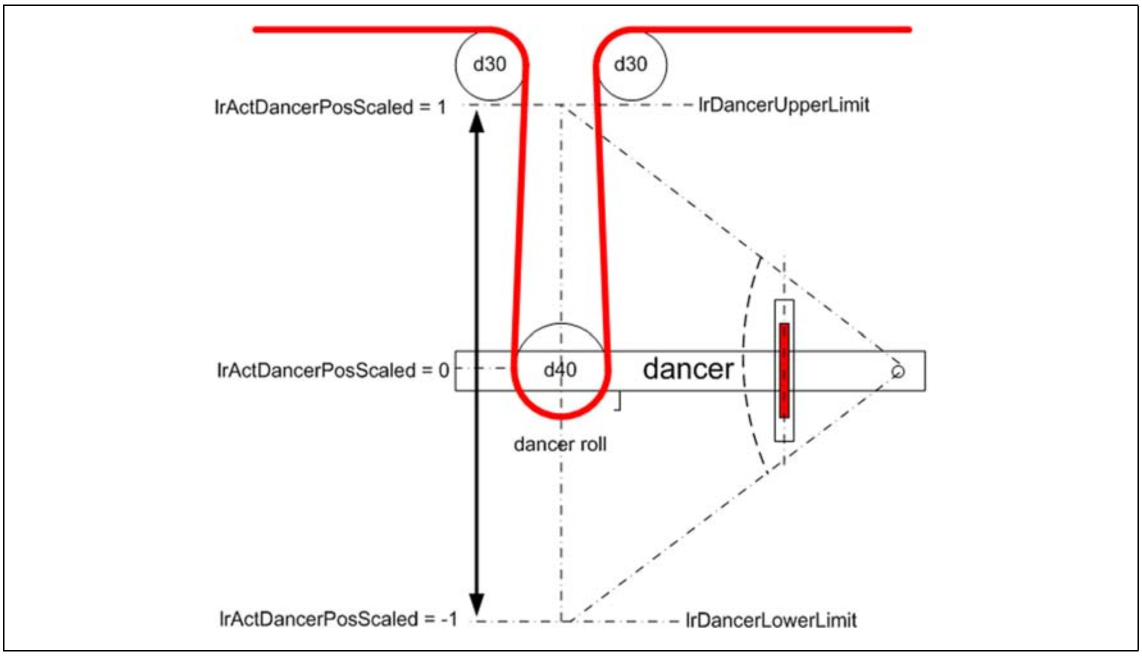
The dancer position is filtered via a PT1 filter with the *rFiltTimeActDancerPosIn* time constant. The filtered position is scaled from -1 to 1 and provided at the *lrActDancerPosScaled* output. This makes the selection of the setpoint and the monitoring of the dancer position easier.

The limit values for the upper and lower dancer position (dancer limit positions) are defined via the parameters *lrDancerUpperLimit* and *lrDancerLowerLimit*. The source for the dancer position limitations is determined by the *xTeachDancerLimits* parameter.

Parameters to be set

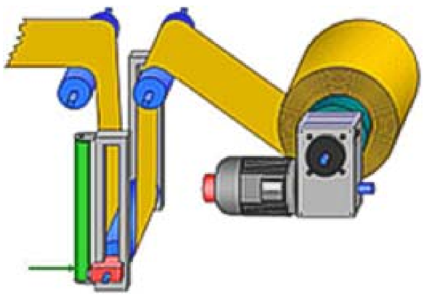
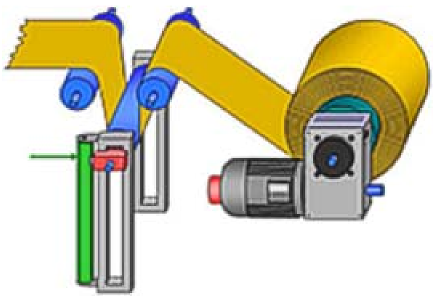
The parameters for scaling the dancer position are located in the L_TT1P_scPar_WinderDancerCtrl[Base/State/High] (📖 21) parameter structure.

```
rFiltTimeActDancerPosIn : REAL := 0.005;  // Filter time for lrActDancerPosIn [s]
lrDancerUpperLimit : LREAL := 10000000;  // Mechanical upper limit dancer position
lrDancerLowerLimit : LREAL := 1;          // Mechanical lower limit dancer position
xTeachDancerLimits : BOOL  := FALSE;     // Source for dancer position limiting
```



[3-10] Scaling of the dancer position

3.21 Teaching function for dancer limit positions

Dancer in the lower limit position	Dancer in the upper limit position
	

How to execute the teaching manually:

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

   The current input value *lrActDancerPosIn* is saved and written into the retain/persistent memory if this is interconnected via the *PersistentVar* input.
3. Move the dancer manually to the upper limit position so that a minimum of material is saved in the dancer.
4. Set the *xTeachUpperPos* input = TRUE.

