
SIEMENS

# SIMATIC Winder and Tension Control

Manual

https://support.industry.siemens.com/cs/ww/en/view/58565043

Siemens Industry Online Support

Unrestricted

# Legal information

**Use of application examples**

Application examples illustrate the solution of automation tasks through an interaction of several components in the form of text, graphics and/or software modules. The application examples are a free service by Siemens AG and/or a subsidiary of Siemens AG ("Siemens"). They are non-binding and make no claim to completeness or functionality regarding configuration and equipment. The application examples merely offer help with typical tasks; they do not constitute customer-specific solutions. **The application examples are not subject to standard tests and quality inspections of a chargeable product and may contain functional and performance defects or other faults and security vulnerabilities. You are responsible for the proper and safe operation of the products in accordance with all applicable regulations, including checking and customizing the application example for your system, and ensuring that only trained personnel use it in a way that prevents property damage or injury to persons. You are solely responsible for any productive use.**

Siemens grants you the non-exclusive, non-sublicensable and non-transferable right to have the application examples used by technically trained personnel. Any change to the application examples is your responsibility. Sharing the application examples with third parties or copying the application examples or excerpts thereof is permitted only in combination with your own products. Any further use of the application examples is explicitly not permitted and further rights are not granted. You are not allowed to use application examples in any other way, including, without limitation, for any direct or indirect training or enhancements of AI models.

**Disclaimer of liability**

Siemens shall not assume any liability, for any legal reason whatsoever, including, without limitation, liability for the usability, availability, completeness and freedom from defects of the application examples as well as for related information, configuration and performance data and any damage caused thereby. This shall not apply in cases of mandatory liability, for example under the German Product Liability Act, or in cases of intent, gross negligence, or culpable loss of life, bodily injury or damage to health, non-compliance with a guarantee, fraudulent non-disclosure of a defect, or culpable breach of material contractual obligations. Claims for damages arising from a breach of material contractual obligations shall however be limited to the foreseeable damage typical of the type of agreement, unless liability arises from intent or gross negligence or is based on loss of life, bodily injury or damage to health. The foregoing provisions do not imply any change in the burden of proof to your detriment. You shall indemnify Siemens against existing or future claims of third parties in this connection except where Siemens is mandatorily liable.

By using the application examples you acknowledge that Siemens cannot be held liable for any damage beyond the liability provisions described.

**Other information**

Siemens reserves the right to make changes to the application examples at any time without notice and to terminate your use of the application examples at any time. In case of discrepancies between the suggestions in the application examples and other Siemens publications such as catalogs, the content of the other documentation shall have precedence.

The Siemens terms of use (https://www.siemens.com/global/en/general/terms-of-use) shall also apply.

**Cybersecurity information**

Siemens provides products and solutions with industrial cybersecurity functions that support the secure operation of plants, systems, machines and networks.

In order to protect plants, systems, machines and networks against cyber threats, it is necessary to implement – and continuously maintain – a holistic, state-of-the-art industrial cybersecurity concept. Siemens' products and solutions constitute one element of such a concept.

Customers are responsible for preventing unauthorized access to their plants, systems, machines and networks. Such systems, machines and components should only be connected to an enterprise network or the internet if and to the extent such a connection is necessary and only when appropriate security measures (e.g. firewalls and/or network segmentation) are in place.

For additional information on industrial cybersecurity measures that may be implemented, please visit https://www.siemens.com/cybersecurity-industry.

Siemens' products and solutions undergo continuous development to make them more secure. Siemens strongly recommends that product updates are applied as soon as they are available and that the latest product versions are used. Use of product versions that are no longer supported, and failure to apply the latest updates may increase customer's exposure to cyber threats.

To stay informed about product updates, subscribe to the Siemens Industrial Cybersecurity RSS Feed under https://www.siemens.com/cert.

# Table of contents

# 1 Overview

## 1.1 SIMATIC Converting Toolbox

The following library for realizing a winder/ tension control is part of the SIMATIC Converting Toolbox:

https://support.industry.siemens.com/cs/ww/en/view/109744606

The SIMATIC Converting Toolbox was developed with the objective to provide a collection of common standard functions which are necessary for any kind of converting machines. Converting describes the process of processing/finishing any kind of material. In most cases this starts with unwinding a material web which then is processed in one or several steps in machine types like

- Printing machines
- Coating lines
- Laminating machines
- Slitters
- Web Inspection machines
- Wire drawing machines
- Extrusion lines
- Textile machines
- Diaper machines

The open source library contains tested and well-documented functions to provide support to the user during development, programming and commissioning. Well-structured interfaces as well as documented program code provide the possibility to make customer-specific adaptations.



## 1.2 Core functions

The core functions of the SIMATIC Converting Toolbox are functions and functions blocks to control a winding axis respectively tension control axis. This includes mainly speed and torque setpoint calculations which are necessary for different selectable control modes. Further on the Toolbox provides sub-functions used typically in winding and tension control systems, such as

- Diameter calculation
- Torque pre-control
- Gain adaption
- Tension taper characteristic

**Diameter calculation**

Six different techniques are available to calculate the diameter of the roll being wound:

- Thickness addition: The diameter is calculated by adding the material thickness
- Web length calculation: The diameter is calculated using the length of the winded material and the material thickness
- v/n calculation: The diameter is calculated using the ratio between the velocity of the material web and the winder speed
- v/n integral: The diameter is calculated from the ratio between the material length and the revolutions of the winder (integration)
- s/alpha position: Ratio between web position and winder position
- External diameter: The diameter comes from extern (sensor)

### Torque pre-control

The main component of course, is the motor torque which is necessary to build up the desired web tension. Beside of that friction torque, acceleration torque and the torque resulting from a changing roll diameter can be pre-controlled. This pre-control signal is transferred to the drive as additive torque, being calculated inside the Toolbox function blocks.

### Controller Adaptation

The gain of the technology controller can be adapted using a free adaption signal (e.g., diameter, web velocity etc.) The speed controller gain in the drive can be adapted as a function of the moment of inertia or the actual diameter of the roll being wound. This is necessary to guaranty a well optimized and dynamic speed controller even the diameter ratio (full roll to empty roll) is big.

### Tension taper characteristic

A tension taper characteristic might be required at rewinding systems in order to decrease the rewinding tension as a function of the increasing roll diameter. Five different techniques are available to calculate the tension taper characteristic:

- No characteristic
- Linear characteristic – the tension is linearly reduced as a function of the actual diameter
- Hyperbolic characteristic type 1 – the tension decreases according to a hyperbolic function/characteristic
- Hyperbolic characteristic type 2 – the tension decreases according to a hyperbolic function/characteristic
- Interpolated characteristic using a table of points – 10 points can be entered to define the characteristic



## 1.3 Additional functions

Beside the core functions – winding and tension control – the SIMATIC Converting Toolbox provides many useful functions and function blocks which are necessary in converting applications. The following list shows a short overview about some of them:

- PID controller
- Ramp function generator
- Stop distance calculation
- PT1/DT1 filter
- Integrator/Differentiator

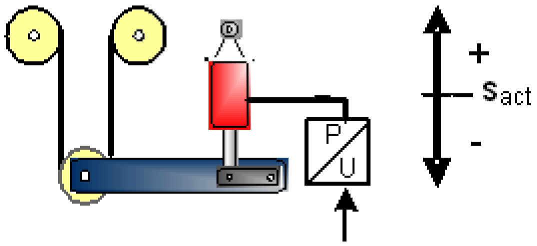
# 2 Basic information

## 2.1 Winder drives

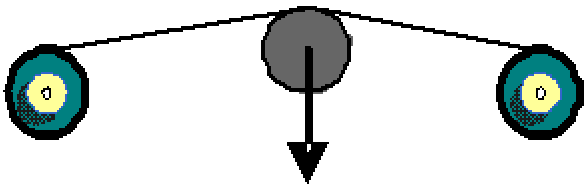
The function of a winder is to rewind or unwind a material web with a defined tension. During the winding process the diameter of the roll changes. The diameter needs to be detected or calculated to adjust the motor speed to keep the tension of the material constant. If the winding system must fulfill higher demands regarding the performance and tension accuracy, the actual web tension can be detected my measuring devices (dancer system or load cell) to use the feedback signal as input of a controller – here named “technology controller” (TPID).

There are two general possibilities to detect the web tension:

The **dancer roll** is a position-controlled measuring system. The material web is threaded through this dancer roll system. A cylinder with a deflection roll presses, with an adjustable counterforce against the material web. If tension is established in the material web, then this acts against the pressure from the dancer roll. The dancer roll is at its center position, if pulling force of the material web and counterforce of the dancer system is in balance. In order to control (closed-loop) the tension in the material web, the technology controller must ensure that the dancer roll remains at its center position. If the tension in the system changes, then the position of the dancer roll also changes – that is in turn coupled with the technology controller. Web tension variations – detected by dancer position variations - can either be compensated by changing the motor speed or by adapting the motor torque.



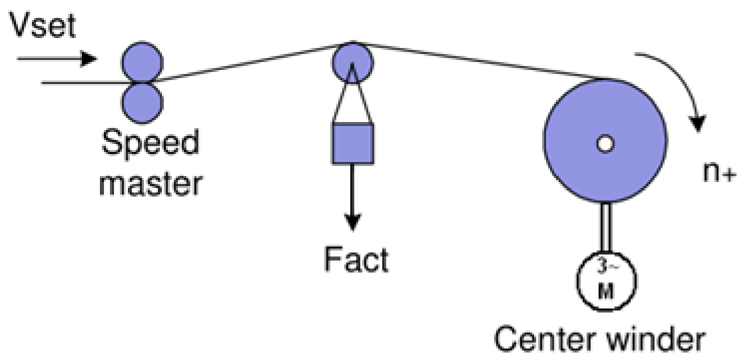

The **load cell** directly measures the tension in the material web and transfers this signal to the technology controller. Tension variations can either be compensated by changing the motor speed or by adapting the motor torque.



In general, there are two types of winding techniques:

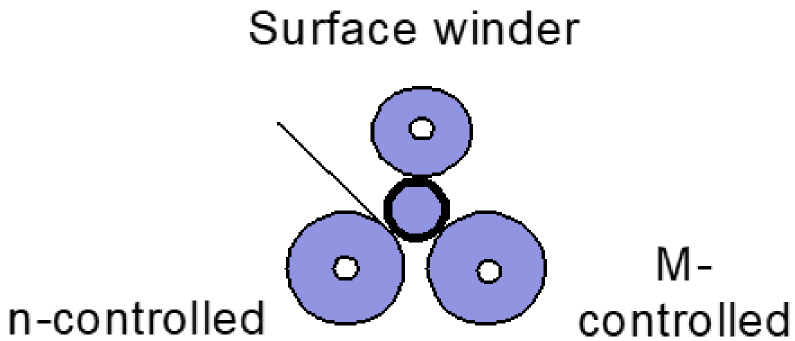
### Center winder

With a center winder, the roll is driven by a central shaft. The diameter range is an important factor when designing this winder type. The reason for this is that for a constant web velocity and constant tension, the speed is inversely proportional to the diameter. This means that the maximum required drive speed is defined by the minimum roll diameter – whereby the maximum required torque is defined by the maximum diameter. The center winder is more complicated and from control perspective more difficult to handle than the surface winder; however, it is still more widely established of the two winder types.




**Surface winder**

For a surface winder, the roll is driven through one or several rolls that are in contact with the roll being wound. The drive speed and power depend on the diameter of the roll being wound. From a mechanical design perspective, this winder technique is more complicated than that of a center winder. The contact winder (surface winder) is essentially used if there are no special requirements placed on the surface quality of the material being wound.




## 2.2 Tension control drives (Sectional drives)

The functionality of a tension control axis is identical to a winder axis (Same possibilities of web tension detection, tension control modes etc.). The only difference is the missing dependencies of a chaning diameter.

**Note** For more information see description on winder drives

## 2.3 Winding/Tension control modes

The following tension control concepts are available for winding and sectional drive functionality:

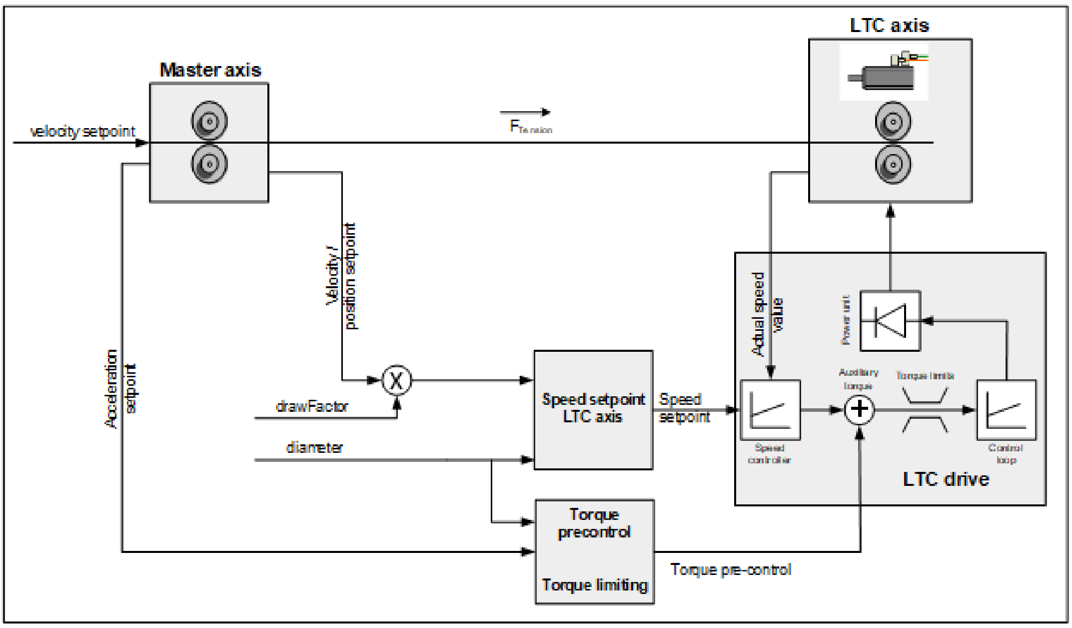
- Open-loop control modes:
  - Draw control (sectional drive only)
  - Indirect Tension Control via torque adaption
  - Indirect tension control via speed adaption
  - V-Constant Control (winder only)

- Closed-loop control modes:
  - Tension Control Torque Adaption using Load-cell
  - Tension Control Speed Adaption using Load-cell
  - Dancer Control Speed Adaption

These control concepts will be briefly explained in the following.

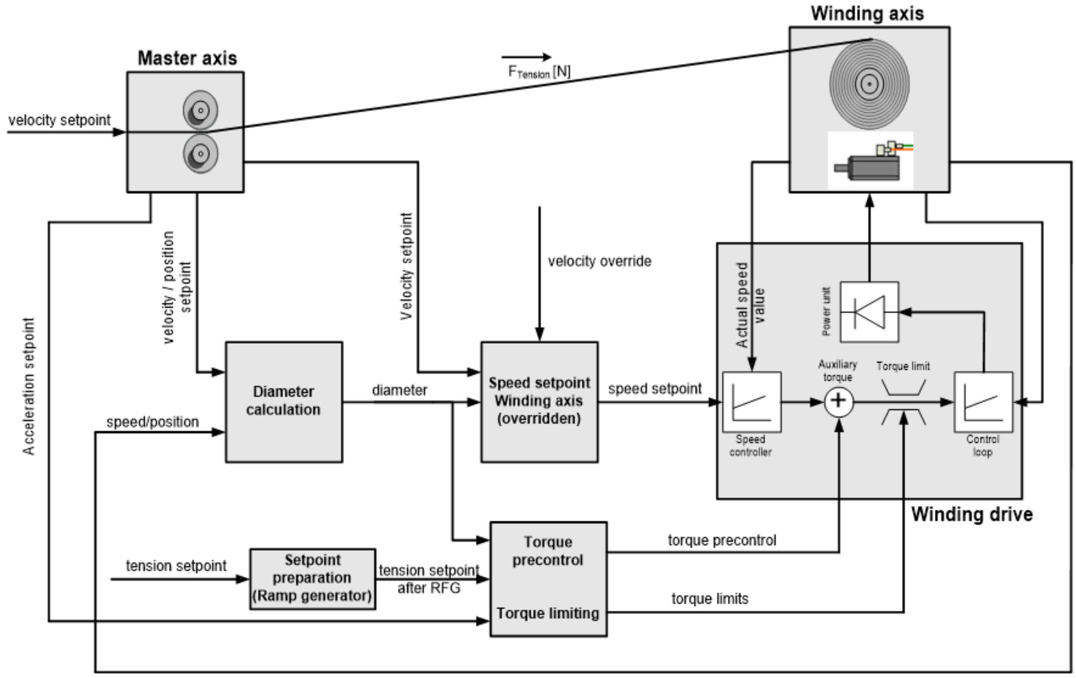
## 2.3.1 Draw control

This method is used for simple tension regulation, especially when any additional measuring system is not available. It does not offer high level tension accuracy. It can't be used for center driven winders. Using this method, the web tension is set by the segment's velocity ratio, means that the velocity of the transport rolls at the segment start is slower than at the segment end. The velocity difference causes material strain which defines the web tension. The resulting tension depends on the material type, the friction between the material and the transport roll (slip) and the transport system itself.




## 2.3.2 Indirect Tension Control via torque limits

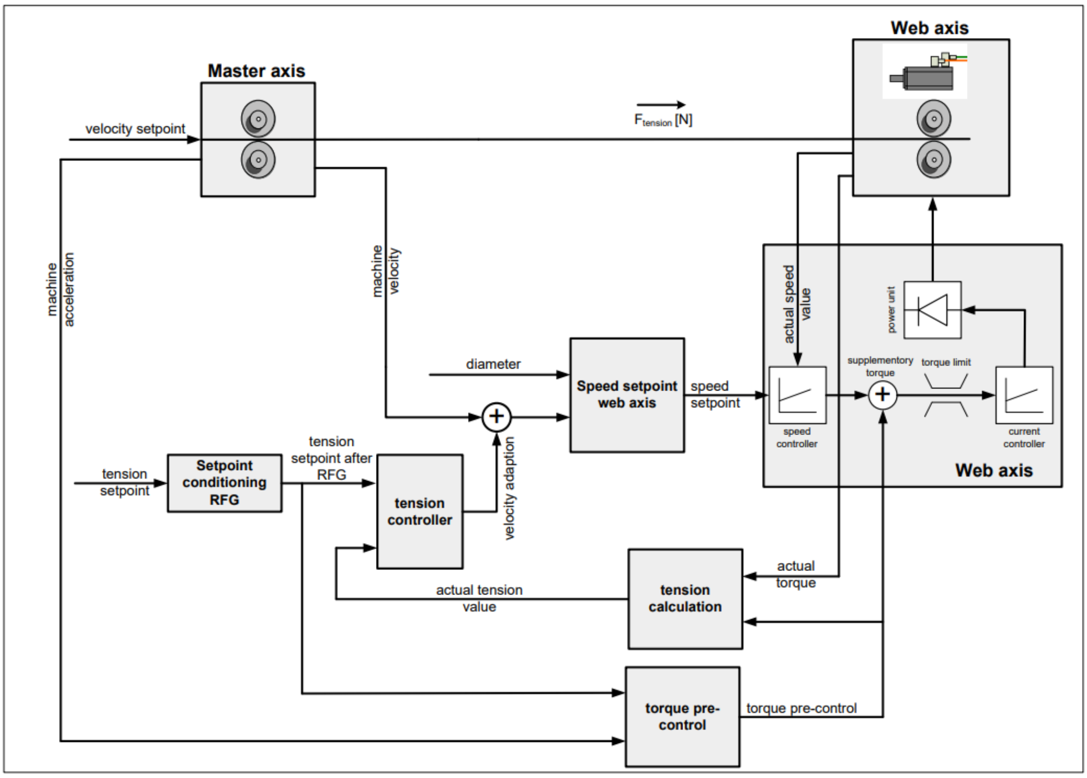
The indirect tension control is very frequently used if a user does not want to use expensive sensor systems as there is no higher-level tension control necessary. There is no measuring system necessary, as there is no feedback used to control the tension - the technology controller is not used. To set the web tension, the tension setpoint is computed into a torque value which is based on the roll diameter and the gear ratio. The speed controller in the drive is overdriven to operate the motor at the torque limit. This torque limit is being adapted as a function of the diameter of the roll being wound or the fix diameter of the roll in case of sectional drive axis. To reach an accurate web tension with this control method it is important that the friction and accelerating torque are precisely compensated in order the pre-controlled tension torque results in the best possible approximation to the required material tension. In this control mode the mechanical losses should be kept as small as possible. This includes an appropriate motor selection and the selection of a highly efficient gearbox with low friction losses.





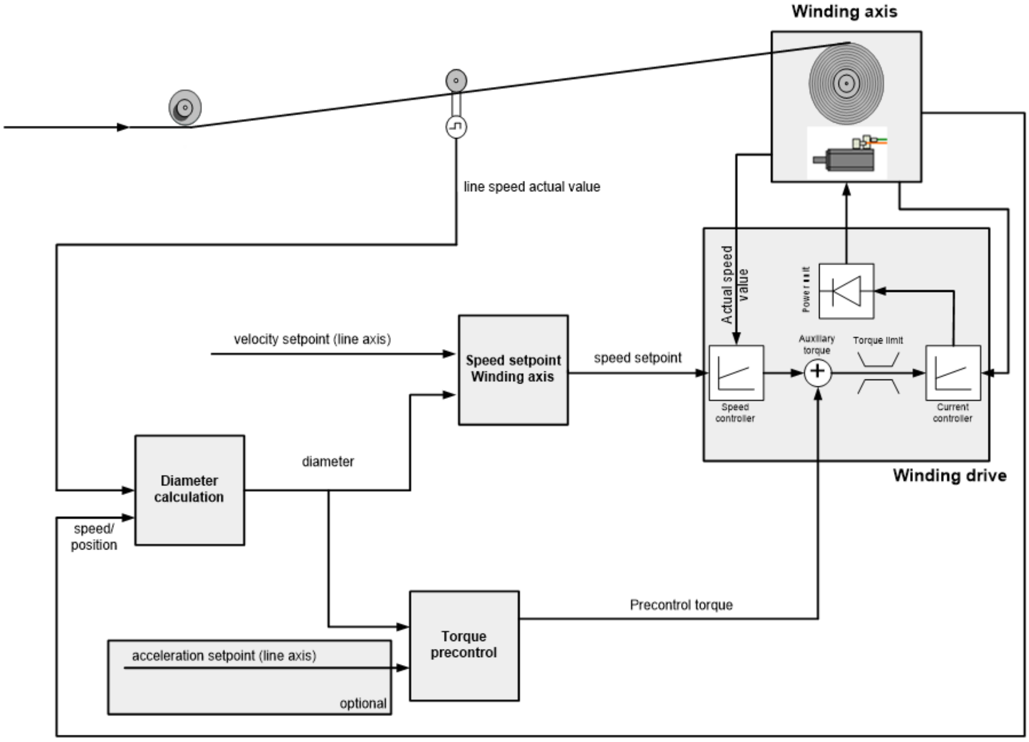

### 2.3.3 Indirect Tension Control via speed adaption

This method is a mixture of the tension control with load cell and of the indirect tension control methods. In this case the actual tension is not measured but calculated from the actual motor torque value. The friction and acceleration compensation is absolutely necessary to reach an as accurate web tension as possible. Using this method the technology controller is activated to control the actual tension value. The calculated tension actual value is taken as feedback signal for the control loop, this will get compared with the ramped tension setpoint and the controllers output is then interppreted as additional velocity setpoint. The accuracy of this method is comparable with the accuracy of indirect tension control




## 2.3.4 V-constant control

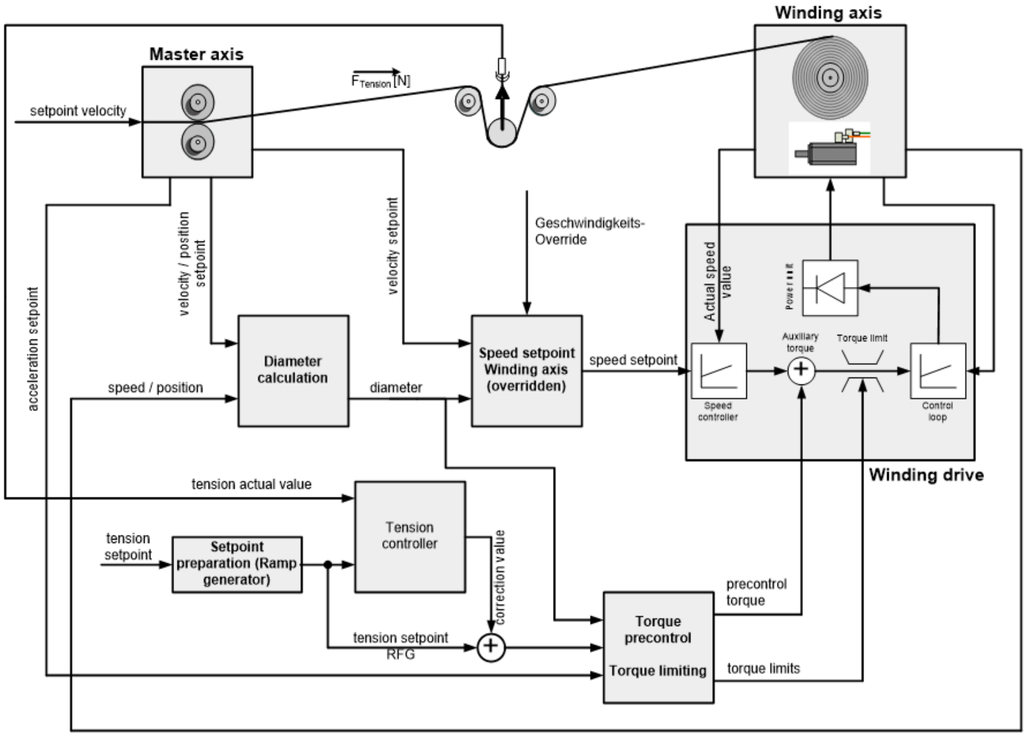
This method is used for simple tension regulation, especially when any additional measuring system is not available. It does not offer high level tension accuracy. In this control mode the winder is controlled to run with constant web speed (web speed command value). No tension control is active.






### 2.3.5 Tension Control Torque Adaption using Load-cell

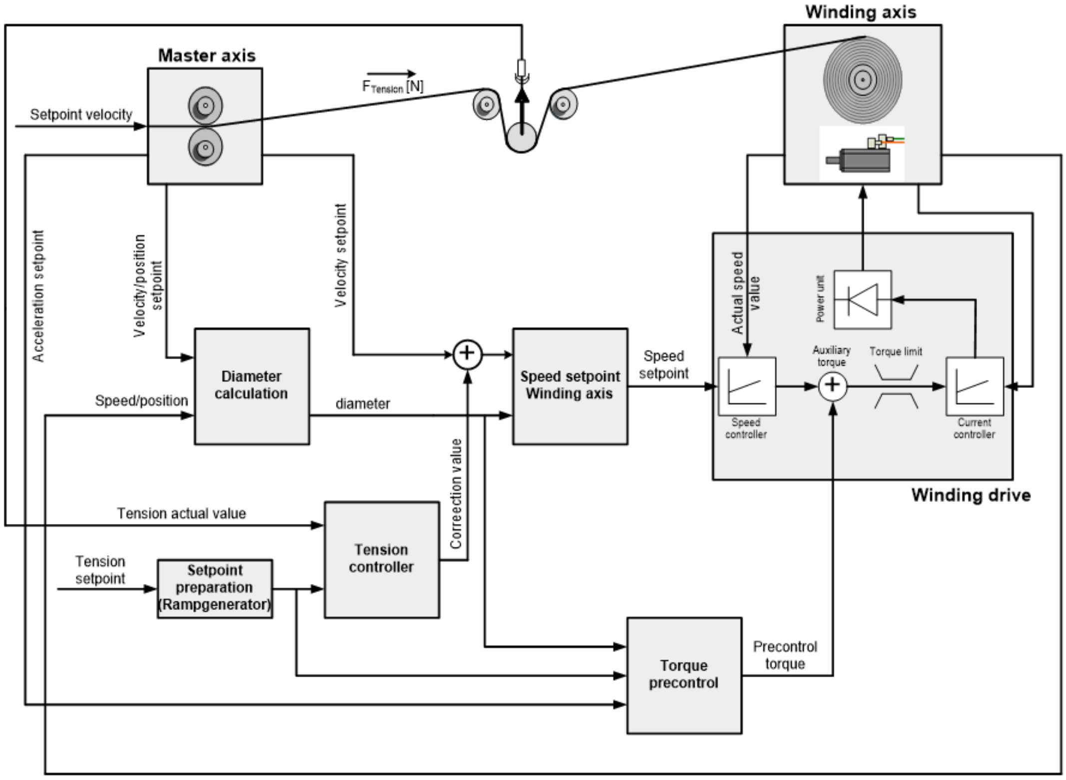
In this control mode the web tension is directly detect by a measuring device – a load cell. Its output signal is used inside the technology controller as actual value signal. The tension control is used if the precision of the open-loop torque control is no longer sufficient or there are disturbances in the system which needs to be eliminated. Just as the same as for indirect tension control, the speed controller in the drive is overdriven to operate the motor at the torque limit. This torque limit is being adapted by the technology controller. Supplementary torque setpoints – e.g., from the acceleration torque pre-control or friction torque pre-control are connected-up so they act additively to the torque limits.
Using a load cell, it is not necessary to interfere the web path, however there is not the advantage of having a buffer like a dancer provides.






### 2.3.6 Tension Control Speed Adaption using Load-cell

In this control mode the web tension is directly detect by a measuring device – a load cell. Its output signal is used inside the technology controller as actual value signal. The tension control is used if the precision of the open-loop torque control is no longer sufficient or there are disturbances in the system which needs to be eliminated. Contrary to the control modes with torque adaption, the output signal of the technology controller is an additional speed value. The speed controller in the drive gets the accurate speed setpoint to achieve the desired web tension. The motor is not operating at the torque limits. Supplementary torque setpoints – e.g., from the acceleration torque pre-control or friction torque pre-control are connected-up to the drive. Using a load cell, it is not necessary to interfere the web path, however there is not the advantage of having a buffer like a dancer provides.



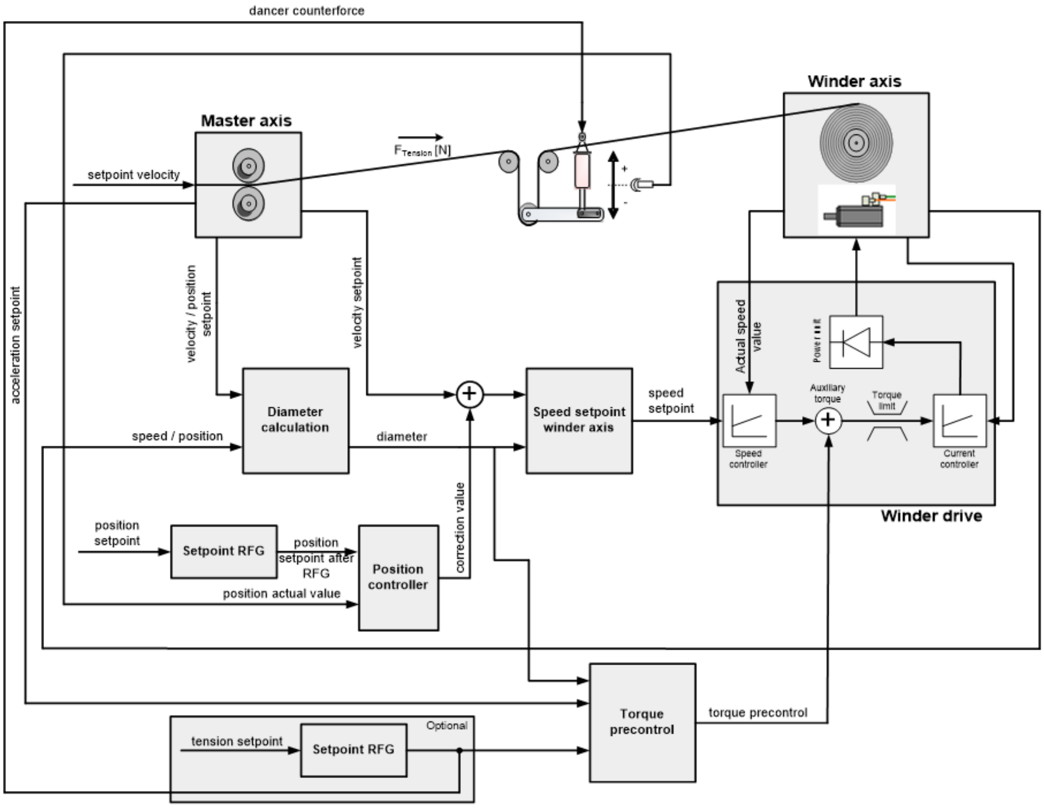



### 2.3.7 Dancer Control Speed Adaption

In this control mode – contrary to the load cell – it is not the web tension that is controlled by the technology controller directly, but the dancer position. If the web tension changes, also the dancer position changes. The position of the dancer roll is determined using a suitable position encoder, which is then compared with the position setpoint in the technology controller. The controller output is an additive velocity setpoint (weighted with the actual diameter) to maintain the setpoint position of the dancer. The dancer roll attempts to deflect the material web with a defined force. The final web tension is a result of the pulling force to the dancer and its counterforce. The counterforce of the dancer – in consequence the web tension can be defined in different ways, dependent on the mechanical construction of the dancer system. An easy way is a counterforce generated with spring power. However, the spring force and due to this the web tension can only be adjusted by the dancer angle. More flexible adjustable counterforce can be achieved with pneumatic or hydraulic dancer system. These systems can keep the counterforce constant even the dancer angle changes. To do this, the tension setpoint signal is connected to an analog output and is used as setpoint for the dancer force. Often, there are additional factors that influence the tension actual value because of the geometrical arrangement (distance to reversing rollers that may be used) and the weight of the dancer roll itself. However, with a careful mechanical design, these effects can be eliminated or sufficiently minimized. The dancer position control has the advantage that brief fluctuations in the web tension – caused by disturbances - can be absorbed due to the material buffer function of the dancer roll. Another advantage of the dancer system is the damping effect on

- Non-round material rolls
- Jumps between layers – e.g., when winding cables
- Influences during flying roll changes (splice)

However, an intervention must be made in the material web routing as a result of the mechanical arrangement. Even though closed-loop dancer roll controls are quite complex, they offer the best control technique.



dancer counterforce
Master axis
Winder axis
F Tension [N]
setpoint velocity
acceleration setpoint
velocity / position setpoint
velocity setpoint
Diameter calculation
Speed setpoint winder axis
speed setpoint
Actual speed value
Auxiliary torque
Torque limit
Speed controller
Current controller
Winder drive
speed / position
diameter
correction value
position setpoint
Setpoint RFG
position setpoint after RFG
Position controller
position actual value
Torque precontrol
torque precontrol
Optional
tension setpoint
Setpoint RFG

# 3 Working with the library

## 3.1 Integrating the library into TIA Portal

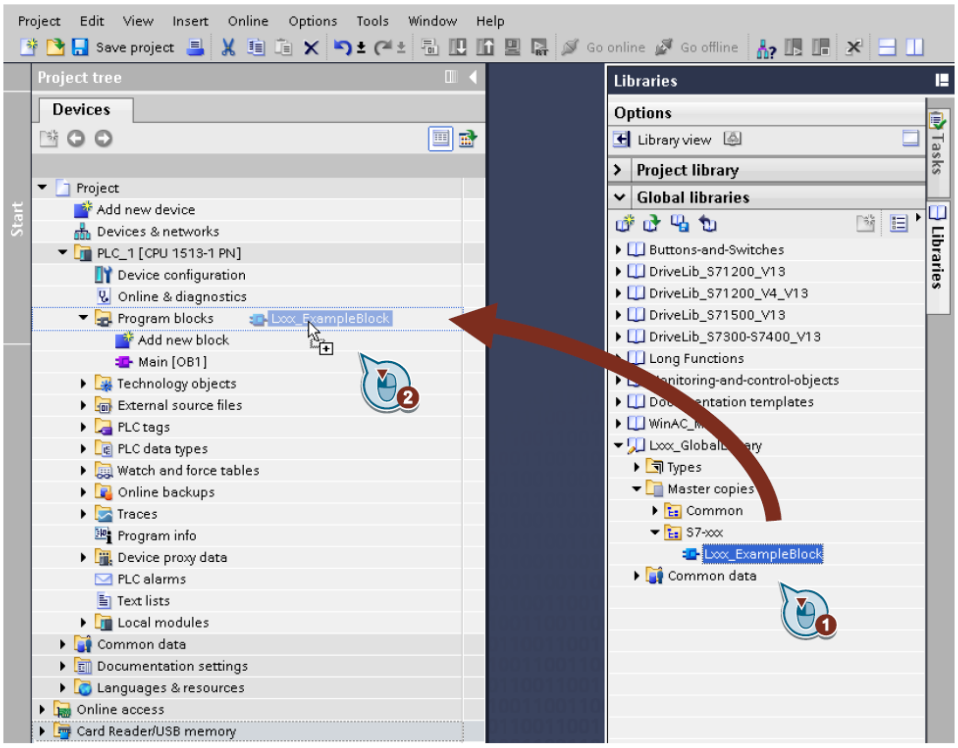
The following steps must be done to integrate the LConSMC library into your STEP 7. Subsequently, you can use the blocks of the LConSMC library.

1. Extract the library LConSMC_Vx_x_x.zip to a local folder.
2. In TIA Portal select “Options” -> “Global libraries” -> “Open library…”
3. Browse to the file LConSMC.alxx
4. It can be found in the subfolder LConSMC of the extracted zip file
5. Open the global library in read-only mode

## 3.2 Integrating the library blocks into TIA Portal

The required function blocks of the library can be found in the “Master copies” folder. Integrate the following folders via Drag&Drop into your project:

- LConSMC_Blocks into Program blocks folder
- LConSMC_Tags into PLC Tags folder
- LConSMC_Types into PLC Types folder




**Note** It is recommended to copy the blocks, tags and types with the appropriate higher-level folder “LConSMC_...”

## 3.3 Calling the function blocks

The winder/ sectional drive function block must be called in a cyclic organization block. For dynamic applications or if torque precontrol is activated it is recommended to call the function blocks in the MC-PreInterpolator [OB68]. Otherwise, it is also possible to call the function blocks in a cyclic interrupt OB.

**Note**

The cycle time in which the function block is called must be connected to the function block input "cycleTime" in milliseconds.

## 3.4 Configuration of the technology objects

### Axis type

For winder and sectional drive applications all kind of axes technology objects are possible:

- TO_SpeedAxis
- TO_PositioningAxis
- TO_synchronousAxis

**Note**

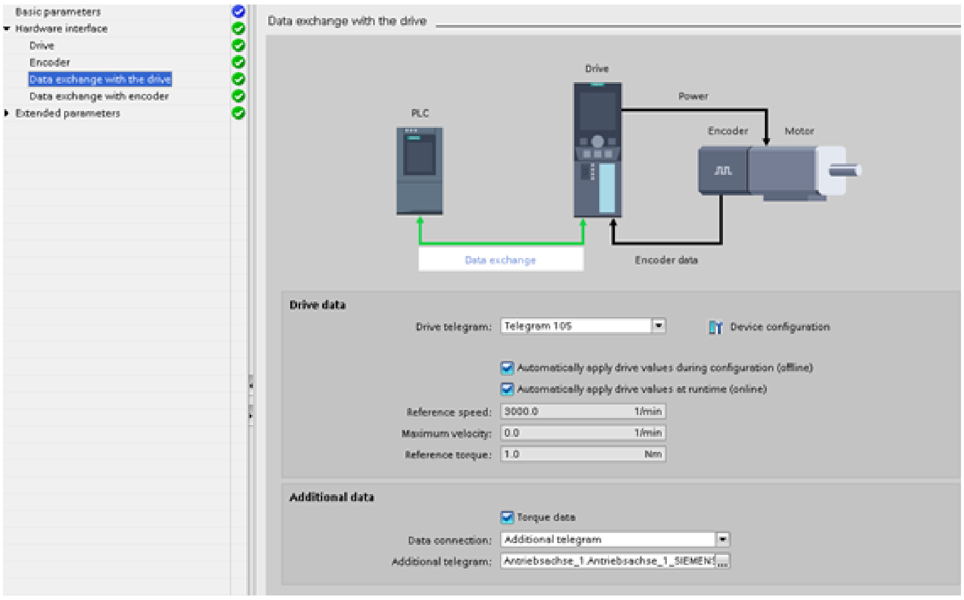
For winder only rotatory axes are possible

For sectional drives rotatory and linear drives are allowed



### Torque data

If the winder/ sectional drive is controlled in a torque-controlled mode, or if the torque pre-control is used, the additional torque data must be activated at the technology object and telegram 750 must be selected:

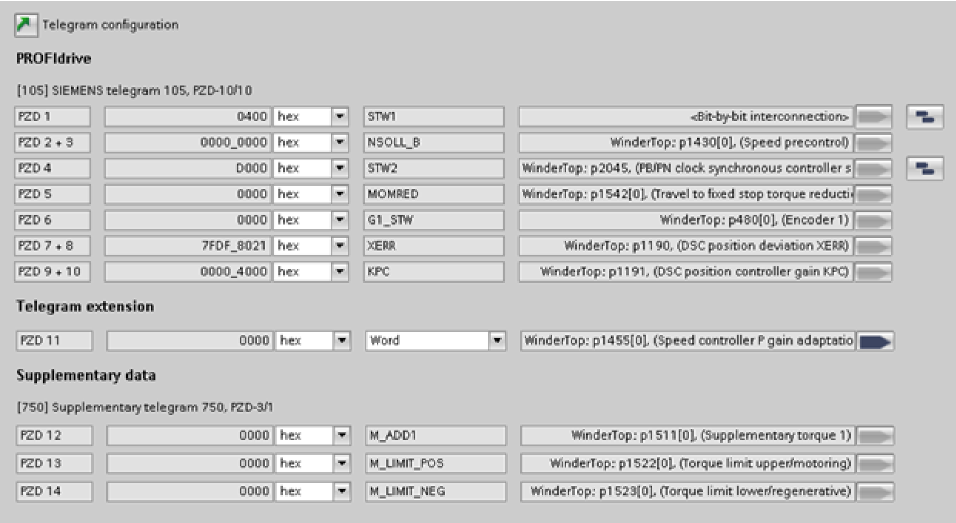


### Kp-Adaption (Only SINAMICS S120)

If it is necessary to activate a Kp-Adaption of the speed controller gain (see description LConSMC_KpAdaption), the adaption value must be sent to the drive using a free telegram:

- Combined with the torque values (Length 1/4)
- Telegram 750 for the torque values and free telegram of the length of one word for the adaption value

If torque and Kp values are sent to the drive using two telegrams the parameter connection for the telegram 750 is done automatically in StartDrive. The additional Word for the Kp-Adaption must be connected to parameter p1455 in the SINAMICS S120.



