Lenze

Manual

Global Drive

Software Package - Winder

Function library

LenzeWinder.lib

The function library **LenzeWinder.lib** can be used for the following Lenze PLCs:

	Type	as from hardware version	as from software version
9300 Servo PLC	EVS93XX-xT	2K	1.3

Important note :

The software is supplied to the user as described in this document. Any risks resulting from its quality or use remain the responsibility of the user. The user must provide all safety measures protecting against possible maloperation.

We do not take any liability for direct or indirect damage, e.g. profit loss, order loss or any loss regarding business.

All information given in this online documentation has been carefully selected and tested for compliance with the hardware and software described. Nevertheless, discrepancies cannot be ruled out. We do not accept any responsibility or liability for any damage that may occur. Required corrections will be included in updates of this documentation.

Version 1.1 02/2003 TD11

Function library LenzeWinder.lib

Contents

1 Preface and general information

1.1 About this Manual

This Manual contains information about the function library **LenzeWinder.lib** for the **Drive PLC Developer Studio**.

- The function library **LenzeWinder.lib** contains functions necessary for winding tasks.
- The function library is a part of the **Software Package - Winder**.

Tip!

The version of the function library can be found under the global constant `C_wLenzeWinderVersion` of the function library **LenzeWinder.lib**.

- The value of this constant is hex-coded. “16#0130” means, for example, “version 1.3”.

1.1.1 Conventions used in this Manual

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

Variable identifiers

are written in italics in the explanation:

- “Use *bReset* ...”

Tip!

Information about the conventions used for the variables of Lenze system blocks, function blocks and functions can be obtained from the appendix of the DDS online documentation “Introduction into IEC 61131-3 programming”. The conventions ensure universal and uniform labelling and make reading the PLC program easier.

Lenze functions/function blocks

can be recognized by their names. They always begin with an “L_”:

- “The FB **L_WndCalcDiameter**...”

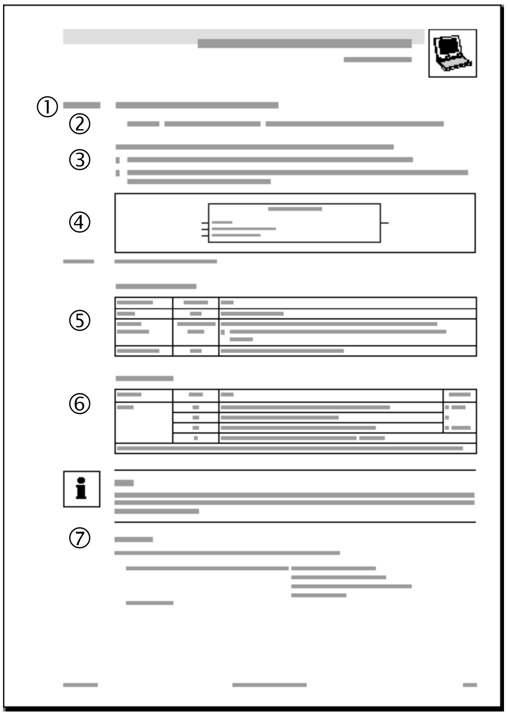
Program listings

are written in “Courier”, keywords are printed in bold:

- “ `IF (ReturnValue < 0) THEN`...”

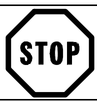
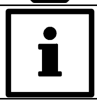
1.1.2 Layout of the descriptions

All function/function block and system block descriptions contained in this Manual have the same structure:



	Function	Function block (FB)/ System block (SB)
①	Heading stating function and function identifier	
②	Declaration of the function: • Data type of the return value • Function identifier • List of transfer parameters	-
③	Short description of the most important properties	
④	Function chart including all associated variables • Transfer parameters • Return value	FB/SB chart including all associated variables • Input variables • Output variables
⑤	Table giving information about the transfer parameters: • Identifier • Data type • Possible settings • Info	Table giving information about the input and output variables: • Identifier • Data type • Variable type • Possible settings • Info
⑥	Table giving information about the return value: • Data type of the return value • Possible return values and their meaning:	-
⑦	Additional information (Comments, tips, application examples, etc.)	

1.1.3 Pictographs in this Manual

	Pictographs	Signal words	
Material damage warning		Stop!	Warns of **potential damage to material**. Possible consequences if disregarded: Damage to the controller/drive system or its environment.
Other notes		Tip! Note!	Indicates a tip or note.

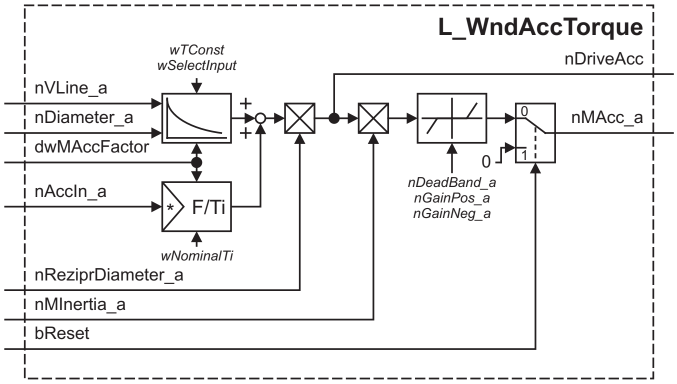
1.1.4 Definitions

Term	In the following text used for
DDS	Drive PLC Developer Studio
FB	Function block
GDC	Global Drive Control (Lenze parameterization program)
Parameter codes	Codes for setting the functionality of a function block
SB	System block

2 Function blocks

2.1 L_WndAccTorque - Acceleration compensation

This FB can be used to generate a precontrol torque for acceleration torque compensation.



⇨ Inputs	(Variable type: VAR_INPUT)		
Identifier	**Data type**	**Signal type**	**Information/possible settings**
nVLine_a	Integer	analog	Current line speed • Referred to the maximum value *wVMax* • 16384 ≡ 100 % *wVMax* • Max. rate of change = 0.5 %/ms
nDiameter_a	Integer	analog	Actual diameter for the calculation of the variable acceleration torque • 16384 ≡ max. diameter • Value is internally limited to 0 ... 16384. • Max. rate of change = 0.5 %/ms
dwMAccFactor	Double word	-	Normalization factor for acceleration torque calculation • Max. moment of inertia applies to the motor side. • Value is internally limited to 10 000 000.
nAccIn_a	Integer	analog	Precontrol signal for acceleration torque
nReziprDiameter_a	Integer	analog	Reciprocal of the current diameter • Referred to *wDMin*. • Value is internally limited to 0 ... 16384.
nMInertia_a	Integer	analog	Current moment of inertia • Referred to the nominal value (Jmax). • Value is internally limited to 0 ... 32767.
bReset	Bool	binary	Reset of acceleration torque TRUE: Resets the acceleration torque to zero

Outputs ⇨	(Variable type: VAR_OUTPUT)		
Identifiers	**Data type**	**Signal type**	**Value/meaning**
nDriveAcc	Integer	-	Display of current motor acceleration
nMAcc_a	Integer	analog	Current acceleration torque (for compensation) • Referred to the max. acceleration torque (16384 ≡ 100 %). • With **9300 Servo PLC**: 16384 ≡ 100 % of *MCTRL_wMmaxC57*

Parameters	(Variable type: VAR_CONSTANT_RETAIN)				
Identifier	**Data type**	**Signal type**	**Info**	**Possible settings**	**Presetting**
wTConst	Word	-	Filter time constant for line speed and diameter differentiation	10 {1 msec} 30000	100 msec
wSelectInput	Word	-	Selection for calculating the motor acceleration	0 Differentiation deactivated. 1 Differentiation of *nVLine_a* . 2 Differentiation of *nVLine_a* and *nDiameter_a*.	0
wNominalTi	Word	-	Nominal ramp time as a reference for acceleration torque control	0.200 {0.001 sec} 30.000	10.000 s
nDeadBand_a	Integer	analog	Dead band for current acceleration torque • 100 % ≡ 16384	0.00 {0.01 %} 100.00	1.00 %
nGainPos_a	Integer	analog	Gain for positive acceleration torque • Gain factor 1 ≡ 100 % ≡ 16384	-199.99 {0.01 %} 199.99	100.00 %
nGainNeg_a	Integer	analog	Gain for negative acceleration torque • Gain factor 1 ≡ 100 % ≡ 16384	-199.99 {0.01 %} 199.99	100.00 %

The acceleration torque is calculated from the current moment of inertia and the speed change.

- For stability reasons, do not differentiate the actual speed signal for the calculation but evaluate the line speed change.
- If the diameter changes comparatively quickly, it might be necessary to consider the diameter change as well.

$$M_{acc} = dwMAccFactor \cdot \left(\frac{dV_{line}}{dt} - \frac{V_{line}}{D} \cdot \frac{dD}{dt} \right) \cdot \frac{1}{D} \cdot \left(J_{machine} + J_{reel} \right)$$

Fig. 2-1 Formula for acceleration torque calculation (with normalization factor)

If the variables are referred to the nominal value, the normalization factor *dwMAccFactor* is calculated as follows:

$$dwMAccFactor = \frac{1}{3} \cdot \frac{J_{max} \cdot V_{max} \cdot i}{M_{max} \cdot D_{min}}$$

Fig. 2-2 Formula for calculating the normalization factor

J_{max} = max. moment of inertia in [0.1 kg cm^2]
V_{max} = max. line speed in [0.1 m/min]
i = gearbox factor
M_{max} = max. motor control torque in [0.1 Nm]
D_{min} = min. reel diameter in [mm]

- The normalization factor *dwMAccFactor* can also be calculated through the FB **L_WndMInertia**.

Material width and material density

With low material widths and densities, the acceleration torque is directly reduced by reducing the current moment of inertia.

Calculation of the drive acceleration

The acceleration of the drive is calculated by differentiating the input signals *nVLinie_a* and *nDiameter_a* and multiplying the result by 1/D (see formula in Fig. 2-1).

- By setting *wSelectInput* to ”2” you can activate the influence of the diameter-dependent component.
- A low-pass filter follows differentiation to damp the effects of the ”noise” of the input signals on the drive acceleration. The filter time constant is set via *wTConst* .
- If the line speed signal cannot be used for differentiation the drive acceleration can be directly controlled via the input *nAccIn_a* .A corresponding precontrol signal can, for example, be generated by the master. The acceleration time referred to by the precontrol signal can be set via *wNominalTi*.

Calculation of the acceleration torque

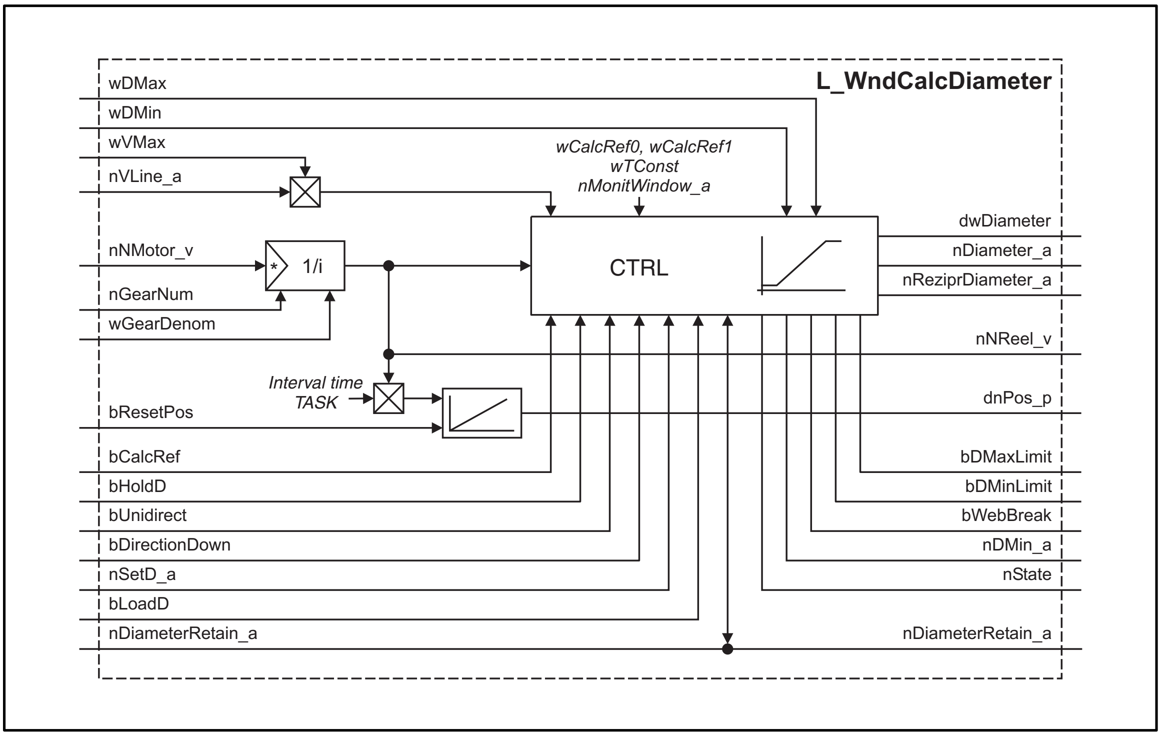
- The acceleration torque is calculated by mulitplying the drive acceleration by the current moment of inertia.
- For noise suppression, low acceleration torques can be suppressed by means of a configurable dead band (*nDeadBand_a*).
- The acceleration torque can be increased or reduced as a function of the direction of acceleration (acceleration or deceleration) by adjusting the gain settings (*nGainPos_a* and *nGainNeg_a*) accordingly.

Deactivation of the acceleration compensation

By setting *bReset* to TRUE you can deactivate the acceleration compensation, i.e. the acceleration torque is set to zero.

2.2 L_WndCalcDiameter - Diameter calculation

This FB can be used to calculate the reel diameter from the line speed and reel speed.