   The current input value *lrActDancerPosIn* is saved and written into the retain/persistent memory if this is interconnected via the *PersistentVar* input.

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

The *xTeachDancerLimits* parameter serves to select the source of the dancer position limitations:

Parameter value	Description
TRUE	The limit positions **lrDancerLowerLimit** and **lrDancerUpperLimit** 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 **lrDancerLowerLimit** and **lrDancerUpperLimit** are used.

Parameters to be set

The parameters for the teaching function are located in the L_TT1P_scPar_WinderDancerCtrl[Base/State/High] (📖 21) parameter structure.

```
lrDancerUpperLimit : LREAL := 10000000;  // Mechanical upper limit dancer position
lrDancerLowerLimit : LREAL := 0;         // Mechanical lower limit dancer position
xTeachDancerLimits : BOOL  := FALSE;     // Source for dancer position limiting
```

3.22 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

The parameters for monitoring the dancer position are located in the L_TT1P_scPar_WinderDancerCtrl[Base/State/High] (📖 21) parameter structure.

```
lrDancerMaxPosScaled : LREAL := 0.95;  //Upper limit of dancer position 0..1 [x100 %]
lrDancerMinPosScaled : LREAL := -0.95; //Lower limit of dancer position 0..-1 [x100 
%]
lrDancerInPosWindowScaled : LREAL := 0.2; //Window of dancer in position [x100 %]
```

3.23 PI controller for the dancer position control

Activating the dancer position control with the *xDancerCtrl* input = TRUE, causes the control to an actual dancer position value.

The *lrDancerPosInfluenceScaled* input serves to define the influence of the PI controller on the motor control.

The signal of the current dancer position can be filtered with a PT1 characteristic. The filter time can be set with the *rFiltTimeActDancerPosIn* parameter (standard setting: 5 ms).

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

The controller gain is set with the *lrDancerPosCtrlGain* parameter.

When the dancer position control has been activated, the dancer has to be brought in the setpoint position. In order that the dancer will be lifted in a controlled way, the ramp generator is loaded for the position setpoint with the current actual position value. The ramp is set with the *lrDancerPosRamp* parameter (standard setting: 1 = 100 %/s). Thus, the influence of the dancer controller does not need to be displayed.

When the input *xResetICtrl* = 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

The parameters for the PI controller and the dancer position control are located in the L_TT1P_scPar_WinderDancerCtrl[Base/State/High] (📖 21) parameter structure.

```
rFiltTimeActDancerPosIn : REAL := 0.005  // PT1 filter time for lrActDancerPosIn [s]
lrDancerPosRamp : LREAL := 1;             // Ramp of lrSetDancerPosScaled [x100 %/s]
lrDancerPosCtrlGain : LREAL := 1;          // Gain of dancer position control
lrDancerPosCtrlResetTime : LREAL := 0;     // Reset time of dancer position control
```

3.24 Tension control via characteristic function (Base version)

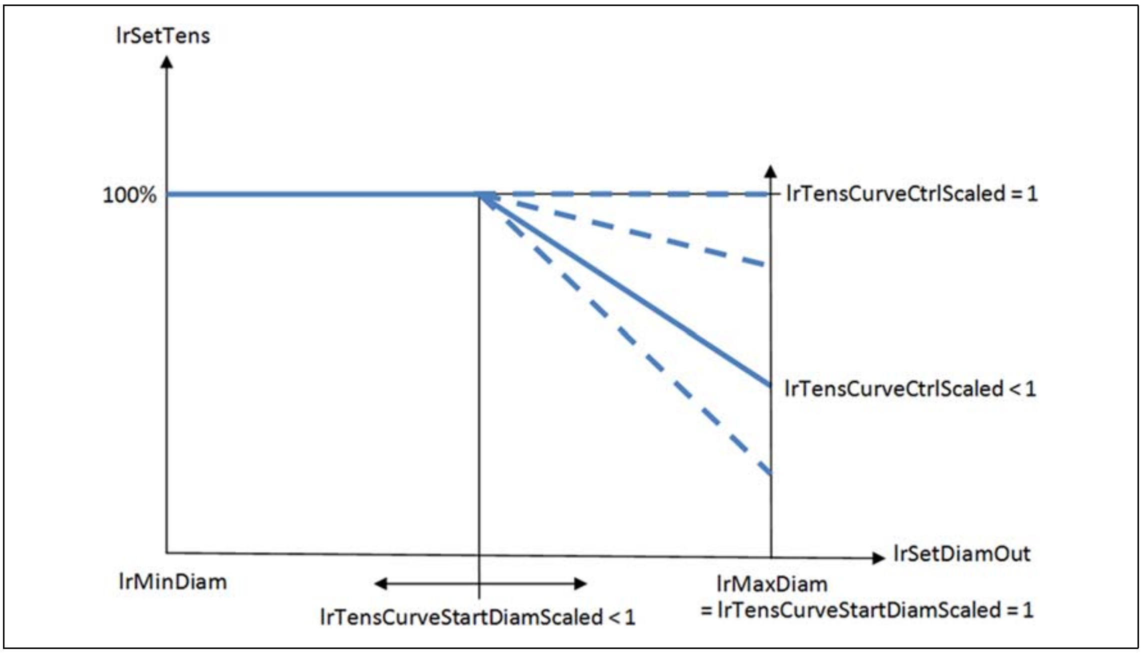
Depending on the surface and the type of the winding material, many rewinding processes require a reduction of the tensile force with increasing diameter in order to prevent a shift of the reel. This is referred to as a winding characteristic or tensile force characteristic. The dancer-position-controlled winder has no direct influence on the web tension. It is determined by the pressure or the weight at the dancer.