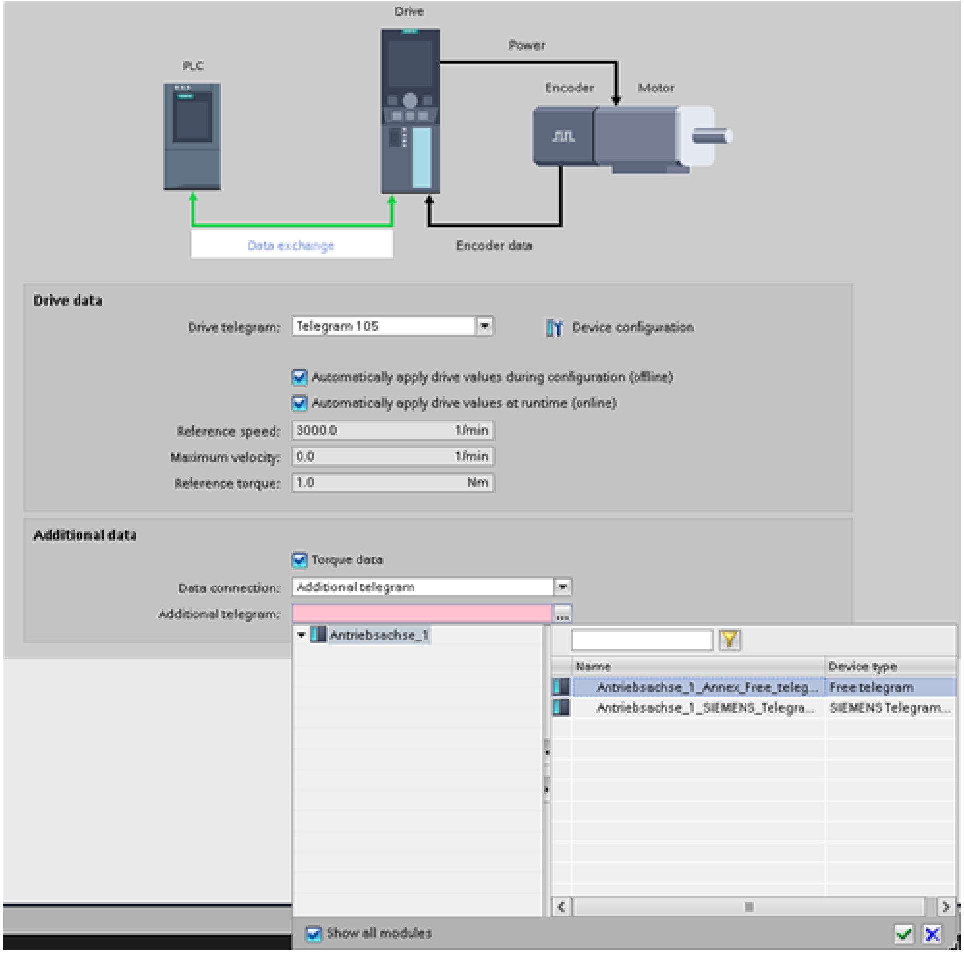

If only one free telegram is used to forward the Kp-Adaption together with further data (e.g. torque data) the connection must be done manually according to the following table:

| Element | Offset (Byte) | SINAMICS parameter |
|---|---|---|
| Additional torque | 0 | p1511 (as telegram 750) |
| Upper torque limit | 2 | p1522 (as telegram 750) |
| Lower torque limit | 4 | p1523 (as telegram 750) |
| Kp-Adaption | 6 | p1455 (only S120) |

| Element | Offset (Byte) | SINAMICS parameter |
|---|---|---|
| Actual torque | 0 | p80 (as telegram 750) |

**Note** The hardware-ID of the free telegram and the byte offset of the Kp-Adaption with the telegram (6 Bytes) must be connected to the function block LConSMC_KpAdaption.

If the connection has been done according to the upper table (similar with telegram 750), the additionl torque data of the TO can still be used






## 3.5 Configuration of the SINAMICS

| Para meter | Description | Effect | Value |
|---|---|---|---|
| p1227 | Setting the monitoring time of the standstill detection. Slowing down with AUS1 or AUS3, the standstill is detected after this time and after the set rotation speed has fallen below the speed limit (p1226). After that, the brake control is started, the closing time in p1217 is awaited and then the pulses are deleted | Deactivation of the standstill monitoring of the setpoint -> no run out of the motor | 300s |
| p1135 | Setting the ramp return runtime from the maximum speed to the standstill for the AUS3 command. | Slowing down at torque limit | 0s |
| p1551 | Setting the signal source for switching the torque limits between variable and fix torque limits. | With AUS3 active (r0899.5), the torque limits are switched | r0899.5 |
| r0899.5 | No quick stop active (state AUS3) | | p1551 |
| p1520 | Setting the fix upper or the motor torque limit | Parameterizable upper torque value for AUS3 | - |
| p1521 | Setting the fix lower or the dynamic torque limit | Parameterizable lower torque value for AUS3 | - |
| p1082 | Setting the highest possible speed | Speed limit for torque control | max. |
| p1145 | Setting the ramp function generator tracking. The initial value of the | Only with extended setpoint channel: Ramp | 0.0 |

| Para meter | Description | Effect | Value |
|---|---|---|---|
| | ramp function generator is tracked according to the maximum possible<br>drive acceleration. Reference value is the difference at the speed / velocity control<br>which is necessary to ensure a run-up at the torque / force limit of | flattening at ramp function generator is<br>deactivated | |
| p1400 .14 | Setting the configuration of the speed control. Torque pre-control VECTOR | In vector mode: No deactivation of the torque<br>pre-control with overdriven speed controller | Alway s active |
| p1580 | Efficiency optimization (Flux setpoint adaption) VECTOR | Drive speed is more stable at low speeds | 50%-100% |
| p1821 | Inversion of the direction of the rotating field:<br>0: no inversion of direction<br>1: inversion of direction | Changes the direction of the motor with<br>retention of the sign of the setpoint | 0 |
| p505 | Selecting the system of units | With this parameter the system of units can be<br>chosen for the drive object, it must be consistent<br>with the winder application | |



### Behavior OFF3 / Braking at torque limit

The pulse enable of the winder axis is controlled by the observation of the standstill threshold besides the control signals. This observation provides an analysis of the actual speed value as well as of the setpoint. If the one of these values falls below the speed threshold, this causes a pulse disable after a delay time (FP2701). In the case of an activation of the OFF3 function, it may cause the braking of the drive with the maximal torque, but the drive torque doesn’t suffice to decelerate the winder according to the ramp. If the setpoint falls below the speed threshold of the standstill detection, the pulses are inhibited activated after the parameterized delay time (p1227) and the motor runs down slowly. To avoid this behavior the standstill observation of the setpoint may be deactivated by setting the parameter p1227 = 300s. In the parameter p1135 the ramp down time from the maximum speed to the standstill in case of an activation of OFF3 is parameterized. For a winder axis it may be useful to set the value to 0, to achieve a braking at the torque limit, because the ramp down time does not fit in all operating points to the speed of the winder axis and rapid braking not warranted. If in the case of tension control by torque limits the function OFF3 is activated, the limits must be reset to the maximum torque, if necessary, to be able to brake with the adequate deceleration. This may be parameterized in SINAMICS by connecting the parameter p1551 to the binector r0899.5. In the case of a fast stop the active torque limits are configurable with the parameter p1520 and p1521. Using the extended set-point channel and the torque control by override of the speed controller, the speed controller is given a velocity setpoint that cannot be reached. If the torque limit is switched with the activation of the AUS3 function, the speed controller makes up the velocity difference between the actual (overridden) setpoint and the actual value and can accelerate the motor before shutting down at the AUS3 ramp. To avoid this behavior, either the AUS3 ramp may be set to 0 or the ramp function generator is to be set with the activation of the AUS3 command.

### Setting the maximum speed

Using a torque-controlled winder control mode it is recommended to limit the maximum motor speed with parameter p1082. E.g. machine speed + offset or maximum speed of the motor gear.

### Ramp flattening

Using overdriven speed controller and extended setpoint channel, the default setting of the ramp flattening leads to the speed setpoint values is not being transferred from SIMOTION to the drive speed controller. In cause of that the speed controller is not overridden. This behavior can be switched off setting p1145 to 0.

### Interconnection additional torque using overdriven speed controller

Using a drive with vector mode the additional torque is only added if the speed controller is active. If the speed controller is overridden, this signal goes to FALSE and so no more additional torque is added. To prevent this behavior and adding the addition torque always, the parameter p1400.14 need to be set to 1.

### Speed inversion

To invert the direction of rotation the parameter p1821 (inversion of the direction of the rotating field) can be used. The encoder will be inverted automatically. In that way the sign of the setpoint remains.

## 3.6 Unit settings

The winder/ sectional drive function block can be operated in the metric or the Anglo-American system. The setting is done in the corresponding configuration structure *winderConfig/sectionalDriveConfig.unitConfiguration.metricUnits* The following table shows the mandatory units the function blocks are working with:

| Inputs | Metric | Anglo-American |
|---|---|---|
| Length unit [LU] | m | ft |
| Line velocity | m/min | ft/min |
| Line acceleration | m/s^2 | ft/s^2 |
| Diameter | m | inch |
| Inertia | kgm^2 | lbft^2 |
| Density | kg/m^3 | lb/ft^3 |
| Tension | N | Lbf |



**Note**

The winder/ sectional drive function block automatically converts the setpoints (Speed, Torque) into the necessary axis units

The parameterizable dynamics of the winder or sectional drive must be specified in the unit of the respective TO at *WinderConfig.winderDynamics* or *SectionalConfig.SectionalDriveDynamics*!

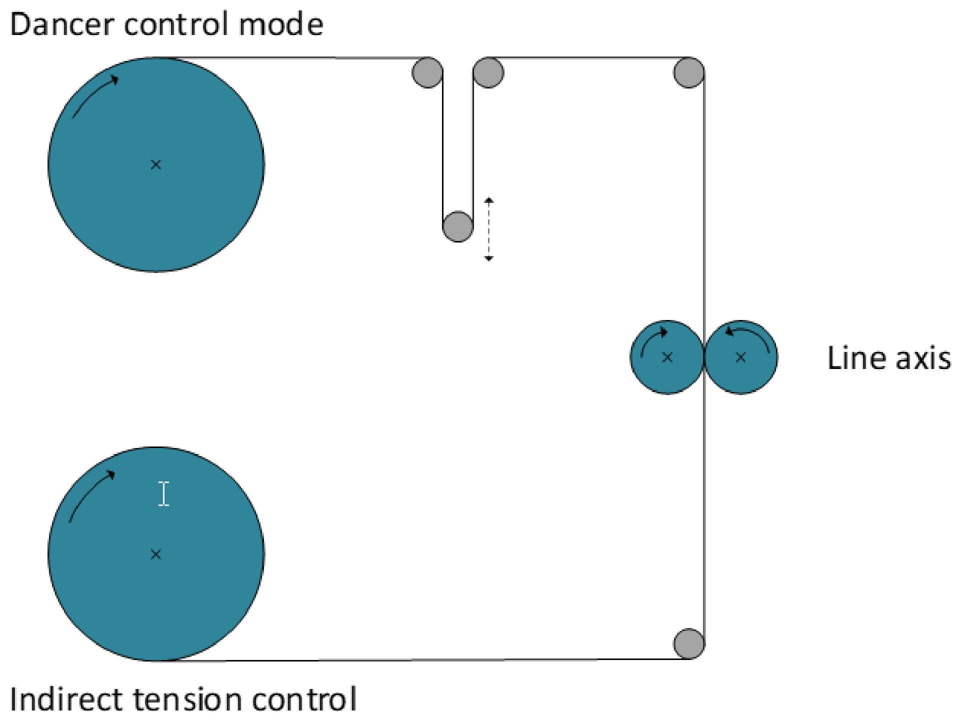
If the units of the line axis are in a different, the values can be converted using the function block LConSMC_LineUnitConversion.

# 4 Example program

## 4.1 Structure of the example

To demonstrate the usage of the application an example project is provided consisting of the following components:

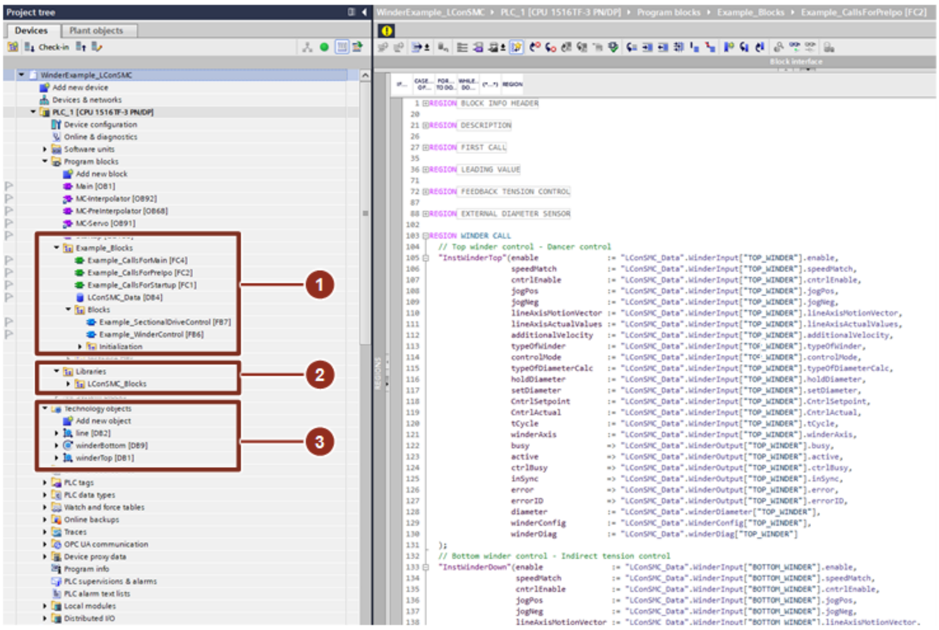
- Unwinder in dancer control mode (winder top)
- Rewinder in indirect tension control (winder down)
- Line axis (Leading value for winders)




Within the example project the basic functions to control a winder are demonstrated:

- Resetting and enabling the axes
- Enable/Disable the winder function blocks (e.g., set starting diameter, enable tension control)
- Start and stop the line axis

**Note** The project is a simple example program on how to control the winder function block. It must be adapted/extended according to the machine project (e.g., error handling, further winder functions etc.)



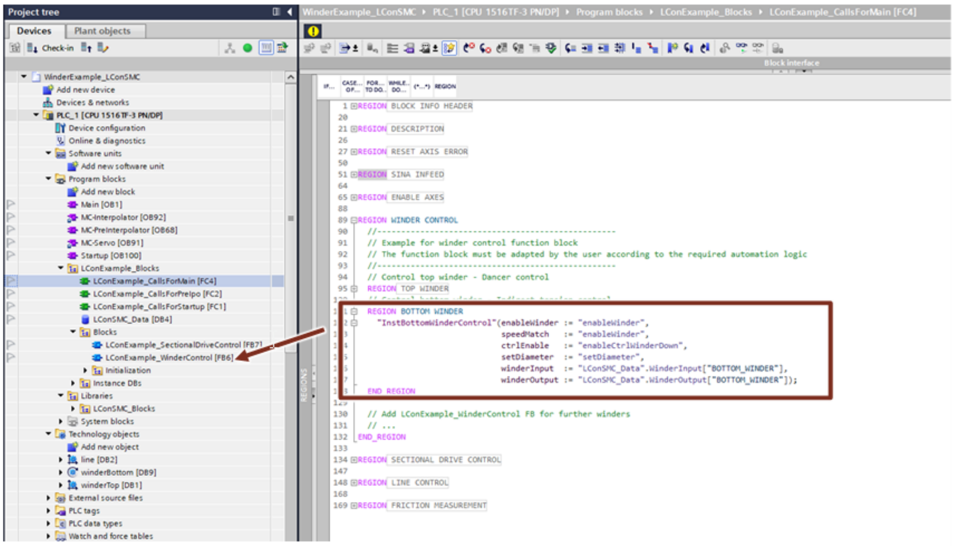

1. Example program
   - Example_CallsForXXX Wrapping FCs that are called in the Startup OB, OB1 and MC_PreInterpolator OB and contain the necessary FB calls
   - LConSMC_Dat Global data block with the necessary data types (winder configuration/diagnostic data etc.)
   - Blocks Contains the FBs/FCs with the example program to control/ initialize the winder/sectional drives
2. LConSMC library blocks
3. Used technology axes within the example project
   - Line = Positioning axis (Leading value)
   - WinderDown = Speed axis (Indirect tension control)
   - WinderTop = Positioning axis (Dancer control mode)

**Winder/Sectional drive control**

An example to set the winder/sectional drive function block inputs correct is provided with the function block Example_WinderControl/ SectionalDriveControl:

- Program blocks -> Example_Blocks -> Blocks

It is called in the OB1 (Example_CallsForMain) once for each winder drive. The REGIONS should indicate the user the necessary inputs and order of the inputs that need to be set.

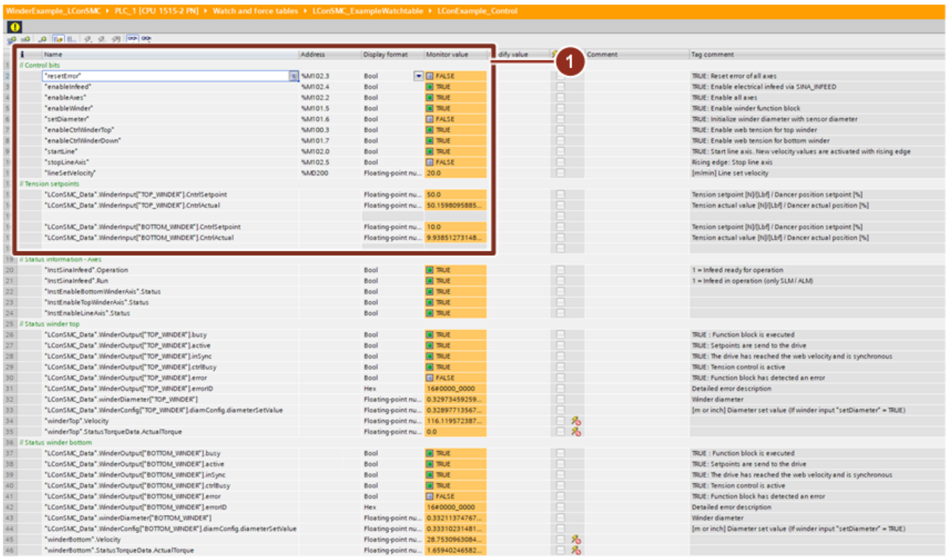


**Note** The FB is just an example and covers not all available inputs of the winder function block. It must be adapted by the user to the final use case (e.g., holdDiameter, jogPos, jogNeg etc.).



## 4.2 Operation of the example

With the help of the provided watch table “Example_Control” the example project can be controlled and monitored:

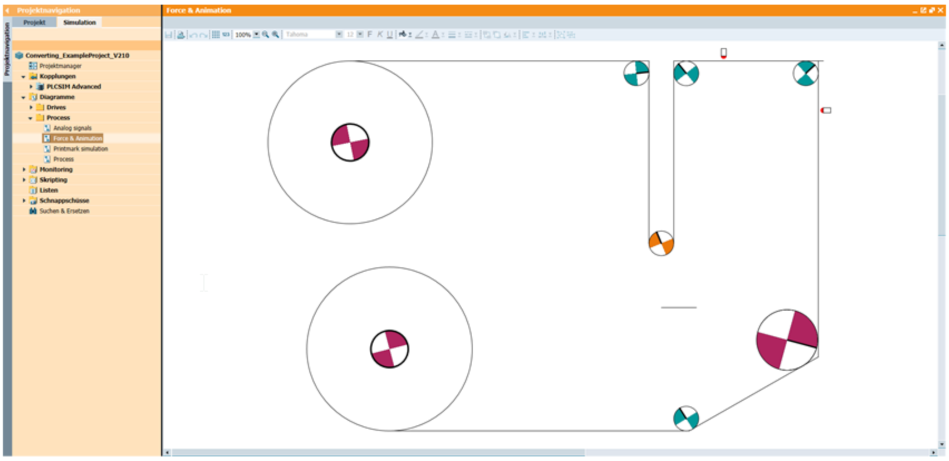


With the help of the control bits the example project can be operated. All variables are connected to the example program in the sequence folder, called in the OB1. A description of the variables can be found in the comment of the variable. To run the program the variables must be set from top to bottom, for disabling the other way around.

## 4.3 SIMIT Converting library

The example project can be simulated with S7-PLCSIM Advanced and the SIMIT Converting model for continuous web. A SIMIT project is provided which can be used without any further modifications. On SIOS you can download the project and the manual in addition to other useful information:

https://support.industry.siemens.com/cs/ww/en/view/109801697



# 5 Program blocks

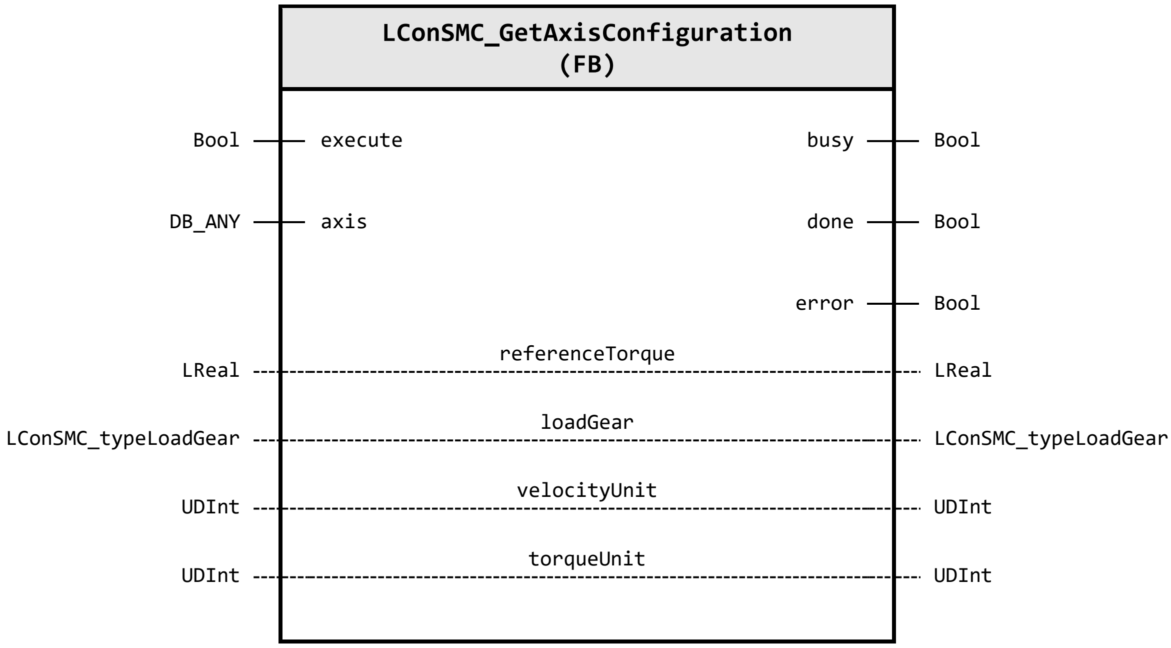
## 5.1 LConSMC_GetAxisConfiguration (FB)

### Short description

This function blocks reads out necessary axis settings from the technology object. It can be called in any cyclic task  Function block ID: 3110

### Interface description

### Block interface




### Input parameter

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| execute | Bool | FALSE | Read axis configuration with rising edge of enable |
| axis | DB_ANY | --- | Reference of axis |

### Output parameter

| Identifier | Data type | Description |
|---|---|---|
| busy | Bool | TRUE: Funtion block is activated |
| done | Bool | TRUE: Configuration has been read successfully |
| error | Bool | TRUE: An error has occured during the reading |

### In/Out parameter

| Identifier | Data type | Description |
|---|---|---|
| referenceTorque | LReal | Reference Torque of axis (see p2003 in the SINAMICS) |
| loadGear | LConSMC_typeLoad Gear | Load gear |
| velocityUnit | UDInt | Velocity unit of axis (TO_Axis.Units.VelocityUnit) |
| torqueUnit | UDInt | Torque unit of axis (TO_Axis.Units.TorqueUnit) |

**Functional description**

This function block reads the necessary axis settings from the technology object “axis”, that are needed for the configuration of a winder or a sectional drive:

- ReferenceTorque
- LoadGear
- Velocity unit
- Torque unit

The IN_OUT variables can directly be connected to the configuration variables:

| IN_OUT variable | Configuration paramter |
|---|---|
| referenceTorque | winderConfig/sectionalDriveConfig.torquePrecontrolConfig.referenceTorque |
| loadGear | winderConfig/sectionalDriveConfig.torquePrecontrolConfig.loadGear |
| velocityUnit | winderConfig/sectionalDriveConfig.UnitConfiguration.axisVelocityUnit |
| torqueUnit | winderConfig/sectionalDriveConfig.UnitConfiguration.axisTorqueUnit |

**Note**

To optimize the controller performance this function block can be used once to read the axis configuration within the main program cycle instead with a rising edge of the winder/ sectional drive function block in a synchronous organization block.

To use this, adjust the settings in the winder/sectional drive configuration structure, to prevent access to TO-variables during the initialization:
*winder/sectionalDriveConfig.getAxisConfigAtStart* := FALSE;

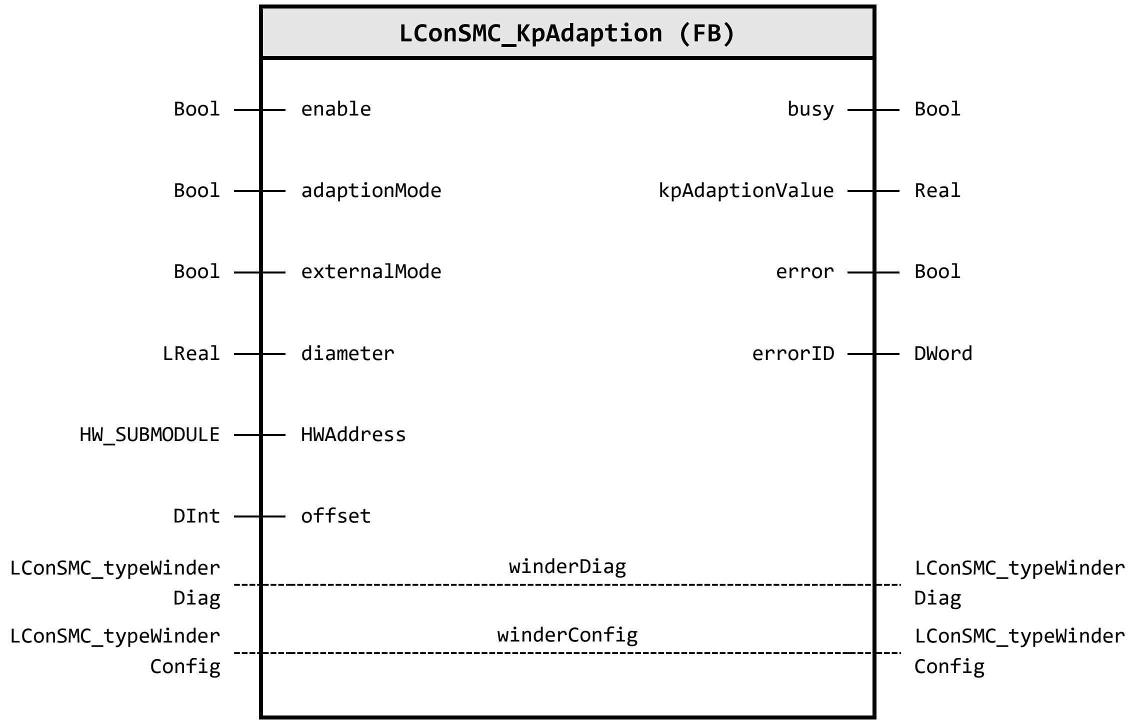
## 5.2 LConSMC_KpAdaption (FB)

### Short description

The function block is used adapt the Kp value of the speed controller as a function of the current diameter or the current intertia of a winder. Function block ID: 3109

### Interface description

### Block interface






### Input parameter

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| enable | Bool | FALSE | TRUE: Start function block |
| adaptionMode | Bool | FALSE | TRUE: User diameter ratio for adaption (dAct/dMax)<br>FALSE: User inertia ratio for adaption (JAct/JMax) |
| externalMode | Bool | FALSE | TRUE: Calculation of adaption value but value will not be written to telegram (input "HWAddress" not needed)<br>FALSE: Adaption value will be calculated and written to the telegram (input "HWAddress" necessary) |
| diameter | LReal | 0.0 | [m]/[inch] Current diameter |
| HWAddress | HW_SUBMODUL E | --- | HW-Address of telegram extension which is used to send adaption value to the drive (Only needed if externalMode = FALSE) |
| offset | DInt | 6 | Byte offset with the telegram to write the Kp-Adaption value (if externalMode = FALSE).<br>If "offset" = 0, the value will be written at the begining of the telegram |

### Output parameter

| Identifier | Data type | Description |
|---|---|---|
| busy | Bool | TRUE: Function block is in operation |

| Identifier | Data type | Description |
|---|---|---|
| kpAdaptionValue | Real | [0-100%] Adaption value<br>This value will be multiplied with the speed controler Kp value |
| error | Bool | TRUE: Error has occured |
| errorID | DWord | Detailed error description |

**In/Out parameter**

| Identifier | Data type | Description |
|---|---|---|
| winderDiag | LConSMC_type<br>WinderDiag | Winder diagnostic structure |
| winderConfig | LConSMC_type<br>WinderConfig | Winder configuration structure |

**Functional description**

The diameter – and therefore also the moment of inertia of the wound roll – permanently changes as the roll is wound. This is the reason that it is necessary to adapt the controller gain of the speed controller to the mechanical relationships at the winder (especially for speed controlled winding types).This function block calculates a Kp-Adaption value for the speed controller as a function of the current diameter or the current inertia according to the following formula (depends on input parameter *adaptionMode*):

- $kpAdaptionValue = \frac{current\ Diameter}{maximum\ Diameter} \cdot 100\%$
- $kpAdaptionValue = \frac{current\ Inertia}{maximum\ Inertia} \cdot 100\%$

In general, the speed controller of the winder must be optimized once for a fully wounded roll and once for an empty roll. The interconnection to the drive depends on the input parameter *externalMode*:

- externalMode = TRUE

The function block is just calculating an adaption value, which is displayed at the output *kPAdaptionValue*. The output equals 100% if the current diameter/inertia equals the maximum diameter/inertia. The value itself will not be forwarded to the drive via the function block.

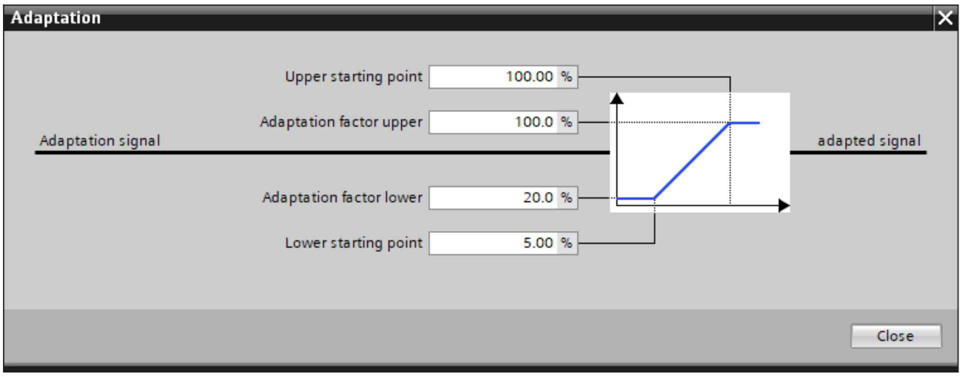
- externalMode = FALSE

For the mode externalMode = FALSE the Kp adaption value is additionally sent to the drive. Therefore, a telegram extension with a sending length of at least two bytes is necessary. The HW-Address and the byte offset within the telegram must be forwarded to the function block.

**Settings in the SINAMICS S120**

The SINAMICS S120 provides a speed controller Kp-Adaption parameter (p1455) which can be used. Follow the described steps to activate the adaption in the drive:

1. Connect the Kp-Adaption value in the telegram to parameter p1455
2. The Kp adaptation must be parameterized in the configuration dialog box “Adaptation”; in this case, the x axis reflects the diameter/inertia ratio and the y axis, the controller gain. These should be parameterized corresponding to your application or determined by optimizing the controller. The ratio becomes 100% if the diameter/inertia has reached the maximum value. The Kp (p1460) for the maximum diameter/inertia must be correspondingly set-up. In order to achieve a certain Kp at the minimum diameter, the scaling must be calculated.
   The following figure shows an example of the Kp-adaption settings in the S120:




For an adaption signal of 5% (e.g. current diameter/inertia = 5% of maximum diameter/inertia) the scaling factor for the entered speed controller Kp value is 20%:

$$Effective\ Kp = 20\% \cdot Kp(p1466)$$

For the maximum diameter/inertia, 100% of the entered speed controller Kp value is effective.

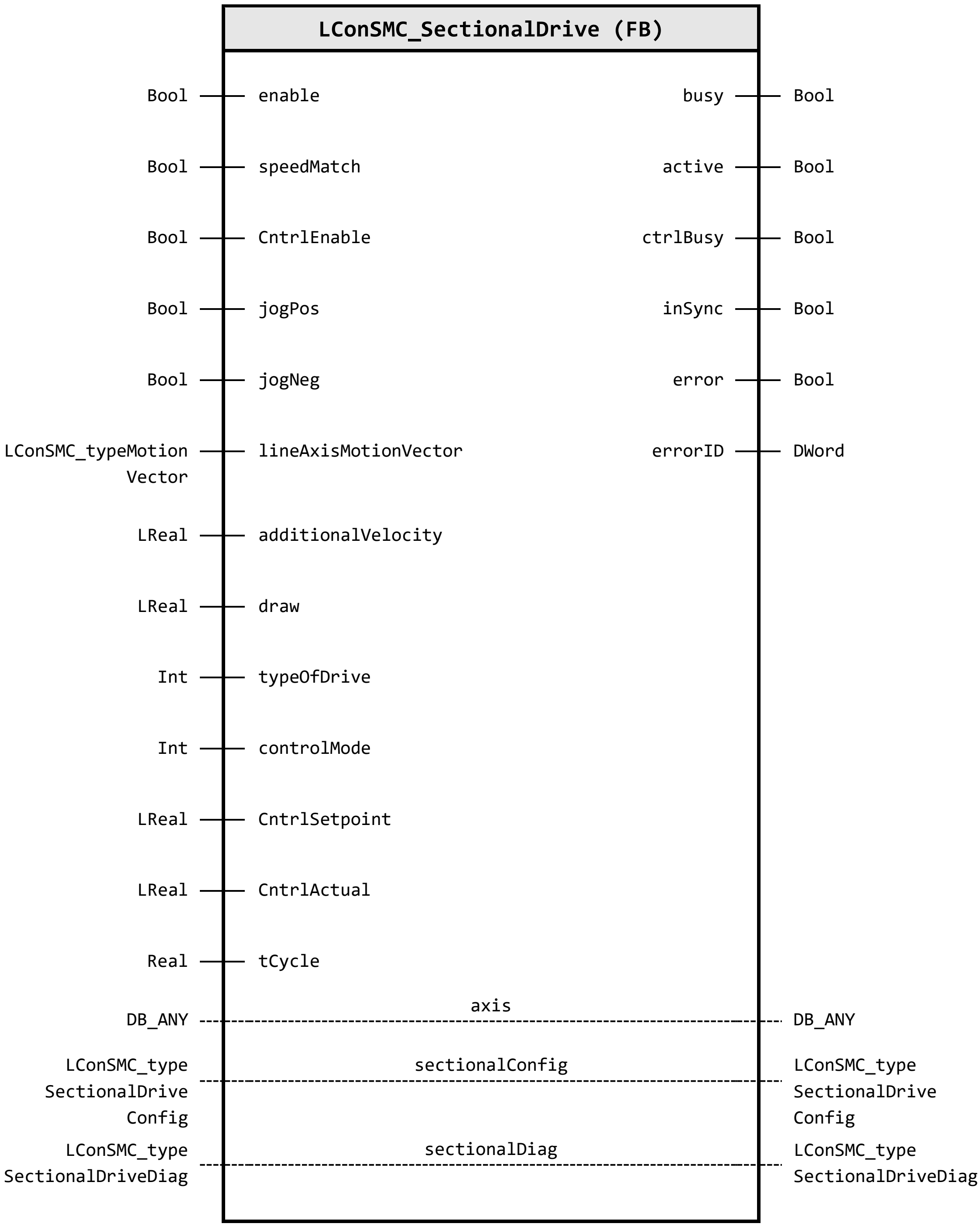
## 5.3 LConSMC_SectionalDrive (FB)

### Short description

This function block is used to control a converting line sectional drive from the PLC program. The function block was designed to handle the requirements of a tension or ratio adjusted driven section. The function block must be called in a cyclic interrupt, e.g. OB30, OB95.
Function block ID: 7102

### Interface description

### Block interface



### Input parameter

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| enable | Bool | FALSE | TRUE: Enable function block. WIth rising edge a validation of parameter is done and communication with drive is started<br>FALSE: Disable the function block |
| speedMatch | Bool | FALSE | True: Enable web velocity setpoint ("lineAxisMotionVector"). The drive will synchronize up to web velocity using syncronization dynamics. If the velocity is reached the output "inSync" will become TRUE.<br>FALSE: Drive will decelerate to standstill using stop dynamics |
| CntrlEnable | Bool | FALSE | TRUE: Enable web tension control (enable torque limits, dancer position setpoint etc.) |
| jogPos | Bool | FALSE | TRUE: Drive is jogged in positive direction (see config.dynamics.jogDynamics) |
| jogNeg | Bool | FALSE | TRUE: Drive is jogged in negative direction (see config.dynamics.jogDynamics) |
| lineAxisMotion Vector | LConSMC_type MotionVector | --- | Setpoint of the web (position/velocity/acceleration) |
| additional Velocity | LReal | 0.0 | [m/min] / [ft/min] Additional web velocity (always enabled) |
| draw | LReal | 1.0 | Draw factor for control mode 4 (DRAW CONTROL).<br>Velocity setpoint = webVelocity * draw |
| typeOfDrive | Int | 0 | Definition of web tension direction<br>0: Web tension before the roll is controlled (Pull)<br>1: Web tension after the roll is controlled (Held) |
| controlMode | Int | 0 | Tension control mode<br>0: Dancer via speed setpoint adaption<br>1: Load cell via speed setpoint adaption<br>2: Load cell via torque limits<br>3: Indirect tension control<br>4: Speed setpoint relation (Draw)<br>5: Indirect tension control vis speed setpoint adaption |
| CntrlSetpoint | LReal | 0.0 | Tension setpoint [N]/[Lbf] / Dancer position setpoint [%] |
| CntrlActual | LReal | 0.0 | Tension actual value [N]/[Lbf] / Dancer actual position [%] |
| tCycle | Real | 0.0 | [ms] Cycle time the function block is called |

### Output parameter

| Identifier | Data type | Description |
|---|---|---|
| busy | Bool | TRUE : Function block is executed |
| active | Bool | TRUE: Setpoints are send to the drive |
| ctrlBusy | Bool | TRUE: Tension control is active |
| inSync | Bool | TRUE: The drive has reached the web velocity and is synchronous |
| error | Bool | TRUE: Function block has detected an error |
| errorID | DWord | Further information on occured error (<> 16#0000_0000) |

### In/Out parameter

| Identifier | Data type | Description |
|---|---|---|
| axis | DB_ANY | Section drive axis |

| Identifier | Data type | Description |
|---|---|---|
| sectionalConfig | LConSMC_type SectionalDrive Config | Sectional drive configuration structure |
| sectionalDiag | LConSMC_type SectionalDrive Diag | Sectional drive diagnostic structure |

### Functional description

This function block utilizes the LConSMC_TorquePrecontrol and LConSMC_TechPID function blocks of the SIMATIC Converting library for a sectional drive. This function block checks for valid configuration data and calls the individual function blocks to minimize application development time. With the *enable* set to TRUE, a plausibility check is done on the configuration data and will display an error with corresponding error ID should invalid data be entered.