⇨ Inputs	(Variable type: VAR_INPUT)		
Identifier	**Data type**	**Signal type**	**Information/possible settings**
wDMax	Word	-	Max. diameter in [mm] • Value is internally limited to 0 ... 10000 mm.
wDMin	Word	-	Min. diameter in [mm] • Value is internally limited to 0 ... 10000 mm.
wVMax	Word	-	Max. line speed in [0.1 m/min] • Value is internally limited to 0 ... 3000 m/min. • 2500 ≡ 250.0 m/min
nVLine_a	Integer	analog	Current line speed • Referred to the maximum value *wVMax* • 16384 ≡ 100 % *wVMax*
nNMotor_v	Integer	Velocity	Current motor speed in [inc/ms]
nGearNum	Integer	-	Current gearbox factor (numerator) • Value is internally limited to -32767 ... -1 / 1 ... 32767.
wGearDenom	Word	-	Current gearbox factor (denominator) • Value is internally limited to 1 ... 32767.
bResetPos	Bool	binary	Reset of angle of rotation *dnPos_p* TRUE: Resets the angle of rotation
bCalcRef	Bool	binary	Selection of the integration cycle FALSE: Parameters *wCalcRef0* . TRUE: Parameters *wCalcRef1* .
bHoldD	Bool	binary	Holding of the latest diameter value TRUE: The latest diameter value output is not overwritten with new values.

Function library LenzeWinder.lib

Function blocks

L_WndCalcDiameter - Diameter calculation

➪ **Inputs**	**(Variable type: VAR_INPUT)**		
Identifier	**Data type**	**Signal type**	**Information/possible settings**
bUnidirect	Bool	binary	Enabling of only one change direction and activation of the web-break monitoring TRUE: Only the change direction selected via *bDirectionDown* is enabled.
bDirectionDown	Bool	binary	Selection of the enabled change direction • Only to be used with *bUnidirect* = TRUE FALSE: Clock-wise rotation (cw) TRUE: Counter-clockwise rotation (ccw)
nSetD_a	Integer	analog	Selection of an initial value/external diameter signal • Referred to *wDMax*. • 16384 ≡ 100 % *wDMax* • Value is accepted by *bLoadD* = TRUE.
bLoadD	Bool	binary	Acceptance of the selected initial value TRUE: Accept value at *nSetD_a*

Outputs ➪	**(Variable type: VAR_OUTPUT)**		
Identifier	**Data type**	**Signal type**	**Value/meaning**
dwDiameter	Double word	-	Current diameter in [µm] • Value is internally limited to 10 m.
nDiameter_a	Integer	analog	Current diameter • Referred to *wDMax* . • 16384 ≡ 100 % *wDMax*
nReziprDiameter_a	Integer	analog	Reciprocal of the current diameter • Referred to *wDMin* . • 16384 ≡ 100 % *wDMin*
nNReel_v	Integer	Velocity	Current reel speed in [inc/ms]
dnPos_p	Double integer	position	Current angle of rotation of the reel in [inc]
bDMaxLimit	Bool	binary	Limit value monitoring (*wDMax*) TRUE: Limit value *wDMax* reached
bDMinLimit	Bool	binary	Limit value monitoring (*wDMin*) TRUE: Limit value *wDMin* reached
bWebBreak	Bool	binary	Web-break monitoring TRUE: Web-break after passing the monitor window
nDMin_a	Integer	analog	Minimum diameter • Referred to *wDMax* . • 16384 ≡ 100 % *wDMax*
nState	Integer	-	Status: Diameter calculation Status messages: 0: Diameter value was calculated/read from the memory 10: Diameter value was initialized with *wDMin* 20: Diameter value is/was loaded via *bLoadD* / *nSetD* 30: Diameter value is/was maintained (*bHoldD*) Error messages: -1: Input value *wDMin* ≥ *wDMax* -10: Internal reset of diameter calculation (Integration overflow through *nVLine_a*)

Inputs/outputs	(Variable type: VAR_IN_OUT)		
Identifier	**Data type**	**Signal type**	**Value/meaning**
nDiameterRetain_a	Integer	analog	Input/output for permanent saving of the current diameter • For this, declare a variable of type VAR RETAIN or VAR_GLOBAL RETAIN and connect it with the input. • The global variable always receives the current value because input and output are the same. • RETAIN variables are automatically saved during a supply interruption and automatically read after power-on.

Parameters	(Variable type: VAR_CONSTANT_RETAIN)						
Identifier	**Data type**	**Signal type**	**Info**				
			Possible settings				**Presetting**
wCalcRef0	Word	-	Number of revolutions after which a new diameter calculation will be made. • Selected by *bCalcRef* = FALSE.				
			0.001	{0.001 rev}	2.000		1.000 rev
wCalcRef1	Word	-	Number of revolutions after which a new diameter calculation will be made. • Selected by *bCalcRef* = TRUE.				
			0.001	{0.001 rev}	2.000		0.100 rev
wTConst	Word	-	Filter time constant for calculated diameter values				
			0.010	{0.01 sec}	3.000		1,000 s
nMonitWindow_a	Integer	analog	Permissible diameter change in opposite direction (web-break monitoring) • Only relevant if *bUnidirect* = TRUE. • Referred to *wDMax*. • 16384 ≡ 100 % *wDMax*				
			0.00	{0.01 %}	100.00		10.00 %

Setting of the initial value

At input *nSetD_a* you can select an initial value or an external diameter signal.

- This value will be accepted if *bLoadD* = TRUE.
- When the value is accepted, the diameter calculation is reset and the filter used for diameter calculation is loaded with the initial value that has been set.

Diameter calculation

For calculating the reel diameter, line speed and current reel speed have to be cyclically integrated.

- The line speed is selected via *nVLine_a*.
- The reel speed results from the motor speed *nNMotor_v* and the current gearbox factor *nGearNum* / *wGearDenom*.
- The calculation cycle is defined via *wCalcRef0* and / or *wCalcRef1* the corresponding parameter is selected via *bCalcRef*.
- At the end of every integration interval, a new diameter value results from the division of the integrator values.
- A first order low-pass is used to generate and smooth the absolute value. The filter time constant can be set via *wTConst*. The output *dwDiameter* is not filtered.

Selection of the change direction, web-break monitoring

Set *bUnidirect* to TRUE to enable only one change direction for diameter calculation and activate the web-break monitoring.

- The permissible change direction is selected under *bDirectionDown*.
- New calculation values will only be accepted if they are higher than the latest storage value in the permissible direction.

After a web break, the calculated diameter values usually change very fast into the opposite direction.

- Use *wMonitorWindow* to select the maximum permissible deviation in opposite direction with which the output *bWebBreak* will be set to TRUE when exceeded.
- If *bUnidirect* is set to FALSE both change directions are enabled and the internal memory is always overwritten with the new calculation value, the web-break monitoring is deactivated.

Holding the current value

Set *bHoldD* to TRUE to hold the latest diameter value *nDiameter_a* internally.

- The diameter calculation is reset and the filter used for diameter calculation is loaded with the held diameter value.

Saving the current value

Use the input/output *nDiameterRetain_a* for permanently saving the current diameter to ensure that the value will be available after a supply interruption or power-off.

- For this, declare a variable of type VAR RETAIN or VAR_GLOBAL RETAIN and connect it with the input.
- The RETAIN variable always receives the current value because the input is also the output.

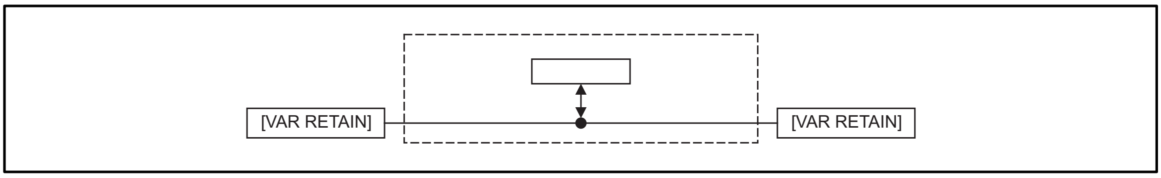


Fig. 2-3 Permanent saving of the current value in a RETAIN variable

Limit value monitoring

Use *wDMin* and *wDMax* to define the limit values for the selected diameter values with which - when reached/exceeded - the output *bDMaxLimit* / *bDminLimit*) is set to TRUE.

- You can enter limit values for the unfiltered diameter value *dwDiameter* and for the filtered diameter value *nDiameter_a*.
- The limit values are entered in [mm].
- The hysteresis for the reset of *bDMaxLimit*/ *bDminLimit* is permanently set to 1% of *wDMax*.
- For further FB parameterization, the value in *wDMin* is referred to *wDMax* and provided at the output *nDMin_a*.

Diameter conversion into 1/D

For line speed evaluation with 1/D (speed-controlled winders), the reciprocal of the reel diameter is provided via *nReziprDiameter_a*.

- this value refers to the limit value *wDMin*.

Physical units

Use *dwDiameter* to output the current diameter as a physical unit (1 LSB ≡ 1 µm).

Use *nNReel_v* to output the reel speed calculated from the motor speed *nNMotor_v* and the current gearbox factor (*nGearNum* / *wGearDenom*).

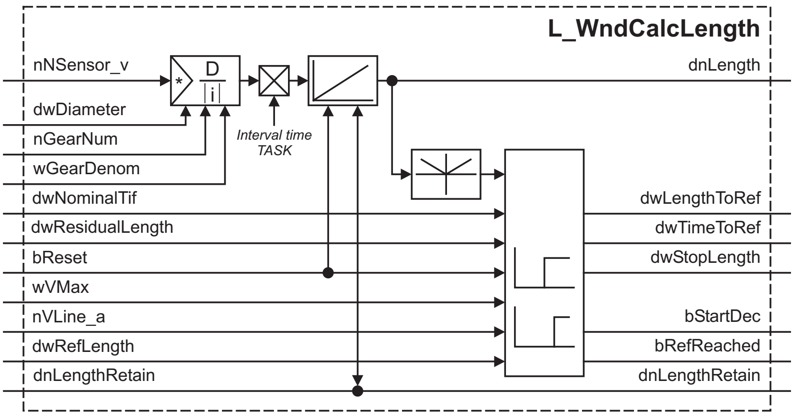
If additional functions are to be supported (e.g. a traversing control) the reel speed is also integrated.

- The angle of rotation is output by *dnPos_p*.
- The angle of rotation is reset by setting *bResetPos* to TRUE.

2.3 L_WndCalcLength - Length calculation

This FB can be used to make a continuous length calculation from the speed and the diameter of the take-up roller.

- If a gearbox is connected between the take-up roller and the speed encoder the sensor speed is converted in accordance with the gearbox factor into the speed of the take-up roller.
- The output signal *dnLength* has a resolution of 1 mm and can represent a maximum of 2000 km.



⇨ Inputs	(Variable type: VAR_INPUT)		
Identifier	**Data type**	**Signal type**	**Information/possible settings**
nNSensor_v	Integer	Velocity	Encoder speed in [inc/ms] • Value is internally limited to -16384 ... 16384.
dwDiameter	Double word	-	Running wheel/roller diameter in [µm] • Value is internally limited to 10 m. • 1000 ≡ 1 mm
nGearNum	Integer	-	Gearbox factor (numerator) of the gearbox between running wheel/rolelr and speed encoder • Value is internally limited to -32767 ... -1 / 1 ... 32767.
wGearDenom	Word	-	Gearbox factor (denominator) of the gearbox between running wheel/roller and speed encoder • Value is internally limited to 128 * |*nGearNum*| in the range of 1 ... 32767.
bReset	Bool	binary	Reset of length detection TRUE: Resets the length detection.
wVMax	Word	-	Max. line speed in [0.1 m/min] • Value is internally limited to 0 ... 3000 m/min • 2500 ≡ 250.0 m/min
nVLine_a	Integer	analog	Current line speed • Referred to the maximum value *wVMax* • 16384 ≡ 100 % *wVMax*
dwRefLength	Double word	-	Length to be wound up during a winding process in [mm] • Value is internally limited to 2000 km. • 1000 ≡ 1 mm
dwNominalTif	Double word	-	Nominal deceleration time for braking in [ms] • Activated by the message"Pre-stop length reached" (*bStartDec* = TRUE). • Value is internally limited to 0 ... 10000 s. • 1000 ≡ 1 sec
dwResidualLength	Double word	-	Residual length to be left after a winding process in [mm] • Value is internally limited to 0 ... 1000000000 mm. • 1000 ≡ 1 m

Outputs ⇨	(Variable type: VAR_OUTPUT)		
Identifier	**Data type**	**Signal type**	**Value/meaning**
dnLength	Double integer	-	Current length in [mm]
dwLengthToRef	Double word	-	Residual length to reference value in [mm] • Value is internally limited to 2000 km.
dwTimeToRef	Double word	-	Residual operating time to reference value in [s] • On the assumption that speed = *wVMax*. • This value is only calculated every second FB call (alternating with *dwStopLength*).
dwStopLength	Double integer	-	Change of length during braking in [mm] • This value is only calculated every second FB call (alternating with *dwTimeToRef*).
bStartDec	Bool	binary	Status signal to start braking TRUE: Starts braking.
bRefReached	Bool	binary	Status signal "Reference length reached" TRUE: Reference length reached.

Inputs/outputs	(Variable type: VAR_IN_OUT)		
Identifiers	**Data type**	**Signal type**	**Value/meaning**
dnLengthRetain	Doubel Integer	-	Input/output for permanent saving of the current length • For this, declare a variable of type VAR RETAIN or VAR_GLOBAL RETAIN and connect it with the input. • The RETAIN variable always receives the current value because the input is also the output. • RETAIN variables are automatically saved during a supply interruption and automatically read after power-on.

Display of the reached length

Use *dnLength* to display the reached length as a physical unit.

- Negative values are also possible.

Display of the residual length/residual operating time

Use *dwLengthToRef* to display the residual length to the reference value.

- If the line speed = *wVMax* the residual operating time *dwTimeToRef* results from the residual length.

Activation of the braking operation

For activating the braking operation, the current length is compared with the residual length after the stop *dwResidualLength*) including the change of length during braking (*dwStopLength*).

- The change of length during braking is output at *dwStopLength* and depends on the current line speed calculated from *wVMax* and *nVLine_a* and on the nominal deceleration time *dwNominalTif* .
- The signal required to activate the braking operation is output at *bStartDec*.

Activation of a final stop

For activating a final stop, the current length is compared with the reference length.

- The signal required to activate the final stop is output at *bRefReached*.

Reset of the length calculation and the two release signals

By setting *bReset* to TRUE you can reset the length calculation and the two digital outputs for activating the braking operation/final stop.