Influencing the tensile force in the winder control is, however, a common practice in order to apply the adapted setpoint to, e.g., a pneumatic actuator.

In order that the material-dependent characteristic is reached, the real tensile force setpoint from the *lrSetTens* input is evaluated dependent on the diameter via a linear characteristic function.

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

The *lrTensCurveStartDiamScaled* parameter serves to define the diameter from which on the tensile force is to be reduced. The *lrTensCurveCtrlScaled* parameter is used to evaluate the tensile force at the maximum diameter.



[3-11] Characteristic for a linear tensile force profile

Parameters to be set

The parameters for the "tensile force via characteristic function" are located in the L_TT1P_scPar_WinderDancerCtrl[Base/State/High] (📖 21) parameter structure.

```
lrTensCurveStartDiamScaled : LREAL := 0; // Initial point of the characteristic line
                                            [x100 %]
lrTensCurveCtrlScaled : LREAL := 0;      // Gapping of the characteristic line [x100
%]
lrMaxDiam : LREAL := 0;                   // Maximal diameter [mm]
lrMinDiam : LREAL := 0;                   // minimum diameter [mm]
```

3.25 Web break monitoring

The technology module provides two options for web break monitoring:

A. In case of a web break, the calculated diameter changes contrary to the winding direction (unwinding or rewinding).
   Monitoring is activated when the *xWebBreakMonit* input = TRUE and the *xWebBreakMode* parameter = 2. Thus, a diameter change contrary to the winding direction is only permissible within 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.

> **Note!**
>
> **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.

B. Evaluation of the dancer position:
   A web break is detected when the dancer position reaches the lower limit position.
   Monitoring is activated with the *xWebBreakMonit* input = TRUE and the *xWebBreakMode* parameter = 1.
   Responses to a web break:
   - Holding the current diameter.
   - Setting the *xWebBreak* output = TRUE.

Parameters to be set

The parameters for web break monitoring are located in the L_TT1P_scPar_WinderDancerCtrl[Base/State/High] (📖 21) parameter structure.

```
lrWebBreakWindow : LREAL :=  0.1; // Window for web break 0..1 [x100 %]
wWebBreakMode : WORD := 0;        // Select web break mode:
            0: Use Dancer Position and lrWebBreakWindow for detecting the Web break.
               1: Use only dancer position for detecting the Web Break.
               2: Use only lrWebBreakWindow for detecting the Web Break.
```