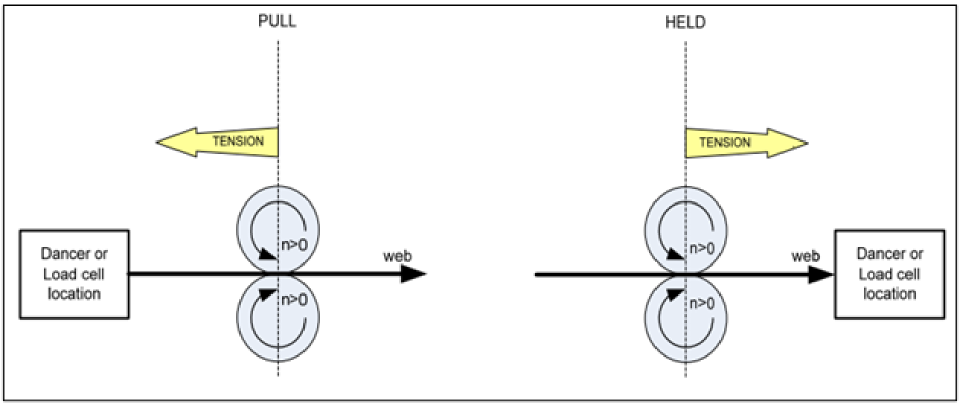
**Note** Enabling of the axis (MC_Power) is not part of the standard application and must be done in the user program.

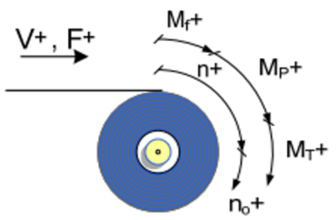
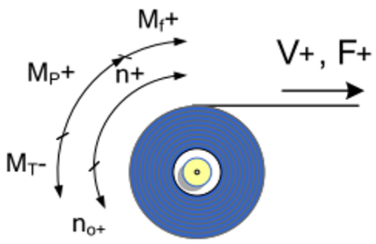
With the *enable* set TRUE the sectional drive will follow the sum of the lineVelocity and additionalVelocity times the draw factor (Only for draw control mode). Sectional drives are typically made to follow the master without any ramps. The programmer is responsible for proper interlocks to prevent unwanted behavior.
The configuration data has diameter, gear ratio and max line speed parameters which are utilized to calculate the correct surface velocity of the driven roll to match the section to the master line velocity.
The configuration data also has parameters for torque pre-control. Torque pre-control is necessary for torque-based tension control modes and is optional for speed-based tension control modes.
With the *enable* set to TRUE and the axis being enabled and *CntrlEnable* then the CntrlSetpoint will be ramped up based on configuration data. The Tension controller correction value will be added to the total speed setpoint or additional torque, scaled and sent to the drive using the telegram extension as required. The function block diagnostic structure allows the user to view the calculated velocity/ torque setpoint as well the tension controller parameters or the status of the web break detection.

**Note** A detailed description of further technological add-ons the function block offers is described at the LConSMC_Winder function block description.

### Tension operating modes

The following overview shows the tension operation modes depending on the position of the dancer or tension measuring device location in accordance to the web axis as well as the orientation so the setting variables (speed, force, torque). The tension operating mode is selected by the *typeOfDrive* input of the function block. The following table shows the different tension operating modes and the orientation of velocity, tension, and torque. At commissioning of the application, it must be ensured that the direction convention and signs of the process variables are aligned. Eventually it might be necessary to change the direction of rotation in the drive system (p1821).




| Operation mode | Representation | Description |
|---|---|---|
| PULL |  | V: Machine velocity<br>F: Tension in material line<br>n: speed of drive<br>$n_o$ : velocity override for speed controller overcontrol<br>$M_F$: Friction torque<br>$M_P$: Pre-control torque<br>$M_T$: Tension torque |
| HELD |  | |

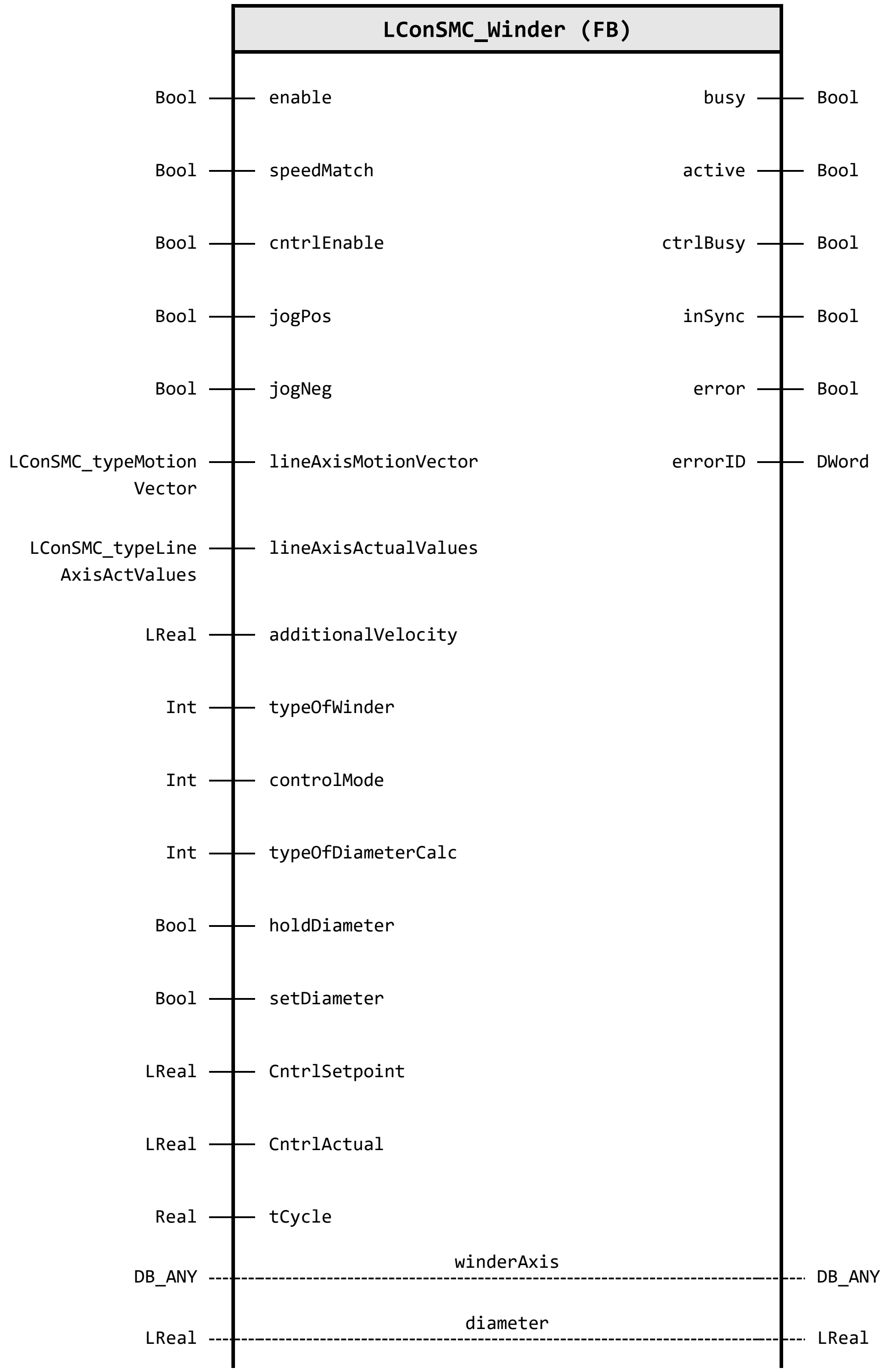
## 5.4 LConSMC_Winder (FB)

### Short description

Function block for the control of an axial winder. The function block must be called in a cyclic interrupt (e.g. OB30) or synchronous operation (e.g. OB60, OB95). Function block ID: 5105

### Interface description

### Block interface



LConSMC_typeWinderConfig — winderConfig — LConSMC_typeWinderConfig

LConSMC_typeWinderDiag — winderDiag — LConSMC_typeWinderDiag

### Input parameter

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| enable | Bool | FALSE | TRUE: Enable function block. With rising edge, a validation of parameter is done and communication with drive is started<br>FALSE: Disable the function block |
| speedMatch | Bool | FALSE | True: Enable web velocity setpoint ("lineAxisMotionVector"). The drive will synchronize up to web velocity using syncronization dynamics. If the velocity is reached the output "inSync" will become TRUE.<br>FALSE: Drive will decelerate to standstill using stop dynamics |
| cntrlEnable | Bool | FALSE | TRUE: Enable web tension control (enable torque limits, dancer position setpoint etc.) |
| jogPos | Bool | FALSE | TRUE: Drive is jogged in positive direction (see config.dynamics.jogDynamics) |
| jogNeg | Bool | FALSE | TRUE: Drive is jogged in negative direction (see config.dynamics.jogDynamics) |
| lineAxisMotion Vector | LConSMC_type MotionVector | --- | Setpoint of the web (position/velocity/acceleration) |
| lineAxisActual Values | LConSMC_typeLin e AxisActValues | --- | Actual values of the web for diameter calculation. Leading value (actual or setpoint values) must be connected to the input "lineAxisMotionVector" |
| additional Velocity | LReal | 0.0 | [m/min] / [ft/min] Additional web velocity (always enabled) |
| typeOfWinder | Int | LCON_TYPE_ OF_WINDER_ REWIND_FRO M_ABOVE | Definition of winding direction<br>0: Rewind from above<br>1: Rewind from below<br>2: Unwind from above<br>3: Unwind from below |
| controlMode | Int | 0 | Tension control mode<br>0: Dancer via speed setpoint adaption<br>1: Load cell via speed setpoint adaption<br>2: Load cell via torque limits<br>3: Indirect tension control<br>4: V-constant<br>5: Indirect tension control vis speed setpoint adaption |
| typeOfDiameter Calc | Int | LCON_DIAM_C ALC_MODE_IN TEGRAL | Type of diameter calculation (Can be changed on the fly)<br>0: V/n<br>1: Thickness addition<br>2: Integral<br>3: External (Sensor value must be connected to "diamConfig.diameterSetValue")<br>4: WebLength<br>5: Position |
| holdDiameter | Bool | FALSE | TRUE: Hold diameter calculation<br>FALSE: Diameter calculation is active |
| setDiameter | Bool | FALSE | TRUE: Set diameter to the value in "winderConfig.diamConfig.diameterSetValue" |
| CntrlSetpoint | LReal | 0.0 | Tension setpoint [N]/[Lbf] / Dancer position setpoint [%] |

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| CntrlActual | LReal | 0.0 | Tension actual value [N]/[Lbf] / Dancer actual position [%] |
| tCycle | Real | 0.0 | [ms] Cycle time the function block is called |

**Output parameter**

| Identifier | Data type | Description |
|---|---|---|
| busy | Bool | TRUE : Function block is executed |
| active | Bool | TRUE: Setpoints are send to the drive |
| ctrlBusy | Bool | TRUE: Tension control is active |
| inSync | Bool | TRUE: The drive has reached the web velocity and is synchronous |
| error | Bool | TRUE: Function block has detected an error |
| errorID | DWord | Detailed error description |

**In/Out parameter**

| Identifier | Data type | Description |
|---|---|---|
| winderAxis | DB_ANY | Winder axis |
| diameter | LReal | [m or inch] Current diameter |
| winderConfig | LConSMC_type WinderConfig | Winder configuration structure |
| winderDiag | LConSMC_type WinderDiag | Winder diagnostic structure |



**Functional description**

This function block utilizes the WITorquePrecontrol, DiamCalc and TechPID, as well as EnableInterface function blocks of the SIMATIC Converting library. This function block checks for valid configuration data and calls the individual function blocks to minimize application development time. With the *enable* set to true, a plausibility check is done on the configuration data and will display an error with corresponding errorId should invalid data be entered. If the function block is enabled and there are no errors the state will be shown by the outputs *busy* and *active*.

**Note** Enabling of the axis (MC_Power) is not part of the function block and must be done by the user program.

With the enables and the input *speedMatch* set true the winder will follow the sum of the lineVelocity and additionalVelocity. Winders are typically made to follow the master without any ramps. Winders with splice are an exception as they need to be able to be detached from the running machine and therefore have separate ramps. The winder configuration offers different dynamic settings therefor. The programmer is responsible for proper interlocks to prevent unwanted behavior. The *operationDynamics* must be configured correctly so that the winder can follow the master.

The configuration data has diameter calculation parameters, gear ratio and line speed parameters which are utilized to calculate the correct surface velocity of the winder to match the section to the master line velocity. The actual diameter of the winder can either be calculated using different calculation modes or can also be supplied from an external source using the configuration structure of the function block. The configuration data also has parameters for torque pre-control. Torque pre-control is necessary for torque-based tension control modes and is optional for speed-based tension control modes. With the *enable*, *speedMatch* set to true and *Cntrlenable* then the CntrlSetpoint will be ramped up based on configuration data. The Tension controller correction value will be added to the total speed setpoint or additional torque, scaled and sent to the drive using the telegram extension as required. Function block output and diagnostic parameters allow the user to view the most important variables calculated within the winder function block (diameter, torque/speed setpoint etc.)

The winder function block offers some additional functionality next to the winding function e.g., for winders with splicing functionality:

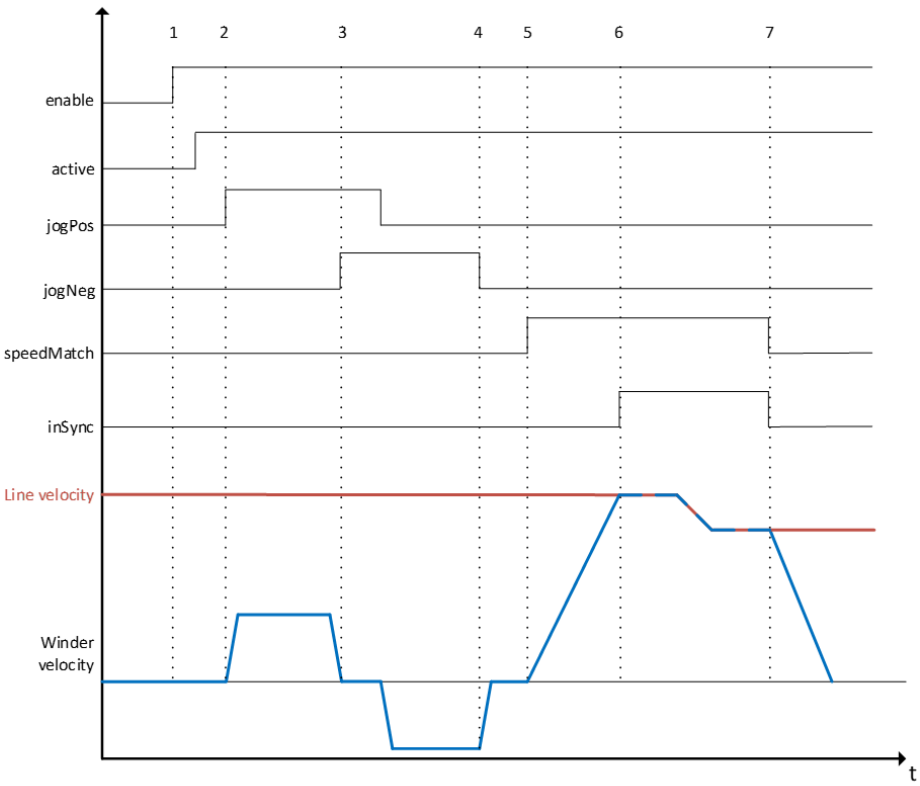
### Jogging

The winder can be jogged with respect to the current diameter via the inputs *jogPos* and *jogNeg*. Therefor some conditions/ limitations must be considered:

- Winder axis must be enabled (Not part of the winder function block)
- Dynamics can be parametrized via the winder configuration (*winderConfig.winderDynamics.jogDynamics*)
- Both inputs *speedMatch* and *ctrlEnable* must be FALSE
- Diameter calculation is not running

### Synchronizing/Stopping

If the winder should be synchronized to a moving line (e.g. in case of a splice) the winder can be accelerated to the line speed setpoint via the synchronization dynamics (*winderConfig.winderDynamics.syncDynamics*). If the winder axis and the winder function block is enabled the winder can by synchronized to line speed via the input *speedMatch* = TRUE. If the winder is in synchronism with the line speed the output *inSync* will become true. If the winder should be stopped, without ramping the line speed setpoint to zero, the *speedMatch* input can be set to FALSE. By this the internal setpoint will be ramped to zero, using the configured stop dynamics (*winderConfig.winderDynamics.stopDynamics*). The output *inSync* is only calculated during the synchronization process. If the tension control and thus the “operationDynamics” is active, inSync is set to TRUE. If the winder axis is un-synchronized, stopped, or mistyped, inSync = FALSE is set. For this reason, it is necessary that the operationDynamics are configured high enough so that the winder axis can follow the master axis.



| State | Description |
|---|---|
| 1. | The winder function block is enabled (enable= TRUE). If no error within the input and configuration parameter is detected and the interface has been enabled successfully (MoveVelocity, TorqueLimit, Additive torque) the ouput active will become TRUE. |
| 2. | With the inputs jogPos and jogNeg the winder can be jogged with respect to the current diameter (winderConfig.winderDynamics.jogDynamics). To use the jogging the inputs speedMatch and ctrlEnable must be FALSE. With jogPos = TRUE the winder will accelerate up to positive jogging velocity. |
| 3. | If both inputs jogPos and jogNeg are set to TRUE, the velocity setpoint will be zero. If the winder is already running it will be ramped down to zero via the jog dynamics. With jogNeg = TRUE and jogPos = FALSE the winder will run with negative jog velocity. |
| 4. | Setting the jog input to FALSE will ramp down the internal velocity setpoint with the jog dynamics. |
| 5. | By switching the input speedMatch = TRUE the winder will synchronize up to line speed via the configured synchronization dynamics (winderConfig.winderDynamics.syncDynamics). |
| 6. | If the winder is in synchronism with the line speed the output inSync will become TRUE. If the line speed setpoint is zero, the inSync will become immediately TRUE with the rising edge of the speedMatch input. |
| 7. | Setting the speedMatch = FALSE with a line speed setpoint unequal to zero the winder will be decelerated to zero using the configured stop dynamics (winderConfig.winderDynamics.stopDynamics). |



### Overview of the dynamics

In following table, you can see an overview of the resulting dynamics and velocity this FB will use. The variables with “x” can be “true” or “false”, the resulting dynamics and velocity will be the same.

| Enable | SpeedMatch | CtrlEnable | JogPos/ JogNeg | Resulting dynamics | Resulting velocity |
|---|---|---|---|---|---|
| 0 | x | x | x | stop | 0 |
| 1 | 0 | 0 | 0 | stop | 0, because of TPID is not active, so tensionspeed = 0 |
| 1 | 1 | 0 | 0 | sync | line + add |
| 1 | 0 | 1 | 0 | stop | tension |
| 1 | 1 | 1 | 0 | operation | line + add + tension |
| 1 | 0 | 0 | 1 | jog | jog |
| x | x | x | 1 | not possible! | not possible! |

### Web break detection

The integrated web break detection monitors the actual value of the technology controller (input value actValuePID) or in case of indirect tension control the actual torque – depending on the configured control mode. The monitoring consists of limits and the following switch-on delay. By the limits a lower and an upper error limit is described.

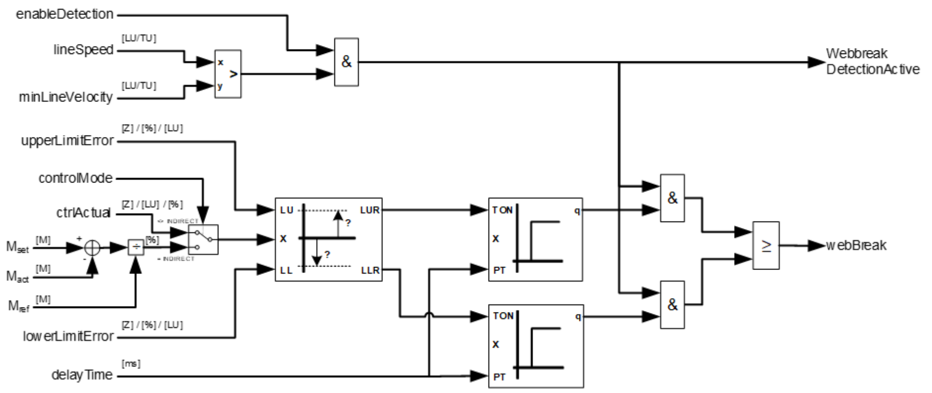
**Note** With controlMode = 4 (V_CONSTANT) no web break detection is possible

The web break detection is activated via the variable *enableDetection* in of the structure *winderConfig.webbreakDetectionConfig*. In addition a velocity threshold (for line axis speed) can be defined by the variable *winderConfig.webbreakDetectionConfig.minLineVelocity*, that activates the monitoring. The active monitoring is displayed using the *winderDiag.webbreakDiag.webbreakDetectionActive* feedback signal. The actual value is provided for the FB in the input variable *cntrlActual*, for indirect tension control the actual torque is determined from the axis object itself. The limits are adjusted via the parameters *lowerLimitError*, and *upperLimitError* in the structure *winderConfig.webbreakDetectionConfig*. In addition, the output *webBreak* is set at the FB when crossing an error limit. The response time of the feedback signals is parameterized using the delayTime response time.

| Note | Internally, the signal deactivates the technology controller (except for indirect tension control); all other responses to the web break detection must be programmed in the user program. |
|---|---|

In case of indirect tension control the difference between the setpoint and actual torque is evaluated. The difference is interpreted as a percentage value of the reference torque value. Therefore the limit values are also defined in percentage of the reference torque.

$$errorSignal = \frac{ABS(torqueSetpoint - torqueActual)}{torqueNominal} \cdot 100\%$$




## External setpoint connection

During normal operation the winder function block forwards the calculated velocity/ torque setpoint internally to the connected technology object automatically. If the function block should just be used as a setpoint generation without forwarding the setpoint to the technology object, the axis control must be disabled by setting *winderConfig.axisCtrl* = FALSE (e.g., when using the print standard library to forward the setpoint).
In that case the winder diagnostic structure offers the following parameters to externally control the winder axis:

- winderDiag.referenceVelocity [TO unit]
- winderDiag.diameterFactor [0-200%]

The effective velocity setpoint can be calculated according to the following formula:

$$velocitySetpoint = referenceVelocity \cdot \frac{diameterFactor}{100.0}$$

There are three possibilities to forward the velocity setpoint to an axis:

1. Via MC_MotionInVelocity

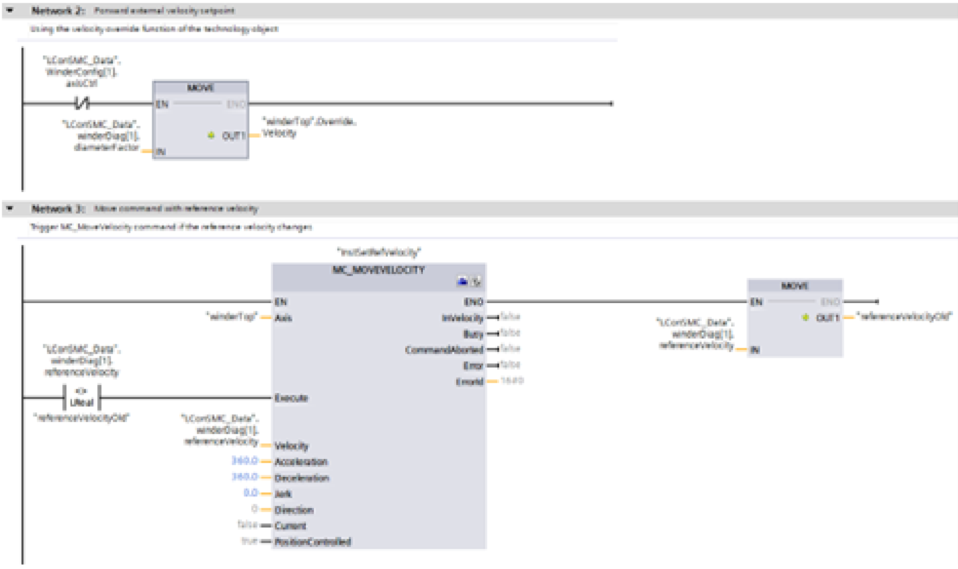
The velocity setpoint is directly forwarded to the axis via the user setpoint interface

| NOTICE | **Dynamic limitations**<br><br>The velocity setpoint contains no dynamic limitation (= ramping). The setpoint will directly follow the leading value (input lineAxisMotionVector) of the winder. This solution is not suitable if the winder needs to synchronize to a moving web (e.g., flying splice) |
|---|---|

2. Using the velocity override of the technology object

The TO offers the functionality to scale an active MC_MoveVelocity command in the range between 0 – 200% using the variable <TO>.Override.Velocity. In that case the effective velocity of an axis can be adjusted without triggering a new motion command. Additionally, the used dynamic limitations of the motion command are considered. The following figure shows an example how to realize this function:

- **Network 2**: Cyclic copy of the diameterFactor to the velocity override parameter of the technology object
- **Network 3**: Trigger MC_MoveVelocity command if the referenceVelocity changes. Additional dynamics can be added to the motion command which are considered.




The torque limits and additive torque value is provided in the diagnostic parameters:

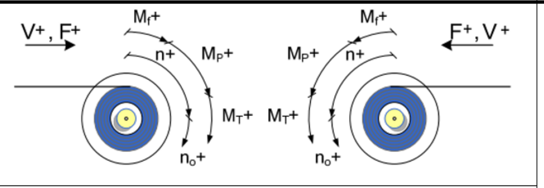
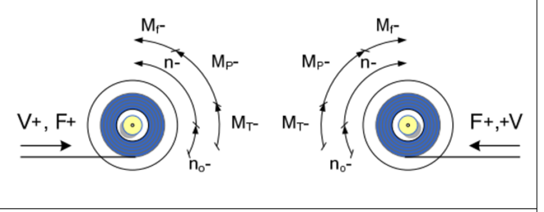
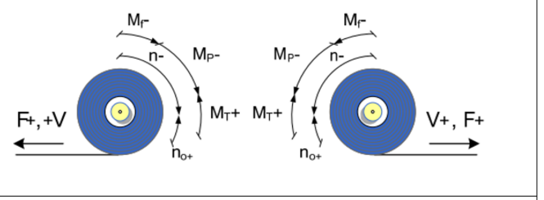
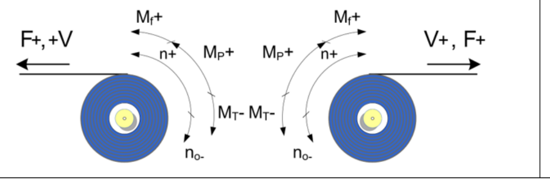
- winderDiag.torquePreControlDiag.upperTorqueLimit
- winderDiag.torquePreControlDiag.lowerTorqueLimit
- winderDiag.torquePreControlDiag.totalTorquePreControl

They can be sent to the drive using the motion control commands MC_TorqueRange and MC_AdditiveTorque.

3. Using the library function block LConSMC_EnableInterface

See function block description.

**Winder operating modes**

| Operating mode | Drawing | Description |
|---|---|---|
| Rewind from above |   | V: Machine velocity<br>F: Tension in material line<br><br>n: speed of winder<br>$n_o$: velocity override for speed<br>controller over control<br><br>$M_f$: Friction torque<br>$M_P$: Pre-control torque<br>$M_T$: Tension torque |
| Rewind from below |   | |
| Unwind from below |   | |
| Unwind from above |   | |

# 5.5 Additional functions / 01_General

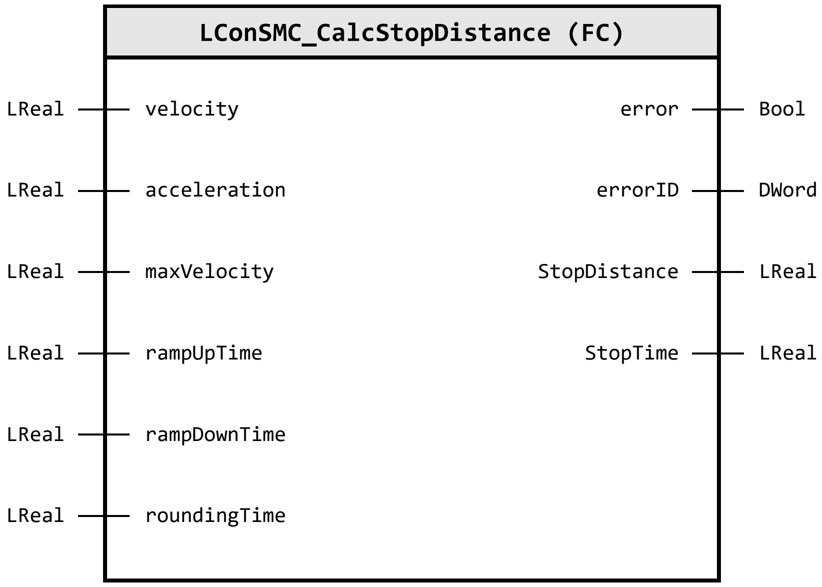
## 5.5.1 LConSMC_CalcStopDistance (FC)

### Short description

This function calculates the distance remaining for braking and associated time to ramp down to zero the Unwinder as per the ramp down time, depending upon the given speed and acceleration. Function block ID: 0005

### Interface description

### Block interface






### Input parameter

| Identifier | Data type | Description |
|---|---|---|
| velocity | LReal | [m/min] / [ft/min] Actual velocity |
| acceleration | LReal | [m/s^2] / [ft/s^2] Actual Acceleration/deceleration |
| maxVelocity | LReal | [m/min] / [ft/min] Reference velocity for ramp time |
| rampUpTime | LReal | [s] Ramp-up time |
| rampDownTime | LReal | [s] Ramp-down time |
| roundingTime | LReal | [s] Rounding time |

### Output parameter

| Identifier | Data type | Description |
|---|---|---|
| error | Bool | TRUE: An error occured |
| errorID | DWord | Detail error description |
| StopDistance | LReal | [m] / [ft] Distance to stop |
| StopTime | LReal | [s] Time to stop |

### Functional description

The function calculates the braking time and braking distance taking the actual motion vector of a drive (velocity, acceleration) as well as the specification of dynamics (maxVelocity, rampUpTime, rampdownTime, roundingTime) into account. The length unit of the calculation corresponds to the unit used as input for the lineSpeed input. The units of the inputs must be consistent as described.

With roundingTime = 0.0 the rounding (jerk limiting, smoothing) is deactivated.

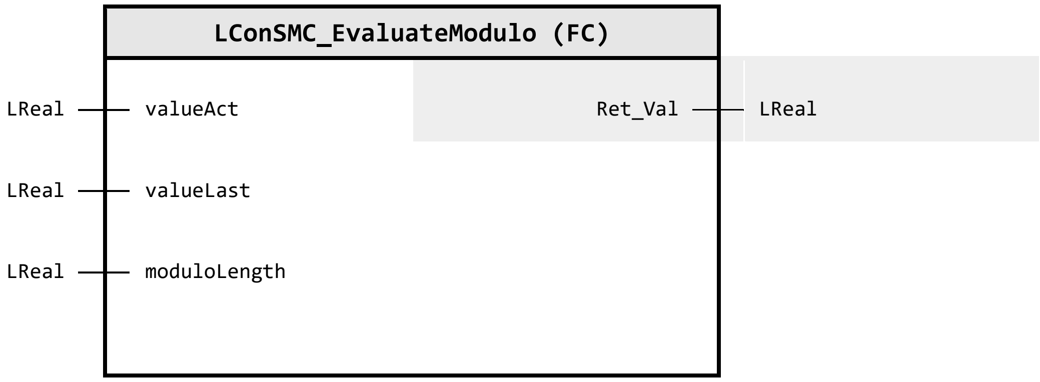
### 5.5.2 LConSMC_EvaluateModulo (FC)

#### Short description

Calculates the difference between two sampling values of a signal with respect to modulo value.
Function block ID: 000B

#### Interface description

#### Block interface






#### Input parameter

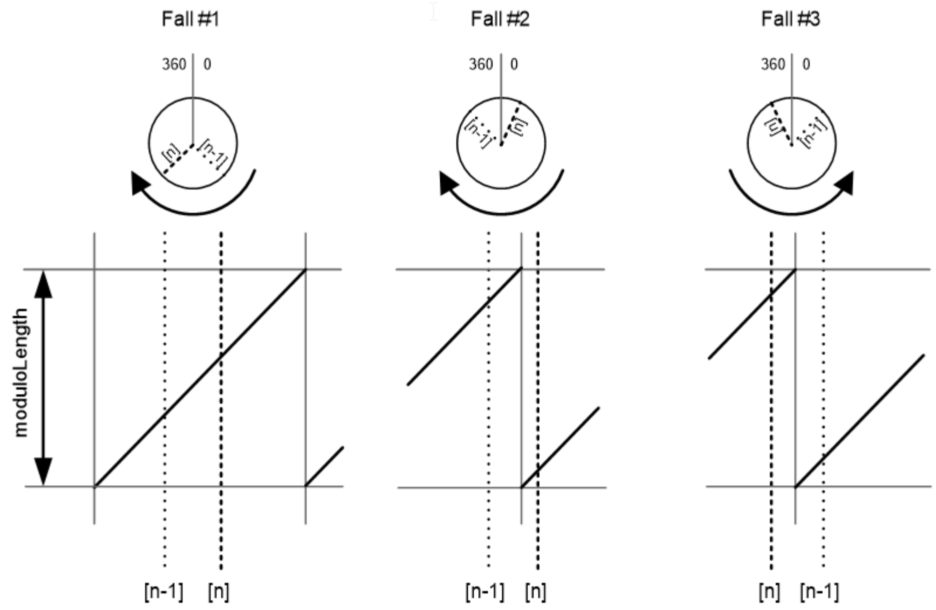
| Identifier | Data type | Description |
|---|---|---|
| valueAct | LReal | Actual value |
| valueLast | LReal | Last value |
| moduloLength | LReal | Modulo length |

#### Output parameter

| Identifier | Data type | Description |
|---|---|---|
| Ret_Val | LReal | Difference to last value with respect to the modulo length |

#### Functional description

The block calculates the position difference between two block calls and corrects the calculated value taking into account the modulo length. The evaluation is realized by comparing the actual and last position value referred to the modulo length. In order that the block correctly operates, it must be ensured that a modulo length is run through in not less than 2 sampling cycles – as otherwise, reliable detection is not possible.




**Case#1**

Value [ n ] > Value [ n – 1 ], no modulo overflow, no modulo correction.
FCLConSMCEvaluateModulo = Value [ n ] – Value [ n – 1 ]

**Case#2**

Value [ n ] < Value [ n - 1 ], modulo overflow in positive direction
FCLConSMCEvaluateModulo = Value [ n ] – Value [ n – 1 ] + moduloLength

**Case#3**

Value [ n ] > Value [ n – 1 ], modulo overflow in negative direction
FCLConSMCEvaluateModulo = Value [ n ] – Value [ n – 1 ] - moduloLength

In the examples, the modulo length is 360:

| valueLast | valuleAct | Raw value | deltaPos | deltaNeg | Result |
|---|---|---|---|---|---|
| 130 | 330 | 200 | 200 | -160 | -160 |
| 330 | 130 | -200 | 160 | -200 | 160 |
| 330 | 150 | -180 | 180 | -180 | 180 |
| 330 | 330 | 0 | 0 | 0 | 0 |
| 150 | 330 | 180 | 180 | -180 | 180 |
| 330 | 170 | -160 | 200 | -160 | -160 |
| 170 | 330 | 160 | 160 | -200 | 160 |

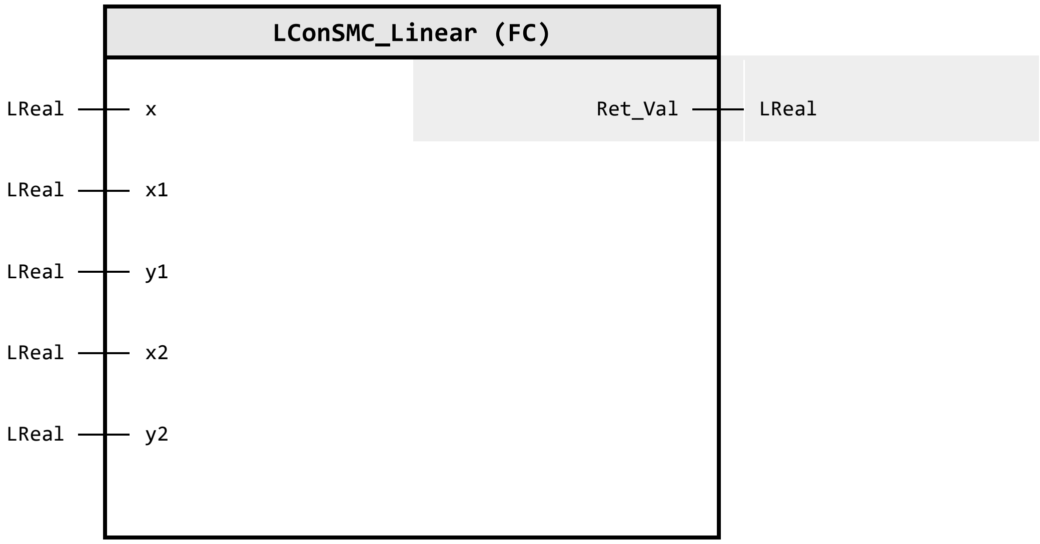
### 5.5.3 LConSMC_Linear (FC)

#### Short description

Linear interpolation between two breakpoints.  Function ID: 0001

#### Interface description

#### Block interface




#### Input parameter

| Identifier | Data type | Description |
|---|---|---|
| x | LReal | Input value |
| x1 | LReal | First breakpoint, x value |
| y1 | LReal | First breakpoint, y value |
| x2 | LReal | Second breakpoint, x-value |
| y2 | LReal | Second breakpoint, y value |

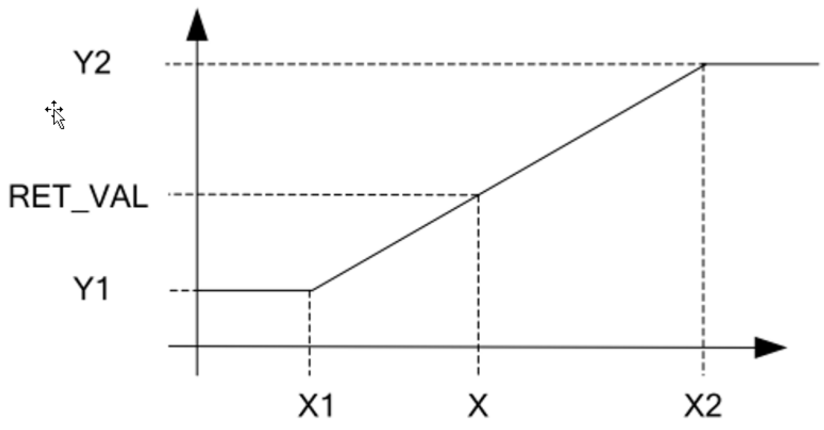
#### Output parameter

| Identifier | Data type | Description |
|---|---|---|
| Ret_Val | LReal | Return value with linear interpolation |

#### Functional description

Using the breakpoints entered, this function determines the matching Y coordinate of the coordinate present at input x. The interpolated Y coordinate is provided as return value of the function. The input point x is limited by the coordinates X1 and X2. If the coordinates X1 and X2 are equal, 0.0 is displayed as return value and no interpolation is carried out.

$$Return\_value = \frac{x - x_1}{x_1 - x_2}\,(y_2 - y_1) + y_1$$


Y2
RET_VAL
Y1
X1
X
X2

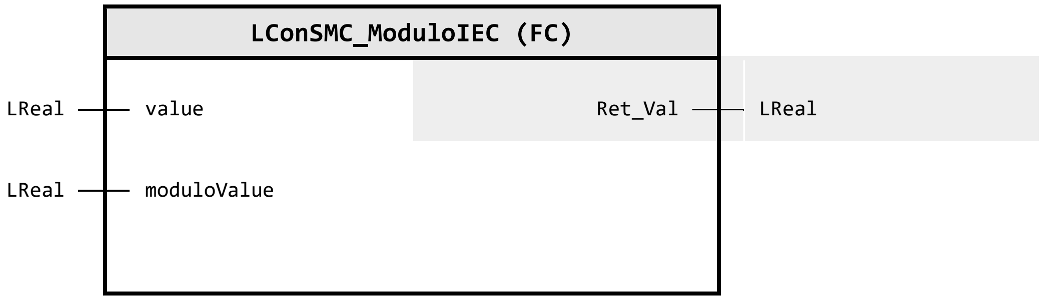
### 5.5.4 LConSMC_ModuloIEC (FC)

**Short description**

This function calculates the modulo division for an input value oft he data type LREAL. Function block ID: 000C

**Interface description**

**Block interface**




**Input parameter**

| Identifier | Data type | Description |
|---|---|---|
| value | LReal | Input value for modulo calculation |
| moduloValue | LReal | Modulo length |

**Output parameter**

| Identifier | Data type | Description |
|---|---|---|
| Ret_Val | LReal | Modulo value |



**Functional description**

Modulo calcuation according to IEC 61131-3:

- positive input value => positive return value
- negative input value => negative return value

If the input parameters are incorrect (moduloValue = 0.0) the return value of the function is zero. The following table describes the functionality of the function:

| Value | moduloValue | Return value |
|---|---|---|
| 180 | 360 | 180 |
| 362 | 360 | 2 |
| -180 | 360 | -180 |
| -362 | 360 | -2 |
| 180 | 0 | 0 |

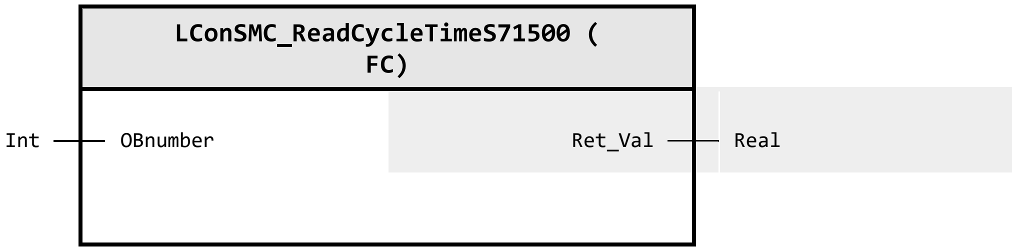
### 5.5.5 LConSMC_ReadCycleTimeS71500 (FC)

**Short description**

Determines the cycle time of a calling OB, called in cyclic interrupt or synchronous operation task. This function can be used in S7-1500 PLC. Function ID: 1001

**Interface description**

**Block interface**




**Input parameter**

| Identifier | Data type | Description |
|---|---|---|
| OBnumber | Int | OB number for which the cycle time should be read |

**Output parameter**

| Identifier | Data type | Description |
|---|---|---|
| Ret_Val | Real | [ms] Cycle time of OB (= 0.0 if invalid OB number at input "OBnumber") |

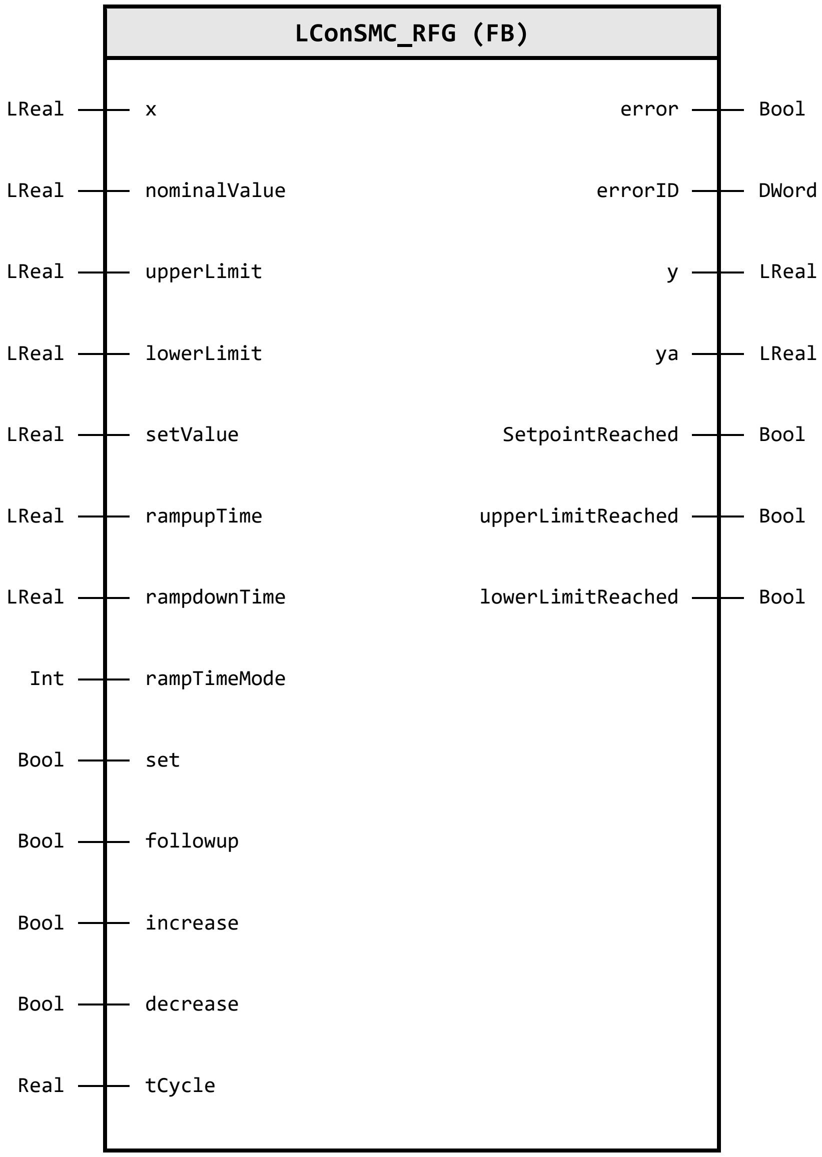
### 5.5.6 LConSMC_RFG (FB)

#### Short description

RFG with MOP and follow-up operation – without rounding. The function block must be called in a cyclic interrupt, e.g. OB30. Function block ID: 1106

#### Interface description

#### Block interface




#### Input parameter

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| x | LReal | 0.0 | Input value |
| nominalValue | LReal | 1.0 | Reference value for ramping times |
| upperLimit | LReal | 0.0 | Upper limit for output value |
| lowerLimit | LReal | 0.0 | Lower limit for output value |
| setValue | LReal | 0.0 | Set value (if "set" = TRUE, output "y" is set to this value) |

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| rampupTime | LReal | 0.0 | [ms] Ramp up time (at least the cycle time) |
| rampdownTime | LReal | 0.0 | [ms] Ramp down time (at least the cycle time) |
| rampTimeMode | Int | #DIFF_TO_ZERO | Select ramping time mode<br>0: Ramp up time if target value moves away from zero<br>1: Ramp up time if target value is bigger than start value |
| set | Bool | FALSE | TRUE: Set output to the value of the input "setValue" |
| followup | Bool | FALSE | TRUE: Integrate output y to setpoint x |
| increase | Bool | FALSE | TRUE: Increase output y until upper limit is reached |
| decrease | Bool | FALSE | TRUE: Decrease output y until lower limit is reached with ramp down time |
| tCycle | Real | 0.0 | [ms] Cycle time the FB is called |

### Output parameter

| Identifier | Data type | Description |
|---|---|---|
| error | Bool | TRUE:<br>An error has occurred, refer to error identification |
| errorID | DWord | Error identification, refer to error messages |
| y | LReal | Ramped output value (Same unit as input value "x") |
| ya | LReal | [Input unit from "x" / tCycle] Changing rate of output "y" |
| SetpointReached | Bool | TRUE: Output value is Input Value |
| upperLimit Reached | Bool | TRUE: upper limit reached |
| lowerLimit Reached | Bool | TRUE: lower limit reached |



### Functional description

The function block includes an integrator with two integration time constants that can be set independently of one another. Output *y* changes according to the following algorithm:

$$y_n = y_{n-1} + ya_n$$

The acceleration value $ya$ are separately calculated for the up ramp and down ramp. Dependent on the input parameter *rampTimeMode*, the *rampUpTime* and *rampDownTime* will be used differently:

- rampTimeMode = DIFF_TO_ZERO:
  An up ramp is when the output value y moves away from zero. A down ramp is when the output value y approaches zero. By this the used ramp time will be switched if the output y crosses zero.
- rampTimeMode = DIFF_OF_CHANGE:
  An up ramp is when the target value is bigger than the output value y. A down ramp is when the target value is smaller than the output value y. By this there is no change between rampUpTime and rampDownTime when the output value y moves through zero.