Saving of the current value

Use the input/output *dnLengthRetain* for permanently saving the current length to ensure that the value will be available after a supply interruption or power-off:

- For this, declare a variable of type VAR RETAIN or VAR_GLOBAL RETAIN and connect it with the input.
- The RETAIN variable always receives the current value because the input is also the output.

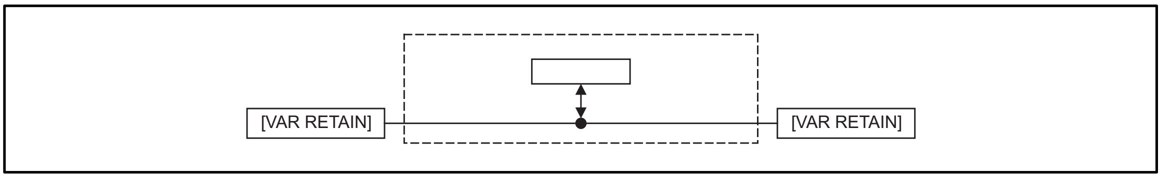
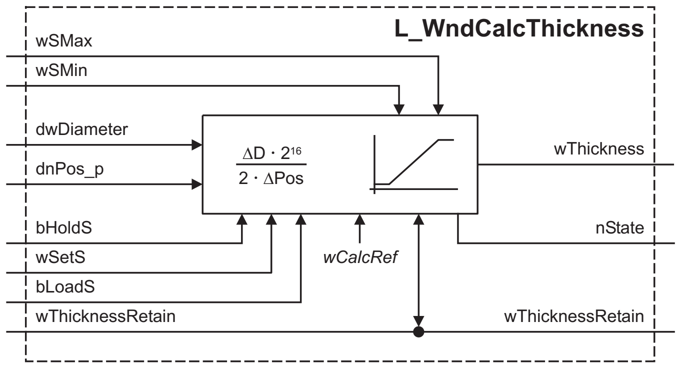


Fig. 2-4 Permanent saving of the current value in a RETAIN variable

2.4 L_WndCalcThickness - Thickness calculation

This FB can be used to calculate the mean material thickness in [µm] from the change of the reel diameter and the change of the angle of rotation of the reel shaft.

- The current diameter and angle of rotation are accepted by the FB **L_WndCalcDiameter**.
- To obtain a high resolution when calculating the material thickness, the number of layers/revolutions after which the calculation is to be made can be selected accordingly.



⇨ Inputs	(Variable type: VAR_INPUT)		
Identifier	**Data type**	**Signal type**	**Information/possible settings**
wSMax	Word	-	Max. material thickness in [µm] • Value is internally limited to 1 ... 30000 µm. • 1000 ≡ 1 mm
wSMin	Word	-	Min. material thickness in [µm] • Value is internally limited to 1 ... 30000 µm or *wSMax*. • 1000 ≡ 1 mm
dwDiameter	Double word	-	Current reel diameter in [µm] • Value is internally limited to 10 m.
dnPos_p	Double integer	position	Current angle of rotation of the reel • 2^{16} ≡ 1 rev.
bHoldS	Bool	binary	Holding the current output value TRUE: Current output value is maintained and material thickness calculation reset.
wSetS	Word	-	Start value for material thickness in [µm] • Value is internally limited to 1 ... 30000 µm.
bLoadS	Bool	binary	Start value acceptance • This input has a higher priority than *bHoldS*. TRUE: Accepts start value *wSetS* and resets material thickness calculation.

Outputs ⇨	(Variable type: VAR_OUTPUT)		
Identifier	**Data type**	**Signal type**	**Value/meaning**
wThickness	Word	-	Current material thickness in [µm] • Value is internally limited to 1 ... 30000 µm.
nState	Integer	-	Thickness calculation status 30: Material thickness is/was maintained (*bHoldS*). 20: Material thickness is/was loaded by *bLoadS* / *nSetS*. 10: Material thickness has been initialized with *wSMin*. 0: Material thickness has been calculated/read from the memory. -1: Error: Input value *wSMin* ≥ *wSMax*

Inputs/outputs	(Variable type: VAR_IN_OUT)		
Identifier	**Data type**	**Signal type**	**Value/meaning**
wThicknessRetain	Word	-	Input/output for permanent saving of the current material thickness • For this, declare a variable of type VAR RETAIN or VAR_GLOBAL RETAIN and connect it with the input. • The global variable always receives the current value because input and output are the same. • RETAIN variables are automatically saved during a supply interruption and automatically read after power-on.

Parameters	(Variable type: VAR_CONSTANT_RETAIN)			
Identifier	**Data type**	**Signal type**	**Info** **Possible settings**	**Presetting**
wCalcRef	Word	-	Number of reel shaft revolutions until next thickness calculation • When 1000 rev (special setting) are selected, the material thickness will be continuously calculated after 100 revolutions. 1 {1 rev} 1000	100 rev

Setting of the initial value

It is possible to define an initial value at the input *wSetS*, e.g. after a reel change with a different material thickness. In this way, it is possible to output an appropriate value until the next calculation will be made.

- The value at *wSetS* is accepted if *bLoadS* is set to TRUE.
- After acceptance, the next calculation cycle starts saving the current diameter and the current angle of rotation when the reel shaft has made two revolutions.

Calculation of the material thickness

After the current input values (*dwDiameter*, *dnPos_p*) have been stored, a preset number of reel shaft revolutions are made (*wCalcRef*) before the material thickness will be calculated from the diameter change.

- The current diameter and the current angle of rotation are stored after each calculation so that the next calculation cycle can start.
- With the setting *wCalcRef* = 1000 rev. the initially saved reference values for diameter and angle of rotation are maintained. The first calculation is made after 100 revolutions. After that, the calculation is repeated whenever the FB is called. After 10000 revolutions, the calculation stops.

Holding the current value

Set *bHoldS* to TRUE to hold the latest material thickness *wThickness*. The current calculation cycle is aborted.

- If *bHoldS* is reset to FALSE, the next calculation cycle starts saving the current input values when the reel shaft has made two revolutions.

Saving of the current value

Use the input/output *wThicknessRetain* for permanently saving the current material thickness to ensure that the value will be available after a supply interruption or power-off.

- For this, declare a variable of type VAR RETAIN or VAR_GLOBAL RETAIN and connect it with the input.
- The RETAIN variable always receives the current value because the input is also the output.

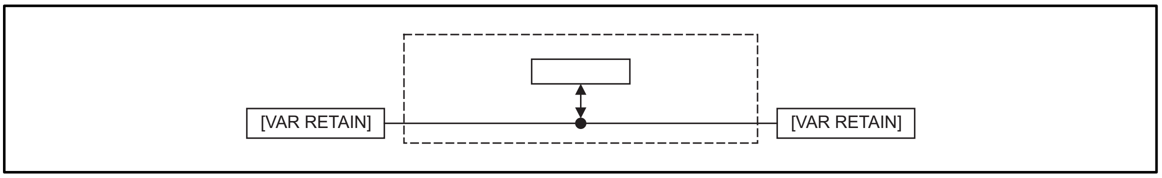


Fig. 2-5 Permanent saving of the current value in a RETAIN variable

- After the stored value has been loaded in, the next calculation cycle starts saving the input values when the reel shaft has made two revolutions.

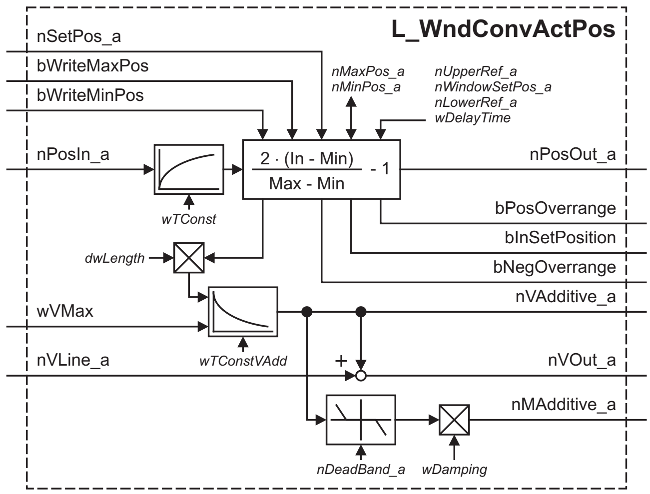
Limiting the material thickness

Use *wSMax* and *wSMin* to select the limit values for the material thickness to avoid higher variations of the output value.

- Big jumps of the output value are, for example, possible if jumps occur with the actual values (reel change with reset of the diameter controller) or the selected calculation cycle is too small.

2.5 L_WndConvActPos - Actual dancer value conditioning

This FB can be used to filter the actual dancer position value and to normalize it to the selected setting range. The setting range is determined by the limit positions that are automatically accepted by means of the “teach-in” function.



⇨ Inputs	(Variable type: VAR_INPUT)		
Identifier	**Data type**	**Signal type**	**Information/possible settings**
nSetPos_a	Integer	analog	Current position setpoint • Value range: -32767 ... 32767
bWriteMaxPos	Bool	binary	“Teach-in” function for the upper limit position TRUE | Current actual position value is stored under *nMaxPos_a* .
bWriteMinPos	Bool	binary	“Teach-in” function for the lower limit position TRUE | Current actual position value is stored under *nMinPos_a* .
nPosIn_a	Integer	analog	Current actual position value • Value range: -32767 ... 32767
wVMax	Word	-	Max. line speed in [0.1 m/min] • Value is internally limited to 0 ... 32767. • 2500 ≡ 250.0 m/min
nVLine_a	Integer	analog	Current line speed • Referred to the maximum value *wVMax* • 16384 ≡ 100 % of *wVMax*

Outputs ⇨	(Variable type: VAR_OUTPUT)		
Identifier	**Data type**	**Signal type**	**Value/meaning**
nPosOut_a	Integer	analog	Converted actual position value • -16384 ≡ -100 % ≡ *nMinPos_a* • 16384 ≡ +100 % ≡ *nMaxPos_a*
bPosOverrange	Bool	binary	Monitoring of the upper reference position TRUE | Upper reference position *nUpperRef_a* reached.
bInSetPosition	Bool	binary	Setpoint monitoring TRUE | Setpoint reached.
bNegOverrange	Bool	binary	Monitoring of the lower reference position TRUE | Lower reference position *nLowerRef_a* reached.
nVAdditive_a	Integer	analog	Material speed, resulting from the dancer movement • Referred to the maximum value *wVMax* • 16384 ≡ 100 % of *wVMax*

Outputs ⇨	(Variable type: VAR_OUTPUT)		
Identifier	**Data type**	**Signal type**	**Value/meaning**
nVOut_a	Integer	analog	Current circumferential speed of the reel when the dancer travel is input • Value is internally limited to -32767 ... 32767.
nMAdditive_a	Integer	analog	Additional torque for dancer control damping • Value is internally limited to -32767 ... 32767.

Parameters	(Variable type: VAR_CONSTANT_RETAIN)						
Identifier	**Data type**	**Signal type**	**Info**				
			Possible settings				**Presetting**
wTConst	Word	-	Filter time constant for current actual position value				
			0	{1 msec}	1000		10 msec
nMaxPos_a	Integer	analog	Upper limit position of the dancer • 100 % ≡ 16384				
			-199.99	{0.01 %}	199.99		100.00 %
nMinPos_a	Integer	analog	Lower limit position of the dancer • -100 % ≡ -16384				
			-199.99	{0.01 %}	199.99		-100.00 %
dwLength	Double word	-	Material length in the dancer				
			0	{1 mm}	100000		0 mm
nUpperRef_a	Integer	analog	Reference value for upper limit position				
			-199.99	{0.01 %}	199.99		90.00 %
nWindowSetPos_a	Integer	analog	Tolerance range for ”Setpoint reached”				
			0.00	{0.01 %}	200.00		10.00 %
nLowerRef_a	Integer	analog	Reference value for lower limit position				
			-199.99	{0.01 %}	199.99		-90.00 %
wDelayTime	Word	-	Delay time for “Dancer in limit position” / “Dancer no longer in setpoint position”				
			0.000	{0.001 sec}	50.000		0.100 s
wTConstVAdd	Word	-	Filter time constant for speed change through dancer				
			0	{1 msec}	1000		10 msec
nDeadBand_a	Integer	analog	Dead band for current additional torque				
			0.00	{0.01 %}	100.00		1.00 %
wDamping	Word	-	Damping factor				
			0.00	{0.01}	10.00		0.00

Selection of limit positions

The limit positions are selected via the parameters *nMaxPos_a* and *nMinPos_a* .

- The actual position at which the material stored in the dancer reaches its maximum length is to be entered under *nMinPos_a* .
- The actual position at which the material stored in the dancer reaches its minimum length is to be entered under *nMaxPos_a* .

Tip!

Alternatively, it is possible to move the dancer to the individual limit positions and to accept the current values in the corresponding parameters by setting *bWriteMaxPos* and / or *bWriteMinPos* to TRUE (”teach-in” function).

- If the dancer detection supplies an inverse signal the value of *nMinPos_a* may be higher than the value of *nMaxPos_a*.

Dancer position monitoring

If the actual dancer value reaches the current setpoint (*nSetPos_a*), *bInSetPosition_b* is set to TRUE.

- You can use *nWindowSetPos_a* to set a tolerance range for reaching the setpoint. If the actual value leaves the tolerance range the output variable *bInSetPosition* will only be reset to FALSE when the time *wDelayTime* has expired (OFF-delayed).
- You can use *nUpperRef_a* and *nLowerRef_a* to select reference values for the limit position monitoring. If the actual dancer value reaches the set reference value approaching the limit position, the corresponding output (*bPosOverrange* and / or *bNegOverrange*) will only be set to TRUE when the time *wDelayTime* has expired (ON-delayed).

Circumferential speed of the reel

If the storage capacity of the dancer is entered via *dwLength* the additional speed of the material resulting from the dancer movement can be obtained via *nVAdditive_a*. When adding this value to the line speed *nVLine_a* you will obtain the circumferential speed of the reel via *nVOut_a*.