3.26 Persistent variables

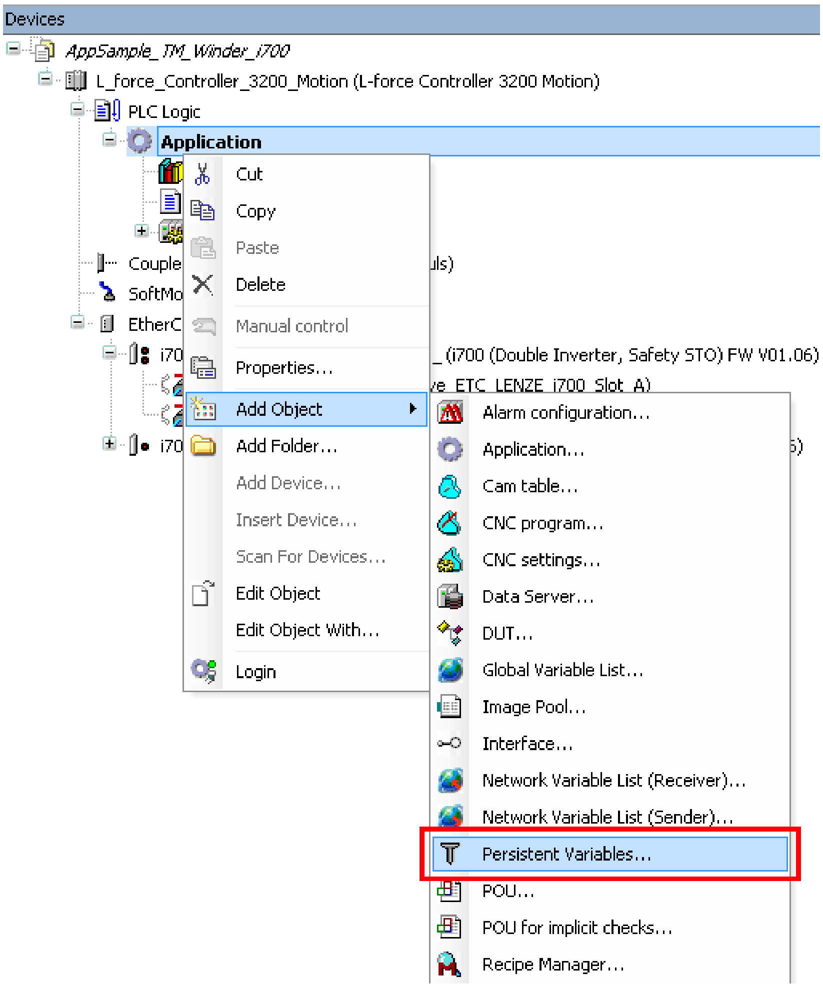
The technology module provides the possibility of persistently saving the parameters determined such as the winding diameter. For this purpose, the following setting has to be made in the »PLC Designer« for the technology module.

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

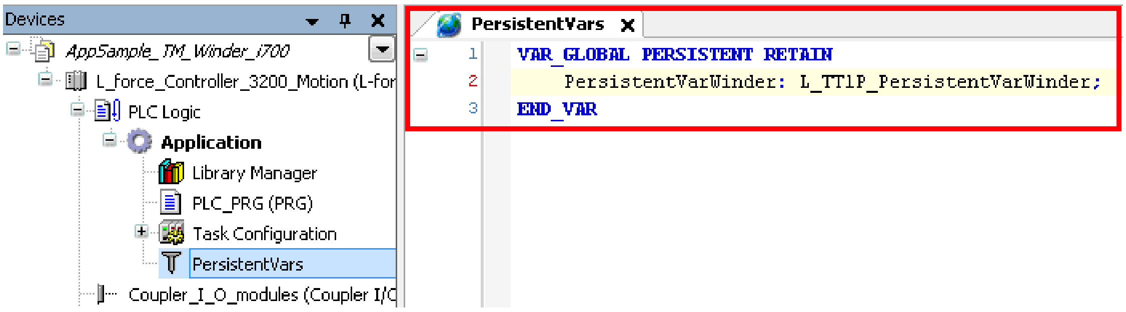
Note!

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

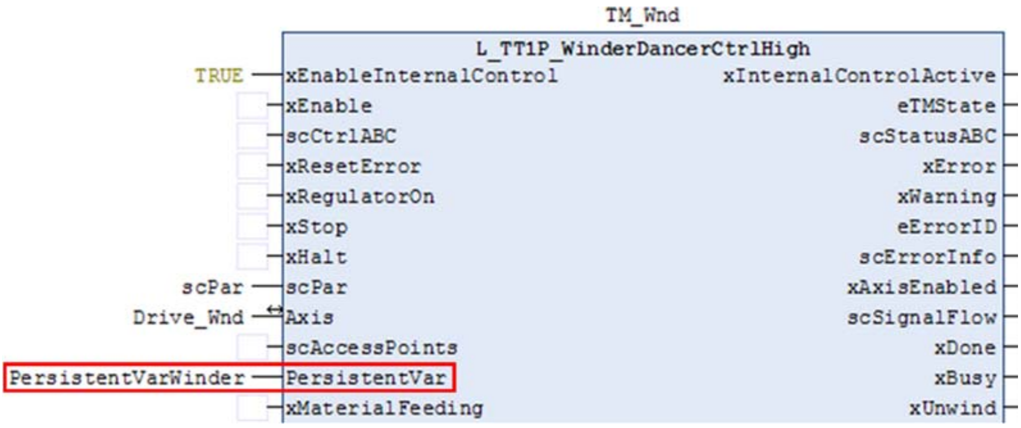
1. Add the global list for managing persistent variables in the context menu to **Application** with the **Add object → Persistent variables...** command.