For the acceleration value during ramping-up the following applies:

$ya = \frac{CycleTime}{rampUpTime} \cdot nominalValue$ (For y > 0)

$ya = -\frac{CycleTime}{rampUpTime} \cdot nominalValue$ (For y < 0)

For the acceleration value when ramping-down the following applies:

$ya = -\frac{CycleTime}{rampDownTime} \cdot nominalValue$ (For y > 0)

$ya = -\frac{CycleTime}{rampDownTime} \cdot nominalValue$ (For y < 0)

*CycleTime* is the sampling time in which the function block is processed.The changeover between ramp-up time and ramp-down time is realized when the direction changes or at the zero crossover of the transfer function. Depending on the logical states of control inputs set, followup, increase and decrease, the operating mode is specified by the control logic. The output quantity can be limited using inputs upperLimit and lowerLimit. When the set limits are reached, through y, binary outputs upperLimitReached or lowerLimitReached are set to TRUE. Binary output setpointReached goes TRUE, if y=x.

### Ramp-up time and ramp-down times

The ramp-up time *rampupTime* is the time in which the absolute value of the output quantity increases by *nominalValue*. The ramp-down *rampdownTime* is the time in which the absolute value of the output quantity decreases by nominalValue. The ramp-up time and ramp-down time can be selected to be different. The lower $\frac{T_A}{rampupTime}$ or $\frac{T_A}{rampDownTime}$, the lower the amplitude change at $y$ from one sampling instant to the next.

### Priority of the control inputs

For the control inputs, the following sequence of priority applies:
*set* before *followup* before *increase* and *decrease*

| Priorit y | Control input | Description |
|---|---|---|
| 1 | set =1 | Setting value setValue is loaded into the integrator, non-integrating |
| 2 | followUp=1 | Output y integrating to setpoint X, tracking |
| 3 | increase=1 | Output y integrating torques upperLimit, tracking |
| 4 | decrease=1 | Output y integrating torques lowerLimit, tracking |

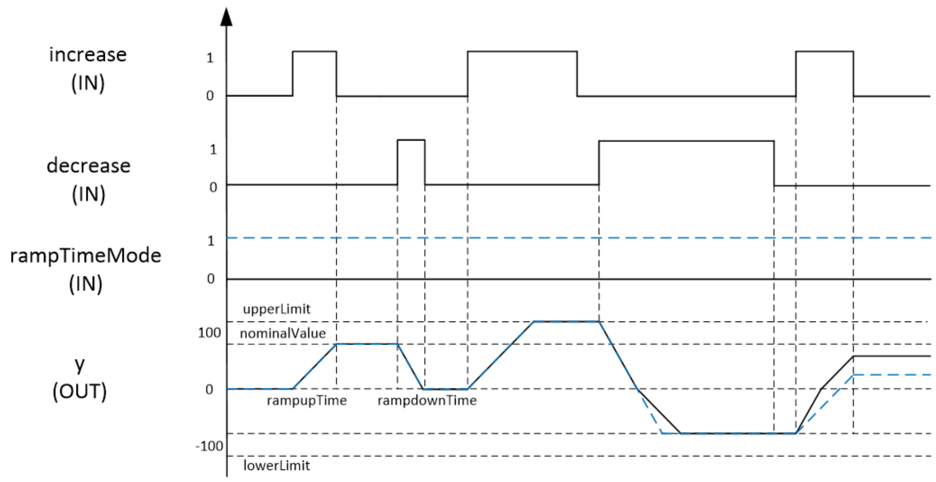
### Operating modes and control of the ramp-function generator

The combination of commands at the control inputs and the possible operating modes can be taken from the truth tables. Standard ramp-function generator operation involves lowerLimit ≤ 0 ≤ upperLimit and lowerLimit < y_n < upperLimit. However, other settings are also possible that are explained in the following. For the setting with lowerLimit ≥ upperLimit, the following applies: Limit upperLimit is dominant with respect to limit lowerLimit. If output y reaches one of the set limits lowerLimit or upperLimit, during the control sequence, the integrator value is held. Output value y is then kept constant until the integrator value leaves the limit due to a change of the input quantities. If the integrator is at the limit and the limit value is changed the integrator behaves differently depending on the direction in which the limit value was changed. If the absolute value of a limit value is increased, and if the control logic specifies that the ramp-function generator should run in the same direction, then the integrator integrates from the previously defined value according to the selected ramp-up time until the output again reaches the limit value. If the absolute value of a limit value is reduced, the integrator integrates from the previously held value – according to the selected ramp-down time – until the output again reaches the limit value. The integrator operates internally with an increased accuracy so that even for small setpoint-actual value differences it still integrates. However, it should be carefully noted that the sampling time is sufficiently low referred to the ramp-up or ramp-down time. rampupTime and rampdownTime are internally limited: rampupTime ≥ cycleTime, rampdownTime ≥ cycleTime

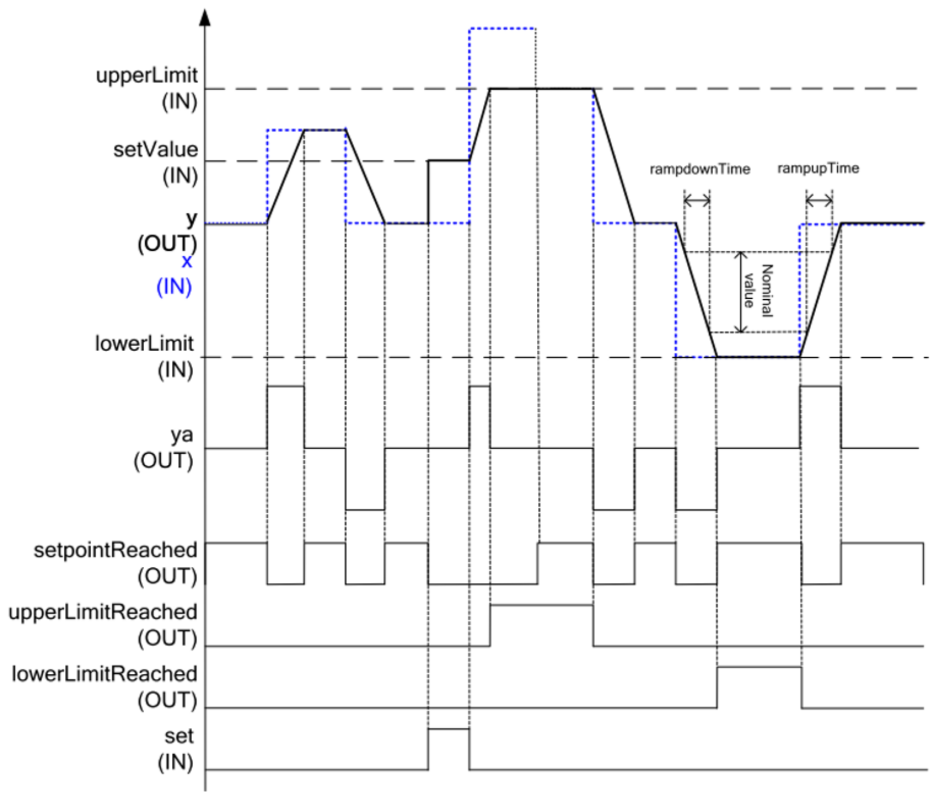
| set | foll ow up | inc. | dec. | ya_n | y_n | mode | Comment |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | $y_{n+1}$ | Stop | Y is constant |
| 1 | 1 | 1 | 1 | Step | $setValue_n$ | Sets output to setValue | setValue is any value, fixed or variable setValue is any value, fixed or variable |
| 0 | 1 | 1 | 1 | cycleTime/rampupTime<br>cycleTime/rampdownTi me | $y_{n+1} + ya_n$ | Normal operation<br>$y \rightarrow x$ | rampupTime for:<br>x>y & y >= 0<br>x<y & y <= 0<br>rampdownTime for: |

| set | follow up | inc. | dec. | ya_n | y_n | mode | Comment |
|---|---|---|---|---|---|---|---|
| | | | | | | | x>y & y <= 0<br>x<y & y >= 0 |
| 0 | 0 | 1 | 0 | cycleTime/rampupTime<br>cycleTime/rampdownTime | $y_{n+1}$ $+ ya_n$ | Approach the upper limit value<br>$y \rightarrow upperLimit$ | rampupTime, rampdownTime as before, dependent on the starting (initial) position |
| 0 | 0 | 0 | 0 | cycleTime/rampdownTime<br>cycleTime/rampupTime | $y_{n+1}$ $+ ya_n$ | Approach the lower limit value<br>$y \rightarrow lowerLimit$ | rampupTime, rampdownTime as before, dependent on the starting (initial) position |

**Timing Diagram - Potentiometer**




**Timing Diagram - Normal Operation**



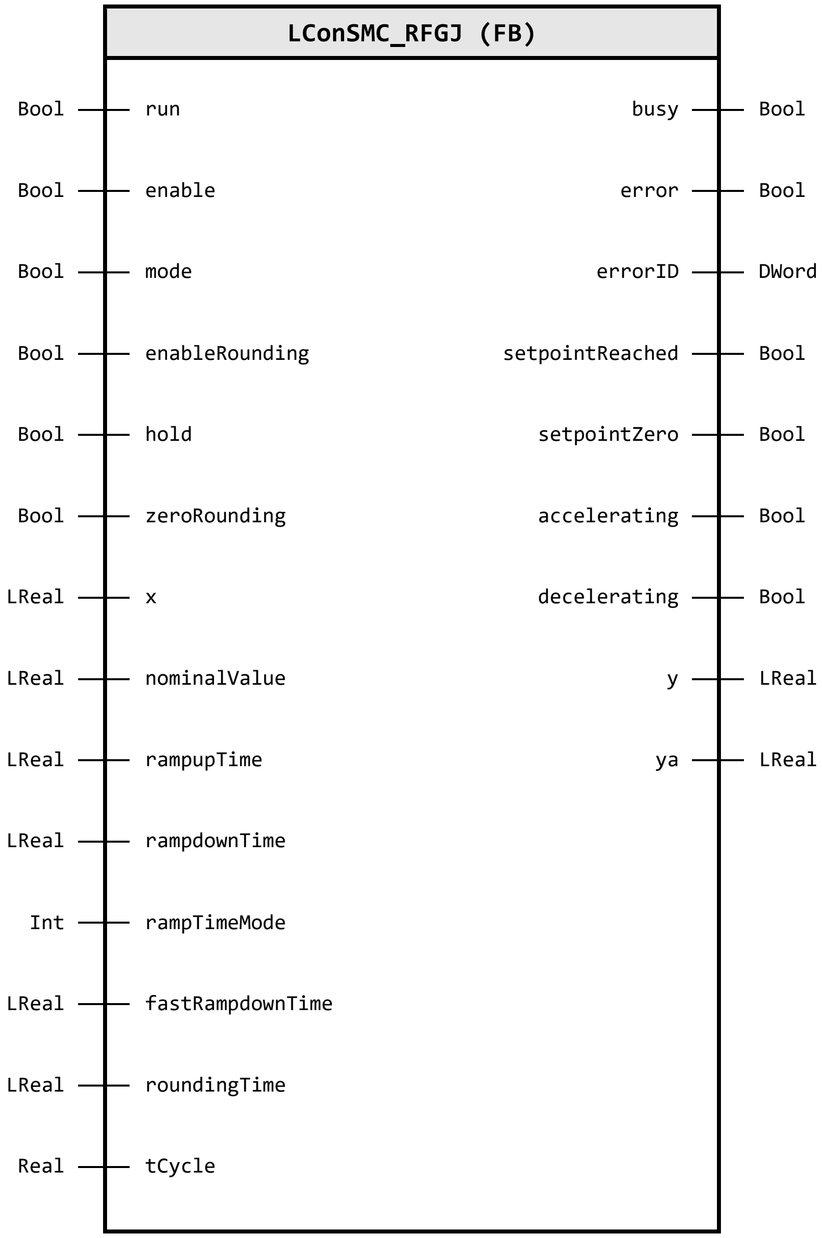
## 5.5.7 LConSMC_RFGJ (FB)

### Short description

Ramp-function generator with ramp-up, ramp-down times and rounding-off. The function block has to be called in a cyclic interrupt, e.g. OB30.  Function block ID: 1107

### Interface description

### Block interface




### Input parameter

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| run | Bool | FALSE | TRUE: Enable function block |
| enable | Bool | FALSE | TRUE: Enable ramp function |
| mode | Bool | TRUE | TRUE: Use ramp up-time/ramp down-time<br>FALSE: Use emergency ramp down-time |

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| enableRounding | Bool | TRUE | TRUE: Enable Rounding |
| hold | Bool | FALSE | TRUE: Hold actual output value constant (with jerklimit!) |
| zeroRounding | Bool | FALSE | TRUE: Enable Rounding at crossing 0.0<br>FALSE: Disable Rounding at crossing 0.0 |
| x | LReal | 0.0 | Input value |
| nominalValue | LReal | 1000.0 | Reference value for ramping times |
| rampupTime | LReal | 1.0 | [s] Ramp up time (at least the cycle time) |
| rampdownTime | LReal | 1.0 | [s] Ramp down time (at least the cycle time) |
| rampTimeMode | Int | #DIFF_TO_ZERO | Select ramping time mode<br>0: Ramp up time if target value moves away from zero<br>1: Ramp up time if target value is bigger than start value |
| fastRampdownTime | LReal | 0.1 | [s] Emergency Ramp-down time |
| roundingTime | LReal | 0.5 | [s] Rounding time |
| tCycle | Real | 0.0 | [ms] Cycle time the FB is called |

### Output parameter

| Identifier | Data type | Description |
|---|---|---|
| busy | Bool | TRUE: Functionblock is busy |
| error | Bool | TRUE:<br>An error has occurred, refer to error identification |
| errorID | DWord | Error identification, refer to error messages |
| setpointReached | Bool | TRUE: Output value is Input Value |
| setpointZero | Bool | TRUE: Output Value = 0 |
| accelerating | Bool | TRUE: Output is increasing |
| decelerating | Bool | TRUE: Output is decreasing |
| y | LReal | Ramped output value (Same unit as input value "x) |
| ya | LReal | [Input unit from "x" / second] Changing rate of output "y" |

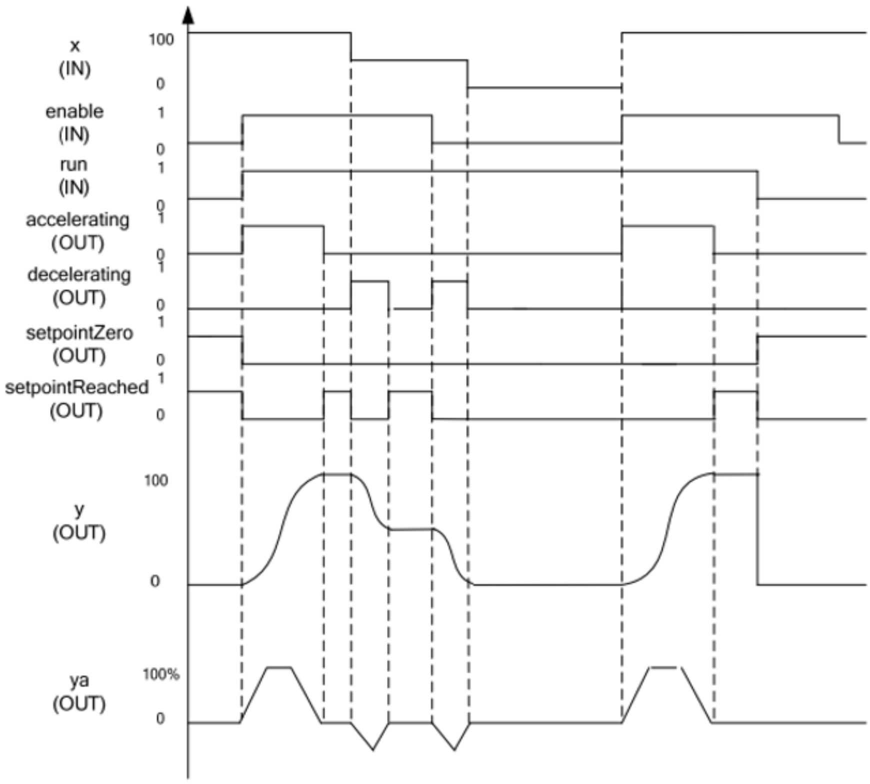
### Functional description

The function block is active with the input run at TRUE and the calculation can be started or stopped with input enable. Depending on the parameterization of the inputs, a ramp function is calculated. Various options can be set by selecting the inputs. If mode is at False, the ramp is braked using a special entered deceleration time (fastRampdownTime) and without jerk limiting. This means that the ramp can be effectively decelerated for an Emergency Stop – this is also possible while accelerating. If mode is at True, then the drive is braked normally with the entered deceleration time (rampdownTime). The acceleration is always calculated with the entered acceleration time (rampupTime) independent of mode.
Dependent on the input parameter *rampTimeMode*, the *rampUpTime* and *rampDownTime* will be used differently. A detailed description can be found in the function block description of the function block LConSMC_RFG.

In order to reach the setpoint (x), the appropriate interval values are output (at y) and the actual change rate is output (at ya). If input enableRounding is set to TRUE, the ramp is calculated with jerk limiting. The rounding-off is determined by the rounding-off time (roundingTime). The zero crossing (when the input x changes direction) can be parameterized with or without rounding. As soon as the output value is zero this is displayed using the setpointZero output. Further, when accelerating output acceleration is set, when decelerating, output deceleration is set and for a constant value setpointReached is set. If the block is in the enable mode, then output busy is set to TRUE. If the hold input is set to TRUE, the ramp-function generator is stopped (held) and the actual acceleration is ramped-down to zero. As long as hold is TRUE, setpoint x is ignored and output ya is set to zero.

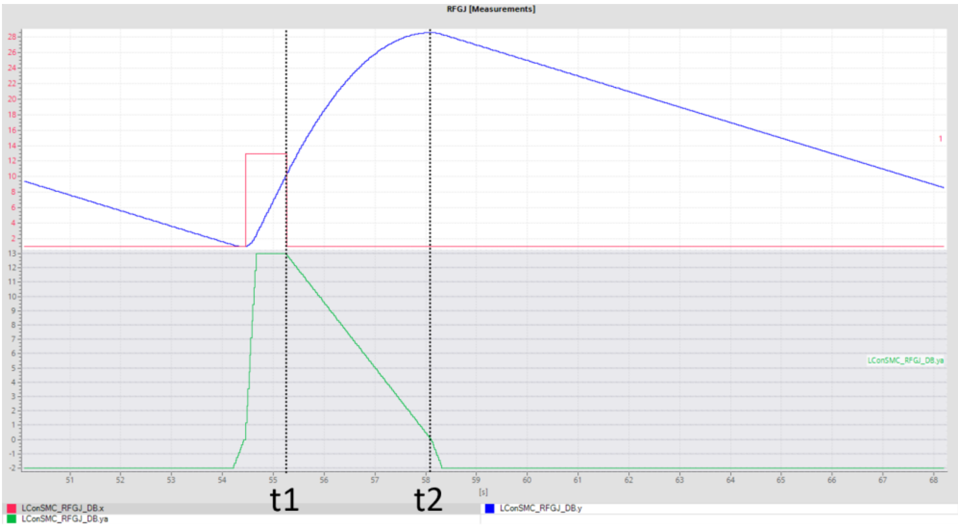
**Note** In case of rampTimeMode=DIFF_TO_ZERO the zero-crossing is always rounded if the values for rampUpTime and rampDownTime are different.

**Normal operation with rounding**

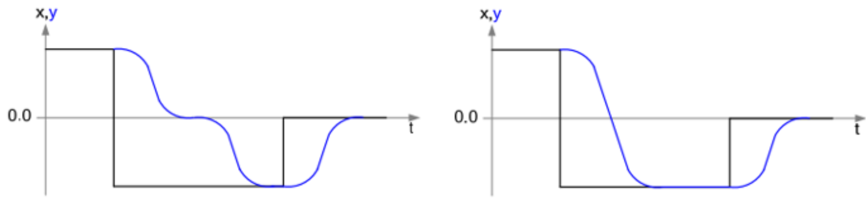



**Overshoot with active rounding**

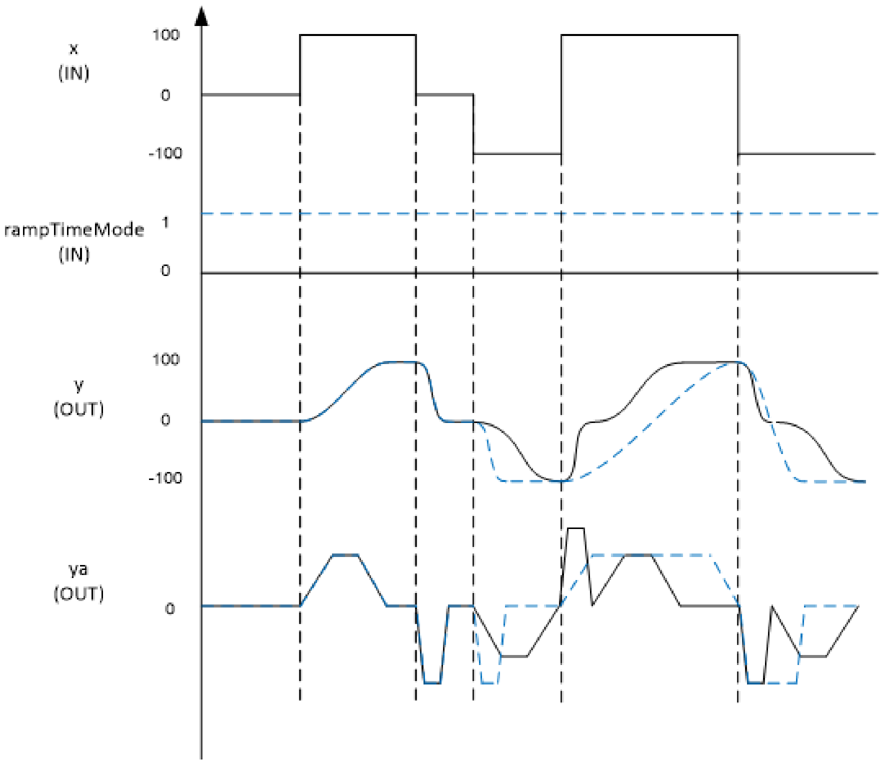
Changing the input dynamics (nominalValue, roundingTime, rampupTime, rampdownTime) together with a new setpoint during acceleration/ deceleration can lead to an overshoot of the output signal before reversing. Depending on the settings the current acceleration/ deceleration must be ramped to zero first before ramping up into the opposite direction. An example is displayed in the following figure. At time stamp t1 the dynamics and the setpoint (red) are changed during acceleration. The output (blue) is increasing until time stamp t2 before the current acceleration is fully ramped to zero and is changing its sign.



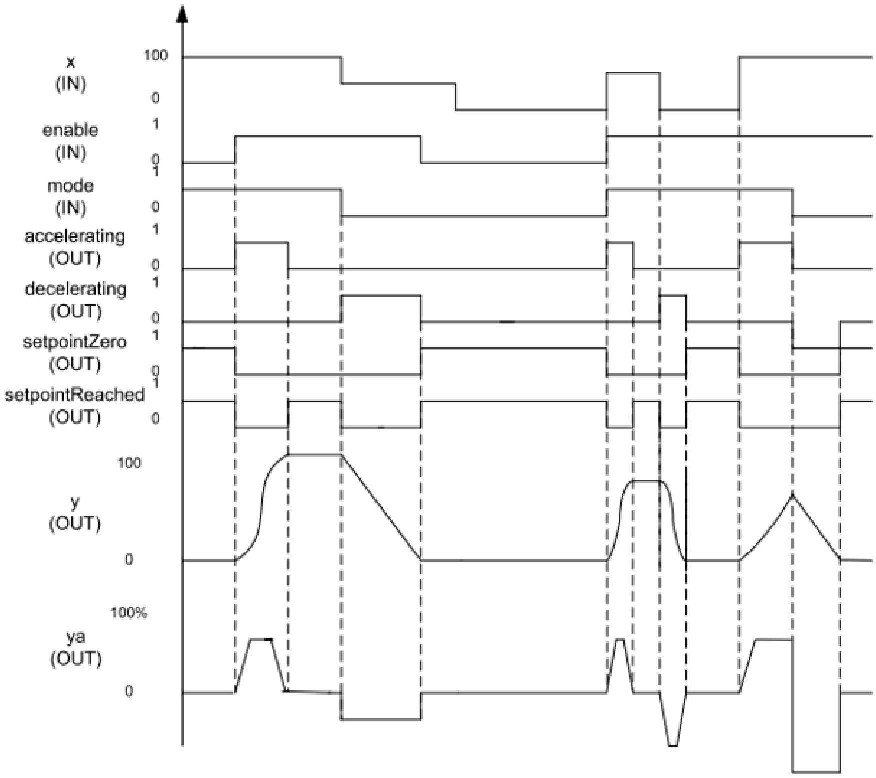

**With and without zero rounding**




**Using different rampTimeModes**




**Operation with fast stop mode**


x
(IN)
100
0
enable
(IN)
1
0
mode
(IN)
1
0
accelerating
(OUT)
1
0
decelerating
(OUT)
1
0
setpointZero
(OUT)
1
0
setpointReached
(OUT)
1
0
100
y
(OUT)
0
100%
ya
(OUT)
0

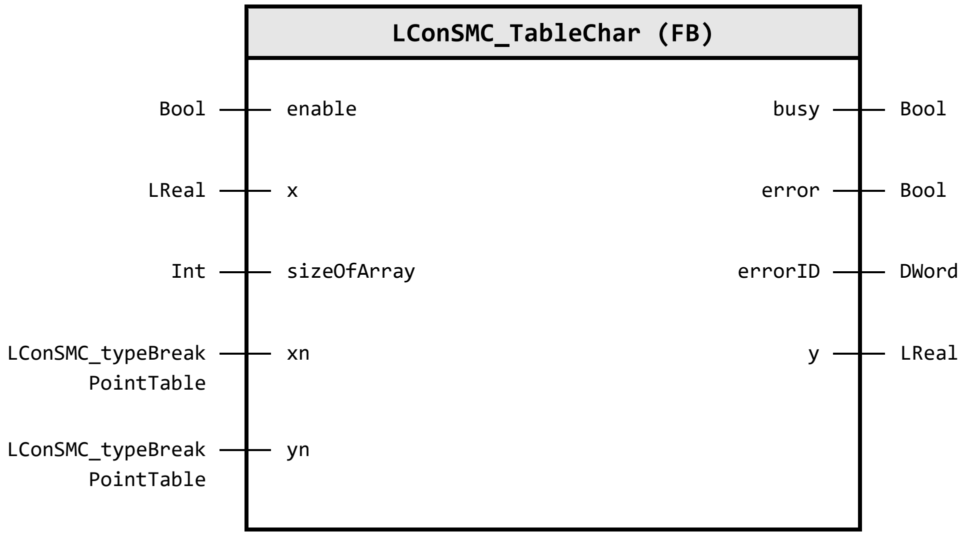
### 5.5.8 LConSMC_TableChar (FB)

#### Short description

Interpolation from breakpoint table, the function block can be called in any cyclic task. Function block ID: 0106

#### Interface description

#### Block interface






#### Input parameter

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| enable | Bool | FALSE | TRUE: FB is activated |
| x | LReal | 0.0 | Input value (x-axis) |
| sizeOfArray | Int | #MAX_SIZE_OF_ARRAY | Number of breakpoints that should be used |
| xn | LConSMC_typeBreak PointTable | --- | breakpoint table for x-axis |
| yn | LConSMC_typeBreak PointTable | --- | breakpoint table for y-axis |

#### Output parameter

| Identifier | Data type | Description |
|---|---|---|
| busy | Bool | TRUE: FB is active |
| error | Bool | TRUE: An error has occurred, refer to "errorID" |
| errorID | DWord | Error identification, refer to error messages |
| y | LReal | Output value |

#### Functional description

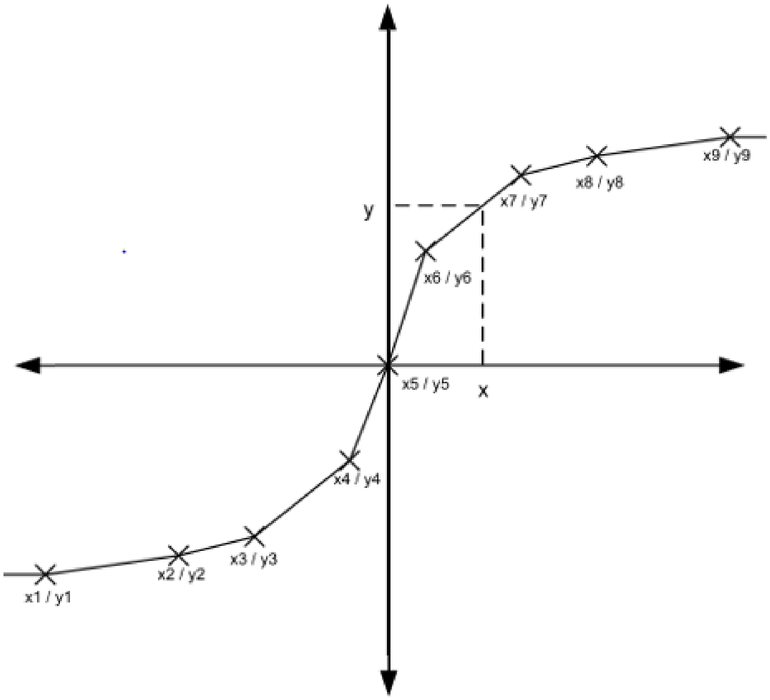
The function blocks adapts the output value y corresponding to the input value x based on a breakpoint table of flexible size. Between the breakpoints the values are interpolated linearly.

Purpose of the function block is to e.g.:

- Linear characteristic curves

- Simulation of non-linear transfer elements
- Minimum number of breakpoint elements is two

The x values of the curve must increase monotonically; the y values have no limitation. The total number of breakpoints is defined by the size of the data reference. The data type of the addressed data area must be LREAL. Is x lower than the x value of the first coordinate, the output y value will maintain the level of the y value of the first coordinate. The same applies if x is bigger than the last x coordinate of the table.



## 5.6 Additional functions / 02_Closed-loop-control

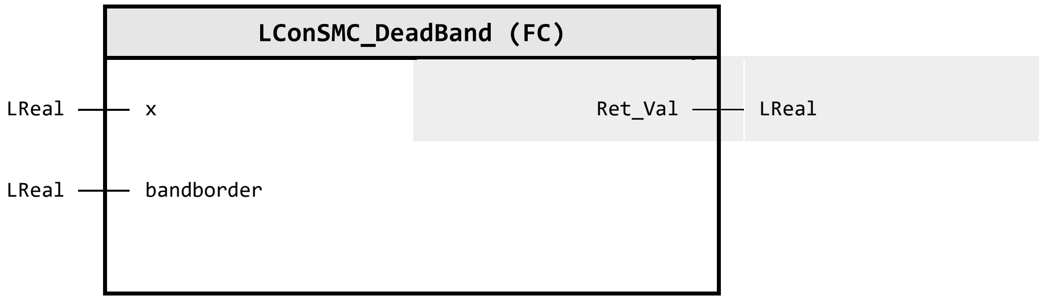
### 5.6.1 LConSMC_DeadBand (FC)

#### Short description

Dead band filter within a specified value. The function can be called in any task. Function ID: 0002

#### Interface description

#### Block interface




#### Input parameter

| Identifier | Data type | Description |
|---|---|---|
| x | LReal | Input value |
| bandborder | LReal | Value of the dead band symmetrical around 0<br>If "x" is within the deadband the output is 0. If "x" is outside the output is x +- bandborder |

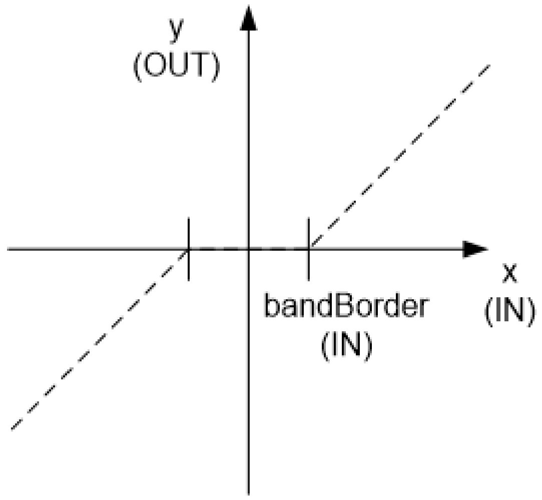
#### Output parameter

| Identifier | Data type | Description |
|---|---|---|
| Ret_Val | LReal | Output |

#### Functional description

If the input value (x) is located in the dead band range (-bandBorder < x < +bandBorder), the output value is set to zero. However, if it is at or below (–bandBorder) the output will be x – bandBorder. If the input value is at or above +bandBorder, x + bandBorder is set as output value. The dead band – symmetrical around the zero point – can be set using the bandBorder parameter.

| Input value x | Return value |
|---|---|
| x <= -bandBorder | x + bandBorder |
| -bandBorder < x < bandBorder | 0 |
| x >= bandBorder | x - bandBorder |


y
(OUT)
bandBorder
(IN)
x
(IN)

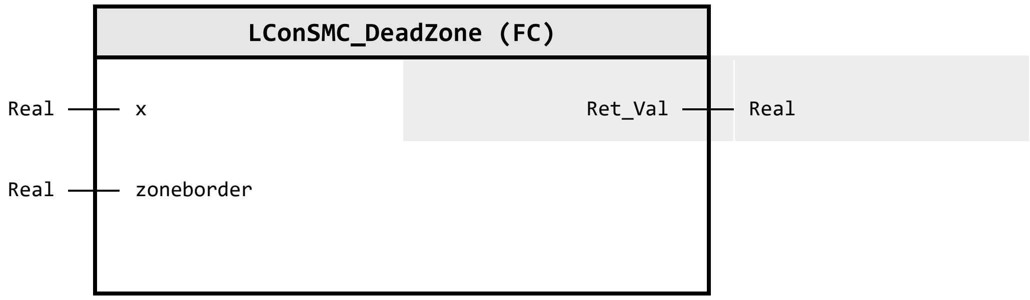
### 5.6.2 LConSMC_DeadZone (FC)

**Short description**

Dead zone filter The function can be called in any task.  Function ID: 0003

**Interface description**

**Block interface**




**Input parameter**

| Identifier | Data type | Description |
|---|---|---|
| x | Real | Input value |
| zoneborder | Real | Value of the dead band symmetrical around 0<br>If "x" is within the zoneborder the output is 0. If "x" is outside the output is x |

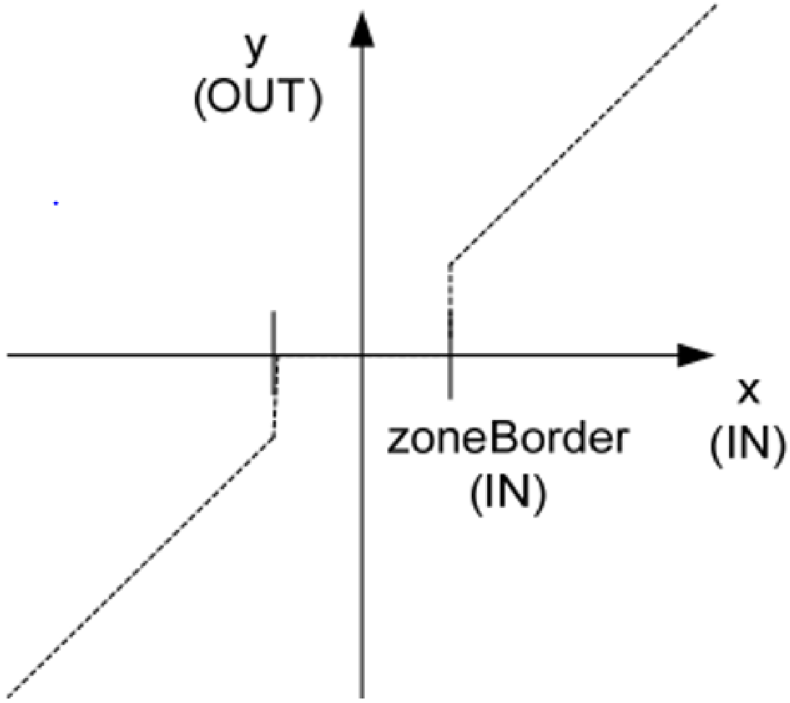
**Output parameter**

| Identifier | Data type | Description |
|---|---|---|
| Ret_Val | Real | Output value |

**Functional description**

If the input value (x) is in the range of the dead zone (-zoneBorder < x < + zoneBorder), the output value is set to zero. If the input value is at / below (–zoneBorder) or at / above +zoneBorder, input x is the output value. The dead zone – symmetrical around the zero point – can be set using the value zoneBorder.

| Input value x | Return value |
|---|---|
| x <= -zoneBorder | x |
| -zoneBorder < x < zoneBorder | 0 |
| x >= zoneBorder | x |


y
(OUT)
x
(IN)
zoneBorder
(IN)

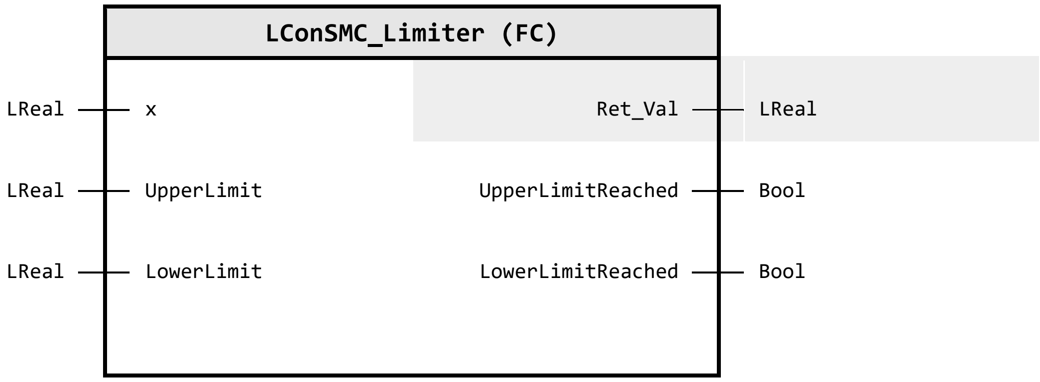
### 5.6.3 LConSMC_Limiter (FC)

**Short description**

Limiting the input value between specified limits, the function can be called in any task.
Function block ID: 0004

**Interface description**

**Block interface**






**Input parameter**

| Identifier | Data type | Description |
|---|---|---|
| x | LReal | Input value |
| UpperLimit | LReal | Upper limit |
| LowerLimit | LReal | Lower limit |

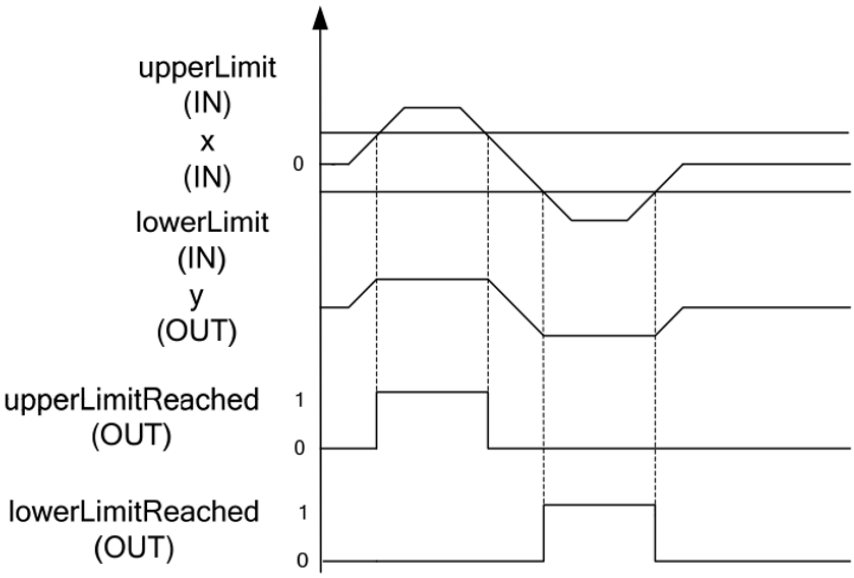
**Output parameter**

| Identifier | Data type | Description |
|---|---|---|
| Ret_Val | LReal | Limited output value |
| UpperLimit Reached | Bool | TRUE: Upper limit reached |
| LowerLimit Reached | Bool | TRUE: Lower limit reached |

**Functional description**

The function block transfers the input value x to its return value. The input value is limited dependent on the input parameters upperLimit and lowerLimit. If the input quantity reaches upper limit upperLimit, output upperLimitReached is set to TRUE. If the input quantity reaches lower limit value lowerLimit, output lowerLimitReached is set to TRUE. If the lower limit value is greater or equal to the upper limit value, output y is set to the upper limit value upperLimit.