- The storage capacity is, for example, twice the distance between the two limit positions multiplied by the number of material windings.
- The line speed is calculated from *wVMax* and *nVLine_a*.
- The current circumferential speed of the reel is output via *nVOut_a*.
- The additional speed is filtered to suppress the effects of the natural “noise” of the actual value signal. The time constant of the filter can be selected via *wTConstVAdd*.

Additional torque for dancer control damping

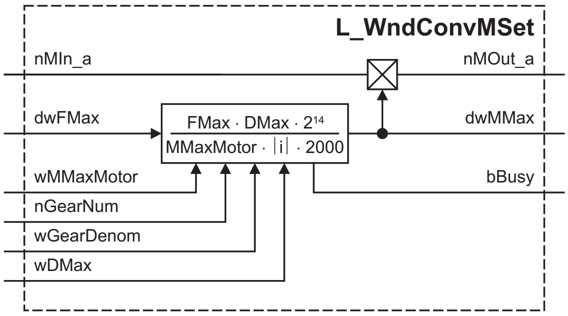
If the storage capacity of the dancer is entered via *dwLength* it is possible to generate an additional precontrol torque *nMAdditive_a* to damp the dancer control.

- The precontrol torque is derived from the inverted additional speed (twice the dancer speed) evaluated with the damping factor *wDamping*.
- The additional speed is filtered and, additionally, passes a dead band element to suppress the effects of the natural ”noise” of the actual value signal on the additional torque. The dead band range can be set via *nDeadBand_a*.

2.6 L_WndConvMSet - Torque setpoint converter

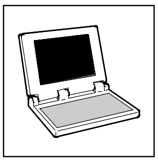
This FB can be used to convert the torque setpoint resulting from the tension multiplied by the diameter into a setpoint that can be used for the motor control (F/F_{max} * D/D_{max} ⇨ $M/M_{max.motor}$).

The maximum output current of the controller and the maximum motor torque can be freely selected without having to adapt the setpoint calculation manually.



⇨ Inputs	(Variable type: VAR_INPUT)		
Identifier	**Data type**	**Signal type**	**Information/possible settings**
nMIn_a	Integer	analog	Torque setpoint • Referred to the max. torque *dwMMax* resulting from the data of the winding drive. • 16384 ≡ 100 % of *dwMMax*
dwFmax	Double word	-	Max. tension in [0.1 N] • Value is internally limited to 10000 kN. • 3000 ≡ 300.0 N
wMMaxMotor	Word	-	Max. motor control torque in [0.1 Nm] • Value is internally limited to 0 ... 3000 Nm. • 1250 ≡ 125.0 Nm • For the **9300 Servo PLC** this input is to be connected with the output *MCTRL_nMmaxC57* of the SB **MCTRL**.
nGearNum	Integer	-	Current gearbox factor - numerator • Value is internally limited to -32767 ... -1 / 1 ... 32767.
wGearDenom	Word	-	Current gearbox factor - denominator • Value is internally limited to 2 * |*nGearNum*| in the range of 1 ... 32767.
wDMax	Word	-	Max. diameter in [mm] • Value is internally limited to 10000 mm.

Outputs ⇨	(Variable type: VAR_OUTPUT)		
Identifier	**Data type**	**Signal type**	**Value/meaning**
nMOut_a	Integer	analog	Current torque setpoint • Referred to the max. motor control torque. • 16384 ≡ 100 % of max. torque • For **9300 Servo PLC** : max. torque = *MCTRL_nMmaxC57*
dwMMax	Double word	-	Max.torque on the motor side from *FMax* * (*DMax* /2) / |i | • Referred to the max. motor control torque. • Value is internally limited to 819200 (5000 %). • Updating interval: 4 FB call cycles
bBusy	Bool	binary	Status signal “New calculation active” TRUE: Factor *dwMMax* is re-calculated.



Conversion factor

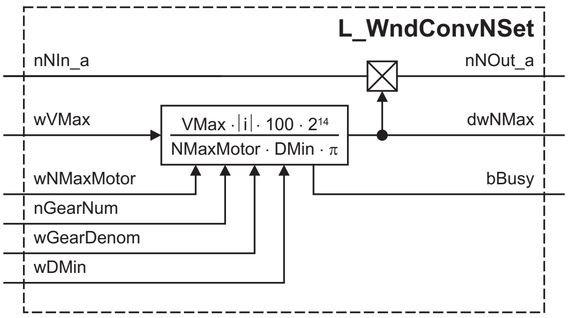
The conversion factor *dwMMax* calculated from the the physical parameters shows the utilization of the motor control torque setting range.

- Values higher than 16384 (> 100 %) indicate that the torque required for max. tension is higher than the max. motor control torque when the largest diameter is used.

2.7 L_WndConvNSet - Speed setpoint converter

This FB can be used to convert the speed setpoint resulting from the line speed multiplied by the reciprocal diameter into a setpoint that can be used for the motor control (V/V_{max} * $1/(D/D_{min})$ ⇨ $N/N_{max.motor}$).

The maximum controller speed can be freely selected without having to adapt the setpoint calculation manually.



⇨ Inputs	(Variable type: VAR_INPUT)		
Identifier	**Data type**	**Signal type**	**Information/possible settings**
nNIn_a	Integer	analog	Speed setpoint • Referred to the max. speed *dwNMax* resulting from the data of the winding drive. • 16384 ≡ 100 % of *dwNMax*
wVMax	Word	-	Max. line speed in [0.1 m/min] • Value is internally limited to 0 ... 3000 m/min. • 2500 ≡ 250.0 m/min
wNMaxMotor	Word	-	Max. motor control speed in [rpm] • Value is internally limited to 15000 rpm. • For the **9300 Servo PLC** this input is to be connected with the output *MCTRL_nNmaxC11* of the SB **MCTRL** .
nGearNum	Integer	-	Current gearbox factor - numerator • Value is internally limited to 2000 * |*wGearDenom*| in the range of -32767 ... -1 / 1 ... 32767.
wGearDenom	Word	-	Current gearbox factor - denominator • Value is internally limited to 2 * |*nGearNum*| in the range of 1 ... 32767.
wDMin	Word	-	Min. diameter in [mm] • Value is internally limited to 10000 mm.

Outputs ⇨	(Variable type: VAR_OUTPUT)		
Identifier	**Data type**	**Signal type**	**Value/meaning**
nNOut_a	Integer	analog	Current speed setpoint • Referred to the max. motor control speed. • 16384 ≡ 100 % of max. speed • For **9300 Servo PLC** : max. speed = *MCTRL_nNmaxC11*
dwNMax	Double word	-	Max. speed on the motor side from *VMax* * |i| / (*DMin* * π) • Referred to the max. motor control speed. • Value is internally limited to 819200 (5000 %). • Updating interval: 4 FB call cycles
bBusy	Bool	binary	Status signal “New calculation active” TRUE: Factor *dwNMax* is re-calculated.

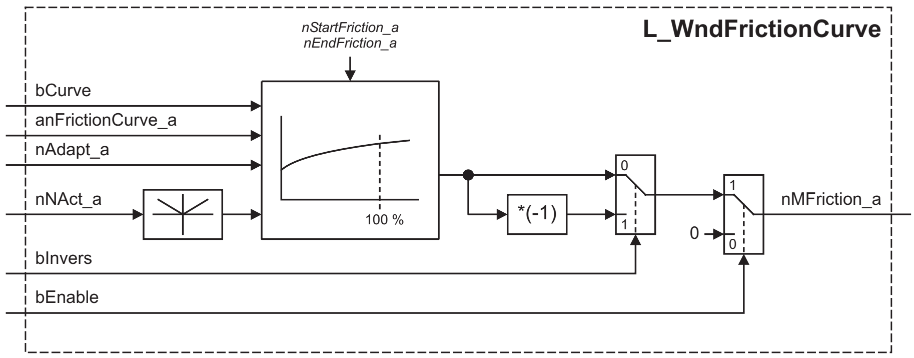
Conversion factor

The conversion factor *dwNMax* calculated from the the physical parameters shows the utilization of the motor control speed setting range.

- Values higher than 16384 (> 100 %) indicate that the speed required for max. line speed is higher than the max. motor control speed when the smallest diameter is used.

2.8 L_WndFrictionCurve - Friction compensation

This FB can be used to generate a speed-dependent precontrol signal to compensate the friction torque. As an alternative, you can select a linear characteristic or a characteristic of 65 table values.



⇨ Inputs	(Variable type: VAR_INPUT)		
Identifier	**Data type**	**Signal type**	**Information/possible settings**
bCurve	Bool	binary	Selection of the compensation method FALSE: Speed-dependent compensation TRUE: Characteristic-based compensation
anFrictionCurve_a	Array of integers [0..64]	analog	Array for characteristics table • The 65 values are distributed over the speed range of *nNAct_a* = 0 ... 16384. • Values are internally limited to 0 ... 32767.
nAdapt_a	Integer	analog	Control signal for proportional characteristics evaluation • 16384 ≡ 100 % • Value is internally limited to 0 ... 32767.
nNAct_a	Integer	analog	Actual speed as a reference signal for the characteristic • Referred to the max. motor control speed (16384 ≡ 100 %). • For **9300 Servo PLC** : 16384 ≡ 100 % of *MCTRL_nNmaxC11*
bInvers	Bool	binary	Output signal inversion TRUE: Output signal *nMFriction_a* is inverted.
bEnable.	Bool	binary	Activation of the friction compensation FALSE: Friction torque is reset to "0". TRUE: Friction compensation is activated.

Outputs ⇨	(Variable type: VAR_OUTPUT)		
Identifier	**Data type**	**Signal type**	**Value/meaning**
nMFriction_a	Integer	analog	Current friction torque • Referred to the max. motor control torque (16384 ≡ 100 %). • For **9300 Servo PLC** : 16384 ≡ 100 % of *MMax*

Parameters	(Variable type: VAR_CONSTANT_RETAIN)						
Identifier	**Data type**	**Signal type**	**Info**				
			Possible settings			**Presetting**	
nStartFriction_a	Integer	analog	Initial friction torque				
			0.00	{0.01 %}	199.99	0.00 %	
nEndFriction_a	Integer	analog	Friction torque at max. speed				
			0.00	{0.01 %}	199.99	0.00 %	

Friction behaviour

Starting from the breakaway torque, the friction generally increases with the speed.

Apart from the speed, the friction behaviour can also be influenced by temperature and torque (e.g. when using a gearbox).

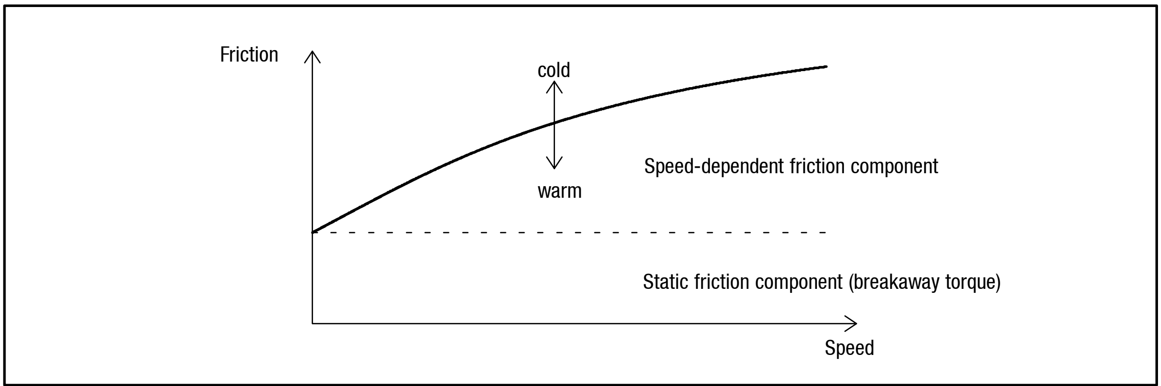


Fig. 2-6 Friction behaviour

- The current speed is preset via *nNAct_a*.
- If the actual speed exceeds 100 %, the characteristic is extrapolated linearly.

Characteristic selection

By setting *bCurve* to TRUE you can select a characteristic of 65 values instead of a linear characteristic for friction compensation.

- The linear characteristic is selected via the parameters *nStartFriction_a* and *nEndFriction_a*.
  – The characteristic is internally limited to 0 ... 32767.
- The FB **L_WndIdentFriction** can be used to select a table value-based characteristic. The FB automatically calculates the friction.

Tip!

For a table-value based characteristic:

1. Create a variable (ARRAY [0..64] of INT) and assign it to a code. Now, you can enter the characteristic values and save them permanently.
2. Connect the variable with the input *anFrictionCurve_a*.

Temperature compensation

You can use *nAdapt_a* to select a characteristic gain that allows you to compensate, for example, for friction torque changes caused by temperature changes.

Output of the precontrol signal

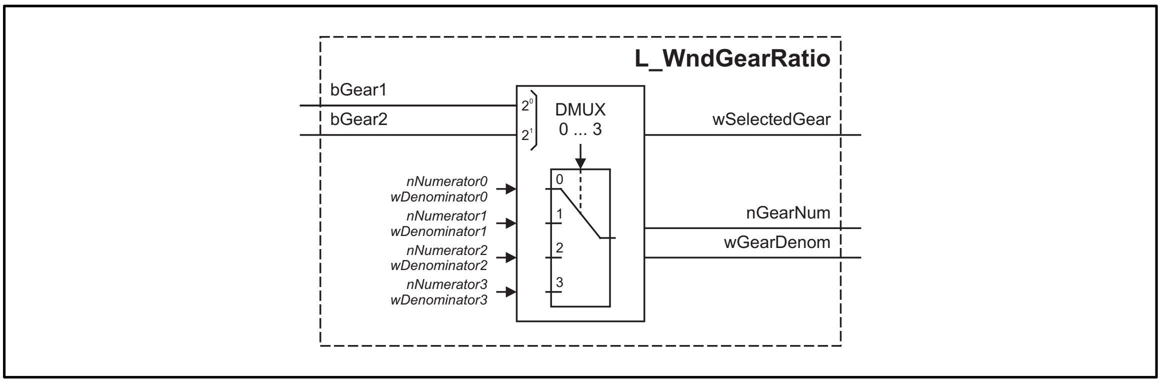
The generated precontrol signal is output at *nMFriction_a*.