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.



3.27 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 δ	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

In this connection, the change of the speed value (new value – old value) corresponds to the acceleration of the winder. The winder speed is calculated from the line velocity.

In practice, it must be expected that the line velocity signal will not be ideally constant. The parameters *lrAccCmpsGainAcc* and *lrAccCmpsGainDec* can be used to set the resolution of the signal to be differentiated. Additionally, the signal first can be smoothed via a PT1 functionality. The PT1 time constant is set via the *rFiltTimeAccSpd* parameter. For noise suppression, a lagging range can be interconnected via the calculated acceleration torque. The lagging range is set via the *lrAccCmpsDeadBandTrqScaled* parameter in the unit [x 100 %].

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

In order to generate acceleration, the line velocity must be differentiated. Depending on the resolution and the stability of the signal, it may be required to reduce the sensitivity of the differentiation to prevent that master value fluctuations cause sudden acceleration changes. Different material widths or material densities can be taken into account as percentage via the *lrMInertiaAdapt* input.

Defining moments of inertia

Note!

The defined mass inertia has to refer to the winder shaft and not to the motor shaft.

The inertia (J) of motor shaft to the winder shaft can be calculated using the following equation:

Calculation of the inertia (J) from the motor shaft to the winder shaft

$$J_{Winder} = i^2 \cdot J_{Motor}$$

with

$$i = \frac{n_{Motor}}{n_{Winder}}$$

Symbol	Description	Dimension unit
J_{Winder}	Moment of inertia of the winder shaft	$kgcm^2$
J_{Motor}	Moment of inertia of the motor shaft	$kgcm^2$
i	Gearbox factor	
n_{Motor}	Motor speed	revs/s
n_{Winder}	Winder speed	revs/s

The moment of inertia consists of a constant part and a diameter-dependent part. The constant part is mainly determined by the moment of the inertia of the motor (code C00273/1 for Lenze motors). The variable part is determined by the diameter as well as the maximum and constant moment of inertia in the technology module.

The constant moment of inertia is set using the *lrConstMInertia* parameter.

The maximum moment of inertia (full reel) is set via the *lrMaxMInertia* parameter.

Parameters to be set

The parameters for the acceleration compensation are located in the L_TT1P_scPar_WinderDancerCtrl[Base/State/High] (📖 21) parameter structure.

```
rFiltTimeAccSpd : REAL := 0.005;  // Filtertime ActReelSpeed during AccComp [s]
lrAccCmpsDeadBandTrqScaled : LREAL := 0.10;  // Dead-band of winder torque [Nm]
lrAccCmpsGainAcc : LREAL := 1.05;  // [x100 %]
lrAccCmpsGainDec : LREAL := 0.95;  // [x100 %]
lrConstMInertia : LREAL := 9;  // Constant MInertia J_min [kgcm^2]
lrMaxMInertia : LREAL := 50;   // Maximum MInertia J_max [kgcm^2]
```

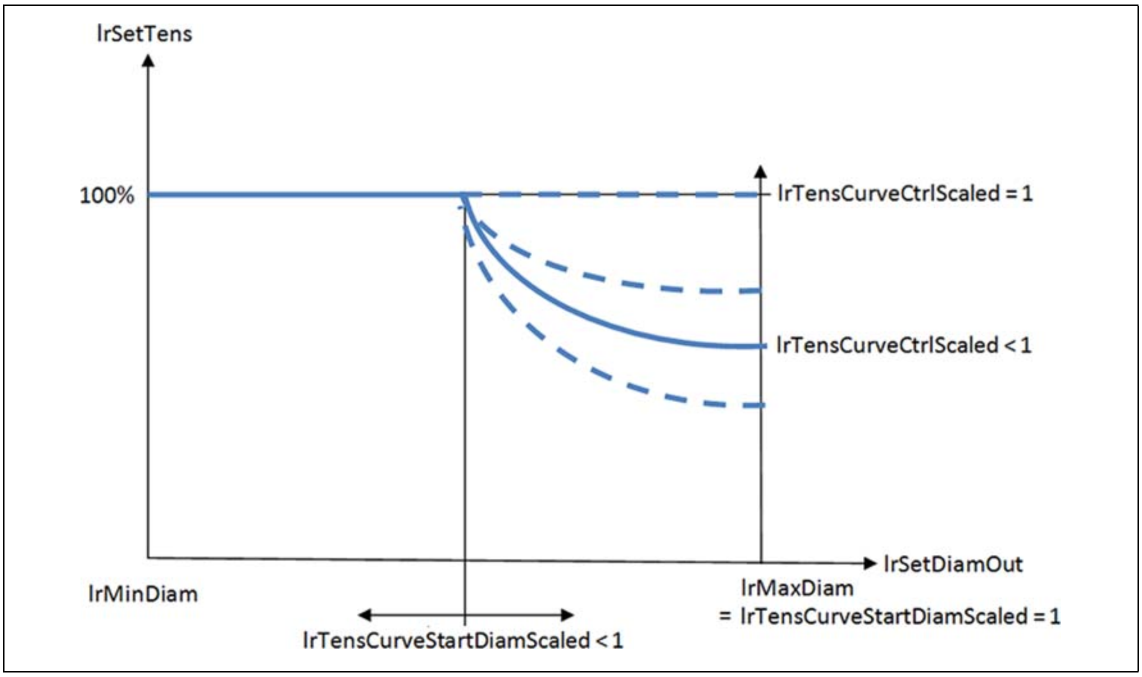
3.28 Tension control via characteristic function/winding characteristic