**Timing diagram**


upperLimit
(IN)
x
(IN)
0
lowerLimit
(IN)
y
(OUT)
upperLimitReached
(OUT)
1
0
lowerLimitReached
(OUT)
1
0

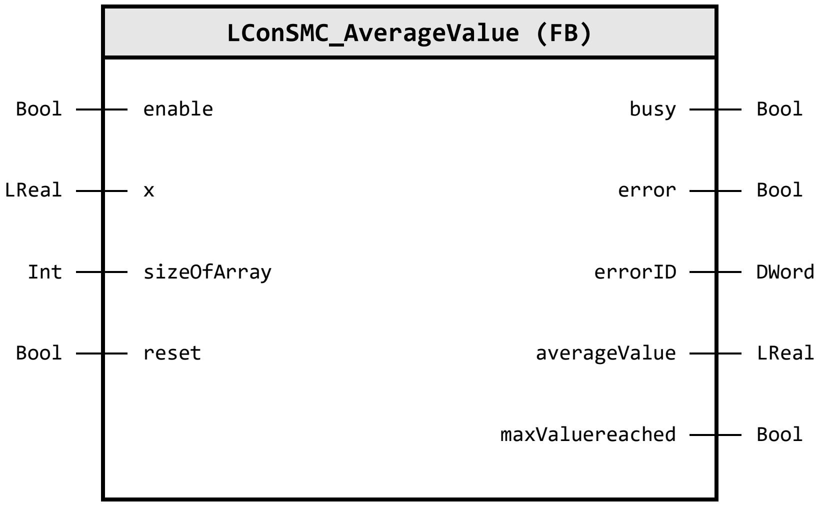
## 5.6.4 LConSMC_AverageValue (FB)

### Short description

Average value filter. The function block must be called in a cyclic task. Function block ID: 0105

### Interface description

### Block interface




### Input parameter

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| enable | Bool | FALSE | TRUE: Enable function block |
| x | LReal | 0.0 | Input value |
| sizeOfArray | Int | 20 | Number of values to calculate avarage value of [1-20] |
| reset | Bool | FALSE | TRUE: Reset avarage value calculation |

### Output parameter

| Identifier | Data type | Description |
|---|---|---|
| busy | Bool | TRUE: Function block is active |
| error | Bool | TRUE: An error occured |
| errorID | DWord | Detailed error description |
| averageValue | LReal | Average value |
| maxValuereached | Bool | Defined number of values are used for average value calculation (internal array is filled) |

### Functional description

The function block calculates the moving average value (averageValue) of the input values that are entered. Each time the block is called, the value (x) currently present is saved in an array. The values stored in the array are deleted upon deactivation of the function block. The function block is activated via the input enable and calculates values from that instant. The output maxValueReached is set if the amount of values in an array reaches the defined number sizeOfArray.

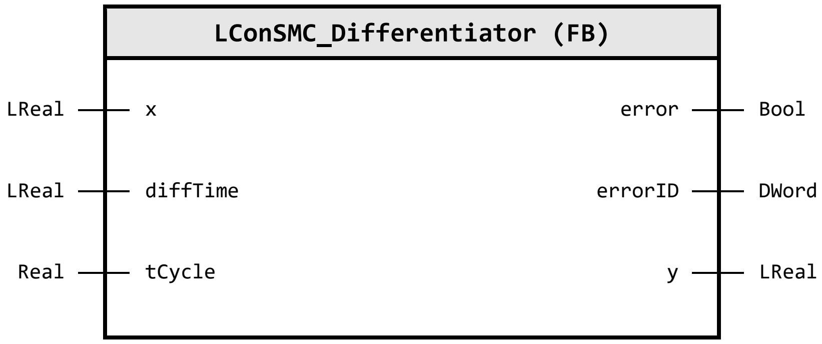
### 5.6.5 LConSMC_Differentiator (FB)

#### Short description

Numerical differentiator function block must be called in a cyclic interrupt, e.g. OB30. Function block ID: 1103

#### Interface description

#### Block interface




#### Input parameter

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| x | LReal | 0.0 | Input value |
| diffTime | LReal | 0.0 | [ms] Time constant for differentator |
| tCycle | Real | 0.0 | [ms] Cycle time the function block is called |

#### Output parameter

| Identifier | Data type | Description |
|---|---|---|
| error | Bool | TRUE: An error occured |
| errorID | DWord | Detailed error description |
| y | LReal | Output value = (xn-xn-1) * diffTime/cycleTime |

#### Functional description

The output value y changes as a function of the rate of change of input value x, multiplied by the differential time constant diffTime. The input *diffTime* is internally limited: cycleTime ≤ diffTime ≤ 1000 Discrete values are calculated using the following algorithm:

$$y_n = (x_n - x_{n-1}) \cdot \frac{diffTime}{cycleTime}$$

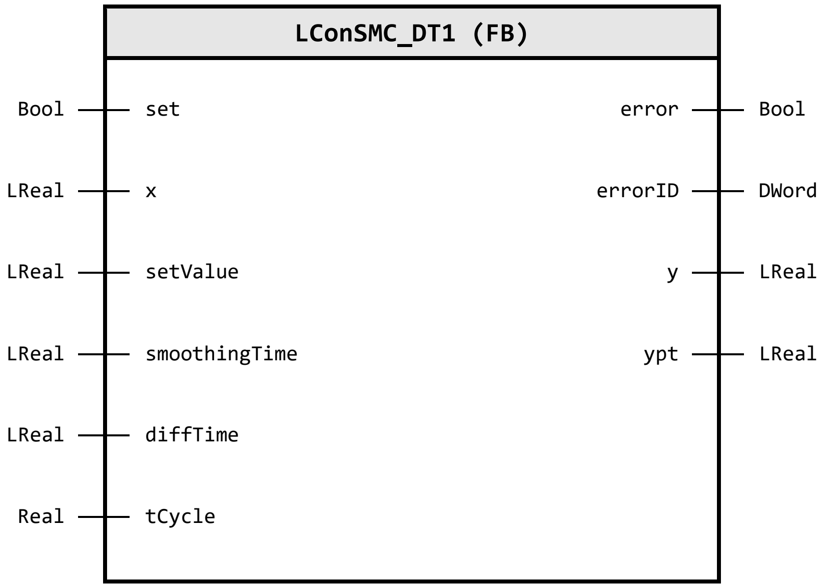
### 5.6.6 LConSMC_DT1 (FB)

#### Short description

DT1-filter The function block has to be called in a cyclic interrupt, e.g. OB30.  Function block ID: 1104

#### Interface description

#### Block interface






#### Input parameter

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| set | Bool | FALSE | TRUE: Set output = "setValue" |
| x | LReal | 0.0 | Input value |
| setValue | LReal | 0.0 | Set value (if "set" = TRUE) |
| smoothingTime | LReal | 0.0 | [ms] Smoothing time (tCycle - 5000) |
| diffTime | LReal | 0.0 | [ms] Time constant for differentator |
| tCycle | Real | 0.0 | [ms] Cycle time the function block is called |

#### Output parameter

| Identifier | Data type | Description |
|---|---|---|
| error | Bool | TRUE: An error occured |
| errorID | DWord | Detailed error description |
| y | LReal | Output value |
| ypt | LReal | Output value of smoothing element |

#### Functional description

Input value x is dynamically delayed and the smoothing time constant *smoothingTime* is entered at a differentiating element and at block output *ypt*. The output value *y* of the complete DT1 element is proportional to the change rate of *ypt* multiplied with differential time constant *diffTime*. The input *diffTime* is internally limited: cycleTime ≤ diffTime ≤1000. The input *smoothingTime* defines the gradient of the down ramp of the output value. This specifies the time after which the transfer function – after smoothing and differentiation – has dropped to 37%

of x-diffTime/ smoothingTime. For a sufficiently high smoothingTime/cycleTime (smoothingTime /cycleTime > 10), the transfer function corresponds to a characteristic of:

$$y(t) = x\left(\frac{diffTime}{smoothingTime}\right)e^{\frac{-t}{smoothingTime}} \qquad where\ t = n(cycleTime)$$

The input smoothingTime is internally limited: cycleTime ≤ smoothingTime ≤ 5000. The discrete values are calculated using the algorithm:

$$y1_n = \frac{diffTime}{smoothingTime} \cdot (x_n - ypt_{n-1})$$

$$ypt_n = ypt_{n-1}\frac{cycleTime}{smoothingTime} \cdot (x_n - ypt_{n-1})$$

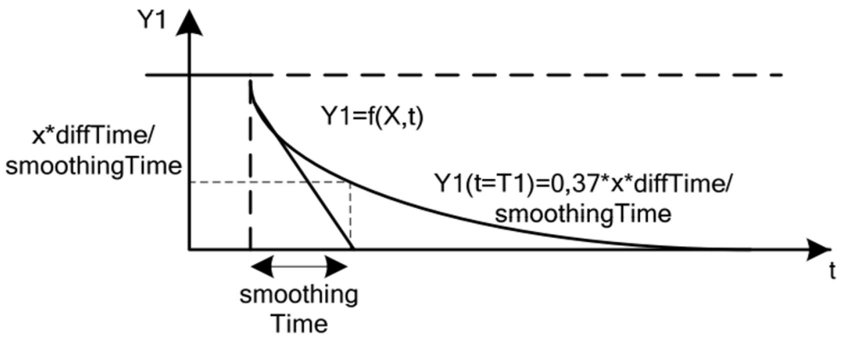
The higher smoothingTime/cycleTime, the lower the amplitude changes at y and at ypt from one sampling instant to the next.
The higher the diffTime/cycleTime, the higher the amplitude changes at y from one sampling instant to the next.

**Setting function**

When the setting function is active, setting value setValue is transferred to output y; the output of the smoothing element is obtained as follows:

$$ypt_n = x_n - \frac{smoothingTime}{diffTime} \cdot setValue_n \qquad for\ diffTime \neq 0$$

In this case, the internal limits for smoothingTime and diffTime apply. For diffTime = 0, the output values remain unchanged as long as set = TRUE.



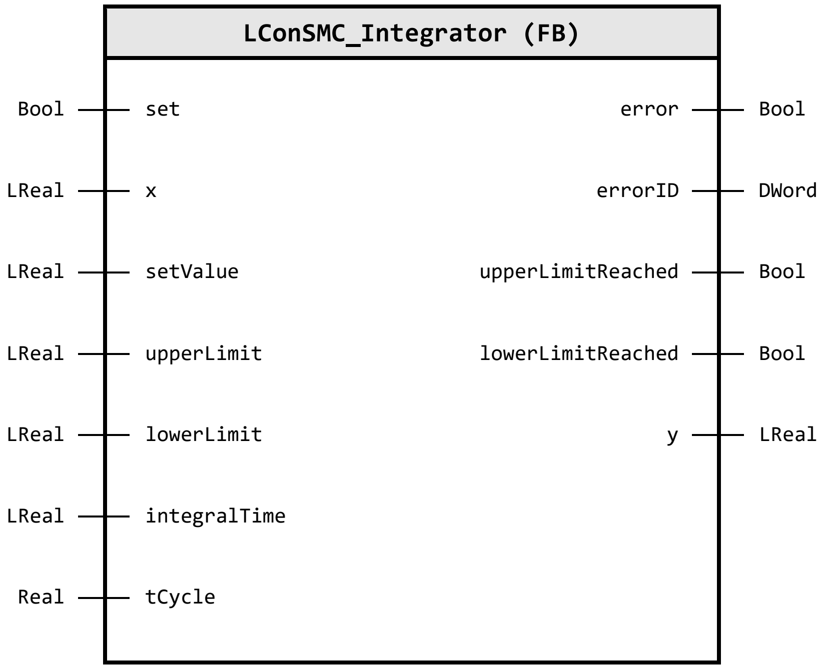
## 5.6.7 LConSMC_Integrator (FB)

### Short description

Numerical integrator function block has to be called in a cyclic interrupt, e.g. OB30. Function block ID: 1102

### Interface description

### Block interface






### Input parameter

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| set | Bool | FALSE | TRUE: Set output = "setValue" |
| x | LReal | 0.0 | Input value |
| setValue | LReal | 0.0 | Set value (if "set" = TRUE) |
| upperLimit | LReal | 1.0 | Upper limit |
| lowerLimit | LReal | 0.0 | Lower limit |
| integralTime | LReal | 5.0 | [ms] Time constant for integrator |
| tCycle | Real | 0.0 | [ms] Cycle time the function block is called |

### Output parameter

| Identifier | Data type | Description |
|---|---|---|
| error | Bool | TRUE: An error occured |
| errorID | DWord | Detailed error description |
| upperLimit Reached | Bool | TRUE: Upper limit reached |
| lowerLimit Reached | Bool | TRUE: Lower limit reached |
| y | LReal | Output value |

### Functional description

he function block integrates the input value x with integration time integralTime. The result is output at y. The integrator is limited by the upper limit upperLimit and the lower limit lowerLimit. If the integrator reaches one of the limits, then this is connected to the output. Further, when the particular limit at the output is reached, this is indicated using upperLimitReached and lowerLimitReached. Using the setting function, which is activated with set =True, the output can be set to a pre-defined value (setValue). If *set* input is FALSE, then starting from the set value the integrator continues to integrate in the direction specified using input value x.
The input integralTime is internally limited: cycleTime ≤ integralTime ≤ 60000
The discrete values are calculated using the following algorithm:

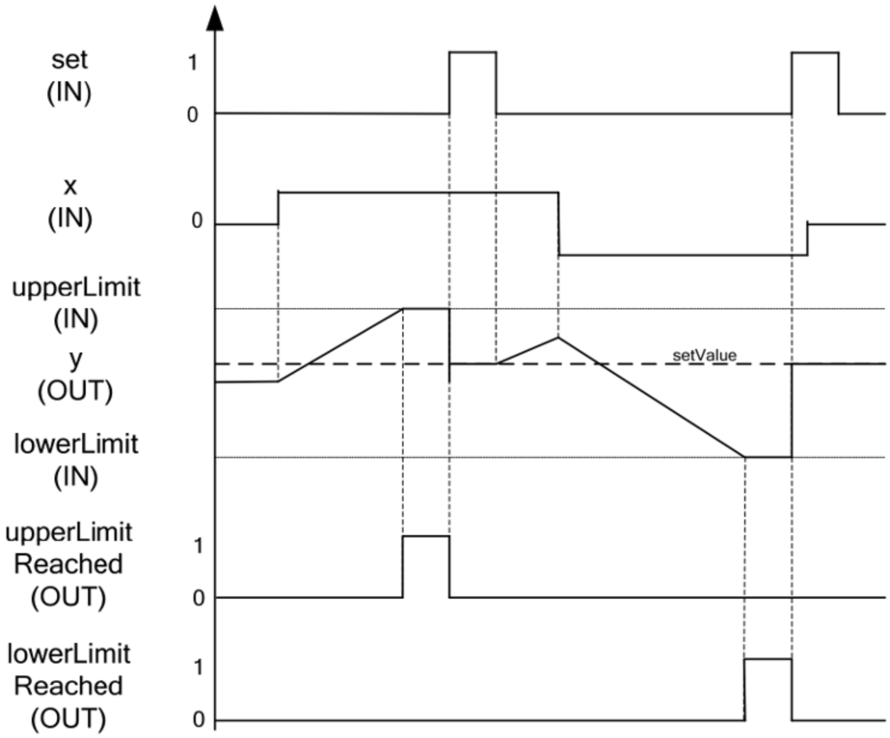
$$yn = y_{n-1} + \frac{cycleTime}{integralTime} \cdot x_n$$

### Truth Table for lowerlimit < upperlimit

| Set | Condition | Y | Upper limit reached | Lower limit reached | Mode |
|---|---|---|---|---|---|
| 0 | lowerLimit < integration result <upperLimit | yn | 0 | 0 | Integrating |
| 0 | integration result >= upperLimit | UpperLimit | 1 | 0 | Integrator at the upper limit |
| 0 | integration result <= lowerLimit | LowerLimit | 0 | 1 | Integrator at the lower limit |
| 1 | lowerLimit < setValue < upperLimit | setValue | 0 | 0 | Setting |
| 1 | setValue >= upperLimit | upperLimit | 1 | 0 | Integrator at the upper limit |
| 2 | setValue =< lowerLimit | lowerLimit | 0 | 1 | Integrator at the lower limit |

### Truth Table for lowerlimit > upperlimit

| Set | Condition | Y | Upper limit reached | Lower limit reached | Mode |
|---|---|---|---|---|---|
| any | lowerLimit < integration result <upperLimit | yn | 0 | 0 | Integrator limited by upper Limit |


set
(IN)
1
0
x
(IN)
0
upperLimit
(IN)
y
(OUT)
setValue
lowerLimit
(IN)
upperLimit
Reached
(OUT)
1
0
lowerLimit
Reached
(OUT)
1
0

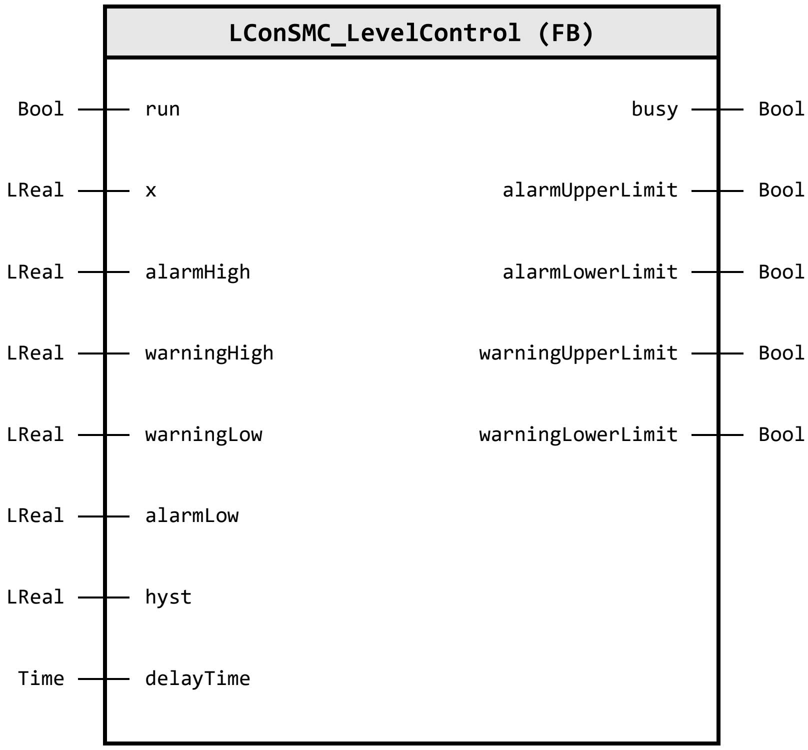
## 5.6.8 LConSMC_LevelControl (FB)

### Short description

Limit value monitor. The function block can be called in any cyclic task.  Function block ID: 0104

### Interface description

### Block interface




### Input parameter

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| run | Bool | FALSE | TRUE: Enable function block |
| x | LReal | 0.0 | Input value |
| alarmHigh | LReal | 100.0 | Alarm, upper limit value |
| warningHigh | LReal | 80.0 | Warning, upper limit value |
| warningLow | LReal | -80.0 | Warning, lower limit value |
| alarmLow | LReal | -100.0 | Alarm, lower limit value |
| hyst | LReal | 0.0 | Hysteresis (Absolute value is used) |
| delayTime | Time | T#3S | [ s ] Output delay time<br>delayTime = 0, signal delay is disabled |

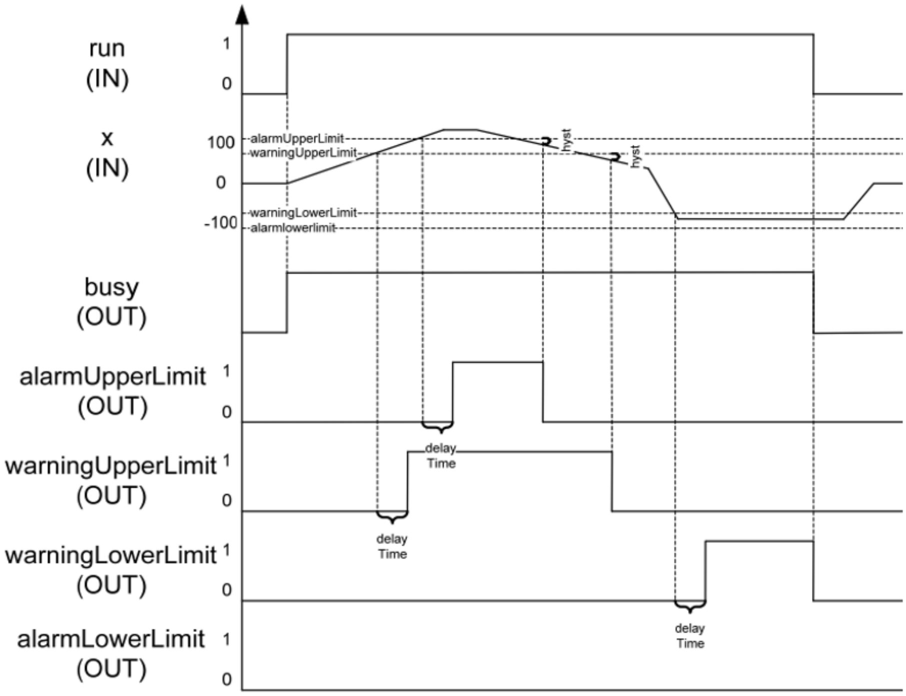
### Output parameter

| Identifier | Data type | Description |
|---|---|---|
| busy | Bool | TRUE: Function block is activated |
| alarmUpperLimit | Bool | TRUE: Upper alarm limit value exceeded |
| alarmLowerLimit | Bool | TRUE: Lower alarm limit value exceeded |
| warningUpper Limit | Bool | TRUE: Upper warning limit value exceeded |

| Identifier | Data type | Description |
|---|---|---|
| warningLower Limit | Bool | TRUE: Lower warning limit value exceeded |

**Functional description**

This function block monitors an input value (x) to identify whether it violates (exceeds, falls below) parameterized limit values as long as it is active with run = TRUE. If the block is enabled, then this can be identified using the busy output. When the warningHigh value is exceeded, output warningUpperLimit is set after the delayTime. The same applies when the alarmHigh value is exceeded with the associated output alarmUpperLimit. If the input value drops below the limit value warningHigh or alarmHigh minus the hysteresis value (hyst), the particular output is reset. The same applies for the lower limit monitoring with the limits warningLow and alarmLow as well as the associated outputs warningLowerLimit and alarmLowerLimit. However, in this case, the hysteresis (hyst) is added to the limit value. Warnings are not reset as a result of alarms.



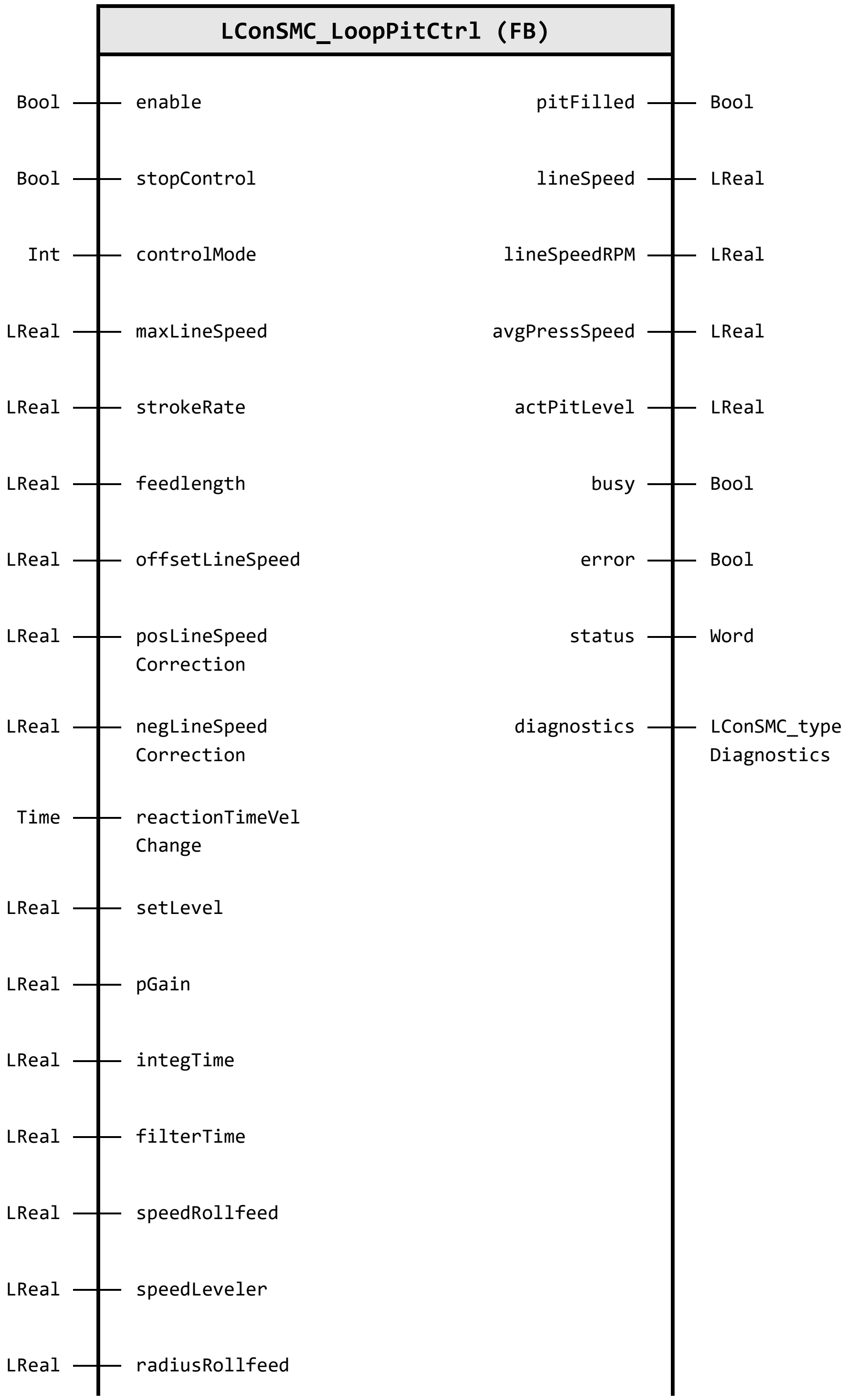
### 5.6.9 LConSMC_LoopPitCtrl (FB)

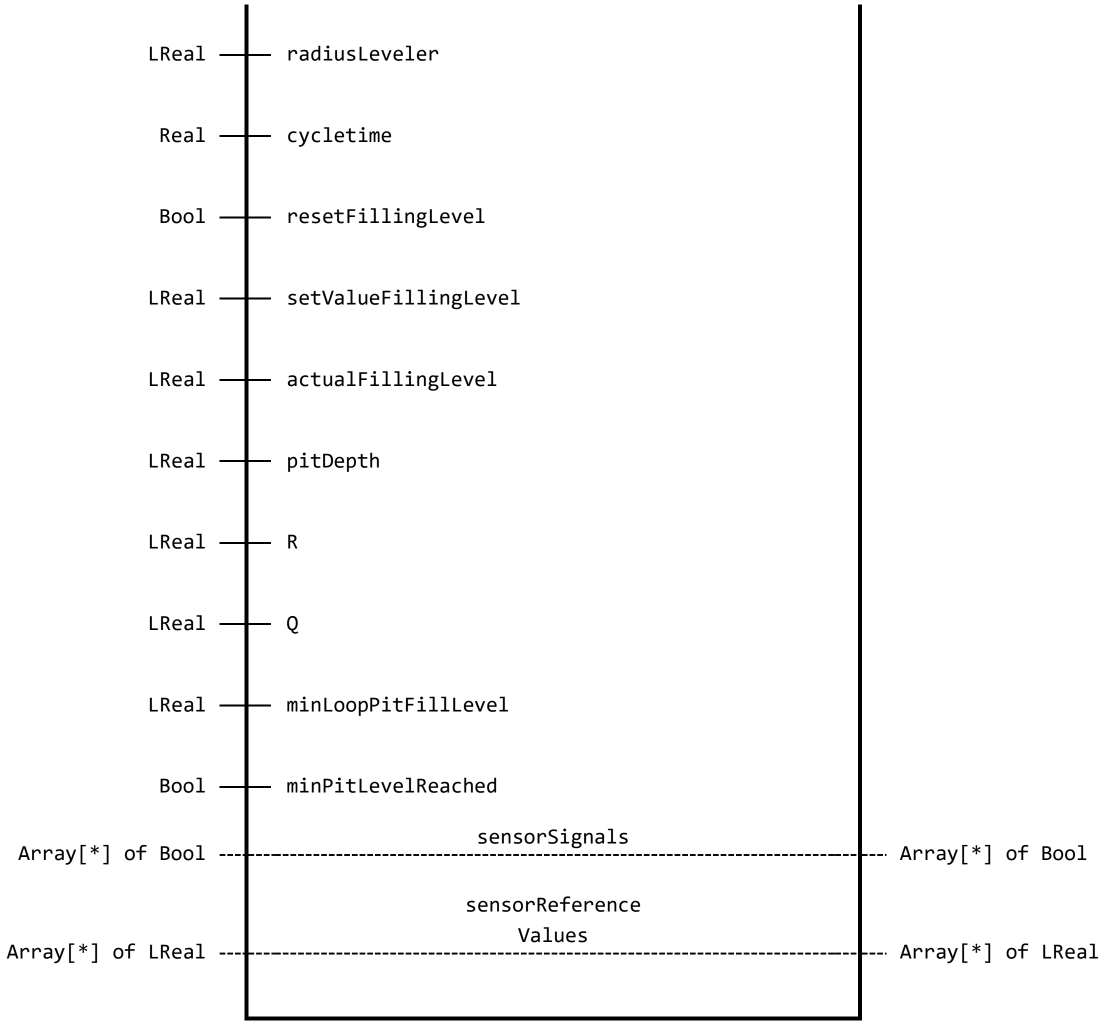
**Short description**

The function block controls the line speed of a strip feeding machine in order to keep the fill level of a loop pit within the specified range

**Interface description**

**Block interface**






### Input parameter

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| enable | Bool | FALSE | TRUE: Enable function block. With rising edge a validation of parameter is done.<br>FALSE: Disable the function block |
| stopControl | Bool | FALSE | TRUE: Controller are on hold and pit can be filled manually while fill level estimation stays active |
| controlMode | Int | 0 | 0: simple two-point-control with 2 binary signals (e.g. light barriers); 1: adaptive binary-control with min. 3 binary signals; 2: PI control with estimated fill level (Integral of velocities); 3: PI control with estimated fill level (Integral & reference values from binary sensors); 4: PI control with actual fill level (analog sensor); 5: LQC with estimated fill level (Integral); 6: LQC with estimated fill level (Integral & reference values from binary sensors); 7: LQC with actual fill level (analog sensor); 8: Control depending on press behavior (no sensors for fill level detection) |
| maxLineSpeed | LReal | 0.0 | [m/min] maximal allowed line speed |
| strokeRate | LReal | 0.0 | [spm] stroke rate of press in strokes per minute |
| feedlength | LReal | 0.0 | [mm] feed length of e.g. rollfeed in millimeter |
| offsetLineSpeed | LReal | 0.0 | [%] additional line speed if precontrol speed is too slow / fast (use only in mode 0 and 8) |
| posLineSpeed Correction | LReal | 0.0 | [%] Line Speed adaption if line speed is too low. (use only in mode 1) |

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| negLineSpeed Correction | LReal | 0.0 | [%] Line Speed adaption if line speed is too high. (use only in mode 1) |
| reactionTimeVel Change | Time | T#0s | [ms] reaction time for speed changes (use only in mode 1) |
| setLevel | LReal | 0.0 | [mm] filling level setpoint (used in mode 2-7) |
| pGain | LReal | 0.0 | proportional gain of PI controller (used in mode 2-4) |
| integTime | LReal | 0.0 | [ms] time constant for integrator of PI controller (used in mode 2-4) |
| filterTime | LReal | 0.0 | [ms] PT1 - Filter for actual value. A value of 0 disables the filter. (used in mode 2-4) |
| speedRollfeed | LReal | 0.0 | [1/min] actual speed of rollfeed (used in mode 2, 3, 5, 6) |
| speedLeveler | LReal | 0.0 | [1/min] actual speed of leveler (used in mode 2, 3, 5, 6) |
| radiusRollfeed | LReal | 0.0 | [mm] radius of rollfeed roll (used in mode 2, 3, 5, 6) |
| radiusLeveler | LReal | 0.0 | [mm] radius of leveler roll (used in mode 2, 3, 5, 6) |
| cycletime | Real | 0.0 | [ms] cycle time of OB in which this function is called |
| resetFilling Level | Bool | FALSE | reset filling level to value of setValueFillingLevel (used in mode 2, 3, 5, 6) |
| setValueFilling Level | LReal | 0.0 | [mm] Fill level value to which the calculated value is to be set when resetFillingLevel is TRUE<br>(used in mode 2, 3, 5, 6) |
| actualFilling Level | LReal | 0.0 | [mm] actual pit filling level<br>(used in mode 4 and 7) |
| pitDepth | LReal | 0.0 | [mm] Depth of pit (used in mode 2, 3, 5, 6) |
| R | LReal | 0.0 | factor to set energyefficient bahaviour of Linear-quadratic regulator (ratio between Q and R is decisive) (used in mode 5, 6, 7) |
| Q | LReal | 0.0 | factor to set dynamic behaviour of Linear-quadratic regulator (ratio between Q and R is decisive) (used in mode 5, 6, 7) |
| minLoopPitFill Level | LReal | 0.0 | [mm] minimal fill level of pit to allow start of press |
| minPitLevel Reached | Bool | FALSE | TRUE: pit is filled, press can start (output "pitFilled" is set to true); used in mode 8 |

### Output parameter

| Identifier | Data type | Description |
|---|---|---|
| pitFilled | Bool | TRUE: pit is filled and press can be started |
| lineSpeed | LReal | [m/min] line speed in meter per minute |
| lineSpeedRPM | LReal | [1/min] line speed in rounds per minute |
| avgPressSpeed | LReal | [m/min] average press speed depending on stroke rate and feed length |
| actPitLevel | LReal | [mm] actual pit level (depending on chosen mode it is estimated or sensor data) |
| busy | Bool | TRUE: function block is busy |
| error | Bool | TRUE: An error occurred during the execution of the FB |
| status | Word | 16#0000 - 16#7FFF: Status of the FB, 16#8000 - 16#FFFF: Error identification |
| diagnostics | LConSMC_type Diagnostics | Diagnostics information of FB |

### In/Out parameter

| Identifier | Data type | Description |
|---|---|---|
| sensorSignals | Array[*] of Bool | Signals of binary sensors. Lower bound: pit is empty, Upper bound: pit is full (TRUE = no material detected) (used in mode 1, 3, 6) |
| sensorReference Values | Array[*] of LReal | Reference values of binary sensors describing sensor position as filling level. Lower bound: pit is empty, Upper bound: pit is full (used in mode 3 and 6) |

### Status & Error codes

| Code / Value | Identifier / Description |
|---|---|
| 16#0000 | STATUS_FINISHED_NO_ERROR<br>NO error - execution finished without errors |
| 16#8000 | ERR_UNDEFINED<br>Error: due to an undefined case |
| 16#8002 | ERR_IN_PRECONTROL_SR<br>Error: negative strokerate not allowed |
| 16#8003 | ERR_IN_PRECONTROL_FL<br>Error: negative feedlength not allowed |
| 16#8004 | ERR_IN_PRECONTROL_LS<br>Error: calculation of average line speed exceeded min or max line speed. Check the speed limit settings. |
| 16#8005 | ERR_IN_ADAPTBINCTRL<br>Error: This control mode needs more than two sensor signals. |
| 16#8006 | ERR_IN_LQR<br>Error: LQ-Control Parameter Q and R are not allowed to be zero or negative |
| 16#8201 | ERR_PARAM_CONTROLMODE<br>Error: undefined control mode chosen |
| 16#8202 | ERR_PARAM_PRECONTROL_SR<br>Error: invalid value of strokerate |
| 16#8203 | ERR_PARAM_PRECONTROL_FL<br>Error: invalid value of feedlength |
| 16#8204 | ERR_PARAM_PRECONTROL_LS<br>Error: maximal allowed line speed is negative |
| 16#8205 | ERR_PARAM_RADIUS<br>Error: radius of leveler or rollfeed is zero or negative. Please change the value and restart the function block. |
| 16#8401 | ERR_BINARY_SENSOR_SIGNAL<br>Error: impossible sensor state. Check signal connection and sensor for defects. |
| 16#8402 | ERR_NO_SENSORSIGNAL<br>Error: No signals connected. Number of Sensors is 0 or below. |
| 16#8403 | ERR_SENSORSIGNAL<br>Error: Lower bound auf SensorSignal-Array is not 0. |
| 16#8404 | ERR_MAX_SENSOR_NBR_REACH<br>Error: Number of used sensor signals is exceeding parameterized maximal number of sensors (MAX_SENSORS) |
| 16#8601 | ERR_PID<br>Error in PID-Controller. More information in diagnostics.subfunctionStatus |
| 16#8602 | ERR_INTEG<br>Error in Integrator for pit fill level determination. More information in diagnostics.subfunctionStatus |
| 16#8603 | ERR_DIFF<br>Error in Differentiator for average pit fill level determination. More information in diagnostics.subfunctionStatus |

**Functional description**

This function block controls the line speed of a strip feeding machine in order to keep the fill level of a loop pit within the specified range. The function block is activated by a TRUE signal at the *enable* input. By setting *stopControl* true, you can switch off the controllers without deactivating the function block. This is useful if you want to change the filling level manually and still receive an estimate of the filling level. The *controlMode* input can be used to switch between different control modes. It is only possible to change the control mode if the function block is deactivated and reactivated after the change. The function block has to be called in a cyclic OB (OB3x or OB6x) if control mode 2 - 7 are used. The input *maxLineSpeed* limits the *lineSpeed* output to the applied value so that excessive speeds are prevented.
The output *pitFilled* indicates whether the loop pit is sufficiently full to pull material out of it. This can be used to prevent the material from being pulled out if there is not enough. When the pit is full enough is determined either by the first light barrier *sensorSignals[0]* or by *minLoopPitFillLevel*, depending on the mode.