- By setting *bInvers* to TRUE you can invert the signal (change of direction of rotation)

2.9 L_WndGearRatio - Gearbox factor changeover

This FB can be used to select one of four gearbox factors via two binary signals. In this way, you can easily adjust the internal signal processing when using winders with changeover gearboxes.

- The four gearbox factors to choose from are provided as integer fractions (numerator and denominator).
- The gearbox factor selected for the Lenze FBs is also output as an integer fraction.



⇨ Inputs	(Variable type: VAR_INPUT)		
Identifier	**Data type**	**Signal type**	**Information/possible settings**
bGear1	Bool	binary	Binary coded “1 out of 4“ gearbox factor selection
bGear2	Bool	binary	

Outputs ⇨	(Variable type: VAR_OUTPUT)		
Identifier	**Data type**	**Signal type**	**Value/meaning**
wSelectedGear	Word		Index (0 ... 3) of the selected gearbox factor
nGearNum	Integer	-	Current gearbox factor - numerator
wGearDenom	Word	-	Current gearbox factor - denominator

Parameters	(Variable type: VAR_CONSTANT_RETAIN)			
Identifier	**Data type**	**Signal type**	**Info**	
			Possible settings	**Presetting**
			Gearbox factor for selection index 0	i = 1.000
nNumerator0	Integer	-	Numerator	
			-32767 {1} 32767 • If ”0” is selected, the value ”1” is used internally.	1000
wDenominator0	Word	-	Denominator	
			1 {1} 32767	1000
			Gearbox factor for selection index 1	i = 2.000
nNumerator1	Integer	-	Numerator	
			-32767 {1} 32767 • If ”0” is selected, the value ”1” is used internally.	2000
wDenominator1	Word	-	Denominator	
			1 {1} 32767	1000

Parameters	(Variable type: VAR_CONSTANT_RETAIN)				
Identifier	Data type	Signal type	Info		
			Possible settings		Presetting
			Gearbox factor for selection index 2		i = 5.000
nNumerator2	Integer	-	Numerator		
			-32767 {1} 32767 • If ”0” is selected, the value ”1” is used internally.		5000
wDenominator2	Word	-	Denominator		
			1 {1} 32767		1000
			Gearbox factor for selection index 3		i = 8.000
nNumerator3	Integer	-	Numerator		
			-32767 {1} 32767 • If ”0” is selected, the value ”1” is used internally.		8000
wDenominator3	Word	-	Denominator		
			1 {1} 32767		1000

Gearbox factor selection

The gearbox factor is binary coded. It is selected via the digital inputs *bGear1* and *bGear2* :

bGear1	bGear2	Selected gearbox factor	wSelectedGear (output)
0	0	nNumerator0 / wDenominator0	0
0	1	nNumerator1 / wDenominator1	1
1	0	nNumerator2 / wDenominator2	2
1	1	nNumerator3 / wDenominator3	3

Output of the selected gearbox factor

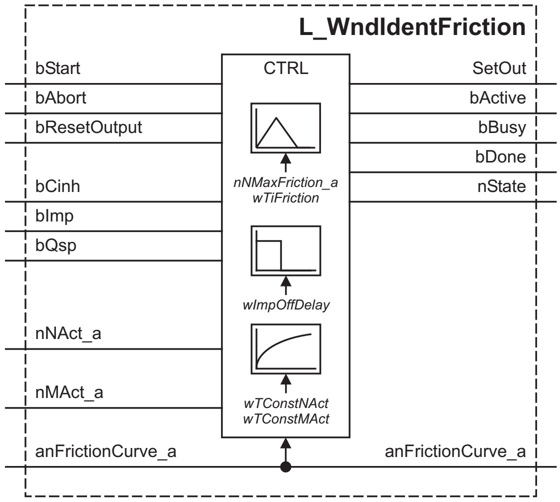
The selected gearbox factor is output as an integer fraction via *nGearNum* (numerator) and *wGearDenom* (denominator). Many Lenze FBs expect the gearbox factor entries in this standard form.

- Via *wSelectedGear* you can also display the index (0 ... 3) of the selected gearbox factor. The index can, for example, be used for parameter changeovers.

2.10 L_WndIdentFriction - Identification of the friction characteristic

With this FB you can automatically identify the friction characteristic of the controller. When the controller is inhibited, the FB takes over drive control on request and carries out the identification after controller enable.

- The motor control receives the corresponding setpoints in the form of a structure (*SetOut*).
- Via *nState* you can display the current status of the identification run or the cause that led to an early abort.
- As soon as the identification run has been successfully completed, the friction characteristic is output by *anFrictionCurve_a*.



⇨ Inputs	(Variable type: VAR_INPUT)			
Identifier	**Data type**	**Signal type**	**Information/possible settings**	
bStart	Bool	binary	Starting the identification of the friction characteristic • Only in connection with controller enable.	
			FALSE ↗TRUE	Starts identification run
bAbort	Bool	binary	Aborting the identification of the friction characteristic	
			TRUE	Aborts identification run, sets quick stop (QSP).
bResetOutput	Bool	binary	Reset of detected values	
			TRUE	Resets values
bCinh	Bool	binary	Message input for controller inhibit	
			TRUE	Aborts identification run, sets quick stop (QSP).
bImp	Bool	binary	Message input for pulse inhibit	
			TRUE	Aborts identification run, sets quick stop (QSP).
bQsp	Bool	binary	Message input for quick stop (QSP)	
			TRUE	Aborts identification run
nNAct_a	Integer	analog	Current motor speed	
nMAct_a	Integer	analog	Current motor torque	

Function library LenzeWinder.lib

Function blocks

L_WndIdentFriction - Identification of the friction characteristic

Outputs ⇨	(Variable type: VAR_OUTPUT)				
Identifier	**Data type**	**Signal type**	**Value/meaning**		
SetOut	Struct	-	Structure including motor control setpoints (*bQspOut* , *bNMSwt* , *bILoad* , *bPosOn* , *nHiMLim_a* , *nLoMLim_a* , *nNSet_a* , *nNStartMLim_a* , *nMAdd_a* , *nFldWeak_a* , *nNAdapt_a*)		
bActive	Bool	binary	Status signal “Identification activated”		
				TRUE	Identification run activated.
bBusy	Bool	binary	Status signal “Measured-value detection active”		
				TRUE	Measured-value detection active.
bDone	Bool	binary	Status signal “Identification completed successfully”		
				TRUE	Identification run completed successfully.
nState	Integer	-	Identification run status		
				Status messages:	
				0	Identification run not active.
				20	Waiting for controller enable for friction characteristic identification.
				21	Drive is decelerated along the quick stop ramp to standstill.
				22	Friction starts during run-up.
				23	Friction starts during run-down.
				24	Calculation of the friction characteristic (*anFrictionCurve_a*).
				30	Identification run completed. • Waiting for controller inhibit or restart.
				Error messages (waiting for controller inhibit or restart):	
				-2	Cancelled by stop command (*bAbort*).
				-3	Cancelled by pulse inhibit (*bImp*).
				-4	Cancelled by quick stop (*bQsp*).
				-5	Acceleration time too short to start friction.
				-6	Drive could not follow when friction started.

Inputs/outputs	(Variable type: VAR_IN_OUT)		
Identifier	**Data type**	**Signal type**	**Value/meaning**
anFrictionCurve_a	Array of integers	analog	Input/output for permanent saving of the friction characteristic • For this, declare a variable of type VAR RETAIN or VAR_GLOBAL RETAIN and connect it with the input. • The RETAIN variable always receives the current value because the input is also the output. • RETAIN variables are automatically saved during a supply interruption and automatically read after power-on.

Parameters	(Variable type: VAR_CONSTANT_RETAIN)					
Identifier	Data type	Signal type	Info			
			Possible settings			Presetting
nNMaxFriction_a	Integer	analog	Max. speed for identification of the friction characteristic • Referred to the max. motor control speed. (100 % ≡ 16384 ≡ *MCTRL_nNmaxC11*).			
			-100.00 5.00	{0.01 %} {0.01 %}	-5.00 100.00	100.00 %
wTiFriction	Word	-	Acceleration/deceleration time for identification of the friction characteristic			
			1	{1 sec}	3600	30 s
wImpOffDelay	Word	-	Start delay for identification after pulse enable			
			0	{1 msec}	1000	500 msec
wTConstNAct	Word	-	Filter time constant for actual speed			
			0	{1 msec}	50	0 ms
wTConstMAct	Word	-	Filter time constant for actual torque			
			0	{1 msec}	1000	0 ms

Identification of the friction characteristic

For friction characteristic identification, the drive is slowly accelerated to a defined limit speed *nNMaxFriction_a*) and then decelerated to standstill.

- The friction characteristic comprises 65 values that are uniformly distributed over the speed range of 0 to N_{max} (for **9300 Servo PLC**: N_{max} = code C0011).
- The rate of change of the speed is selected via *wTiFriction*.

Functional sequence of identification

Status (nState)	Info
0	The identification run is not active. To start the identification run: 1. Set controller inhibit. 2. Set *bStart* from FALSE to TRUE.
20	The identification run has started. • The output *bActive* switches to TRUE. • To start moving the drive, controller inhibit has to be deactivated.
21	Controller enable has been carried out. • The output *bBusy* switches to TRUE. • If the drive is coasting, the motor will be stopped for a while.
22	The preset acceleration time (*wTiFriction*) is used to accelerate the drive to the selected limit speed (*nNMaxFriction_a*).
	The drive has reached the limit speed and is decelerated until it comes to a standstill using the preset acceleration/deceleration time (*wTiFriction*).
24	The 65 values of the friction characteristic are calculated (*anFrictionCurve_a*). • The friction characteristic is normalized to the max. motor control torque and shows the torque required for the drive due to friction.
30	Friction characteristic identification has been completed. • The output *bDone* switches to TRUE. • The output *bBusy* switches to FALSE.

Status machine for the identification run

The identification is carried out according to the following status machine:

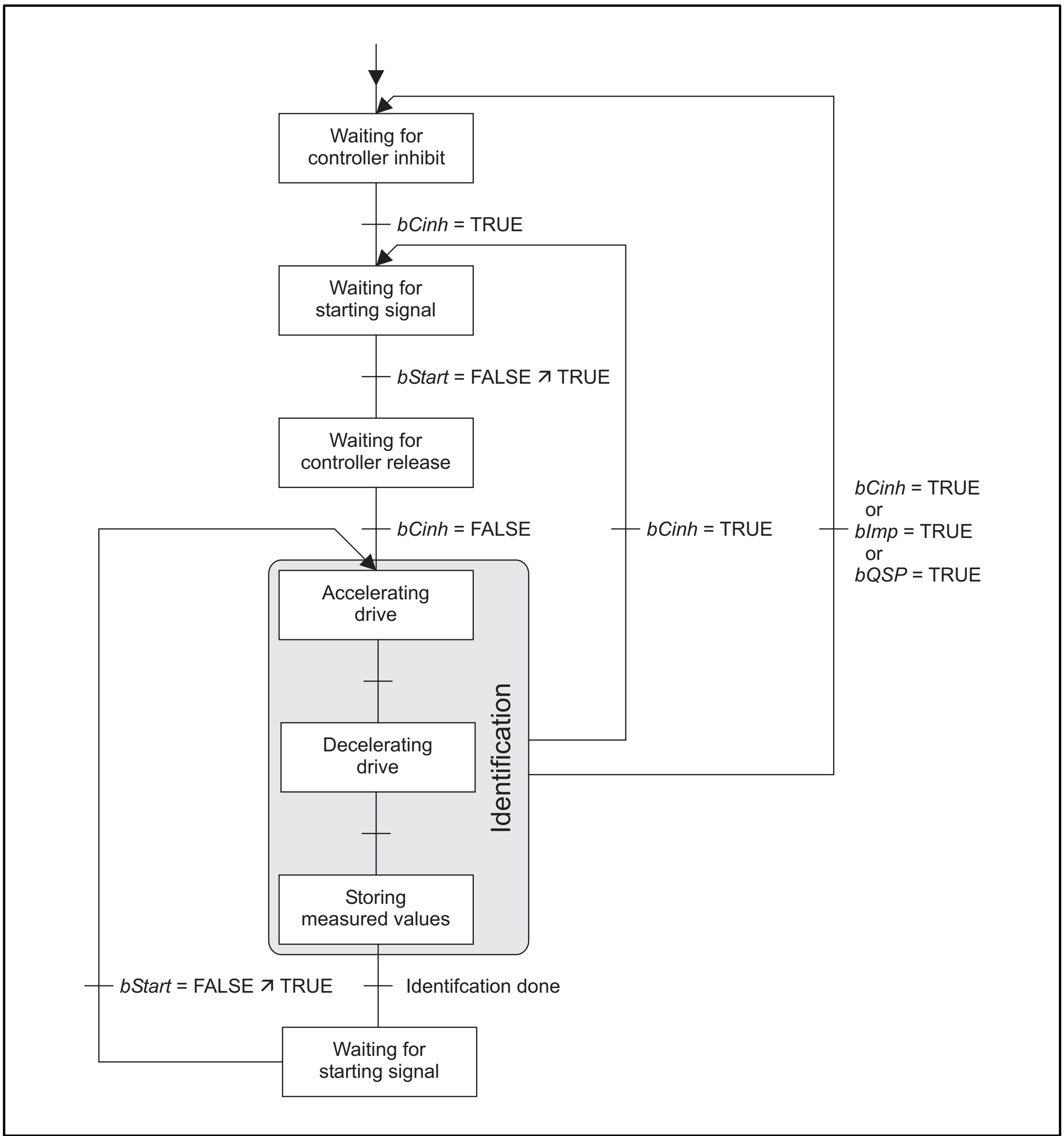


Fig. 2-7 Status machine for the identification run

Permanent storage/correction of the friction characteristic

Use the input/output *anFrictionCurve_a* for permanently saving the friction characteristic to ensure that the values will be available after a supply interruption or power-off.

- For this, declare a variable of type VAR RETAIN or VAR_GLOBAL RETAIN and connect it with the input.
- The variable always receives the current values because input and output are the same.