The characteristic function for tensile control is extended in the State version. In order that the material-dependent characteristic is reached, the real tensile force setpoint from the *lrSetTens* input is evaluated dependent on the diameter via a characteristic function.

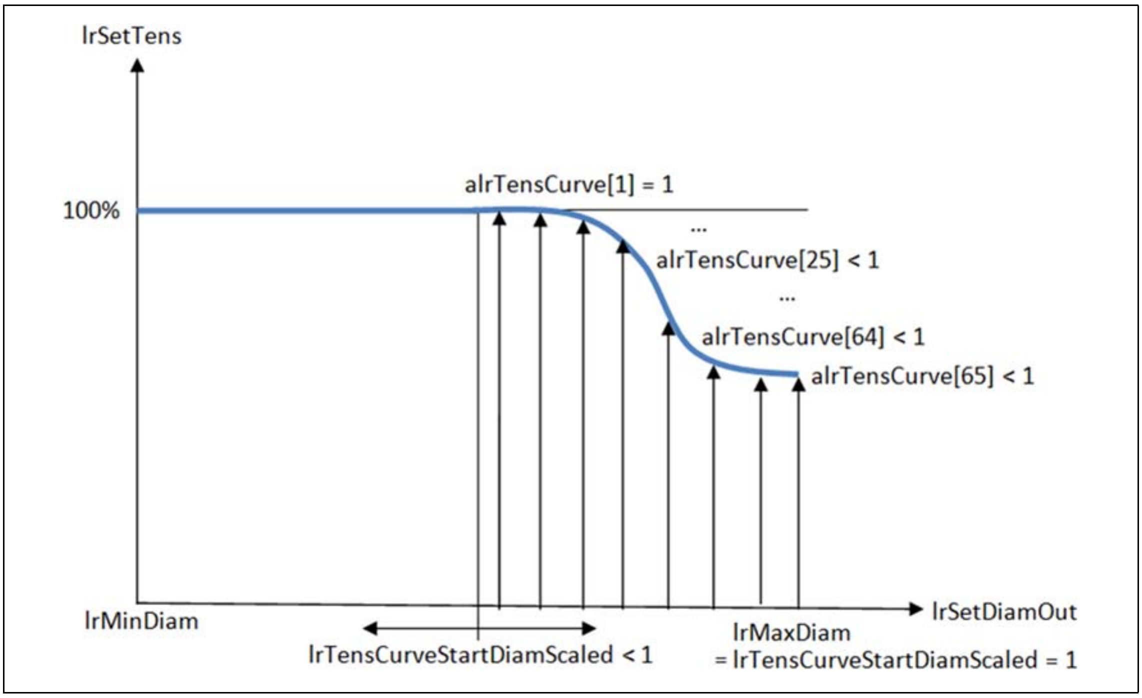
For the adaptation, different methods are available:

- Characteristic for a linear tensile force profile (*dwSelectTensCurve* = 0)
- Characteristic for a linear torque profile (*dwSelectTensCurve* = 1)
- User-definable characteristic with 64 grid points (*dwSelectTensCurve* = 2)

The characteristic has an initial area with constant evaluation (100 %) and a second area in which the tensile force is adapted to the diameter. The *lrTensCurveStartDiamScaled* parameter is used to specify the diameter from which on the tensile force will be reduced. The *lrTensCurveCtrlScaled* parameter serves to specify the percentage of the tensile force at the maximum diameter.