## 5.7 Description of the different control modes

**Control mode 0: Simple two-point controller**

This mode uses two binary signals to determine the fill level of the loop pit. If the loop pit is empty (*sensorSignal[0]* = TRUE and *sensorSignal[Last]* = TRUE), the speed is determined based on the current stroke rate and feed length of the press. This determined average speed of the press can be adjusted using the *offsetLineSpeed* input. If the loop pit is full (*sensorSignal[0]* = FALSE and *sensorSignal[Last]* = FALSE), the line speed is set to 0.



**Control mode 1: Adaptive binary control**

Mode 1 can be used with several binary signals. These must all be connected to the *sensorSignals* input as an Array of Bool. The first element of the array describes the behavior of the topmost light barrier (pit is empty) and the last element describes the lowest light barrier (pit is full). A TRUE signal means that the light barrier is not interrupted. So if *sensorSignal[0]* = TRUE, the pit is empty.

Depending on which light barrier detects a falling or rising edge, the line speed is adjusted according to a specific algorithm. The inputs *posLineSpeedCorrection* and *negLineSpeedCorrection* are used to specify the desired speed correction in relation to the average press speed as a percentage. The *reactionTimeVelChange* input is used to set the reaction time of the speed adjustments in order to ignore flickering of the light barrier signals and avoid frequent speed changes.

**Control mode 2: PI control with estimated fill level**

Mode 2 uses a PI controller whose parameters can be set at the *pGain*, *integTime* and *filterTime* inputs.
The target fill level is specified at the *setLevel* input.
The current fill level is determined by integrating and subtracting the speeds of the leveler (pushing into the pit) and the rollfeed (pulling out of the pit). To do this, the current speeds must be linked to the respective inputs *speedLeveler* and *speedRollfeed* and the radii of the rollers must be specified (*radiusLeveler*, *radiusRollfeed*). For the integration to work properly, the cycle time of the function block in which the function block is called must be specified at the *cycleTime* input. The maximum fill level must be specified at the *pitDepth* input.

If the calculated filling level differs from the actual filling level, the current filling level can be specified at the *setValueFillingLevel* input.
For this value to be accepted, both *enable* and *stopControl* must be true so that the controllers do not change the filling level during parameterization. If this is the case, the value *setValueFillingLevel* is accepted as soon as *resetFillingLevel* is true. Once this has been done, *resetFillingLevel* must be set to FALSE again so that the calculation of the filling level is continued.

Do not forget to set the current fill level accordingly if the system has been moved while this function block was not active! This could lead to considerable damage, as an incorrect actual value is assumed!

**Control mode 3: PI control with estimated fill level and reference signals**

Mode 3 supplements the level determination of mode 2 by referencing the calculated value with sensor signals. If an edge is detected on a sensor signal, the determined level is set to the corresponding reference value and the calculation is continued from there. For this function, the binary signals at the *sensorSignals* input must be linked as for mode 1. In addition, an array with the corresponding fill levels must be linked at the *sensorReferenceValues* input.

**Control mode 4: PI control with actual fill level**

Control mode 4 also uses a PI controller (see mode 2), but does not determine the fill level by calculation, but uses an analog sensor signal. To do this, the currently measured filling level must be specified in millimeters at the *actualFillingLevel* input.

**Control mode 5: Linear-quadratic regulator with estimated fill level**

The linear-quadratic regulator uses the optimal control method. Only weight factors are specified, which determine how dynamically or energy-efficiently the regulator should act. This is used to minimize a cost function. LQ regulators have an inherent robustness with guaranteed gain and phase margin. In this mode, the equations of this method have been adapted to the application case of loop pit control. The weighting factors can be set at the inputs *Q* and *R*. The value of *Q* describes the speed at which the control deviation is to be driven towards zero. *R* determines the behavior of the control variable change. The larger *R* is, the smaller and slower the changes in the control variable. The ratio between *Q* and *R* therefore describes the controller behavior.
In this mode, the current fill level is determined in the same way as in control mode 2.
The target fill level is specified at the *setLevel* input.

Do not forget to set the current fill level accordingly if the system has been moved while this function block was not active! This could lead to considerable damage, as an incorrect actual value is assumed!

**Control mode 6: Linear-quadratic regulator with estimated fill level and reference signals**

This control mode uses the Linear-quadratic regulator (see mode 5) with the same fill level estimation as in control mode 3.

**Control mode 7: Linear-quadratic regulator with actual fill level**

As in control mode 4 the LQ Regulator can be used with an actual fill level, too.

**Control mode 8: Open loop control without use of sensors for pit level determination**

This mode calculates the line speed depending on the press stroke rate and feed length of the press. The determined line speed can be adjusted using the *offsetLineSpeed* input. When the pit is filled with enough material to enable the following process, the user can set *minPitLevelReached* true to enable the output pitFilled.

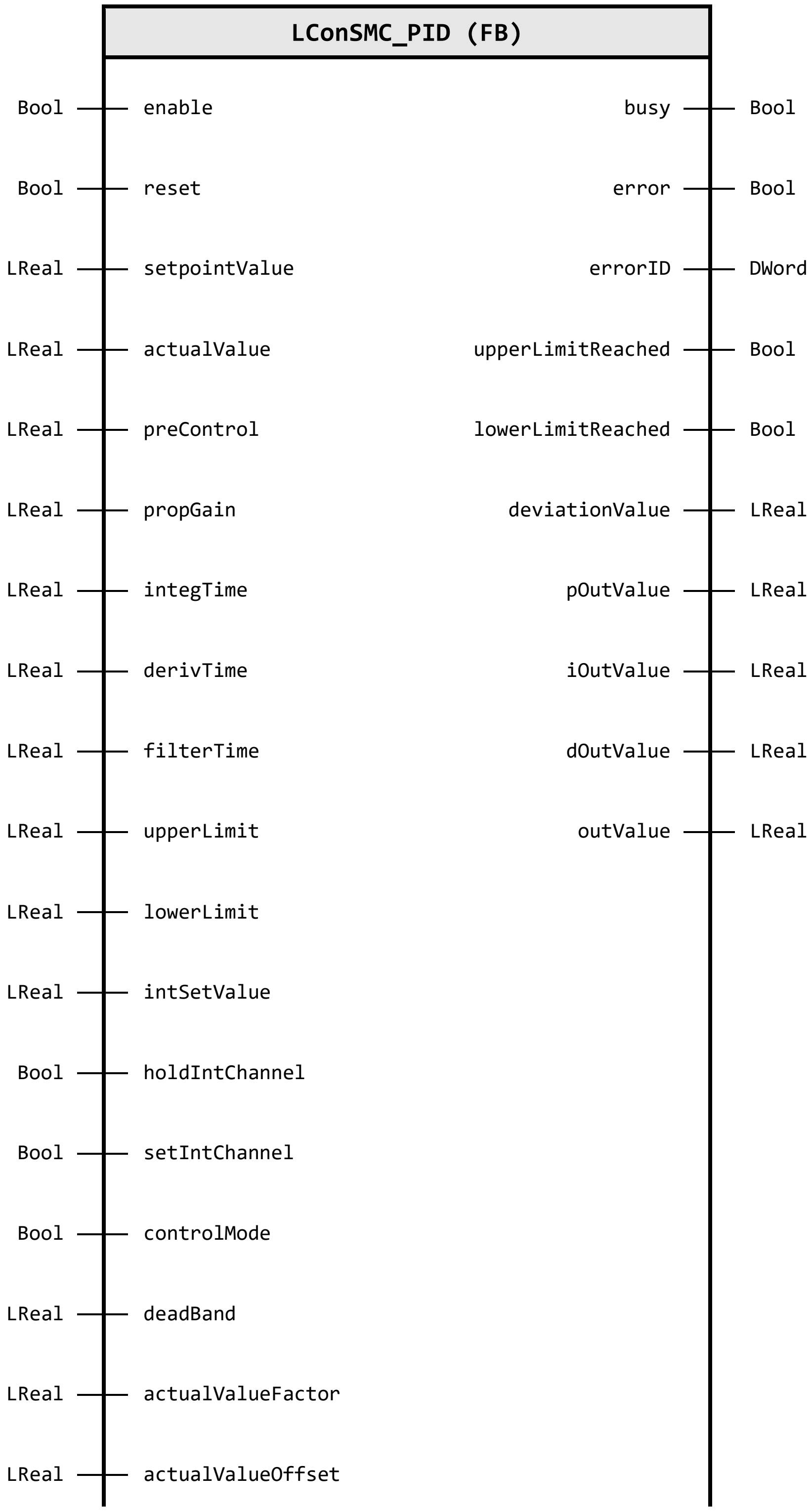
### 5.7.1 LConSMC_PID (FB)

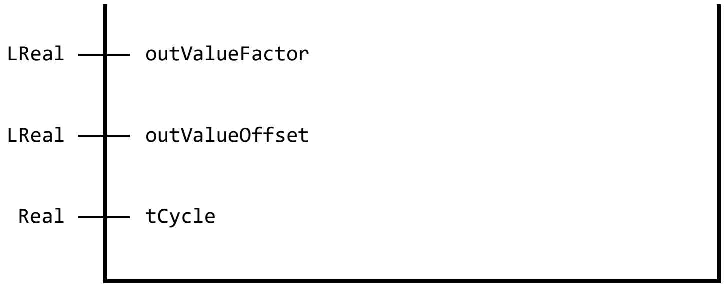
**Short description**

PID controller with actual value smoothing and optional D component in the actual value branch, the function block has to be called in a cyclic interrupt, e.g. OB30. Function block ID: 1105

**Interface description**

**Block interface**






### Input parameter

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| enable | Bool | FALSE | TRUE: Enable function block |
| reset | Bool | FALSE | TRUE: Reset function block |
| setpointValue | LReal | 0.0 | [-] Setpoint |
| actualValue | LReal | 0.0 | [-] Actual value |
| preControl | LReal | 0.0 | Controler pre control value |
| propGain | LReal | 2.0 | Proportional gain. A value of 0 disables the P-part |
| integTime | LReal | 999.0 | [ms] Time constant for integrator. A value of 0 disables the I-part |
| derivTime | LReal | 0.0 | [ms] Time constant for differentiator. A value of 0 disables the D-part |
| filterTime | LReal | 10.0 | [ms] PT1- Filter for actual value. A value of 0 disables the filter |
| upperLimit | LReal | 100.0 | Upper limit of the controller output |
| lowerLimit | LReal | 0.0 | Lower limit of the controller output |
| intSetValue | LReal | 0.0 | Integrator set value (if "setIntChannel" = TRUE) |
| holdIntChannel | Bool | FALSE | TRUE: Integrator is hold |
| setIntChannel | Bool | FALSE | TRUE: The output of the integrator is set to "intSetValue" |
| controlMode | Bool | FALSE | TRUE: D component active in actual value branch<br>FALSE: D component in controller deviation |
| deadBand | LReal | 0.0 | Value of the dead band (symmetrical). A value of 0 disables the deadband |
| actualValue Factor | LReal | 1.0 | Multiplicative scaling of the actual value |
| actualValue Offset | LReal | 0.0 | Additive scaling of the actual value |
| outValueFactor | LReal | 1.0 | Multiplicative scaling of the controller output |
| outValueOffset | LReal | 0.0 | Additive scaling of the controller output |
| tCycle | Real | 0.0 | [ms] Cycle time the function block is called |

### Output parameter

| Identifier | Data type | Description |
|---|---|---|
| busy | Bool | TRUE: Function block is active |
| error | Bool | TRUE: An error occured |
| errorID | DWord | Detailed error description |
| upperLimit Reached | Bool | TRUE: Upper limit reached |
| lowerLimit Reached | Bool | TRUE: Lower limit reached |
| deviationValue | LReal | Controller deviation after the deadband filter |

| Identifier | Data type | Description |
|---|---|---|
| pOutValue | LReal | P component output before limitation |
| iOutValue | LReal | I component output before limitation |
| dOutValue | LReal | D component output before limitation |
| outValue | LReal | Controller output after limitation and scaling |

**Functional description**

The function block is used to implement a universal PID controller that can be switched over to the P controller or I controller modes. The D component can either be calculated in the actual value branch or in the system deviation. The integrator can be set using the inputs intSetValue (setting value) and setIntChannel. In addition, the instantaneous value of the integrator can be held using holdIntChannel (P controller). Internally, the I component is limited, based on the controller output limits (“anti-windup”). The integrator is set to zero at switch-on. All of the block parameters can be changed in operation. The scaled and smoothed actual value (actualValue) is subtracted from the setpoint (setpointValue) corresponding to the following equation:

$$deviationValue = [(actualValue \cdot actualValueFactor) + actualValueOffset] \cdot \left(1 - e^{-\frac{t}{T}}\right) - setpointValue$$

The result is the system deviationValue (after the deadband) and is then multiplied with the proportional coefficient propGain. The result is transferred to the integrator – and optionally to the D component. The settable integral time integTime as well as the time constant of the D component derivTime (optional) define the controller behavior.

- P component

$$pOutValue_n = propGain * deviationValue_n$$

- I component

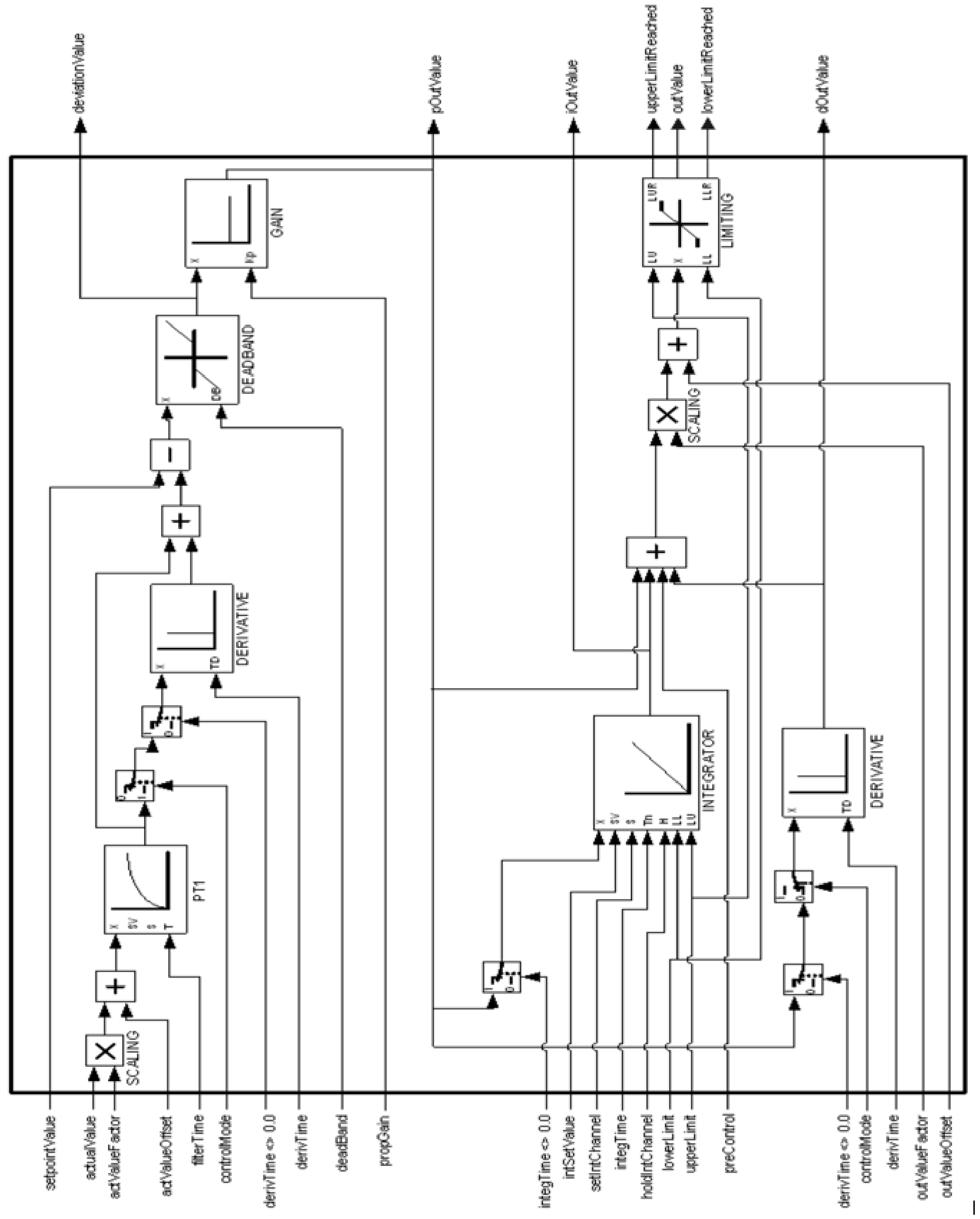
$$iOutValue_n = iOutValue_{n-1} + \left(\frac{pOutValue_n + pOutValue_{n-1}}{2}\right) \cdot \frac{cycleTime}{integTime}$$

The integrator includes “anti-windup” functionality, i.e. if the controller output is limited, the integrator cannot continue to integrate in the same direction. The inputs setIntChannel and holdIntchannel can be used to set the integrator (intSetValue) – or manually hold it. The integrator is set to zero using reset or with a rising edge for enable.

- D component (if activated)

$$dOutValue_n = (pOutValue_n + pOutValue_{n-1}) \cdot \frac{derivTime}{cycleTime}$$

The controller output is first scaled and is then limited.


deviationValue
pOutValue
iOutValue
upperLimitReached
outValue
lowerLimitReached
dOutValue
GAIN
DEADBAND
DERIVATIVE
PT1
SCALING
INTEGRATOR
LIMITING
SCALING
DERIVATIVE
setpointValue
actualValue
actValueFactor
actValueOffset
filterTime
controlMode
derivTime <> 0.0
derivTime
deadBand
propGain
integTime <> 0.0
intSetValue
setIntChannel
integTime
holdIntChannel
lowerLimit
upperLimit
preControl
derivTime <> 0.0
controlMode
derivTime
outValueFactor
outValueOffset

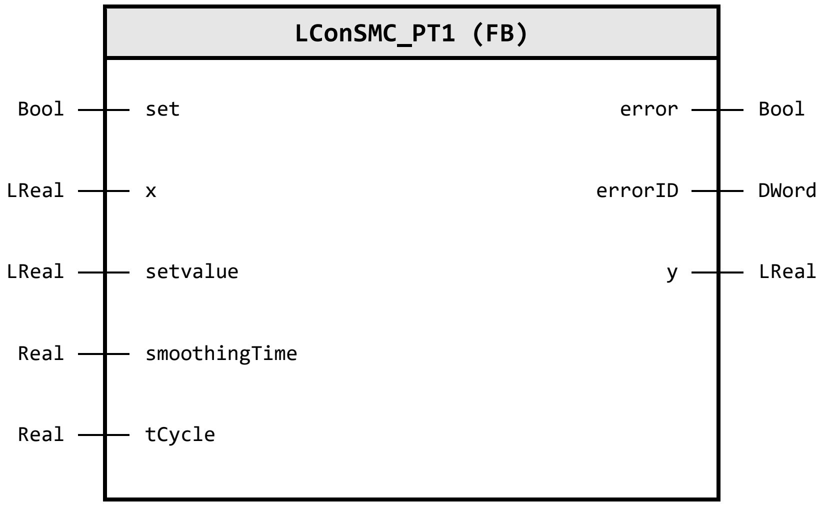
### 5.7.2 LConSMC_PT1 (FB)

#### Short description

Low-pass filter function block, the function block has to be called in a cyclic interrupt, e.g. OB30.
Function block ID: 1101

#### Interface description

#### Block interface






#### Input parameter

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| set | Bool | FALSE | TRUE: Set output to the value of "setValue" |
| x | LReal | 0.0 | Input Value |
| setvalue | LReal | 0.0 | Set value (if "set" = TRUE) |
| smoothingTime | Real | 100.0 | [ms] Smoothing time constant [tCycle - 5000] |
| tCycle | Real | 0.0 | [ms] Cycle time the function block is called |

#### Output parameter

| Identifier | Data type | Description |
|---|---|---|
| error | Bool | TRUE: An error occured |
| errorID | DWord | Detailed error description |
| y | LReal | Filtered output value |

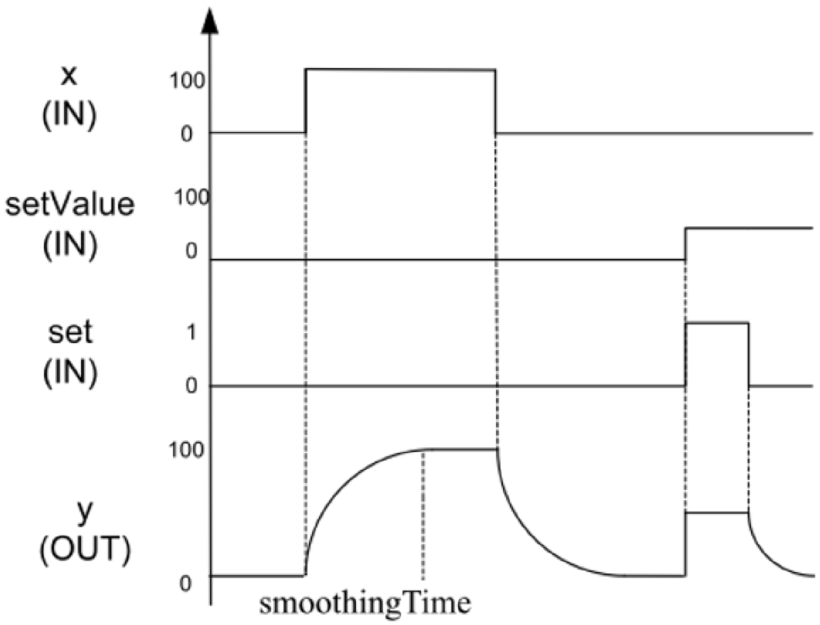
#### Functional description

This input value x is filtered using the time constant smoothingTime and output at the output y. To precisely calculate the internal steps, the sampling time in which the block is processed is internally determined. Further, it is possible to permanently set the output to the value of the input (setValue) by setting the input (set). In order to generate a hold, output y should be connected to the input setValue. This means that when setting set to TRUE, the output value is maintained until set is again set to FALSE. If the time constant is set to zero, the filter is deactivated and the output value is equal to the input value. 63% of the input value is reached at the output after the time constant. 95% of the input value is reached at the output after three times of the time constant. For an appropriately high smoothingTime / cycleTime (smoothingTime / cycleTime > 10) the transfer function corresponds to the following characteristic:

$$y(t) = x \cdot \left(1 - e^{-\frac{t}{smoothingTime}}\right) \qquad with \quad t = n \cdot cycleTime$$

The discrete values are calculated using the following algorithm:

$$y_n = y_{n-1} + \frac{cycleTime}{smoothingTime} \cdot (x_n - y_{n-1})$$



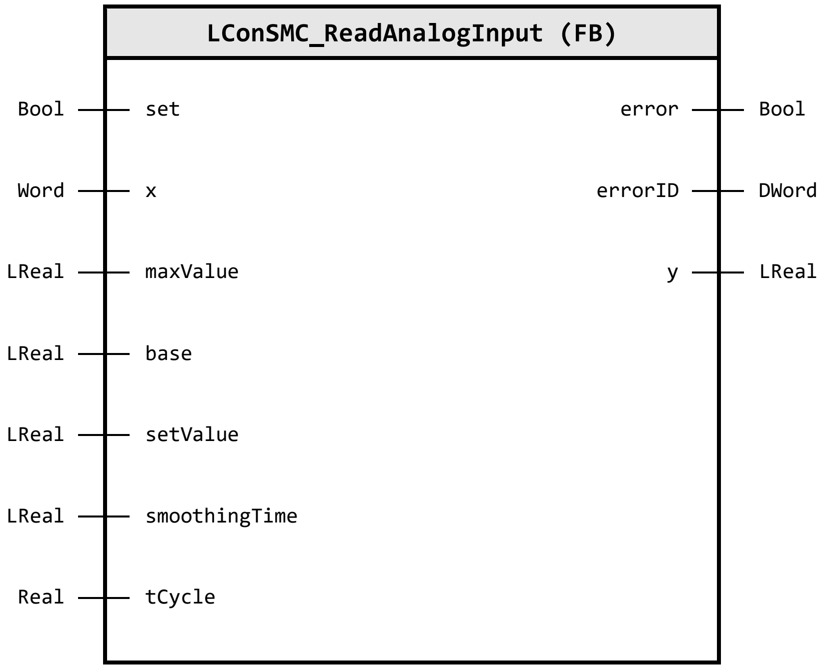
### 5.7.3 LConSMC_ReadAnalogInput (FB)

#### Short description

Analog input evaluation with PT1 filter, must be called in cyclic interrupt task i.e. OB30.
Function block ID: 1109

#### Interface description

#### Block interface






#### Input parameter

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| set | Bool | FALSE | TRUE: Set output to the value of "setValue" |
| x | Word | 16#0000 | Input Value |
| maxValue | LReal | 0.0 | Used as reference value for conversion<br>y = x/base * maxValue<br>The output is limited to +- 2*maxValue |
| base | LReal | 16384.0 | Scaling value for conversion<br>y=x/base*maxValue |
| setValue | LReal | 0.0 | Set value (if "set" = TRUE) |
| smoothingTime | LReal | 100.0 | [ms] Smoothing time constant [tCycle - 5000] |
| tCycle | Real | 0.0 | [ms] Cycle time the function block is called |

#### Output parameter

| Identifier | Data type | Description |
|---|---|---|
| error | Bool | TRUE: An error occured |
| errorID | DWord | Detailed error description |
| y | LReal | Output value |

**Functional description**

This function converts an analog input signal in WORD format (x) into a REAL value using a normalization value (base) and the maximum reference value (maxValue). The output value is limited between the values -2 * maxValue - 2 * maxValue. The WORD value will be converted to INT value and afterwards to REAL value.

$$y = \frac{x}{base} \cdot maxValue$$

Optionally the analog value can be filtered with a settable PT1-filter using the filter time smoothingTime. If the input parameter smoothingTime = 0, the filter is disabled. The internal limit for the filter time is 5000 ms.

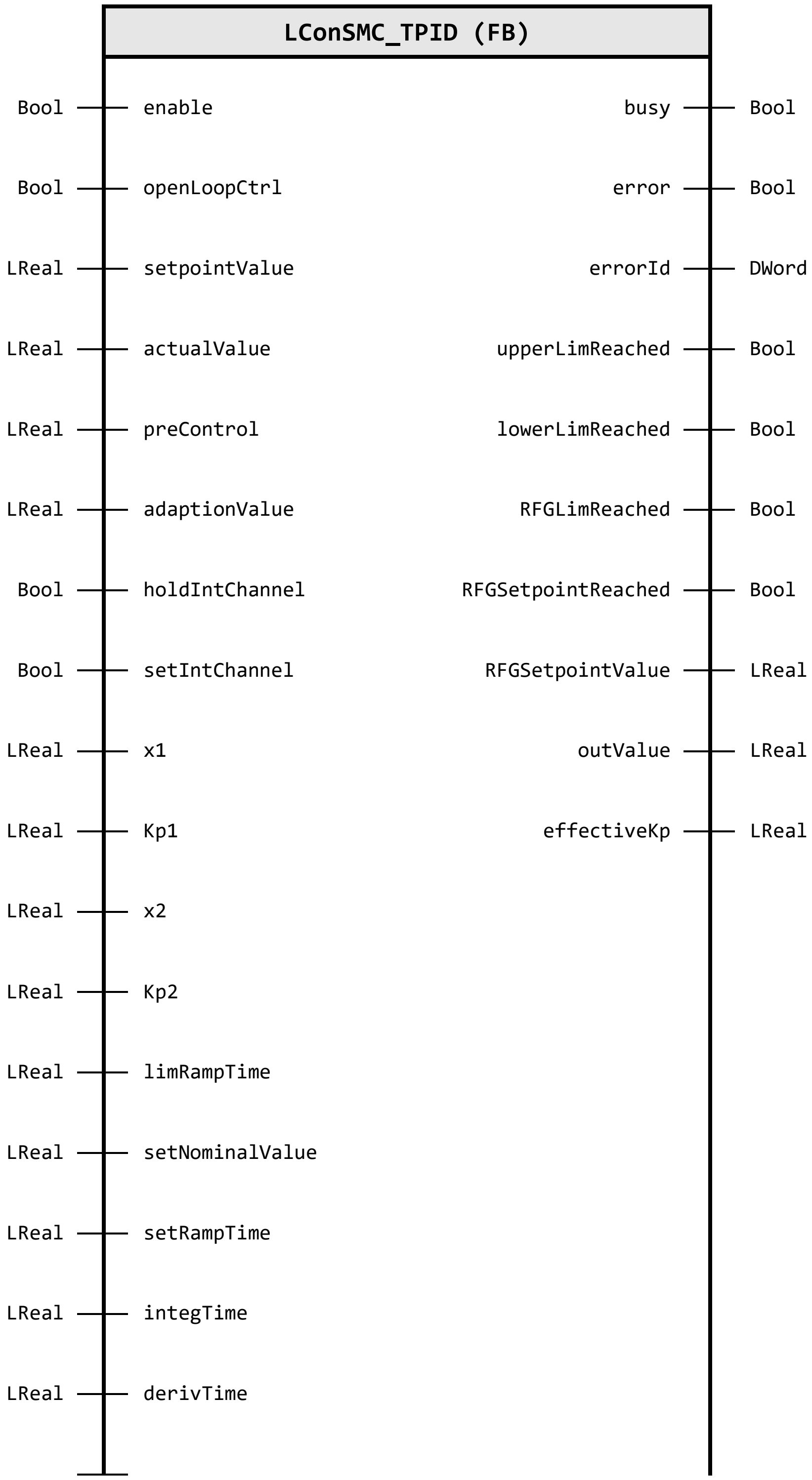
## 5.7.4 LConSMC_TPID (FB)

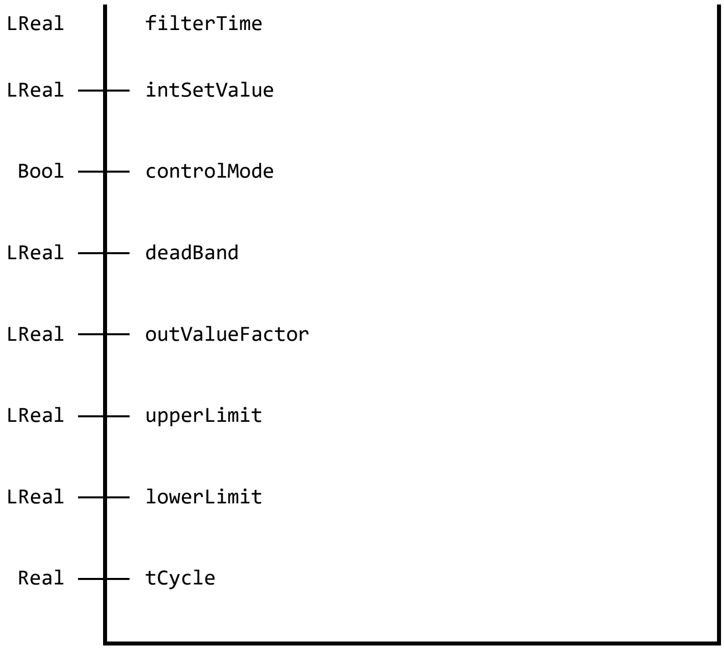
### Short description

Technology controller based on a PID controller for use as tension or dancer roll position controller in closed-loop winder or material web controls The function block has to be called in a cyclic interrupt, e.g. OB30.  Function block ID: 1108

### Interface description

### Block interface






### Input parameter

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| enable | Bool | FALSE | TRUE: Enable function block |
| openLoopCtrl | Bool | FALSE | TRUE: PID-controller is deactivated. Ramp function generator is used to ramp the setpoint (see output "RFGSetpointValue", output "outValue" = 0.0).<br>FALSE: PID controller and ramp function generator is active |
| setpointValue | LReal | 0.0 | [-] Setpoint |
| actualValue | LReal | 0.0 | [-] Actual value |
| preControl | LReal | 0.0 | Controler pre control value |
| adaptionValue | LReal | 0.0 | Adaption value for P-Gain adaption of the controller |
| holdIntChannel | Bool | FALSE | TRUE: The output of the integrator is hold<br>FALSE: The output is integrated |
| setIntChannel | Bool | FALSE | TRUE: The output of the integrator is set to "intSetValue" |
| x1 | LReal | 0.0 | Breakpoint 1 x-axis for P-adaption (if x1=x2 the adaption is deactivated) |
| Kp1 | LReal | 1.0 | Breakpoint 1 P-gain for P-adaption (if x1=x2 the adaption is deactivated) |
| x2 | LReal | 0.0 | Breakpoint 2 x-axis for P-adaption (if x1=x2 the adaption is deactivated) |
| Kp2 | LReal | 1.0 | Breakpoint 2 P-gain for P-adaption (if x1=x2 the adaption is deactivated) |
| limRampTime | LReal | 1000.0 | [ms] Ramp time for output value limitation |
| setNominalValue | LReal | 100.0 | Reference value for ramp time of setpoint channel (Ramp from 0 to "setNominalValue" in "setRampTime") |
| setRampTime | LReal | 1000.0 | [ms] Ramp time for setpoint ramp generator |
| integTime | LReal | 2000.0 | [ms] Time constant for integrator. A value of 0 disables the I-part |
| derivTime | LReal | 0.0 | [ms] Time constant for differentiator. A value of 0 disables the D-part |
| filterTime | LReal | 10.0 | [ms] PT1- Filter for actual value. A value of 0 disables the filter |

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| intSetValue | LReal | 0.0 | Integrator set value (if "setIntChannel" = TRUE) |
| controlMode | Bool | FALSE | TRUE: D component active in actual value branch<br>FALSE: D component in controller deviation |
| deadBand | LReal | 0.0 | Value of the dead band (symmetrical). A value of 0 disables the deadband |
| outValueFactor | LReal | 1.0 | Multiplicative scaling of the controller output |
| upperLimit | LReal | 20.0 | Upper limit of the controller output |
| lowerLimit | LReal | -20.0 | Lower limit of the controller output |
| tCycle | Real | 0.0 | [ms] Cycle time the function block is called |

### Output parameter

| Identifier | Data type | Description |
|---|---|---|
| busy | Bool | TRUE: Function block is active |
| error | Bool | TRUE: An error occured |
| errorId | DWord | Detailed error description |
| upperLimReached | Bool | TRUE: Upper limit reached |
| lowerLimReached | Bool | TRUE: Lower limit reached |
| RFGLimReached | Bool | TRUE: Setpoint is limited internally |
| RFGSetpoint Reached | Bool | TRUE: Ramped internal setpoint has reached the setpoint |
| RFGSetpointValue | LReal | Ramped internal setpoint |
| outValue | LReal | Controller output after limitation and scaling. Only relevant if input "openLoopCtrl" = FALSE. |
| effectiveKp | LReal | Effective p-gain if P-gain adaption is active |



### Functional description

The block is enabled using the enable input; therefore, the setpoint ramp-function generator and the limiting ramp-function generator are started with their respective ramp-up times and the PID controller is activated. In addition, for a rising edge, the setpoint ramp-function generator is set to the actual value that is present (actualValue). This guarantees a jerk-free and smooth transition into the controlled mode. When the edge at the enable input falls, the limiting ramp-function generator is ramped back to zero and therefore the controller output is limited. This results in a bumpless controller shutdown. As soon as the outValue output is zero and enable = FALSE, the block is deactivated and the output busy is set to FALSE. The setpoint is internally ramped to the actual value during the switch-off phase. If input openLoopCtrl = TRUE, then the block operates in the open control circuit mode. As a consequence, the setpoint channel remains active and the controller or the output limits remain inactive. The output busy stays TRUE as long as the setpoint ramp output is not zero. The input actualValue is 0.0.

### Setpoint ramping

The setpoint ramp-function generator is used to avoid setpoint steps in the setpoint channel. The setpoint ramp-function generator is configured using the parameters setRampTime and setNominalValue, whereby setNominalValue defines the reference value for the ramp-up or ramp-down time. The function is always active, if enable = TRUE. If openLoopCtrl = FALSE, enable = FALSE and busy = TRUE (switch-off phase), then the setpoint is internally set to the actual value. The ramp times are internally limited to the cycle time.

### Controller functionality

The controller setpoint is provided via the input setValue, the actual value via the actValue input. The setpoint channel includes a settable setpoint ramp-function generator, via which the setpoint is routed; the effective setpoint can be monitored at output RFGSetpointValue. The control algorithm used corresponds to that described in function block LConSMC_PID.

**Kp adaption**

A linear adaptation is carried out as Kp adaptation. The interpolation line is defined by the value (x1, Kp1) and (x2, Kp2):

- adaptionValue < x1 or x1 = x2

$$effectiveKp = Kp1$$

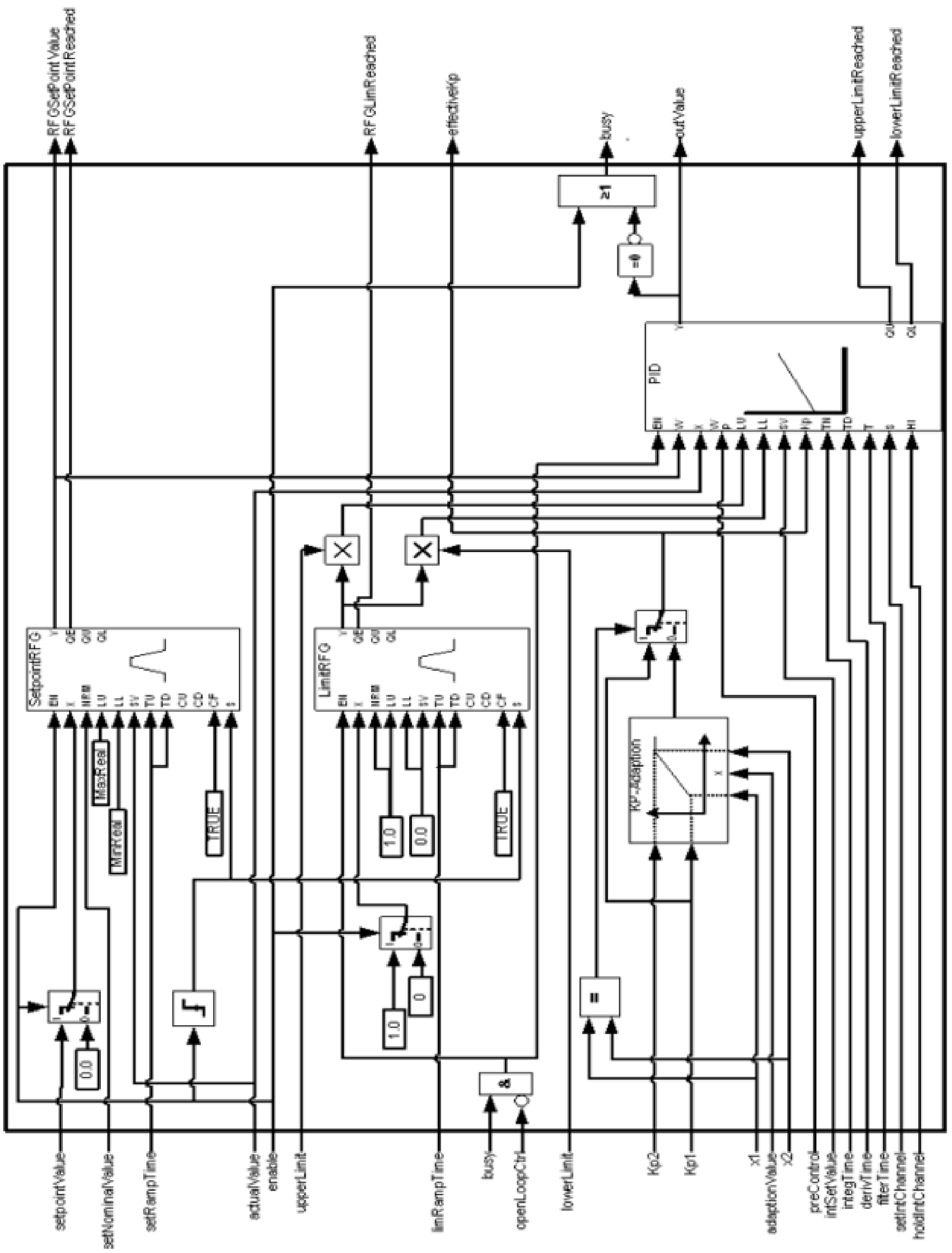
- x1 <= adaptionValue <= x2

$$effectiveKp = Kp1 + \frac{(adaptionValue - x1) \cdot (Kp2 - Kp2)}{x2 - x1}$$

- adaptionValue > x2

$$effectiveKp = Kp2$$

The value Kp1 is used as effective Kp if the Kp adaptation is deactivated (x1=x2).