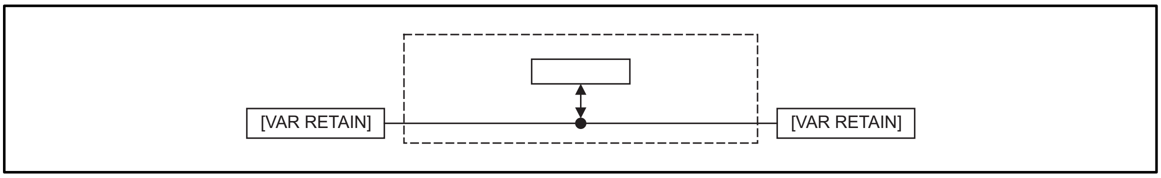


Fig. 2-8 Permanent saving of the current values in a RETAIN variable

If you assign a code to the RETAIN variable, you can easily correct individual characteristic values via the **Parameter monitor**.

Cancellation of the identification run

When the identification run is activated, controller inhibit, pulse inhibit, quick stop and setting *bAbort* to TRUE immediately stop the identification run.

- *bBusy* is set to FALSE and quick stop is used to bring the drive to a standstill.
- The reason for the cancellation is displayed by *nState*.
- When controller inhibit is set, the FB returns to state "0" and *bActive* is reset to FALSE.

Delay of pulse enable

After controller enable, the identification run does not start immediately, but only after expiry of the delay time *wImpOffDelay*. In this way, the magnetic flux of the motor can be completely built up (only required for asynchronous motors).

- When using machines with high rotor time constants it might be necessary to increase the preset values.

Smoothing of actual values

When incremental encoders are used it may be necessary to filter the actual speed signal.

- The filter time constant is selected via *wTConstNAct*, usually 10 ms-settings are enough.
- For smoothing the friction characteristic, it may be helpful to filter the actual torque as well. The corresponding filter time constant is selected via *wTConstMAct*.

Transfer of setpoints to the motor control

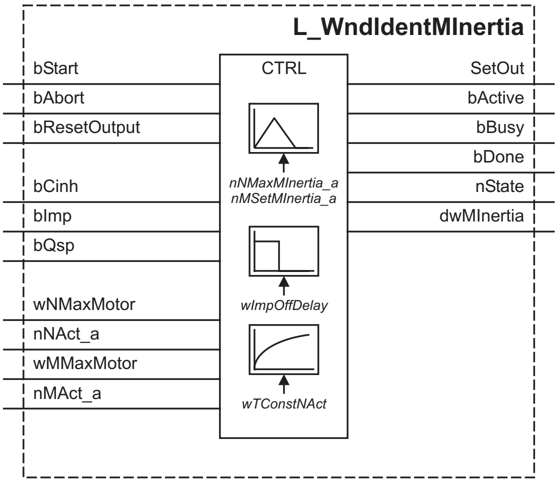
The motor control (MCTRL) receives the corresponding setpoints in the form of a structure (*SetOut*) including the following elements:

Identifier	Data type	Signal type	Value/meaning	
bQSPOut	Bool	binary	Control signal "quick stop"	
			TRUE	Carries out quick stop (QSP).
bNMSwt	Bool	binary	Speed control/torque control selection	
			FALSE	Speed control
			TRUE	Torque control
bILoad	Bool	binary	Control signal "Load I-component"	
			TRUE	Accepts I-component of the n-controller in *MCTRL_nISet_a*.
bPosOn	Bool	binary	Control signal "Activate phase-angle controller"	
			TRUE	Activates phase-angle controller.
nHiMLim_a	Integer	analog	Upper torque limit • Referred to C0057: 16384 ≡ 100 % of C0057	
nLoMLim_a	Integer	analog	Lower torque limit • Referred to C0057: 16384 ≡ 100 % of C0057	
nNSet_a	Integer	analog	Speed setpoint	
nNStartMLim_a	Integer	analog	Lower speed limit for speed restriction	
nMAdd_a	Integer	analog	Additional torque setpoint	
nFldWeak_a	Integer	analog	Motor excitation	
nNAdapt_a	Integer	analog	Adaptive Vp of the speed controller	

2.11 L_WndIdentMInertia - Identification of the moment of inertia

With this FB you can automatically identify the current moment of inertia of the controller. When the controller is inhibited, the FB takes over drive control on request and carries out the identification after controller enable.

- The motor control receives the corresponding setpoints in the form of a structure (*SetOut*).
- Via *nState* you can display the current status of the identification run or the cause that led to an early abort.
- As soon as the identification run has been completed successfully, the moment of inertia is output by*dwMInertia*.



⇨ Inputs	(Variable type: VAR_INPUT)			
Identifier	**Data type**	**Signal type**	**Information/possible settings**	
bStart	Bool	binary	Starting the identification run of the moment of inertia • Only in connection with controller enable.	
			TRUE	Starts identification run
bAbort	Bool	binary	Aborting the identification run of the moment of inertia	
			TRUE	Aborts identification run
bResetOutput	Bool	binary	Reset of the detected value	
			TRUE	Resets value
bCinh	Bool	binary	Message input for controller inhibit	
			TRUE	Aborts identification run
bImp	Bool	binary	Message input for pulse inhibit	
			TRUE	Aborts identification run
bQsp	Bool	binary	Message input for quick stop (QSP)	
			TRUE	Aborts identification run
wNMaxMotor	Word	-	Max. motor control speed in [rpm] • Max. permissible value: 15000 rpm • For the **9300 Servo PLC** this input is to be connected with the output *MCTRL_nNmaxC11* of the SB **MCTRL**.	
nNAct_a	Integer	analog	Current motor speed	

Function library LenzeWinder.lib

Function blocks

L_WndIdentMInertia - Identification of the moment of inertia

⇨ Inputs	(Variable type: VAR_INPUT)		
Identifier	**Data type**	**Signal type**	**Information/possible settings**
wMMaxMotor	Word	-	Max. motor control torque in [0.1 Nm] • Value is internally limited to 3000 Nm. • 1250 ≡ 125.0 Nm • For the **9300 Servo PLC** this input is to be connected with the output *MCTRL_nMmaxC57* of the SB **MCTRL**.
nMAct_a	Integer	analog	Current motor torque in [0.1 Nm] referred to *wMMaxMotor*.

Outputs ⇨	(Variable type: VAR_OUTPUT)			
Identifier	**Data type**	**Signal type**	**Value/meaning**	
SetOut	Struct	-	Structure including motor control setpoints (*bQspOut* , *bNMSwt* , *bILoad* , *bPosOn* , *nHiMLim_a* , *nLoMLim_a* , *nNSet_a* , *nNStartMLim_a* , *nMAdd_a* , *nFldWeak_a* , *nNAdapt_a*)	
bActive	Bool	binary	Status signal “Identification activated”	
			TRUE	Identification run activated.
bBusy	Bool	binary	Status signal “Measured-value detection active”	
			TRUE	Measured-value detection active.
bDone	Bool	binary	Status signal “Identification completed successfully”	
			TRUE	Identification run completed successfully.
nState	Integer	-	Identification run status	
			Status messages:	
			0	Identification run not active.
			20	Waiting for controller enable for moment of inertia identification.
			21	Drive is decelerated along the quick stop ramp to standstill.
			22	Acceleration starts during run-up.
			23	Acceleration starts during deceleration.
			24	Calculation of the moment of inertia (*dwMInertia*).
			30	Identification run completed. • Waiting for controller inhibit or restart.
			Error messages (waiting for controller inhibit or restart):	
			-1	Input value for max. speed or max. torque is not within the permissible limits.
			-2	Cancelled by stop command (*bAbort*).
			-3	Cancelled by pulse inhibit (*bImp*).
			-4	Cancelled by quick stop (*bQsp*).
			-7	Speed reached at the end of acceleration too low.
			-8	Mass moment of inertia too high (overflow).
dwMInertia	Double word	-	Identified moment of inertia in [0.1 kg cm^2]	

Parameters	(Variable type: VAR_CONSTANT_RETAIN)				
Identifier	Data type	Signal type	Info		
			Possible settings		Presetting
nNMaxMInertia_a	Integer	analog	Max. speed for moment of inertia identification • Referred to the max. motor control speed. (100 % ≡ 16384 ≡ *MCTRL_nNmaxC11*).		
			5.00 {0.01 %}	100.00	10.00 %
nMSetMInertia_a	Integer	analog	Torque setpoint for moment of inertia identification • Referred to the max. motor control torque (100 % ≡ 16384 ≡ *MCTRL_nMmaxC57*).		
			-100.00 {0.01 %} 5.00 {0.01 %}	-5.00 100.00	25.00 %
wImpOffDelay	Word	-	Start delay for identification after pulse enable		
			0 {1 msec}	1000	500 msec
wTConstNAct	Word	-	Filter time constant for actual speed		
			0 {1 msec}	50	0 ms

Identification of the moment of inertia

For moment of inertia identification, the drive is accelerated to the defined limit speed (*wNMaxMInertia_a*) and then decelerated to standstill.

- The rate of change of the speed is calculated from the selected torque (*nMSetMInertia_a*) and the existing moment of inertia.
- The reference values for limit speed and motor torque have to be selected through *wNMaxMotor* and *wMMaxMotor*.

Functional sequence of identification

Status (nState)	Info
0	The identification run is not active. To start the identification run: 1. Set controller inhibit. 2. Set *bStart* from FALSE to TRUE.
20	The identification run has started. • The output *bActive* switches to TRUE. • To start moving the drive, controller inhibit has to be deactivated.
21	Controller enable has been carried out. • The output *bBusy* switches to TRUE. • If the drive is coasting, the motor will be stopped for a while.
22	The preset torque (*nMSetMInertia_a*) is used to accelerate the drive to the selected limit speed (*nNMaxMInertia_a*).
23	The drive has reached the limit speed and is decelerated until it comes to a standstill using the inverse torque.
24	The moment of inertia is calculated (*dwMInertia*). • The result is output as a physical unit [0.1 kg cm^2].
30	The identification of the moment of inertia has been completed. • The output *bDone* switches to TRUE. • The output *bBusy* switches to FALSE.

Status machine for the identification run

The identification is carried out according to the following status machine:

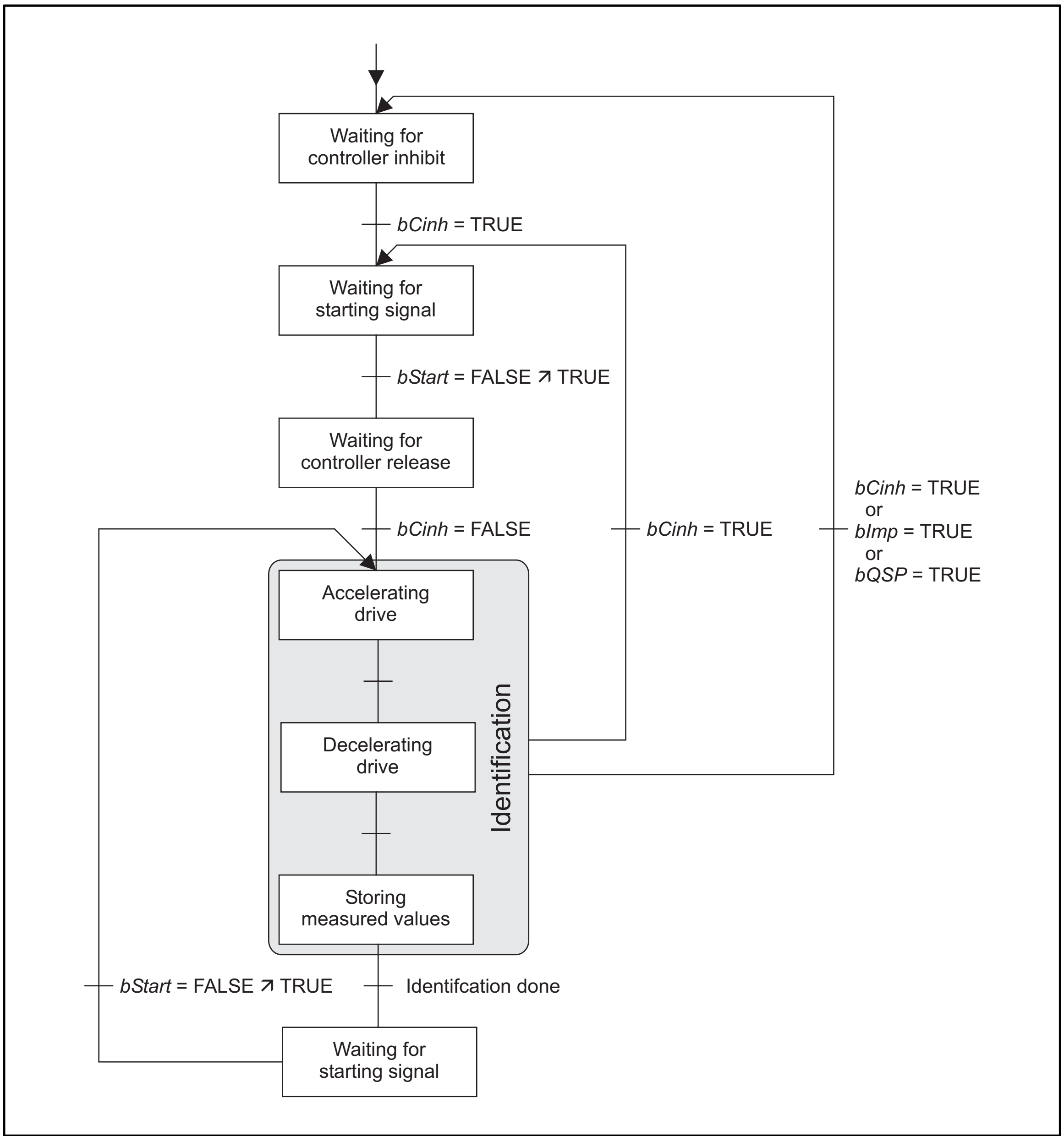


Fig. 2-9 Status machine for the identification run

Cancellation of the identification run

When the identification run is activated, controller inhibit, pulse inhibit, quick stop and setting *bAbort* to TRUE immediately stop the identification run.

- *bBusy* is set to FALSE and quick stop is used to bring the drive to a standstill.
- The reason for the cancellation is displayed by *nState* .
- When controller inhibit is set, the FB returns to state "0" and *bActive* is reset to FALSE.