[3-12] Characteristic for a linear torque profile



[3-13] Characteristic with user-definable grid points

Parameters to be set

The parameters for the characteristic function are located in the L_TT1P_scPar_WinderDancerCtrl[Base/State/High] (📖 21) parameter structure.

```
dwSelectTensCurve : DWORD := 0;            // 0..3
alrTensCurve : ARRAY[1..65] OF LREAL;      // Tension curve
lrTensCurveCtrlScaled : LREAL:= 0;         //  0 .. 1  [x100 %]
lrTensCurveStartDiamScaled : LREAL := 0;   // [x100 %]
lrMaxDiam : LREAL := 0;                    // Maximum diameter [mm]
lrMinDiam : LREAL := 0;                    // Minimum diameter [mm]
```

3.29 Identification of the moments of inertia

In order to compensate the acceleration torque, a parameterisation or identification of the constant moment of inertia (motor + gearbox + winder shaft) and the maximum moment of inertia (with full reel) is required.

Identification of the constant moment of inertia

The winder shaft is empty (no material).

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

The *lrIdentMInertiaMaxTrqScaled* 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 *xExecuteIdentMInertia* input. The *lrIdentMInertia* output displays the detected moment of inertia.

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

The mass moment of inertia is detected with a rising edge (FALSE↗TRUE) at the *xExecuteIdentMInertia* 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 *lrIdentMInertia* 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.

Note!

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

Transfer the values of the identified moments of inertia into the L_TT1P_scPar_WinderDancerCtrl[Base/State/High] (📖 21) parameter structure.

Parameters to be set

The parameters for the identification of the moment of inertia are located in the L_TT1P_scPar_WinderDancerCtrl[Base/State/High] (21) parameter structure.

```
rFiltTimeIdentMInertiaSpd : REAL := 0.01;  // Filter time ActReelSpeed during Ident
                                              MInertia 1 = 1[s]
rFiltTimeIdentMInertiaTrq : REAL := 0.005; // Filtertime ActTorque during Ident
                                              MInertia 1 = 1[s]
lrIdentMInertiaMaxSpdScaled : LREAL := 0.2; // Max Ident Speed [x 100 %]
lrIdentMInertiaMaxTrqScaled : LREAL := 0.2; // Max Ident Torque [x 100 %]
```

Calculation of the maximum moment of inertia

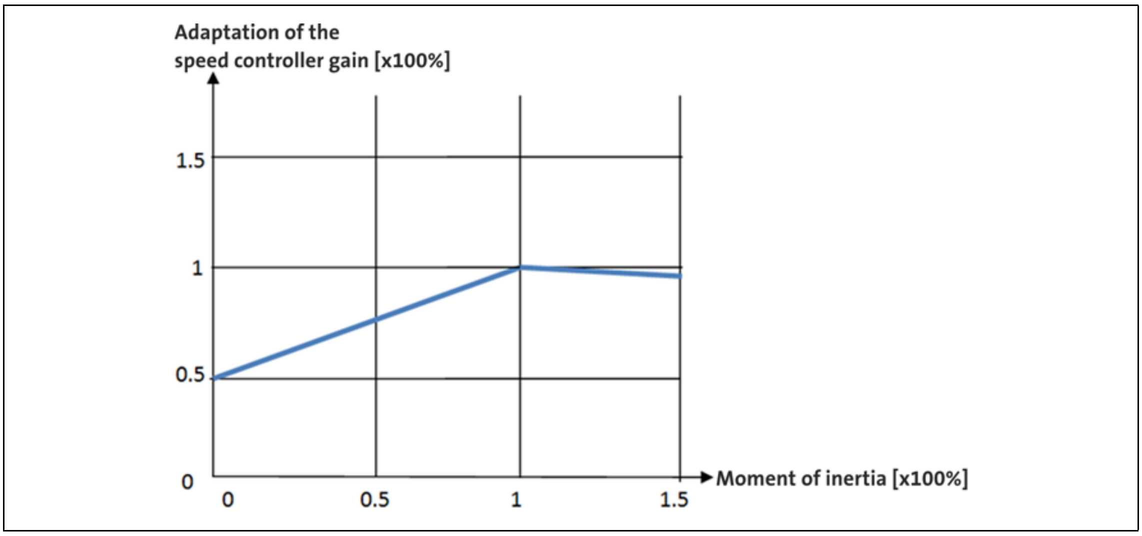
If no reel is available, the maximum moment of inertia can be calculated as follows:

Calculation of the maximum moment of inertia		
The density of the winding material is known: $J_{MaxWinder} = i^2 \cdot J_{Motor} + \left(\frac{\pi}{32 \cdot 10^8}\right) \cdot B \cdot \rho \cdot (d_{max}{}^4 - d_{min}{}^4)$ The mass of the winding material is known: $J_{MaxWinder} = i^2 \cdot J_{Motor} + \frac{m \cdot d_{max}{}^2}{800}$		
Symbol	Description	Dimension unit
$J_{MaxWinder}$	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
r	Material density	kg/dm^3
d_{max}	Maximum diameter	mm
d_{min}	Minimum diameter (sleeve diameter)	mm
m	Mass	kg

3.30 Adaptation of the speed controller gain

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 usually changes significantly during the winding process, for a good control response it may be necessary to continuously adapt the gain of the speed controller to the moment of inertia.