# 5.8 Additional functions / 03_Drive related

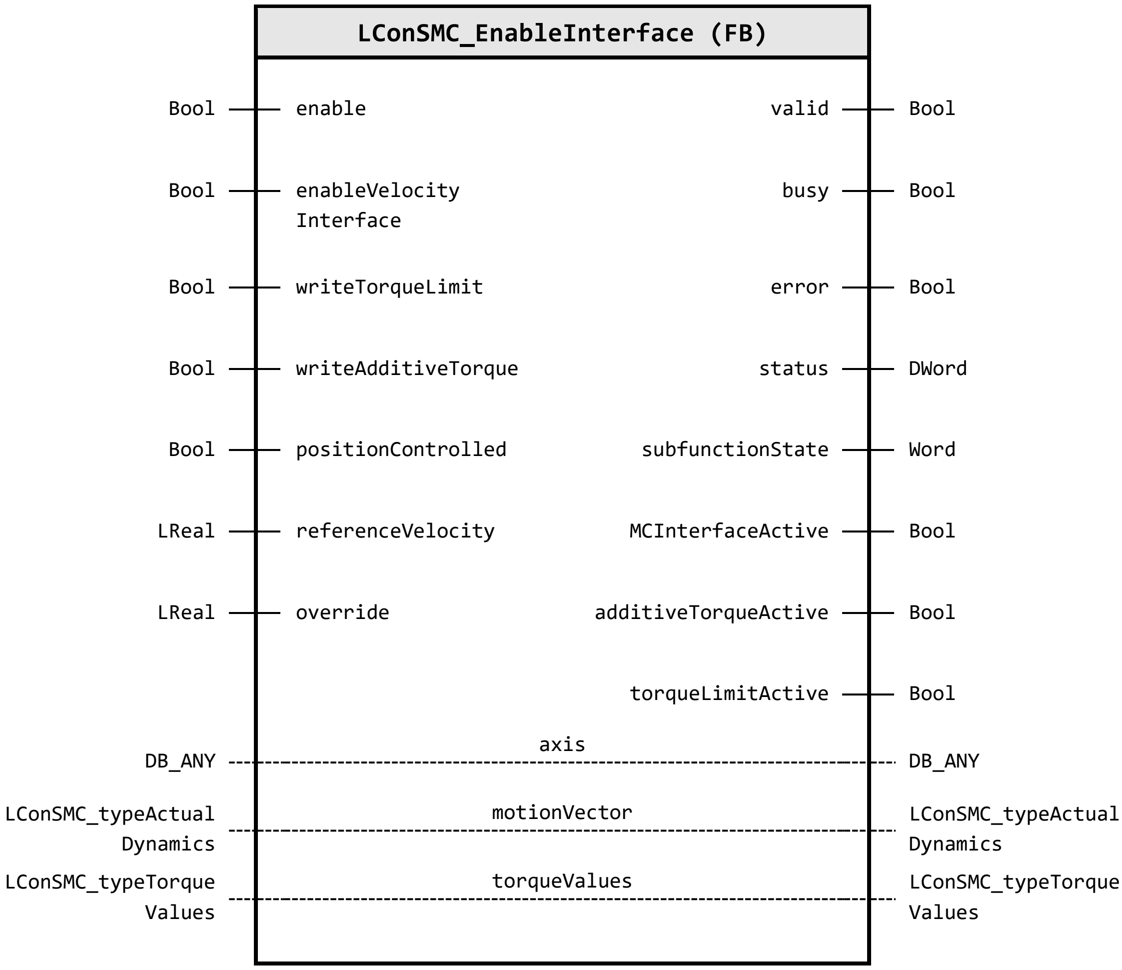
## 5.8.1 LConSMC_EnableInterface (FB)

### Short description

Enable motion and torque interface

### Interface description

### Block interface




### Input parameter

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| enable | Bool | FALSE | TRUE: Enable functionality of FB |
| enableVelocity Interface | Bool | FALSE | TRUE: Enable velocity setpoints<br>Changes in the "referenceVelocity" or the dynamics in "motionVector" are detected automatically |
| writeTorqueLimit | Bool | FALSE | TRUE: Enable torque limits |
| writeAdditive Torque | Bool | FALSE | TRUE: Enable additive torque |
| position Controlled | Bool | FALSE | TRUE: Enable moveVelocity position controlled<br>FALSE: Enable moveVelocity speed controlled |
| reference Velocity | LReal | 0.0 | Reference velocity for input "override" = 100% |

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| override | LReal | 0.0 | [%] Velocity override<br>override*referenceVelocity = setspeed (0-200%) |

**Output parameter**

| Identifier | Data type | Description |
|---|---|---|
| valid | Bool | TRUE: Valid set of output values available at the FB |
| busy | Bool | TRUE: FB is not finished and new output values can be expected |
| error | Bool | TRUE: An error occurred during the execution of the FB |
| status | DWord | 16#0000 - 16#7FFF: Status of the FB, 16#8000 - 16#FFFF: Error identification |
| subfunctionState | Word | ErrorIDs of motion control commands (Further diagnostic) |
| MCInterface Active | Bool | TRUE: A MC command is active (MoveVelocity or stop) |
| additiveTorque Active | Bool | TRUE: Additive torque is active |
| torqueLimit Active | Bool | TRUE: Torque limitation is active |

**In/Out parameter**

| Identifier | Data type | Description |
|---|---|---|
| axis | DB_ANY | TO reference of axis |
| motionVector | LConSMC_type ActualDynamics | Active motion vector |
| torqueValues | LConSMC_type TorqueValues | Active torque values |



**Status & Error codes**

| Code / Value | Identifier / Description |
|---|---|
| 16#7000 | STATUS_NO_CALL<br>No job being currently processed |
| 16#7001 | STATUS_FIRST_CALL<br>First call after incoming new job (rising edge 'enable') |
| 16#7002 | STATUS_SUBSEQUENT_CALL<br>Subsequent call during active processing without further details |

**Functional description**

This function block can be used to forward/manipulate speed setpoints, torque setpoints and torque limits to the drive. All three setpoint channels (Velocity, torque limit and additive torque) can be individually activated and deactivated.

Via the input *override* the velocity setpoint can be scaled: Effective setpoint = referenceVelocity * override/100.0

**Note**

To use the torque pre-control/ limitation functions an additional telegram 750 must be used.

Configuration technology object -> Hardware interface -> Data exchange with the drive -> Additional data

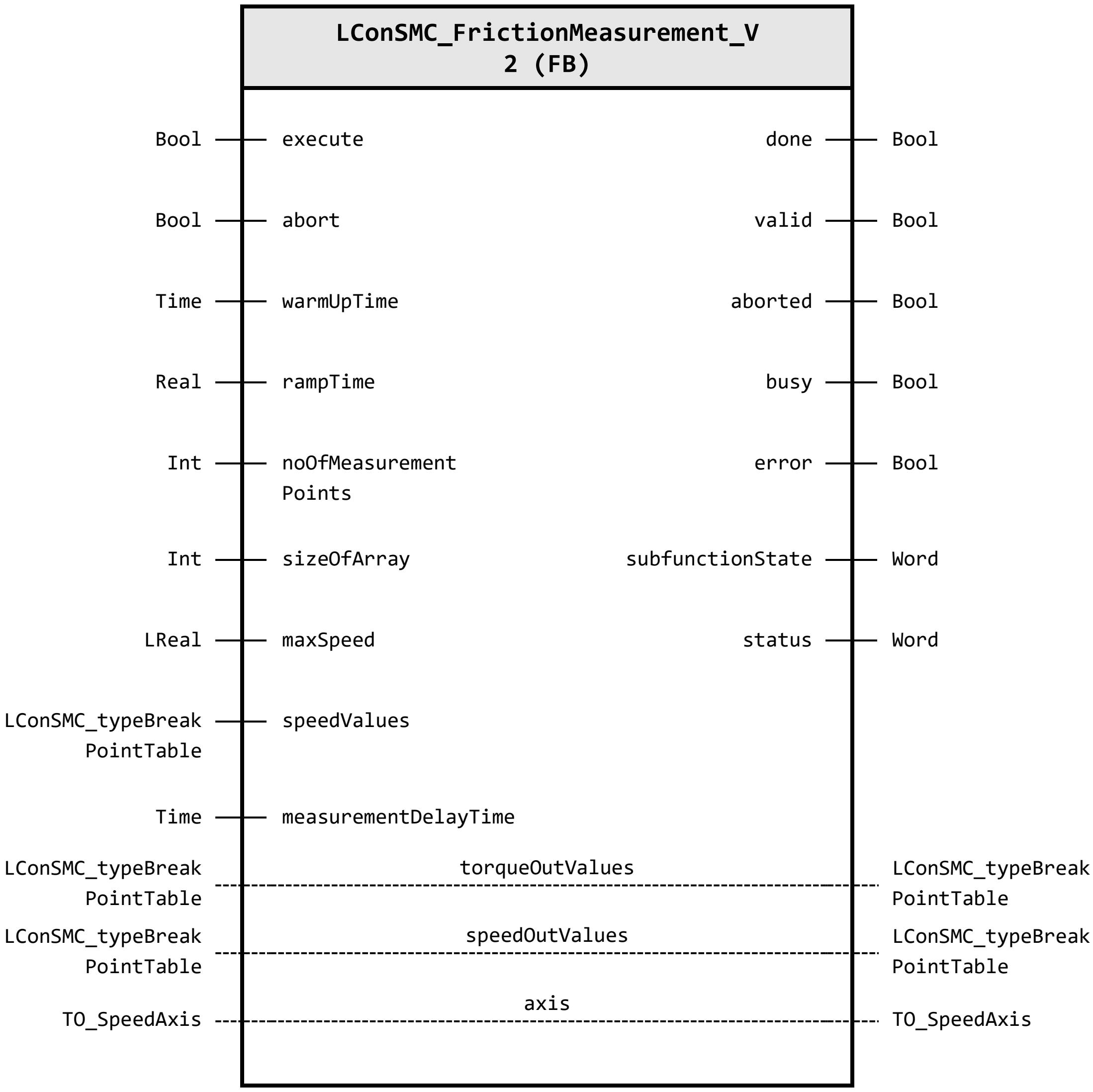
### 5.8.2 LConSMC_FrictionMeasurement_V2 (FB)

#### Short description

The function block is used to record a friction characteristic of the drive. The characteristic can then be used as a component of torque precontrol. This function should be calles in a program cycle OB. Function block ID: 3113

#### Interface description

#### Block interface




#### Input parameter

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| execute | Bool | FALSE | TRUE: Start friction measurement with rising edge |
| abort | Bool | FALSE | TRUE: Rising edge stops measurement and axis |
| warmUpTime | Time | T#0s | Warm up time<br>Before the measurement starts the drive rotates with maximum measurement speed for the parameterized time |
| rampTime | Real | 1000.0 | [ms] Time to ramp up to maximum speed |

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| noOfMeasurement Points | Int | 50 | Number measurement points per speed step (limited between 50-500) |
| sizeOfArray | Int | 20 | maxSpeed = 0:<br>Number of breakpoints (from "speedValues")<br>maxSpeed <> 0:<br>Number of measuring points between 0 and "maxSpeed" |
| maxSpeed | LReal | 0.0 | [RPM] Maximim speed at which the friction should be recorded |
| speedValues | LConSMC_typeBr eak PointTable | --- | [RPM] Breakpoints of speed values for which the friction should be measured. Only if "maxSpeed" = 0. |
| measurement DelayTime | Time | T#1S | Settling time after which the friction is measured at each speed setpoint |

### Output parameter

| Identifier | Data type | Description |
|---|---|---|
| done | Bool | TRUE: Measurement is finished |
| valid | Bool | TRUE: Measurement values are valid |
| aborted | Bool | TRUE: Measuremetn was aborted |
| busy | Bool | TRUE: Measurement is active |
| error | Bool | TRUE: An error has occured |
| subfunctionState | Word | Error identification |
| status | Word | Current status of function block |

### In/Out parameter

| Identifier | Data type | Description |
|---|---|---|
| torqueOutValues | LConSMC_typeBrea k PointTable | Friction torque values |
| speedOutValues | LConSMC_typeBrea k PointTable | Friction speed values |
| axis | TO_SpeedAxis | Axis |

### Status & Error codes

| Code / Value | Identifier / Description |
|---|---|
| 16#0000 | STATUS_EXECUTION_FINISHED |
| 16#7000 | STATUS_NO_CALL |
| 16#7001 | STATUS_FIRST_CALL |
| 16#7002 | STATUS_SUBSEQUENT_CALL |
| 16#7003 | STATUS_ABORTED |
| 16#8600 | ERR_UNDEFINED_STATE |

### Functional description

The function block measures the friction characteristic (speed / torque breakpoints) of a specific mechanical system. Speed setpoint are taken and torque values are saved in the specified data area referenced by an array. The data structure contains breakpoints of the characteristic (x value: speed, y value: torque), while the speed setpoint value has to be specified before starting the function block. The user has the possibility to let the function block preassign the speed values into an array. For this the input parameter maxSpeed must be unequal to zero. The speed values will be distributed evenly according to the length of the data block. The first entry will always be zero speed with zero torque. The function block is started using the input execute. Once started, speedSetpoint outputs the drives speed based on the values of the breakpoint table. If the input warmupTime > 0.0, the drive is moving for this defined period of time at the maximum speed before starting the measurement of the corresponding torque values. The torque values are averaged with defined number of samplings specified by input valueNumber. The ramp times for changing the speed setpoint are specified using the input ramptime. If the sequence is active, done = FALSE and busy = TRUE. If finished, done = TRUE and busy = FALSE. If an error occurs during the sequence, this is displayed at the error output and the associated errorID. If, at the input, the abort signal abort is set to TRUE, the measurement is interrupted and the setpoint ramped to zero.

**Note** The entered speed setpoint are on drive side in the unit 1/min. The setpoint is converted automatically considering the unit of the technology object and the entered gear ratio

Restrictions of the function block:

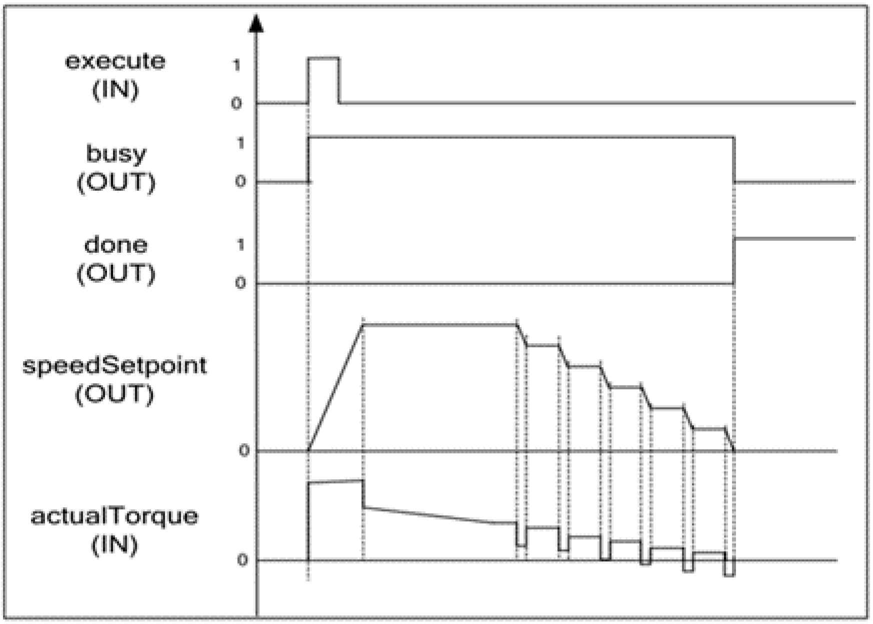
- Only rotatory axes are supported
- The axis must be enabled via the user program

**DANGER**

**Rotating machine parts**

Before starting the measurement it must be ensured that the drive can rotate freely without causing any damage. During the measurement high speed setpoints can occur.





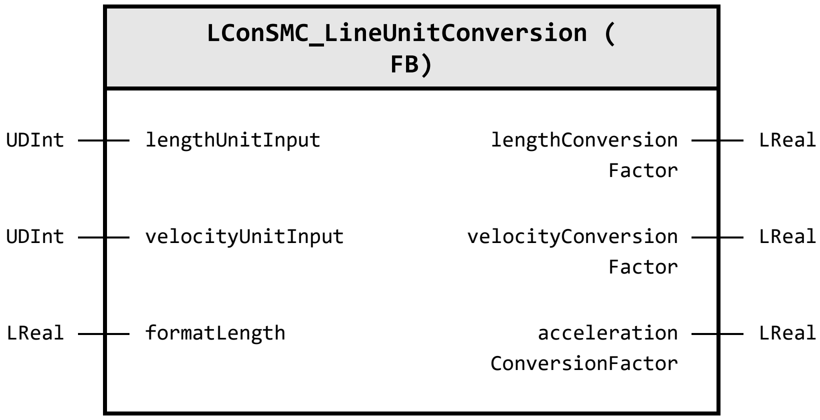
### 5.8.3 LConSMC_LineUnitConversion (FB)

#### Short description

This function block calculates conversion factors for position, velocity and acceleration of the leading axis (line axis) to the necessary units for the winder/ sectional drive function block. The function block can be called in any task. Function block ID: 3114

#### Interface description

#### Block interface






#### Input parameter

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| lengthUnitInput | UDInt | 0 | Configured TO unit at TO.Units.LengthUnit |
| velocityUnit Input | UDInt | 0 | Configured TO unit at TO.Units.VelocityUnit |
| formatLength | LReal | 0.0 | [m] or [ft] Format length<br>Only used for rotatory input units. One rotation correspondents to this formatLength |

#### Output parameter

| Identifier | Data type | Description |
|---|---|---|
| length Conversion Factor | LReal | Multiplicative conversion factor for length. Value of lengthUnitInput multiplied with lengthConversationFactor = [m] or [ft]. It depends on the unit of translatory axis or the unit of the formatLength at rotatory axis. |
| velocity Conversion Factor | LReal | Multiplicative conversion factor for velocity. Value of velocityUnitInput multiplied with velocityConversationFactor = [m]/min or [ft]/min. It depends on the unit of translatory axis or the unit of the formatLength at rotatory axis. |
| acceleration Conversion Factor | LReal | Multiplicative conversion factor for acceleration. Value of the respective accerleration regarding to velocityUnitInput multiplied with accelerationConversationFactor = [m]/s^2 or [ft]/s^2.<br>It depends on the unit of translatory axis or the unit of the formatLength at rotatory axis. |

#### Functional description

This function block is needed if the unit of the leading axis (line axis) is not in the required unit for the winder/sectional drive. In that case the FB can be used to calculate multiplicative conversion factors:

- inputUnit * multiplicative factor = winder/sectional drive unit

Output units for winder/ sectional drive FB:

| unit | metric system | anglo-american system |
|---|---|---|
| length | meter | feet |
| velocity | meter/minute | feet/minute |
| acceleration | meter/second^2 | feet/second^2 |

Example use case:

```
// Converting line position in [m]/[ft]
lineAxisMotionVector.s := lineAxis.position * lengthConversionFactor;
// Converting line velocity in [m/min]/[ft/min]
lineAxisMotionVector.v := lineAxis.velocity * velocityConversionFactor;
// Converting line acceleration in [m/s^2] / [ft/s^2]
lineAxisMotionVector.a := lineAxis.acceleration * accelerationConversionFactor;
```

Is the input unit in the metric system the factor is calculated for the metric system, otherwise for the anglo-american system.

# 5.9 Additional functions / 04_Sectional drive

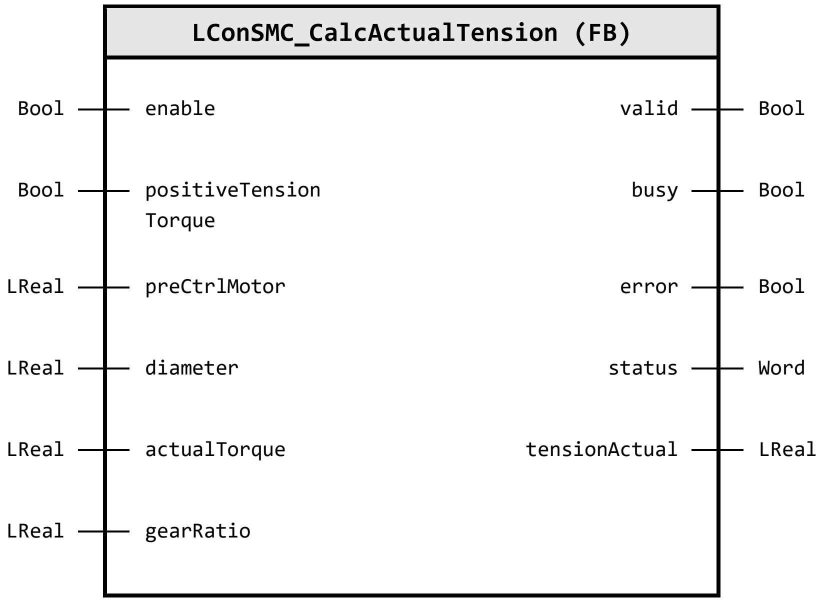
## 5.9.1 LConSMC_CalcActualTension (FB)

### Short description

Calculates the actual tension based on the drive torque

### Interface description

### Block interface






### Input parameter

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| enable | Bool | FALSE | TRUE: Enable functionality of FB |
| positiveTension Torque | Bool | TRUE | TRUE: Tension is generated with a positive torque value |
| preCtrlMotor | LReal | 0.0 | [Nm] Sum of acceleration torque and friction torque precontrol value |
| diameter | LReal | 0.0 | [m or inch] Diameter of sectional drive axis |
| actualTorque | LReal | 0.0 | [Nm] Actual torque value |
| gearRatio | LReal | 1.0 | [-] Gear ratio |

### Output parameter

| Identifier | Data type | Description |
|---|---|---|
| valid | Bool | TRUE: Valid set of output values available at the FB |
| busy | Bool | TRUE: FB is not finished and new output values can be expected |
| error | Bool | TRUE: An error occurred during the execution of the FB |
| status | Word | 16#0000 - 16#7FFF: Status of the FB, 16#8000 - 16#FFFF: Error identification |
| tensionActual | LReal | [N] Actual tension |

### Status & Error codes

| Code / Value | Identifier / Description |
|---|---|
| 16#7000 | STATUS_NO_CALL<br>No job being currently processed |
| 16#7001 | STATUS_FIRST_CALL<br>First call after incoming new job (rising edge 'enable') |
| 16#7002 | STATUS_SUBSEQUENT_CALL<br>Subsequent call during active processing without further details |

### Functional description

The function block calculates the current web tension based on the actual torque of the drive, the gear ratio and the current diameter. For the best calculation results, the disturbances coming from acceleration and friction effects must be pre controlled and the torque pre control value must be forwarded to the function block. The function block calculates the tension torque based on the following formula:

$$TensionTorque = (actualTorque - torquePreControl) * \frac{gearRatio}{radius}$$

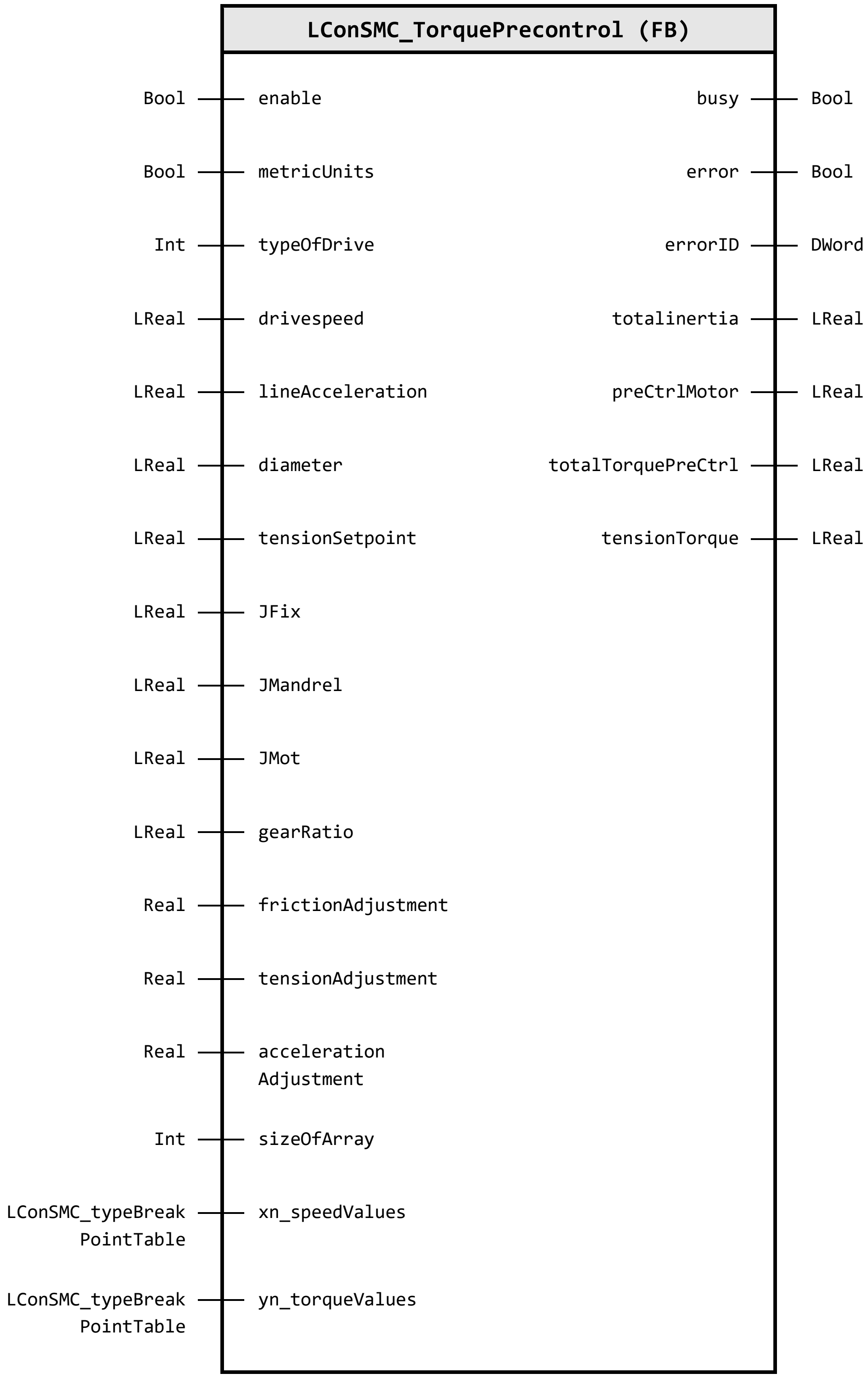
## 5.9.2 LConSMC_TorquePrecontrol (FB)

### Short description

Torque pre-control calculation. The function block must be called in a cyclic task. Function block ID: 6101

### Interface description

### Block interface



**Input parameter**

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| enable | Bool | FALSE | TRUE: Enable function block |
| metricUnits | Bool | TRUE | TRUE: Use metric system [m] / [m/min] / [m/s^2] / [N]<br>FALSE: Use angloamerican system [inch] / [ft/min] / [ft/s^2] / [Lbf] |
| typeOfDrive | Int | 0 | Definition of web tension direction<br>0: Web tension bevor the roll is controlled (Pull)<br>1: Web tension after the roll is controlled (Push) |
| drivespeed | LReal | 0.0 | [RPM] Drive speed for the friction characteristic |
| lineAcceleration | LReal | 0.0 | [m/s^2]/[ft/s^2] Web acceleration |
| diameter | LReal | 0.0 | [m] / [inch] Diameter of sectional drive |
| tensionSetpoint | LReal | 0.0 | [N] / [lbf] Tension setpoint |
| JFix | LReal | 0.0 | [kgm^2] / [lbft^2] Inertia on drive side (without drive, e.g. gear box) |
| JMandrel | LReal | 0.0 | [kgm^2] / [lbft^2] Inertia on load side (e.g. mandrel) |
| JMot | LReal | 0.0 | [kgm^2] / [lbft^2] Drive inertia |
| gearRatio | LReal | 0.0 | Gear ratio |
| friction Adjustment | Real | 0.0 | Multiplicative adaption factor for friction torque (1.0 = 100%) |
| tension Adjustment | Real | 0.0 | Multiplicative adaption factor for tension torque (1.0 = 100%) |
| acceleration Adjustment | Real | 1.0 | Multiplicative adaption factor for acceleration torque (1.0 = 100%) |
| sizeOfArray | Int | 20 | Numer of breakpoints that should be used from the friction characteristic |
| xn_speedValues | LConSMC_typeBreak PointTable | --- | Speed breakpoints for friction characteristic |
| yn_torqueValues | LConSMC_typeBreak PointTable | --- | Torque breakpoints for friction characteristic |



**Output parameter**

| Identifier | Data type | Description |
|---|---|---|
| busy | Bool | TRUE : Function block is executed |
| error | Bool | TRUE: Function block has detected an error |
| errorID | DWord | Further information on occured error (<> 16#0000_0000) |
| totalinertia | LReal | [kgm^2] / [lbft^2] Total inertia on the drive side |
| preCtrlMotor | LReal | [Nm] / [Lbf ft] Drive torque from friction and acceleration |
| totalTorquePreCtrl | LReal | [Nm] / [Lbf ft] Total torque precontrol value on drive side |
| tensionTorque | LReal | [Nm] / [Lbf ft] Tension torque on drive side |

**Functional description**

The function block determines the pre-controlled torque for sectional drives. The torque pre-control outputs the following torque setpoint to pre-control the drive:

1. A torque setpoint as a function of the acceleration setpoint of the machine/system ramp-function generator and the inertia
2. Friction torque setpoint is generated depending on the drive speed using a characteristic
3. Tension torque based on the tension setpoint and the diameter

In order to quickly compensate for disturbances and to implement the closed-loop control, the torque to be provided from the drive must be calculated as accurate as possible. Depending on the control type, this is either realized additively after the speed controller or by dynamically adjusting the torque limits. In both cases the appropriate torque must be calculated using its moment of inertia and transferred to the drive. As an alternative to the method integrated in this block, the integrated friction characteristic of the drive system (only in SINAMICS S120) can also be used.

**Note**

> The precise setting of the compensation is especially important for indirect tension control. This is so that the torque-generating current emulates, as far as possible, the material Tension.

### Calculations

The function block calculates the following physical quantities:

- Total moment of inertia

$$TotalInertia = \frac{J_{Mandrel}}{GearRatio^2} + J_{Mot} + J_{Fix}$$

- Angular acceleration of the drive

$$\alpha = 2 \cdot \frac{lineAcceleration \cdot GearRatio}{diameter}$$

- Acceleration torque

$$M_{acc} = TotalInertia \cdot \alpha \cdot accelerationAdjustment$$

- Friction torque

$$M_{fric} = f(n) \cdot frictionAdjustment$$

- Tension torque

$$M_{tension} = \frac{tensionSetpoint \cdot tensionAdjustment \cdot diameter}{2 \cdot GearRatio}$$

This friction torque is internally determined using the function block LConSMC_TableChar. The user is responsible for providing valid and sensible values in the appropriate data fields. If the factor FrictionAdjustment is set to 0.0, TableChar is not called and therefore a frictional torque is not added. The pre-control torque of the motor, the sum of all pre-control torques, and the tension torque are calculated depending on the drive type. The total torque is always the sum of the preCtrlMotor and tensionTorque.

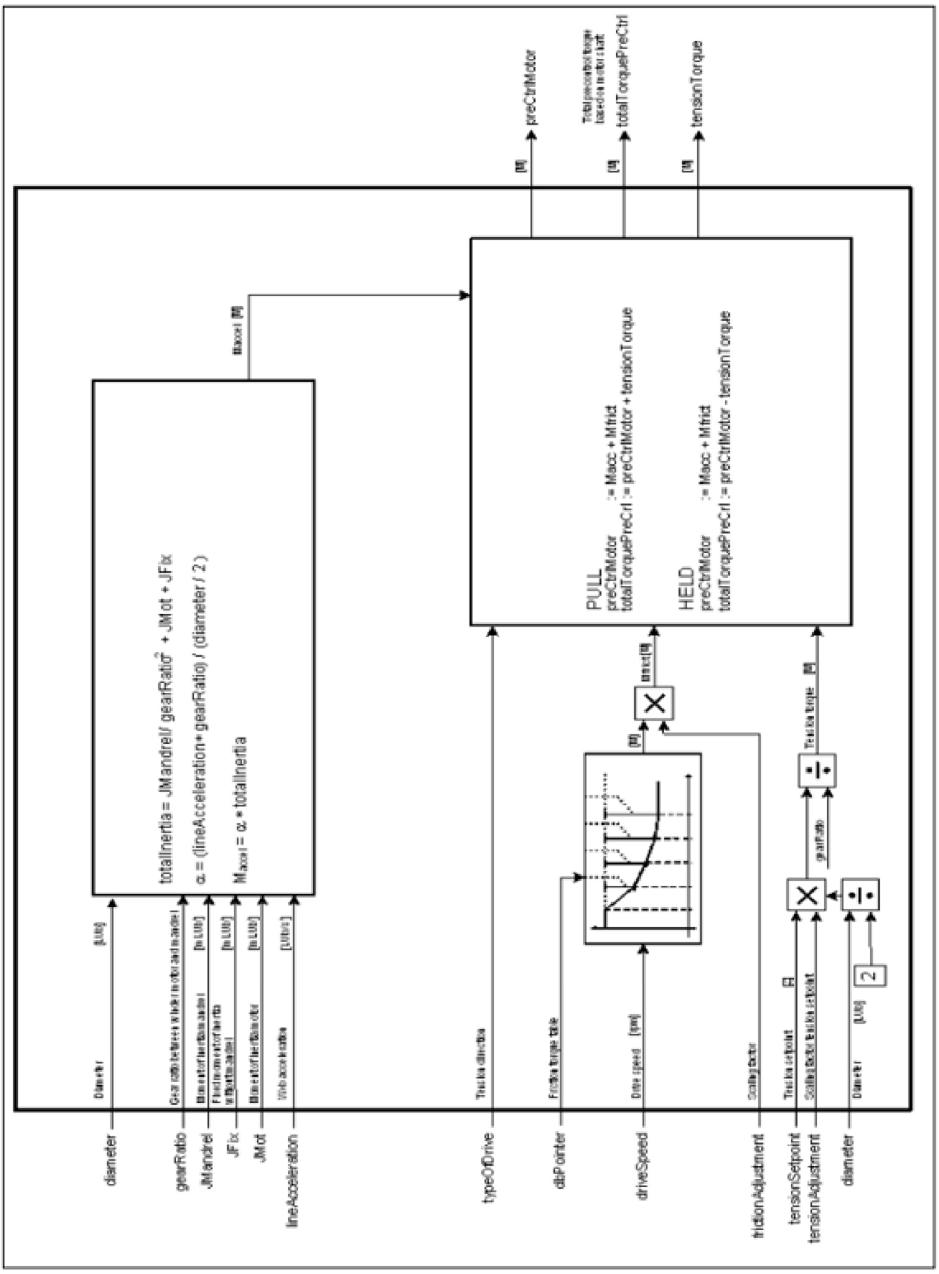
- Drive mode = PULL

$$M_{preCtrlMotor} = M_{acc} + M_{fric} \qquad M_{TensionTorque} = M_{TensionTorque}$$

- Drive mode = HELD

$$M_{preCtrlMotor} = M_{acc} + M_{fric} \qquad M_{TensionTorque} = -M_{TensionTorque}$$

**Block diagram**



## 5.10 Additional functions / 05_Winder

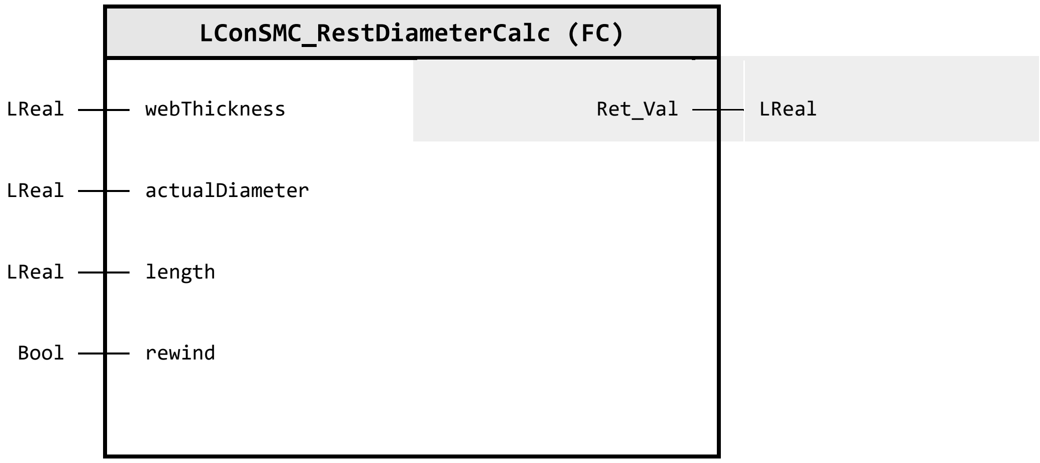
### 5.10.1 LConSMC_RestDiameterCalc (FC)

#### Short description

Calculation of the rest diameter of the coil. The function can be called in any task. Function ID: 4002

#### Interface description

#### Block interface






#### Input parameter

| Identifier | Data type | Description |
|---|---|---|
| webThickness | LReal | [m or inch] Thickness of material |
| actualDiameter | LReal | [m or inch] Actual diameter of coil |
| length | LReal | [m or inch] Length to un/rewind |
| rewind | Bool | TRUE: Rewinder<br>FALSE: Unwinder |

#### Output parameter

| Identifier | Data type | Description |
|---|---|---|
| Ret_Val | LReal | [m or inch] Rest diameter to be coiled |

#### Functional description

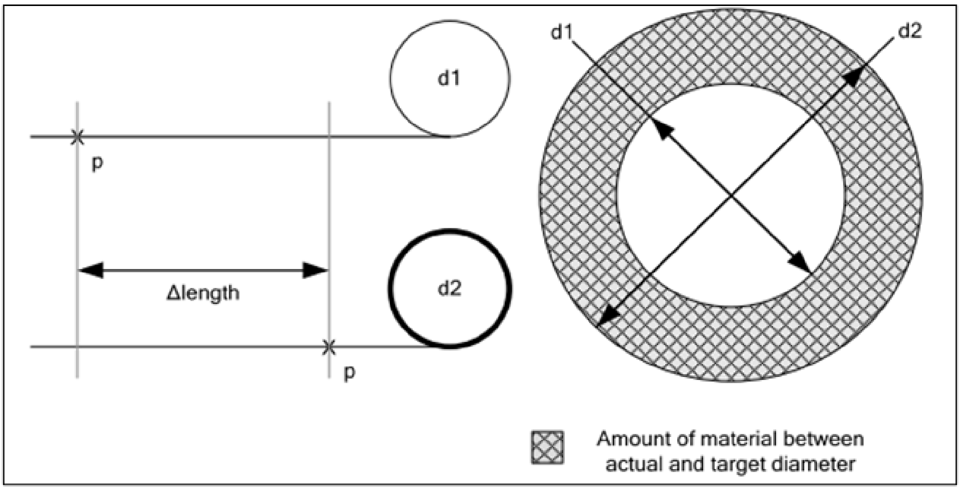
The function uses the input parameters to calculate the target diameter of a roll. The calculation relies on the material thickness (webThickness), the actual roll diameter (actualDiameter) and the material length to be wound or unwound (length). The winding mode is defined using the windingMode input. The material thickness (webThickness) is internally limited (MIN_THICKNESS, MAX_THICKNESS). In case of a too high material length input of the function result is zero.