Delay of pulse enable

After controller enable, the identification run does not start immediately, but only after expiry of the delay time *wImpOffDelay*. In this way, the magnetic flux of the motor can be completely built up (only required for asynchronous motors).

- When using machines with high rotor time constants it might be necessary to increase the preset values.

Smoothing of actual values

When incremental encoders are used it may be necessary to filter the actual speed signal.

- The filter time constant is selected via *wTConstNAct* , usually 10 ms-settings are high enough.

Transfer of setpoints to the motor control

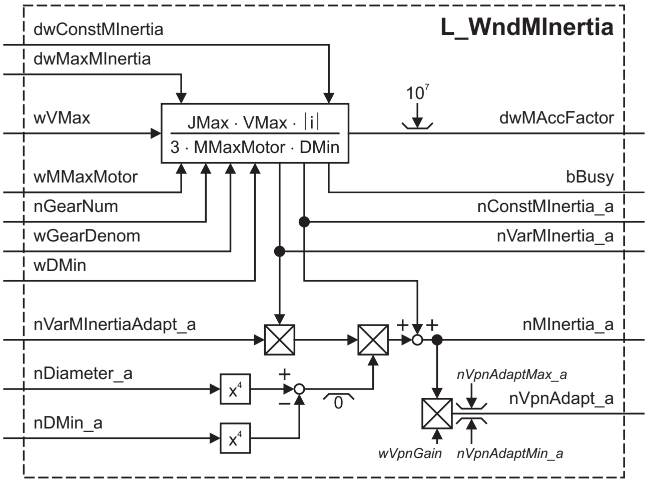
The motor control (MCTRL) receives the corresponding setpoints in the form of a structure (*SetOut*) including the following elements:

Identifier	Data type	Signal type	Value/meaning	
bQSPOut	Bool	binary	Control signal "quick stop"	
			TRUE	Carries out quick stop (QSP).
bNMSwt	Bool	binary	Speed control/torque control selection	
			FALSE	Speed control
			TRUE	Torque control
bILoad	Bool	binary	Control signal "Load I-component"	
			TRUE	Accepts I-component of the n-controller in *MCTRL_nISet_a*.
bPosOn	Bool	binary	Control signal "Activate phase-angle controller"	
			TRUE	Activates phase-angle controller.
nHiMLim_a	Integer	analog	Upper torque limit • Referred to C0057: 16384 ≡ 100 % of C0057	
nLoMLim_a	Integer	analog	Lower torque limit • Referred to C0057: 16384 ≡ 100 % of C0057	
nNSet_a	Integer	analog	Speed setpoint	
nNStartMLim_a	Integer	analog	Lower speed limit for speed restriction	
nMAdd_a	Integer	analog	Additional torque setpoint	
nFldWeak_a	Integer	analog	Motor excitation	
nNAdapt_a	Integer	analog	Adaptive Vp of the speed controller	

2.12 L_WndMInertia - Calculation of the moment of inertia

This FB can be used to calculate the resultant moment of inertia depending on the current reel diameter.

- The value can be passed on to the FB **L_WndAccTorque** to calculate the current acceleration torque.
- The FB can also be used to adapt the speed controller gain.



➩ Inputs	(Variable type: VAR_INPUT)		
Identifier	**Data type**	**Signal type**	**Information/possible settings**
dwConstMInertia	Double word	-	Moment of inertia of the winder without material in [0.1 kg cm^2] • Value is internally limited to 0 ... 2000000000. • 1000 ≡ 100.0 kg cm^2
dwMaxMInertia	Double word	-	Moment of inertia of the winder (Jmax) when the largest reel is used in [0.1 kg cm^2] • Value is internally limited to 0 ... 2000000000. • 1000 ≡ 100.0 kg cm^2
wVMax	Word	-	Max. line speed in [0.1 m/min] • Value is internally limited to 0 ... 3000 m/min • 2500 ≡ 250.0 m/min
wMMaxMotor	Word	-	Max. motor control torque in [0.1 Nm] • Value is internally limited to 0 ... 3000 Nm. • 1250 ≡ 125.0 Nm • For the **9300 Servo PLC** this input is to be connected with the output *MCTRL_nMmaxC57* of the SB **MCTRL**.
nGearNum	Integer	-	Current gearbox factor - numerator • Value is internally limited to -32767 ... 1 / 1 ... 32767.
wGearDenom	Word	-	Current gearbox factor - denominator • Value is internally limited to 2 * |nGearNum | in the range of 1 ... 32767.
wDMin	Word	-	Min. diameter in [mm] • Value is internally limited to 1 ... 10000 mm.
nVarMInertia Adapt_a	Integer	analog	Proportional evaluation of the variable moment of inertia. • Value is internally limited to 0 ... 16384. • 16384 ≡ 100 %
nDiameter_a	Integer	analog	Actual diameter for the calculation of the variable moment of inertia • Value is internally limited to 1 ... 16384. • 16384 ≡ max. diameter
nDMin_a	Integer	analog	Min. diameter for the calculation of the variable moment of inertia • Value is internally limited to 1 ... 16384. • 16384 ≡ max. diameter

Outputs ⇨	(Variable type: VAR_OUTPUT)		
Identifier	**Data type**	**Signal type**	**Value/meaning**
dwMAccFactor	Double word	-	Resultant factor for FB **L_WndAccTorque** • Value is internally limited to 10000000.
nConstMInertia_a	Integer	analog	Constant moment of inertia • Referred to the maximum value. • 16384 ≡ 100 % of *dwMaxMInertia*
nVarMInertia_a	Integer	analog	Variable moment of inertia • Referred to the maximum value. • 16384 ≡ 100 % of *dwMaxMInertia*
nMInertia_a	Integer	analog	Current moment of inertia for the calculation of the acceleration torque using the FB **L_WndAccTorque** • Referred to the maximum value. • 16384 ≡ 100 % of *dwMaxMInertia*
nVpnAdapt_a	Integer	analog	Current signal for speed controller gain adaptation
bBusy	Bool	binary	Status signal “New calculation active” TRUE: Factor *dwMAccFactor* and moments of inertia are re-calculated.

Parameters	(Variable type: VAR_CONSTANT_RETAIN)						
Identifier	**Data type**	**Signal type**	**Info**				
				Possible settings			**Presetting**
wVpnGain	Word	-	Gain to adapt the current moment of inertia to the selected speed controller gain				
				0.00	{0.01}	100.00	1.00
nVpnAdaptMax_a	Integer	analog	Upper limit for speed controller adaptation				
				0.00	{0.01 %}	199.99	100.00 %
nVpnAdaptMin_a	Integer	analog	Lower limit for speed controller adaptation				
				0.00	{0.01 %}	199.99	100.00 %

Composition of the moment of inertia

The moment of inertia is composed of a machine and a material-dependent component.

- The machine-related mass moment of inertia is mainly caused by the mass moment of inertia of the motor. Other machine parts, e.g. a gearbox or a reel shaft can usually be neglected.

$$J_{machine} = J_{motor} + J_{gear} + \frac{J_{shaft}}{i^2}$$

Fig. 2-10 General formula for the machine-dependent moment of inertia

i = Gearbox factor

J = Moment of inertia

- The material-dependent moment of inertia is caused by the reel. Not only the reel diameter, but also the material width and the material density have to be considered for the calculation of the moment of inertia.

$$J_{reel} = \frac{\pi}{32 \cdot 10^8} \cdot B \cdot \rho \cdot \frac{D^4 - D_{Min}{}^4}{i^2}$$

Fig. 2-11 General formula for the material-dependent moment of inertia (referred to the motor end)

J_{reel} = Moment of inertia of the reel in [kg cm^2]

B = Material width in [mm]

ρ = Material density in [kg/dm^3]

D = Current diameter in [mm]

D_{min} = Sleeve diameter in [mm]

i = Gearbox factor

Selection of the moments of inertia

Use *dwConstMInertia* and *dwMaxMInertia* to select the physical values for the machine-dependent and the maximum possible moment of inertia.

- These values are normalized in the FB (*nConstMInertia_a* , *nVarMInertia_a*).
- The FB **L_WndIdentMInertia** is used for an automatic identification of the moment of inertia.

Material width and material density

For low material widths and/or material densities, the proportional reduction of the moment of inertia can be directly selected through *nVarMInertiaAdapt_a* .

Calculation/output of the normalization factor dwMAccFactor

Use *wVMax* , *wMMaxMotor* , *nGearNum* , *wGearDenom* and *wDMin* to select additional physical values for the calculation of the normalization factor *dwMAccFactor* .

$$dwMAccFactor = \frac{1}{3} \cdot \frac{J_{max} \cdot V_{max} \cdot i}{M_{max} \cdot D_{min}}$$

Fig. 2-12 Formula for the calculation of the normalization factor

J_{max} = max. moment of inertia in [0.1 kg cm^2]

V_{max} = max. line speed in [0.1 m/min]

i = Gearbox factor

M_{max} = max. motor control torque in [0.1 Nm]

D_{min} = min. reel diameter in [mm]

The calculated normalization factor is output at *dwMAccFactor* and can be transferred to the FB **L_WndAccTorque** to calculate the current acceleration torque.

Output of the current moment of inertia

The current moment of inertia is output at *nMInertia_a* and can be transferred to the FB **L_WndAccTorque** to calculate the current acceleration torque.

Adaptation of the speed controller gain

The speed controller setting depends on the mechanical characteristics of the winder. If motor and reel are considered as a rigid single-mass system, the optimum speed controller gain is directly proportional to the resultant moment of inertia:

$$V_{pn} = \pi \cdot \frac{N_{\max}}{M_{\max}} \cdot \frac{J_{\max}}{T_{vMReg}}$$

Fig. 2-13 Formula for the ideal speed controller gain

r_{max}	= max. speed of C0011
M_{max}	= max. torque of C0057
J_{max}	= max. moment of inertia in [kg m^2]
T_{vMReg}	= Equivalent time constant in [s]

Since the moment of inertia usually changes considerably during winding, the speed controller gain has to be adapted to ensure constant control characteristics.

- For this, the FB outputs a corresponding value at *nVpnAdapt_a* which can be transferred to the motor control.
  (For **9300 Servo PLC** : SB **MCTRL**, control input *MCTRL_nNAdapt_a*)

Tip!

In applications with a limited increase of the speed controller gain (e.g. two-mass system), it is either necessary to select a lower speed controller gain or to limit the output signal *nVpnAdapt_a* .

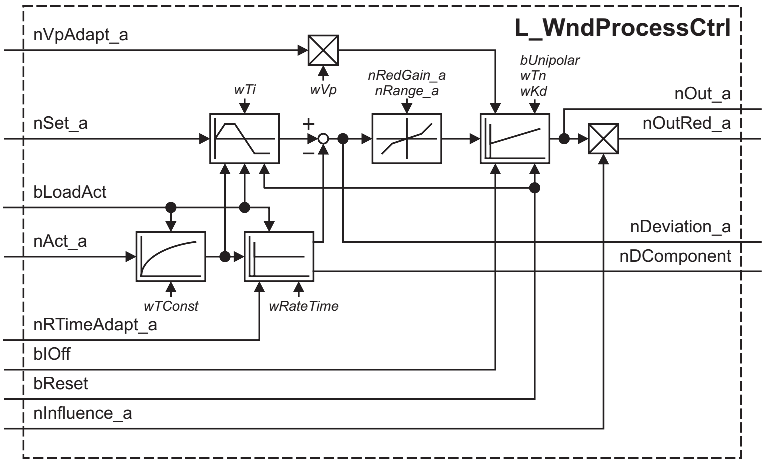
- Use *nVpnAdaptMax_a* to select the max. output signal value.
- It is also possible to define a minimum output signal value through *nVpnAdaptMin_a* .
- If a value lower than the ideal speed controller gain is selected for Vp, the mismatch between *nMInertia_a* and the selected speed controller gain can be compensated for by a correspondingly higher gain (*wVpnGain*).
  (For **9300 Servo PLC**: code C0070)

2.13 L_WndProcessCtrl - Dancer/tension control

This FB can be used for dancer or tension control.

Special function characteristics:

- PID controller characteristic with adaptable gain
- Reduced dynamic performance of the controller with little deviation
- Setpoint ramp generator can be loaded with actual value
- Low-pass filter and rate-action in the actual value feedback
- Deactivation of the integral component
- Deactivation of the process controller



⇨ Inputs	(Variable type: VAR_INPUT)		
Identifier	**Data type**	**Signal type**	**Information/possible settings**
nVpAdapt_a	Integer	analog	Proportional evaluation of the controller gain (Vp) • 16384 ≡ 100 % • Value is internally limited to 0 ... 16384.
nSet_a	Integer	analog	Controller setpoint
bLoadAct	Bool	binary	Acceptance of actual controller value TRUE: Actual controller value *nAct_a* is accepted into the ramp generator, the low pass and the rate action.
nAct_a	Integer	analog	Actual controller value
nRTimeAdapt_a	Integer	analog	Proportional evaluation of the rate time in the actual value channel • 16384 ≡ 100 % • Value is internally limited to 0 ... 16384.
bIOff	Bool	binary	Reset of the I-component of the controller TRUE: Resets I-component
bReset	Bool	binary	Reset of the entire control TRUE: Resets all outputs to 0
nInfluence_a	Integer	analog	Proportional evaluation of the correcting variable of the controller • With value 16384, the controller output signal is directly output.

Outputs ⇨	(Variable type: VAR_OUTPUT)		
Identifier	**Data type**	**Signal type**	**Value/meaning**
nOut_a	Integer	analog	Correcting variable of controller • Value is internally limited to -16384 ... 16384.
nOutRed_a	Integer	analog	Correcting variable of controller converted to the range of influence
nDeviation_a	Integer	analog	Deviation • Value is internally limited to -32767 ... 32767.
nDComponent	Integer	-	D-component of the rate action • Value is internally limited to -16384 ... 16384.