[3-14] Preset characteristic function for adapting the speed controller gain

The standard setting of this characteristic function is based on the following functions:

- Lower limitation of the adaptation: 50 %
- Upper limitation of the adaptation: 100 %
- Linear increase of the gain up to 100 % of the moment of inertia

Parameters to be set

The parameters for adapting the speed controller gain are located in the L_TT1P_scPar_WinderDancerCtrl[Base/State/High] (📖 21) parameter structure.

```
alrSpdCtrlGainAdaptX : ARRAY[1..9] OF LREAL :=
                       [0.0, 0.2, 0.4, 0.6, 0.8, 1.0, 1.2, 1.4, 1.6];
alrSpdCtrlGainAdaptY : ARRAY[1..9] OF LREAL :=
                       [0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 0.98, 0.95, 0.95];
```

3.31 System deviation within the reduced sensitivity

A reduced controller dynamics at low system deviations serves to have a positive effect on the damping behaviour of the control loop.

The system deviation results from the difference of the values from input *lrSetDancerPosScaled* and output *lrActDancerPosScaled*.

The *lrReducedGainWindow* parameter serves to set a tolerance zone in which the system deviation is transmitted to the controller with a lower gain. The tolerance zone is set above and below around the setpoint (*lrSetDancerPosScaled* input).

The *lrReducedGain* parameter serves to set the value to which the gain is to be reduced in the defined tolerance zone. This means, in the tolerance zone, the reduced gain is active (*lrReducedGain*).

Parameters to be set

The parameters for the characteristic function are located in the L_TT1P_scPar_WinderDancerCtrl[Base/State/High] (📖 21) parameter structure.

```
lrReducedGain : LREAL := 0;
lrReducedGainWindow : LREAL := 0;
```

3.32 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
     The axis travels to the target position *lrPos_Dist* with the velocity (*lrVel*), acceleration (*lrAcc*), deceleration (*lrDec*) and the jerk (*lrJerk*.
   - *eDancerCtrlStopMode* = 2: Move Rel
     The axis is brought to a standstill with the velocity (*lrVel*), acceleration (*lrAcc*), deceleration (*lrDec*) and the jerk (*lrJerk*) after the distance travelled (*lrPos_Dist*).

   State change: DANCERCTRL ==> STOP

Parameters to be set

The parameters to be set for terminating the winding process are located in the L_TT1P_scPar_WinderDancerCtrl[Base/State/High] (📖 21) parameter structure.

```
eTpMode : L_TT1P_TpMode := 0;
lrPos_Dist : LREAL :=0;
lrHaltDec : LREAL := 3600;
lrJerk : LREAL := 100000;
lrVel : LREAL := 50;
lrAcc : LREAL := 100;
lrDec : LREAL := 100;
```

3.33 CPU utilisation (example Controller 3231 C)

The following table shows the CPU utilisation in microseconds using the example of the 3231 C controller (ATOM™ processor, 1.6 GHz).

Versions	Interconnection of the technology module	CPU utilisation	
		Average	Maximum peak

Base	xEnable := TRUE;
	xRegulatorOn := TRUE;
	xSyncLineVel := TRUE;
		75 µs	106 µs

State	xEnable := TRUE;
	xRegulatorOn := TRUE;
	xSyncLineVel := TRUE;
		90 µs	125 µs

High	xEnable := TRUE;
	xRegulatorOn := TRUE;
	xSyncLineVel := TRUE;
		100 µs	137 µs

Index

S

T

V

W

FEEDBACK

Your opinion is important to us

These instructions were created to the best of our knowledge and belief to give you the best possible support for handling 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

Lenze Automation GmbH
Postfach 10 13 52, 31763 Hameln
Hans-Lenze-Straße 1, 31855 Aerzen
GERMANY
HR Hannover B 205381
+49 5154 82-0
+49 5154 82-2800
lenze@lenze.com
www.lenze.com

Service
Lenze Service GmbH
Breslauer Straße 3, 32699 Extertal
GERMANY
008000 24 46877 (24 h helpline)
+49 5154 82-1112
service@lenze.com