Parameters	(Variable type: VAR_CONSTANT_RETAIN)						
Identifier	**Data type**	**Signal type**	**Info**	**Possible settings**			**Presetting**
wTi	Word	-	Acceleration/deceleration time of setpoint ramp generator	0	{0.001 sec}	30	0.000 s
wTConst	Word	-	Filter time constant for actual controller value	0	{0.001 sec}	10	0.000 s
wRateTime	Word	-	Rate time for actual controller value	0	{0.001 sec}	30	0.000 s
nRedGain_a	Integer	analog	Deviation gain in the range of reduced sensitivity	0	{0.01 %}	100	100.00 %
nRange_a	Integer	analog	Deviation range with reduced gain/sensitivity	0	{0.01 %}	199.99	0.00 %
wVp	Word	-	Controller gain	0	{0.01}	100	0.10
wTn	Word	-	Adjustment time of controller	0.001	{0.001 sec}	30	1,000 s
wKd	Word	-	Rate action of controller	0	{1 msec}	1000	0 ms
bUnipolar	Bool	binary	Limiting the correcting variable of the controller to the positive range	FALSE	Limitation not active		FALSE
				TRUE	Limitation active		

Control characteristic

The dancer/tension controller can be operated as a P, PI or PID controller.

- In the default setting, the PI algorithm is active.

Gain Vp (P component)

- With the parameter *wVp* you can set the controller gain.
- The gain can be directly adapted via the input *g_nVpAdapt_a* .

Adjustment time Tn (I component)

- With the parameter *wTn* you can set the adjustment time of the controller.
- Use *bIOff* to activate or deactivate the integral-action component of the controller.

Rate time (D component)

- With the parameter *wKd* you can set the rate time of the controller. A value > ”0 ms” activates the D component of the controller.

Controller influence

With motor speed or motor torque precontrol, there is usually only a little controller influence required to meet the setpoint.

- Use *nPCtrlInfluence_a* to select the controller influence.

Note!

The controller influence evaluates the output signal by multiplying it. When the influence is changed, the dynamic performance of the controller changes as well!

Switching the controller on/off

You can use the input *bReset* to switch on/off the controller.

Loading the setpoint ramp generator with the actual value

If the actual value is loaded in the setpoint ramp generator when the controller is switched off, the deviation will be 0 when the controller is enabled again. In this way, compensation is hardly required.

- Set *bLoadAct* to TRUE to load the actual value in the setpoint ramp generator. The control difference is thus 0 and the I-component of the controller kept.

Low pass and rate action in the actual-value channel

It is possible to activate a low pass in the actual-value channel to improve the filtering of signal interferences.

- With the parameter *wTConst* you can set the filter time constant for the low pass.
- With the parameter *wRateTime* you can set the rate time constant for the feedback channel. In this way, interfering delay times can be compensated.

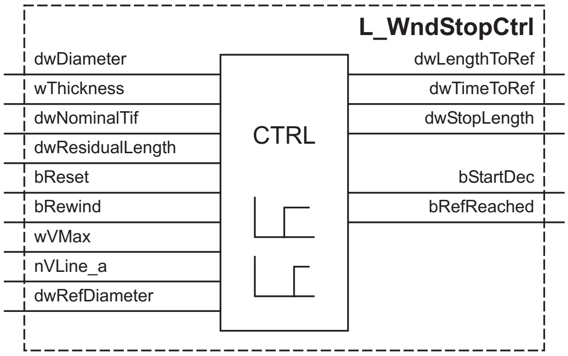
Reduced dynamic controller performance with little control difference

A reduced dynamic controller performance with little control difference usually has a positive influence on the damping characteristics of the control circuit.

- With the parameter *nRange_a* you can set the tolerance range within which the control difference is transferred to the controller with a lower gain.
- With the parameter *nRedGain_a* you can select to how many percent the gain is to be reduced within the preset tolerance range.

2.14 L_WndStopCtrl - Stop controller

This FB can be used to monitor the residual length to the reference diameter based on the current diameter and the material thickness. The output signals for timely braking and final stop are provided.



⇨ Inputs	(Variable type: VAR_INPUT)		
Identifier	**Data type**	**Signal type**	**Information/possible settings**
dwDiameter	Double word	-	Current reel diameter in [µm] • Value is internally limited to 10 m. • 1000 ≡ 1 mm
wThickness	Word	-	Current material thickness in [µm] • Value is internally limited to 1 ... 30000 µm. • 1000 ≡ 1 mm
dwNominalTif	Double word	-	Nominal deceleration time for braking in [s] • Activated through message "Start braking" (*bStartDec* = TRUE). • Value is internally limited to 0 ... 10000 s.
dwResidualLength	Double word	-	Residual length after the stop in [mm] • Value is internally limited to 0 ... 100000 mm. • 1000 ≡ 1 m
bReset	Bool	binary	Reset of the stop controller TRUE: Resets stop controller and digital output signals.
bRewind	Bool	binary	Selection of unwind/rewind operatoion FALSE: Unwind operation TRUE: Rewind operation
wVMax	Word	-	Max. line speed in [0.1m/min] • Value is internally limited to 0 ... 3000 m/min. • 2500 ≡ 250.0 m/min
nVLine_a	Integer	analog	Current line speed • Referred to the maximum value *wVMax*
dwRefDiameter	Double word	-	Reference diameter for pre-stop and stop signal in [µm] • Value is internally limited to 10 m. • 1000 ≡ 1 mm

Outputs ⇨	(Variable type: VAR_OUTPUT)		
Identifier	**Data type**	**Signal type**	**Value/meaning**
dwLengthToRef	Double word	-	Residual length to reference diameter in [mm]
dwTimeToRef	Double word	-	Residual operating time to reference value in [s] • Assuming that speed = *wVMax*. • This value is only calculated every second FB call (alternating with *dwStopLength*).
dwStopLength	Double word	-	Change of length during braking in [mm] • This value is only calculated every second FB call (alternating with *dwTimeToRef*).
bStartDec	Bool	binary	Status signal "Start braking" TRUE: Starts braking.
bRefReached	Bool	binary	Status signal "Reference diameter reached" TRUE: The reference diameter has been reached.

Display of the residual length/residual operating time

Use *dwLengthToRef* to display the residual length to the reference diameter.

- Use *bRewind* to select if the reference diameter is to be lower (unwinder) or higher (rewinder) than the current diameter.
- When the reference diameter has been reached or exceeded, the residual length = 0.
- Assuming that the max.line speed *wVMax* is used the residual operating time *dwTimeToRef* can be caluclated from the residual length.

Activation of the braking operation

For activating the braking operation, the residual length to the reference diameter is compared with the residual length after the stop (*dwResidualLength*) including the change of length during braking (*dwStopLength*).

- The change of length during braking is output at *dwStopLength* and depends on the current line speed calculated from *wVMax* and *nVLine_a* and the nominal deceleration time *dwNominalTif*.
- The signal that releases the braking operation is output at *bStartDec*.

Activation of a final stop

For activating a final stop, the residual length to the reference diameter is compared to zero (current diameter = reference diameter).

- The signal that releases the final stop is output at *bRefReached*.

Reset of the two release signals

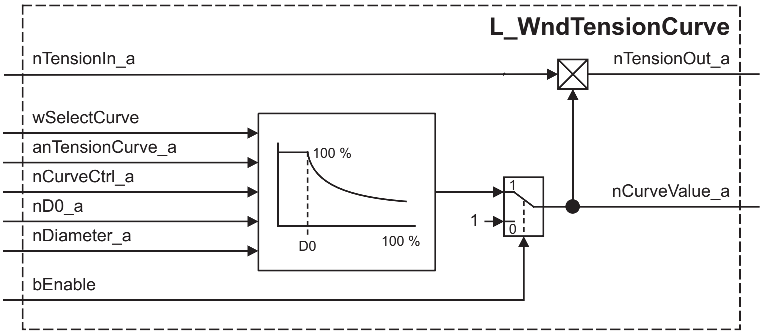
By setting *bReset* to TRUE you can reset the two digital outputs for releasing braking/final stop.

2.15 L_WndTensionCurve - Tension characteristic

With centre winders, the driving torque is transmitted from the centre via the individual layers to the surface of the reel. With smooth materials and large diameter ranges, the tension usually has to be reduced depending on the diameter.

This FB can be used to generate a diameter-dependent tension characteristic.

- The characteristic is divided into an initial range with a constant evaluation (100 %) and a second range in which the tension is adjusted to the diameter.

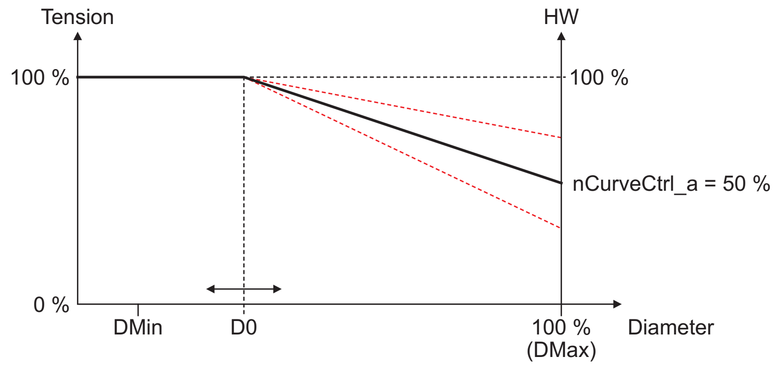


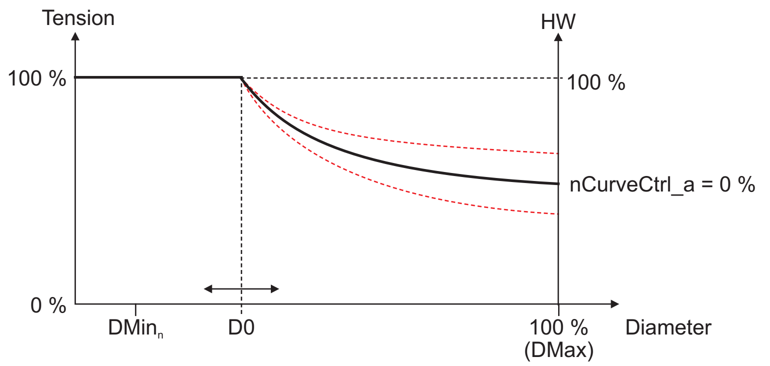
⇨ Inputs	(Variable type: VAR_INPUT)		
Identifier	**Data type**	**Signal type**	**Information/possible settings**
nTensionIn_a	Integer	analog	Max. tension setpoint
wSelectCurve	Word	-	Characteristics selection 0 Linear tension characteristics 1 Linear torque characteristics 2 Tension characteristics according to a set characteristic (*anTensionCurve_a*)
anTension Curve_a[0..64]	Array of integers	analog	Array for characteristics table • The 65 values are distributed over the diameter range of *nDiameter_a* = 0 ... 16384. • The values are internally limited to 1 ... 32767.
nCurveCtrl_a	Integer	analog	Slope of the characteristic (torque characteristics) • Value is internally limited to -16383 ... 32767. • 16384 ≡ 100 % • With 100 %, the tension characteristic is constant and the torque rises proportionally to D.
nD0_a	Integer	analog	Origin of the characteristic • Referred to *DMax* . • 16383 ≡ 100 % of *DMax*
nDiameter_a	Integer	analog	Current diameter • Referred to *DMax* . • 16384 ≡ 100 % of *DMax*
bEnable.	Bool	binary	Activation of tension setpoint FALSE Tension setpoint is not influenced. TRUE Tension setpoint is multiplied by the characteristic value.

Outputs ⇨	(Variable type: VAR_OUTPUT)		
Identifier	**Data type**	**Signal type**	**Value/meaning**
nTensionOut_a	Integer	analog	Current tension setpoint
nCurveValue_a	Integer	analog	Current characteristic value • The characteristic values are internally limited to 1 ... 32767.

Characteristics selection

The part of the characteristic that depends on the diameter can be generated for linear tension characteristics, linear torque characteristics or by means of a preset characteristic. The selection is made through *wSelectCurve* :

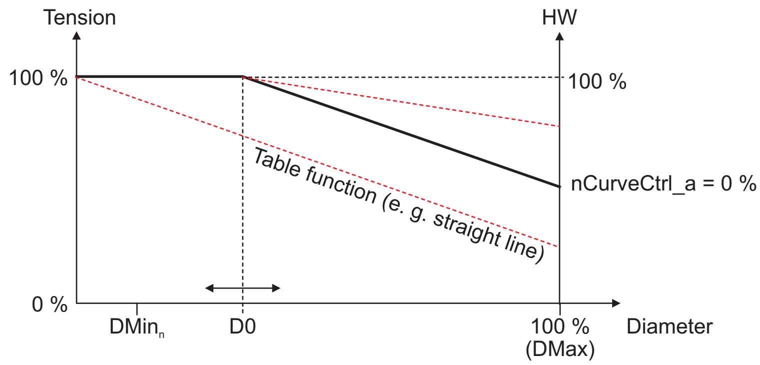
wSelectCurve = 0	**Characteristic for linear tension characteristics**
Characteristic	
Function for D > D0	$16384 - \frac{D - D0}{16384 - D0} \cdot (16384 - HW)$ D Current diameter (*nDiameter_a*) D0 Starting point for diameter dependence (*nD0_a*) HW Tension evaluation (*nCurveCtrl_a*)

wSelectCurve = 1	**Characteristic for linear torque characteristics**
Characteristic	
Function for D > D0	$HW + \frac{(16384 - HW) \cdot D0}{D}$ D Current diameter (*nDiameter_a*) D0 Starting point for diameter dependence (*nD0_a*) HW Tension evaluation (*nCurveCtrl_a*)

Tip!

For a table-value based characteristic:

1. Create a variable (ARRAY [0..64] of INT) and assign it to a code. Now, you can enter the characteristic values and save them permanently.
2. Connect the variable with the input *anTensionCurve_a* .

wSelectCurve = 2	Table-value-based characteristic
	• *nD0_a* defines the beginning of the table range. • Use *nCurveCtrl_a* to change the slope of the characteristic. (If *nCurveCtrl_a* = 0 % the slope of the characteristic remains unchanged.)
Characteristic	
Function for D > D0	$16384 - \frac{(16384 - TAB[D - D0]) \cdot (16384 - HW)}{16384}$ TAB[D-Do] Table value for D-D0 (*anTensionCurve_a[0..64]*) D0 Starting point for diameter dependence (*nD0_a*) HW Tension evaluation (*nCurveCtrl_a*)

3 Index