- Rewinder

$$RestDiameterCalc = 2 \cdot \sqrt{\frac{actualDiameter^2}{4} + \frac{webThickness \cdot length}{\pi}}$$

- Unwinder

$$RestDiameterCalc = 2 \cdot \sqrt{\frac{actualDiameter^2}{4} - \frac{webThickness \cdot length}{\pi}}$$




| Winder mode | Actual diameter | Target diameter |
|---|---|---|
| Rewinder | d1 < d2 | d2 (full roll, max. diameter) |
| Unwinder | d2 < d1 | d1 (empty roll, min. diameter) |

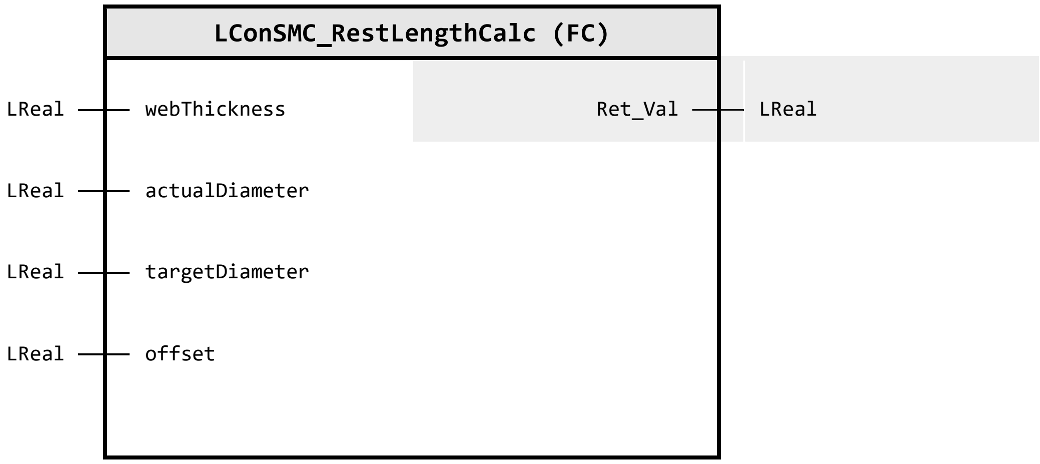
### 5.10.2 LConSMC_RestLengthCalc (FC)

#### Short description

Calculation of the rest length of the coil. The function can be called in any task. Function ID: 4001

#### Interface description

#### Block interface




#### Input parameter

| Identifier | Data type | Description |
| --- | --- | --- |
| webThickness | LReal | [m or inch] Material thickness |
| actualDiameter | LReal | [m or inch] Current diameter |
| targetDiameter | LReal | [m or inch] Target diameter |
| offset | LReal | [m or inch] Offset rest length on the coil |

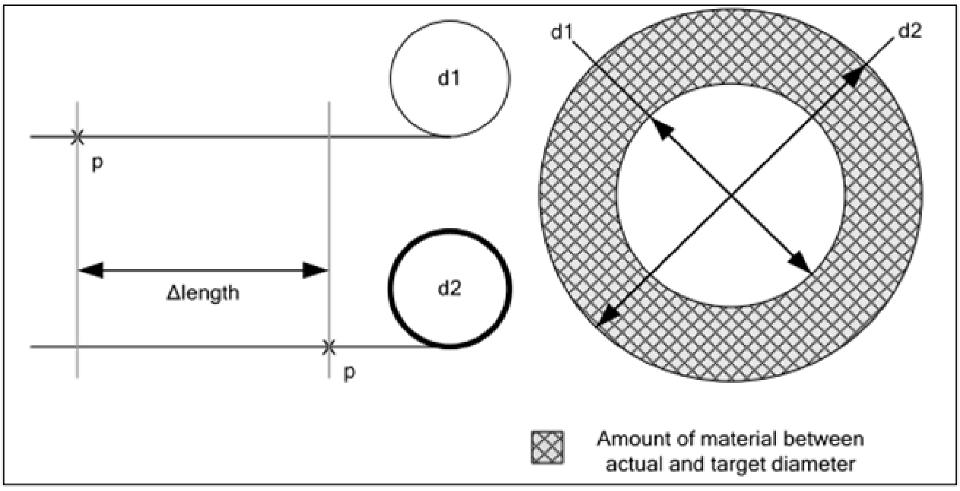
#### Output parameter

| Identifier | Data type | Description |
| --- | --- | --- |
| Ret_Val | LReal | [m or inch] Rest length on the coil |

#### Functional description

Depending on the roll parameters that have been entered, the function calculates the residual length of the material. To do this, the following data is required – the thickness of the material on the roll (webThickness), the actual diameter (actualDiameter) and the diameter to be reached (targetDiameter). In this case, the target diameter represents the end of the roll or the diameter to be reached. Depending on whether the target diameter is greater than or less than the actual diameter, the length of the material on the roll or the length of the material still to be wound is output. The thickness of the material web (webThickness) is limited using the internal constants MIN_THICKNESS and MAX_THICKNESS.

$$ReturnValue = \frac{\pi}{4 \cdot webThickness} \cdot (actualDiameter^2 - targetDiameter^2) + offset$$




| Winder mode | Actual diameter | Target diameter |
| --- | --- | --- |
| Rewinder | d1 < d2 | d2 (full roll, max. diameter) |
| Unwinder | d2 < d1 | d1 (empty roll, min. diameter) |

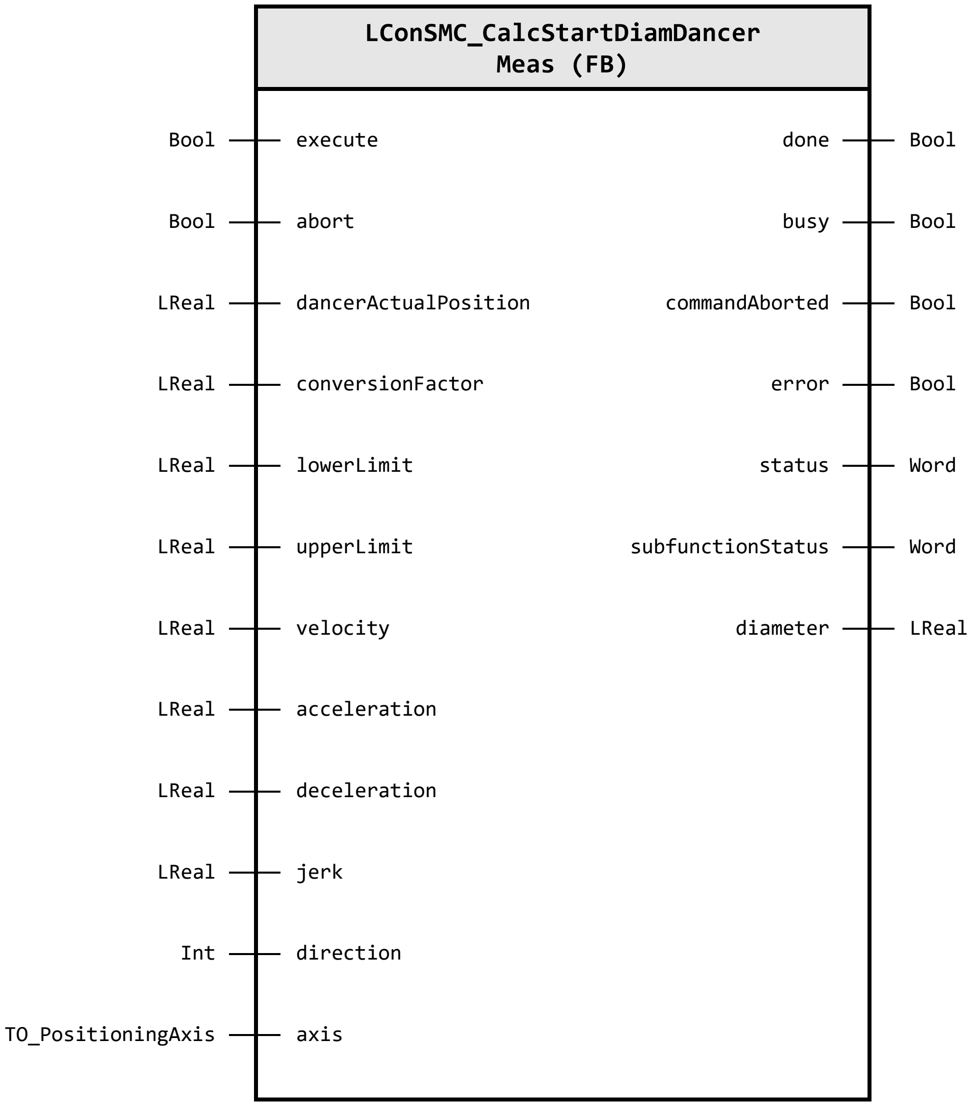
### 5.10.3 LConSMC_CalcStartDiamDancerMeas (FB)

#### Short description

Calculate the initial diameter of a winder roll using a measurement run using the dancer position

#### Interface description

#### Block interface




#### Input parameter

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| execute | Bool | FALSE | Rising edge starts action once |
| abort | Bool | FALSE | TRUE: Stop measurement |
| dancerActual Position | LReal | 0.0 | [Dancer Unit] Actual position of the dancer (e.g. %, Meter, Millimeter, etc.) |
| conversionFactor | LReal | 0.0 | [m/100Dancer Units] Conversion factor to convert the change of the input "dancerActualPosition" into a web length change in meter. The conversion factor describes the change of the total web length for a change of the dancer actual position of 100 units(e.g. %,meter, millimeter, etc.) |

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| lowerLimit | LReal | 0.0 | [Dancer Unit] Lower dancer position limit. If dancer is above this position the measurement is started |
| upperLimit | LReal | 0.0 | [Dancer Unit] Upper dancer position limit. If dancer is above this position the measurement is stoped and the diameter is calculated |
| velocity | LReal | 0.0 | [TO Unit] Axis velocity for the measurement |
| acceleration | LReal | 0.0 | [TO Unit] Axis acceleration for the measurement |
| deceleration | LReal | 0.0 | [TO Unit] Axis deceleration for the measurement |
| jerk | LReal | 0.0 | [TO Unit] Axis jerk for the measurement |
| direction | Int | 0 | Rotation direction for the measurement (Direction in which the dancer is pulled upwards)<br>0: The direction of the rotation is defined by the sign of the velocity input<br>1: To pull up the dancer the winder must rotate in positive direction<br>2: To pull up the dancer the winder must rotate in negative direction |
| axis | TO_PositioningAxis | --- | Winde axis |

### Output parameter

| Identifier | Data type | Description |
|---|---|---|
| done | Bool | TRUE: Commanded functionality has been completed successfully |
| busy | Bool | TRUE: FB is not finished and new output values can be expected |
| commandAborted | Bool | TRUE: Commanded functionality has been aborted by another command<br>(optional) |
| error | Bool | TRUE: An error occurred during the execution of the FB |
| status | Word | 16#0000 - 16#7FFF: Status of the FB, 16#8000 - 16#FFFF: Error identification |
| subfunction Status | Word | Detailed error description |
| diameter | LReal | [m] Calculated diameter value (Can be forwarded to the LConSMC_Winder function block to initialize internal diameter calculation: winderConfig.diamConfig.diameterSetValue) |

### Status & Error codes

| Code / Value | Identifier / Description |
|---|---|
| 16#0000 | STATUS_EXECUTION_FINISHED<br>Execution finished without errors |
| 16#7000 | STATUS_NO_CALL<br>No job being currently processed |
| 16#7001 | STATUS_FIRST_CALL<br>First call after incoming new job (rising edge 'execute') |
| 16#7002 | STATUS_SUBSEQUENT_CALL<br>Subsequent call during active processing without further details |
| 16#7FFF | STATUS_COMMAND_ABORTED<br>Commanded functionality has been aborted by another command |

### Functional description

This function block calculates the initial diameter of a winder roll using a measurement run. During this process, a dancer is moved to two configurable positions (lowerLimit& upperLimit) while both dancer and winder positions are recorded. The diameter is then calculated based on these measurements.

The function block provides a parameterizable measurement speed (velocity, acceleration, deceleration, jerk and direction) and allows for immediate termination of the measurement process via the “abort” input. The calculated diameter serves as a starting value for subsequent winding operations.

**Features**

- Automatic calculation of initial winder roll diameter
- Two-point measurement using dancer positions
- Configurable dancer target positions
- Adjustable measurement speed
- Immediate abort capability
- Comprehensive status and error feedback
- Operation

**Note** The user must take care that the dynamics (acceleration, deceleration and jerk) are suitable to stop the winder axis before the dancer reaches its mechanical limit!

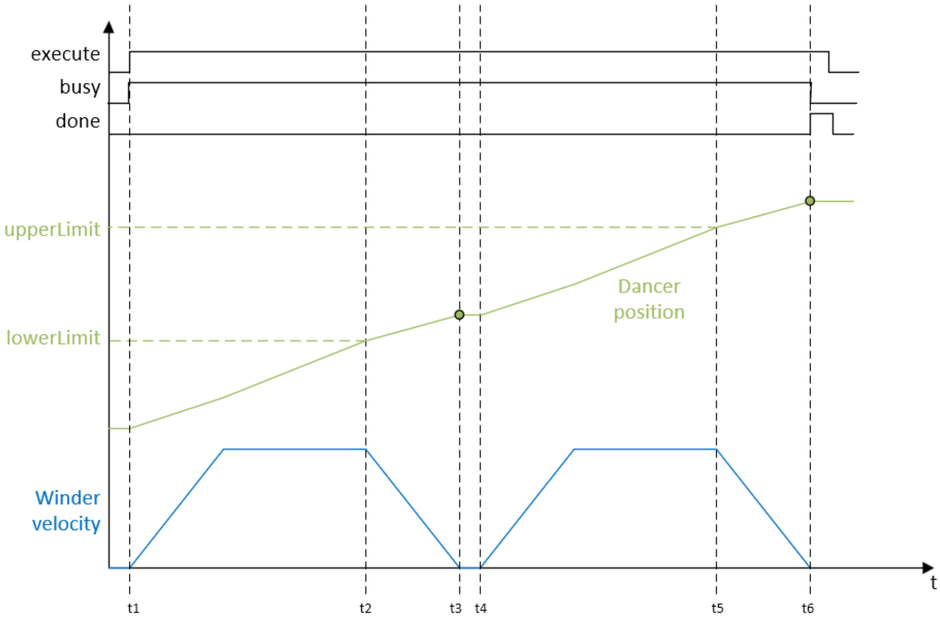


Therefore the maximum permissible rotation angle ($\alpha_{max}$) during the measurement can be considered which occurs at the maximum diameter ($d_{max}$). If the winder rotates with this angle the dancer is fully pulled up from the zero to the maximum position which equivalents the maximum possible web change ($x_{WebMax}$).

$$\alpha_{max} = \frac{360° \cdot x_{WebMax}}{(d_{max} \cdot \pi)}$$

### Timing diagram

The following image displays the sequence of a measurement run to calculate the starting diameter:



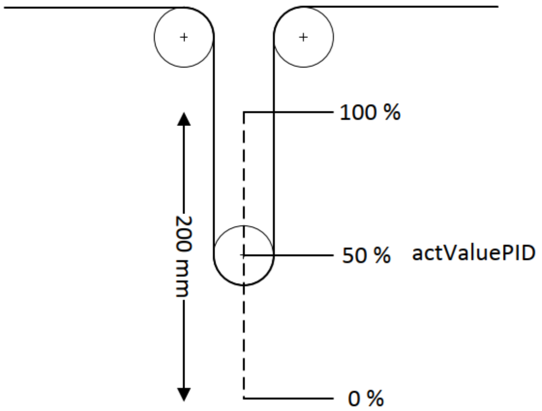
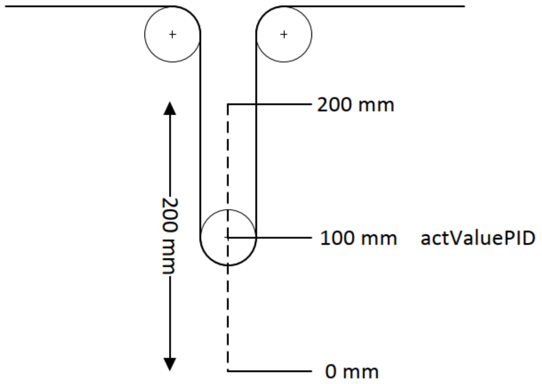

| Step | Description |
|---|---|
| t1 | Start the measurement with a rising edge at the *execute* input. A successful validation of the input parameters and a running measurement is indicated via the output *busy* set to TRUE. The winder will start to rotate with the defined sign and velocity setpoint. The dancer is being pulled up. If during the start of the measurement the dancer actual position is already at or above the lower limit but still below the upper limit the measurement will save the current dancer and winder position and will directly jump to step *t4* |
| t2 | As soon as the defined lower limit of the dancer position is reached the winder will stop. |
| t3 | If the winder has stopped the actual dancer position and the winder position (and in case of a modulo axis also the modulo cycle) is saved. |
| t4 | If the dancer is still below the upper limit the winder will start to run again to further pull up the dancer. If the previous stopping took too long, or the upper dancer limit is too close to the lower limit so that the actual dancer position is already above the upper limit the measurement is aborted and an error is displayed. |
| t5 | If the upper limit is reached the winder will be stopped. |
| t6 | If the winder has stopped the actual dancer position and the winder position is saved. Based on the distance the winder has rotated and the amount of web length that has changed, the actual diameter is calculated. The completion of the measurement is indicated of the output *done* is set to TRUE and the calculated value is displayed at the output *diameter*. |

**Note**

Using the input *abort* the measurement can be aborted at any time. The winder axis will stop with the defined dynamics and the output *commandAborted* is set.

### Conversion Factor

The input *conversionFactor* is required to convert the change of the input *dancerActualPosition* into a web length change in **meter.** The factor depends on the mechanical construction and the unit of the feedback signal of the actual dancer position and represents the web length change in meter of the dancer position changes by 100 units (e.g. %, millimeter, meter etc.). The following two examples should demonstrate how to determine the conversion factor:

| Configuration Example 1 | Description |
|---|---|
|  | The feedback of the dancer position is defined in **%**. The maximum dancer movement is between 0.0 and 100%, which equivalents a linear movement of 200mm. In this example the conversion factor must be set to **0.4**, as a change of 100 dancer units (=100%) leads to a web length change of 400 millimeters → **conversionFactor := 0.4** |
| **Configuration Example 2** | **Description** |
|  | The feedback of the dancer position is defined in **mm**. The maximum dancer movement is between 0.0 and 200mm, which equivalents also a linear movement of 200mm. In this example the conversion factor must be set to **0.2**, as a change of 100 dancer units (=100mm) leads to a web length change of 200 millimeters → **conversionFactor := 0.2** |

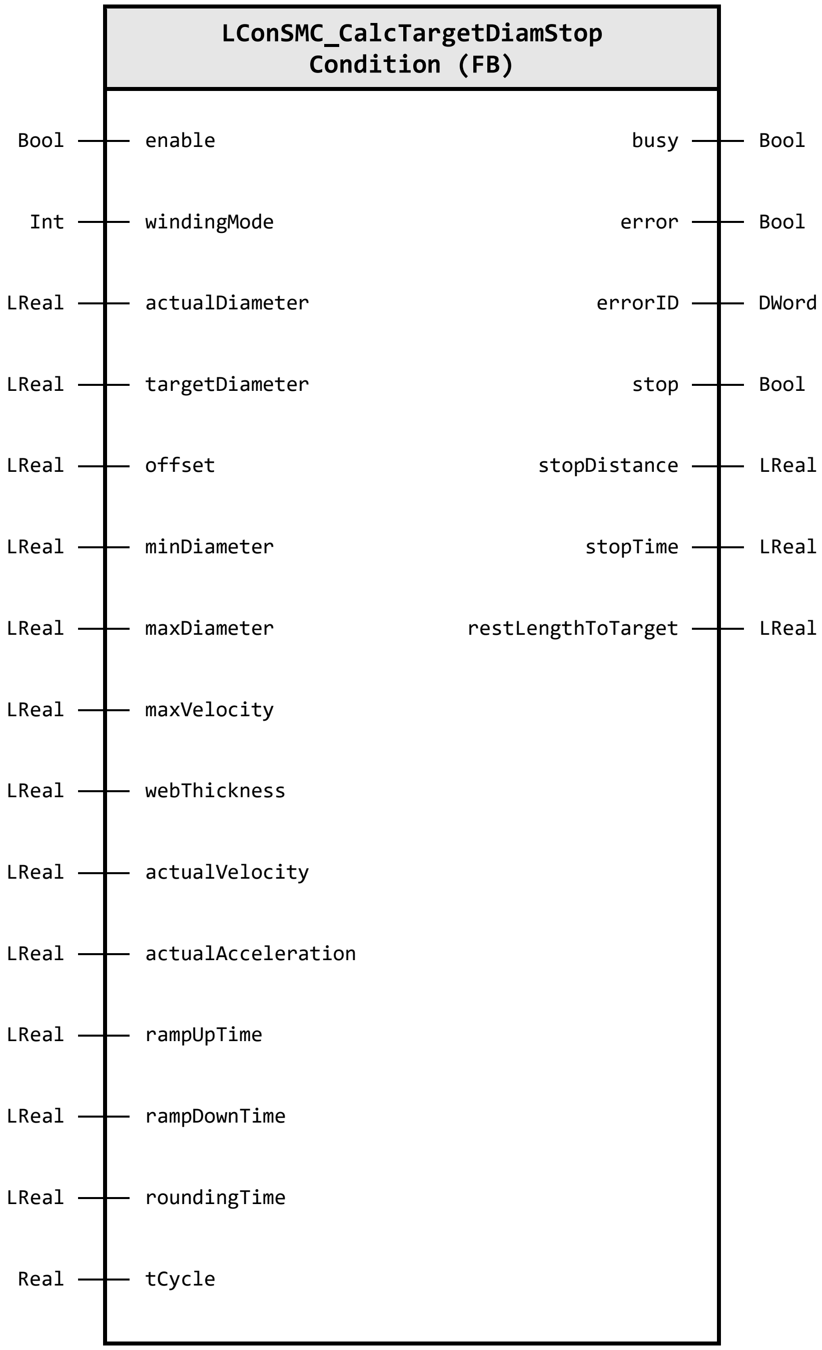
## 5.10.4 LConSMC_CalcTargetDiamStopCondition (FB)

### Short description

Calculation of stop condition at target diameter. The function block must be called in a cyclic interrupt, e.g. OB30. Function block ID: 4004

### Interface description

### Block interface




### Input parameter

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| enable | Bool | FALSE | TRUE: Enable function block |

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| windingMode | Int | 0 | Winding direction<br>0: Rewinding<br>1: Unwinding |
| actualDiameter | LReal | 0.0 | [m or inch] Actual diameter value of winder |
| targetDiameter | LReal | 0.0 | [m or inch] Target diameter for winder |
| offset | LReal | 0.0 | [m or inch] Length offset for material on winder |
| minDiameter | LReal | 0.0 | [m or inch] Minimum diameter of winder |
| maxDiameter | LReal | 0.0 | [m or inch] Maximum diameter of winder |
| maxVelocity | LReal | 0.0 | [m/min or ft/min] Maximum web velocity |
| webThickness | LReal | 0.0 | [m or inch] thickness of material |
| actualVelocity | LReal | 0.0 | [m/min or ft/min] Actual web velocity |
| actual Acceleration | LReal | 0.0 | [m/s^2 or ft/s^2] Actual web acceleration |
| rampUpTime | LReal | 0.0 | [s] Ramp up time of web |
| rampDownTime | LReal | 0.0 | [s] Ramp down time of web |
| roundingTime | LReal | 0.0 | [s] Rounding time of web |
| tCycle | Real | 0.0 | [ms] Cycle time the function block is called |

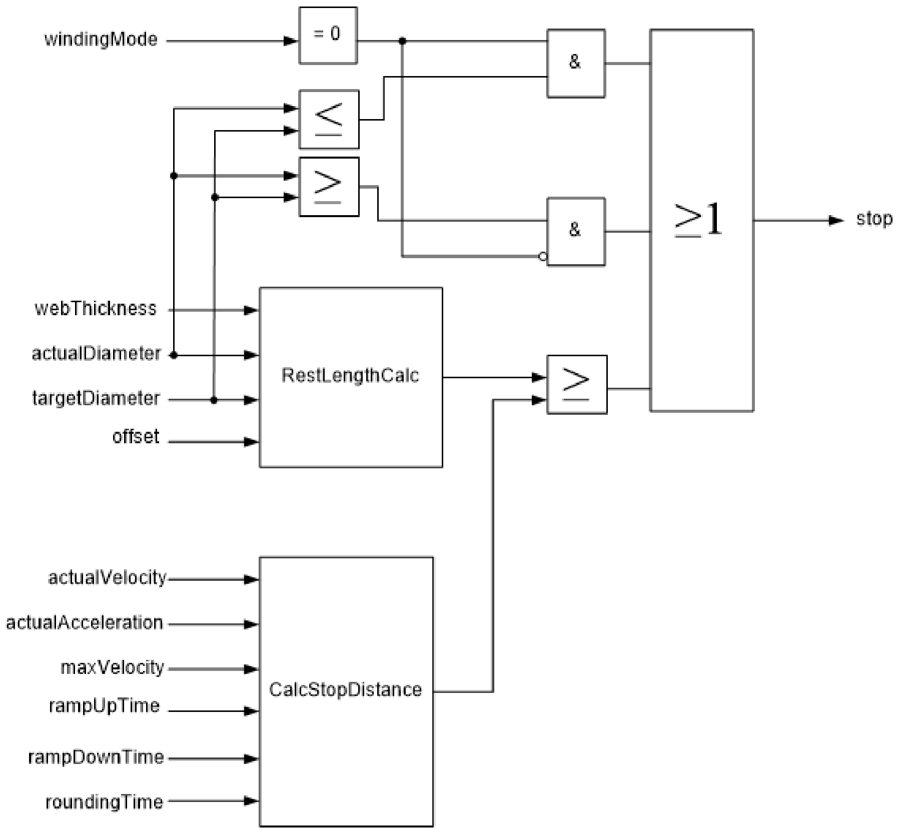
### Output parameter

| Identifier | Data type | Description |
|---|---|---|
| busy | Bool | True: The function block is activated |
| error | Bool | True: An error occurred |
| errorID | DWord | Detailed error description |
| stop | Bool | True: Stop condition reached. The winder must be stopped manually |
| stopDistance | LReal | [m or inch] Actual distance to stop the line |
| stopTime | LReal | [s] Actual time to stop the line |
| restLengthTo Target | LReal | [m or inch] Rest material length between actual diameter and target diameter |



### Functional description

The function block combines the Functions “CalcStopDistance” and “RestLengthCalc”. It calculates based on the input parameters if the line axis needs to be stopped to reach the target diameter. The function block calculates the rest length depending on the actual and target diameter which needs to be re- or unwound. The winding mode and an optional offset will also be used for the calculation. Additionally, the distance the axis needs to stop the motion to 0.0 LU/min will be calculated based on the actual motion vector of the axis. If the rest length is smaller as the stop distance, the stop condition is reached and the output “stop” is set to true. If the actual diameter already exceeded the target diameter – actual diameter is smaller than target diameter in mode unwinding or actual diameter is larger than target diameter in mode rewinding – the output “stop” will be set to true immediately. After setting a new value for actualDiameter, targetDiameter and / or changing the windingMode the output will be reset to false, if a positive rest length to the target diameter is present.


windingMode
= 0
&
&
≥1
stop
webThickness
actualDiameter
targetDiameter
offset
RestLengthCalc
actualVelocity
actualAcceleration
maxVelocity
rampUpTime
rampDownTime
roundingTime
CalcStopDistance

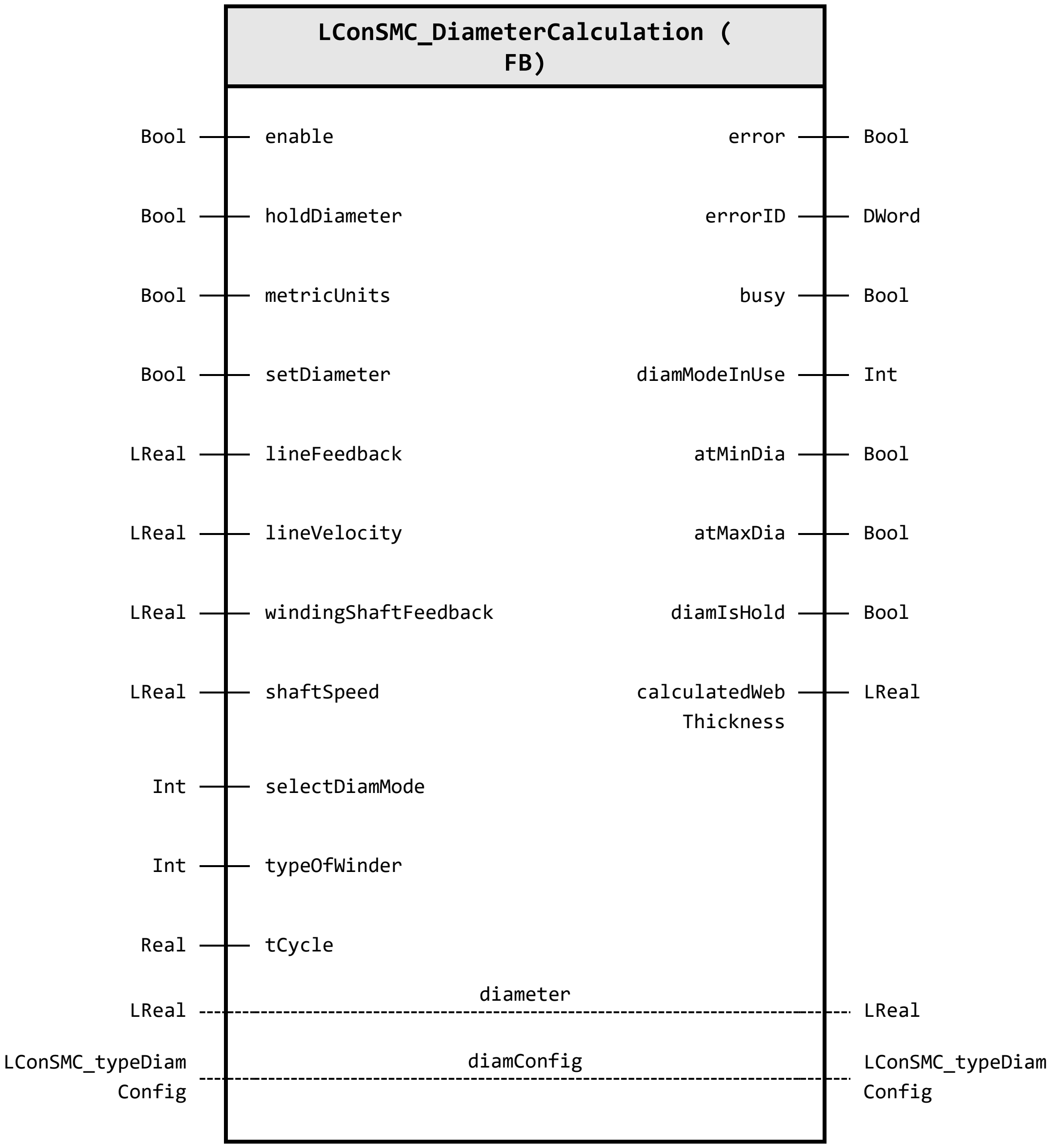
### 5.10.5 LConSMC_DiameterCalculation (FB)

#### Short description

This function block calculates the actual diameter of the roll based on the behavior of the winding drive and line drive. One of six modes can be selected for the calculation. The function block must be called in a cyclic OB, e.g. OB30. Function block ID: 5104

#### Interface description

#### Block interface




#### Input parameter

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| enable | Bool | FALSE | TRUE: Enable function block |
| holdDiameter | Bool | FALSE | TRUE: Holds the actual diameter |
| metricUnits | Bool | TRUE | TRUE: Use metric system [m] / [m/min] / [m/s^2] / [N]<br>FALSE: Use angloamerican system [inch] / [ft/min] / [ft/s^2] / [Lbf] |

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| setDiameter | Bool | FALSE | TRUE: Sets the diameter to a specified value "diamConfig.diameterSetValue" |
| lineFeedback | LReal | 0.0 | [m or ft] Web position |
| lineVelocity | LReal | 0.0 | [m/min oder ft/min] Web velocity |
| windingShaft Feedback | LReal | 0.0 | [°] Winder shaft position |
| shaftSpeed | LReal | 0.0 | [RPM] Speed of the winder shaft on load side |
| selectDiamMode | Int | 0 | Type of diameter calculation (Can be changed on the fly)<br>0: V/n<br>1: Thickness addition<br>2: Integral<br>3: External (Sensor value must be connected to "diamConfig.diameterSetValue")<br>4: WebLength<br>5: Position |
| typeOfWinder | Int | 0 | Definition of winding direction<br>0: Rewind from above<br>1: Rewind from below<br>2: Unwind from above<br>3: Unwind from below |
| tCycle | Real | 0.0 | [ms] Cycle time the function block is called |



### Output parameter

| Identifier | Data type | Description |
|---|---|---|
| error | Bool | TRUE: An error occured |
| errorID | DWord | Detailed error description |
| busy | Bool | TRUE: The function block is active |
| diamModeInUse | Int | Active diameter mode<br>0: V/n<br>1: Thickness addition<br>2: Integral<br>3: External<br>4: WebLength<br>5: Position |
| atMinDia | Bool | TRUE: Lower limit of the actual diameter value reached (see "diamConfig.minDia") |
| atMaxDia | Bool | TRUE: Upper limit of the actual diameter value reached (see "diamConfig.maxDia") |
| diamIsHold | Bool | TRUE: Diameter calculation is stopped. Possible reasons:<br>Input "holdDiameter" = TRUE<br>Minimum velocity reached (see diamConfig.minLineVelocity/minShaftSpeed) |
| calculatedWeb Thickness | LReal | [m or inch] Calculated value of the web thickness |

### In/Out parameter

| Identifier | Data type | Description |
|---|---|---|
| diameter | LReal | [m or inch] Calculated diameter |
| diamConfig | LConSMC_typeDiam Config | Diameter configuration |

## Functional description

The diameter calculator determines the actual diameter value of the roll being wound (or unwound) using the input quantities:

- Web velocity and position
- Shaft speed of the mandrel or angle
- Web thickness

The following calculation modes are realized within the function block:

| Calculation mode | Description |
|---|---|
| Division | Calculates the diameter based on the ratio between the web velocity and winder velocity or speed (speed) |
| Thickness | Integrates the web thickness |
| Integral | Calculates the diameter from the ratio between the winder angle and web length by integrating the winding speed and the web velocity. |
| External | Senses the diameter using a sensor |
| Web length | Diameter calculation based on the wound material via the integration of the line velocity |
| Position | Calculates the diameter using the ratio between the winder angle and web length by measuring the winder position (angle) and the web length. |

At the first run the specified mode is accepted and is issued at output parameter diamModeInUse for the user. A flying change of the calculation mode is possible. The *holdDiameter* input will pause diameter calculation and hold the resulting diameter at the current value when set to true. The *setDiameter* input writes the *diameterSetvalue* as the diameter if the actual shaft speed is below minimum shaft speed. The set value is limited by the minimum and maximum diameter. With actual line velocity above line and actual shaft speed above minimum and not hold, the diameter is calculated. The result of the division is limited and smoothed.

| Quantity | Metric | Imperial |
|---|---|---|
| line velocity | m/min | ft/min |
| winder shaft speed | rpm | rpm |
| Line position | meter | feet |
| Winder position | degree | degree |
| diameter | meter | inch |

The diameter is calculated in two steps:

1. The raw diameter value is calculated
2. The raw diameter value is post-conditioned (limited, interpolated and smoothed)

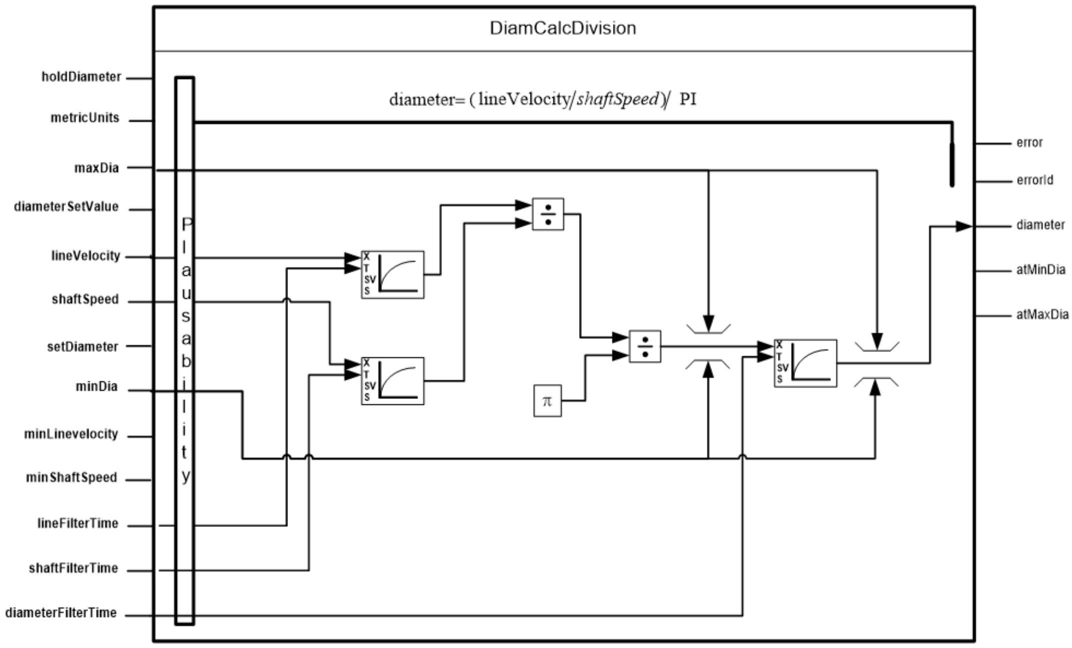
The functionality of each calculation mode is described in the following:

## Division

The function block determines the actual diameter value (diameter) of the roll being wound (or unwound) using the input quantities:

- Web velocity (lineVelocity) - the web velocity is the setpoint velocity of the machine and the velocity with which the main drive moves the material web. In exceptional cases, it is possible to use the actual value from a tachometer measuring the web speed.
- Shaft speed of the mandrel (shaftSpeed)

$$diameter = \frac{v}{n \cdot \pi}$$





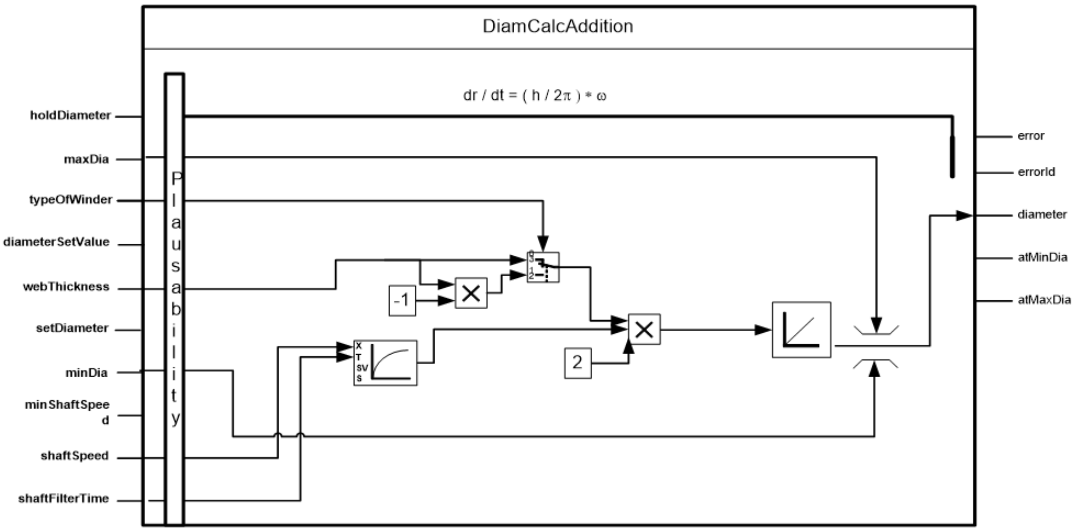

**Thickness addition**

The diameter calculator determines the actual diameter value (diameter) of the roll being wound (or unwound) using the input quantities:

- Web Thickness - the actual web (material) thickness.
- Shaft speed of the mandrel (shaftSpeed) – Actual speed of the winder shaft.

**Note**

This calculation method relies very much on the accuracy of the material thickness and the starting diameter value. Any deviation between the actual effective material thickness and the set material thickness will lead to an invalid diameter which error will get bigger after each revolution. An inaccurate starting value will remain as an offset between the actual and the calculated diameter.
This calculation mode is mostly useful for materials where air entrapment and in wound tension does not influence the effective material thickness.




**Integral mode**

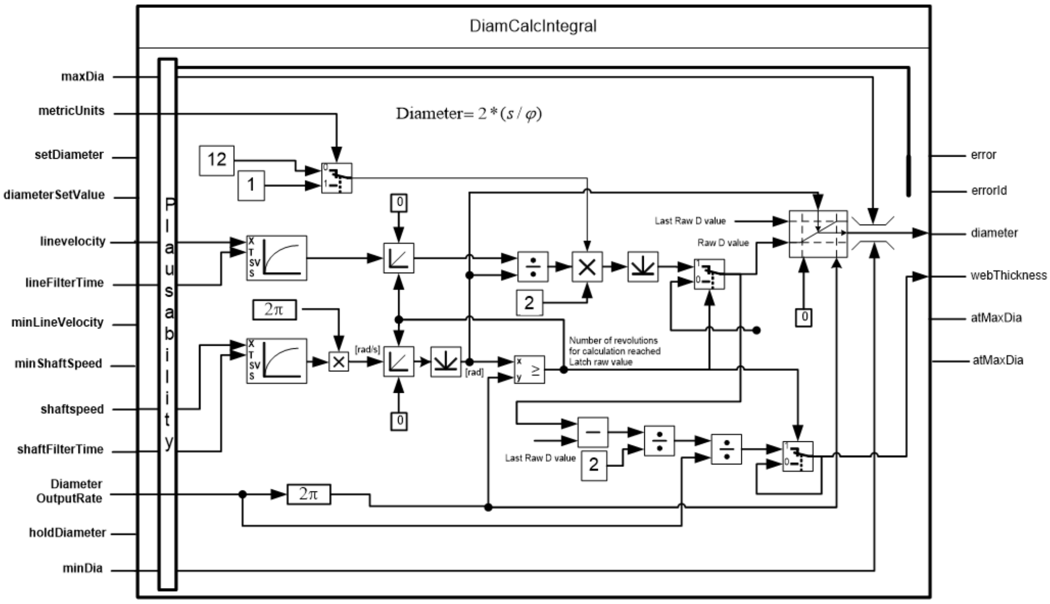
The diameter calculator determines the actual diameter value (diameter) of the roll being wound (or unwound) using the input quantities:

- Web velocity (lineVelocity) - the web velocity is the setpoint velocity of the machine and the velocity with which the main drive moves the material web. In exceptional cases, it is possible to use the actual value from a tachometer measuring the web speed.
- Shaft speed of the mandrel (shaftspeed) – Actual speed of the winder shaft.

The actual motor speed is converted to the roll shaft (by the gear ratio) and is used to calculate the diameter. The integral diameter calculation is also based on the division of the web quantity by the angular quantity. However, quantities v and ω are integrated over several roll revolutions that can be specified by measuringRevolutionCount. This is to minimize the disturbing effect of processing just individual values. The following applies:

$$diameter = 2 \cdot \frac{\int v}{\int n}$$

If the number of revolutions of the winding shaft - specified by parameter diameterOutputRate – is reached, then the integrators are reset. One integrator will give the length change of the material for the specified number of winder shaft revolutions. From these two values the (average) diameter can be calculated.




## External

In this mode an external diameter value, e.g., from a sensor, can be used via the parameter *diamConfig.diameterSetValue*.

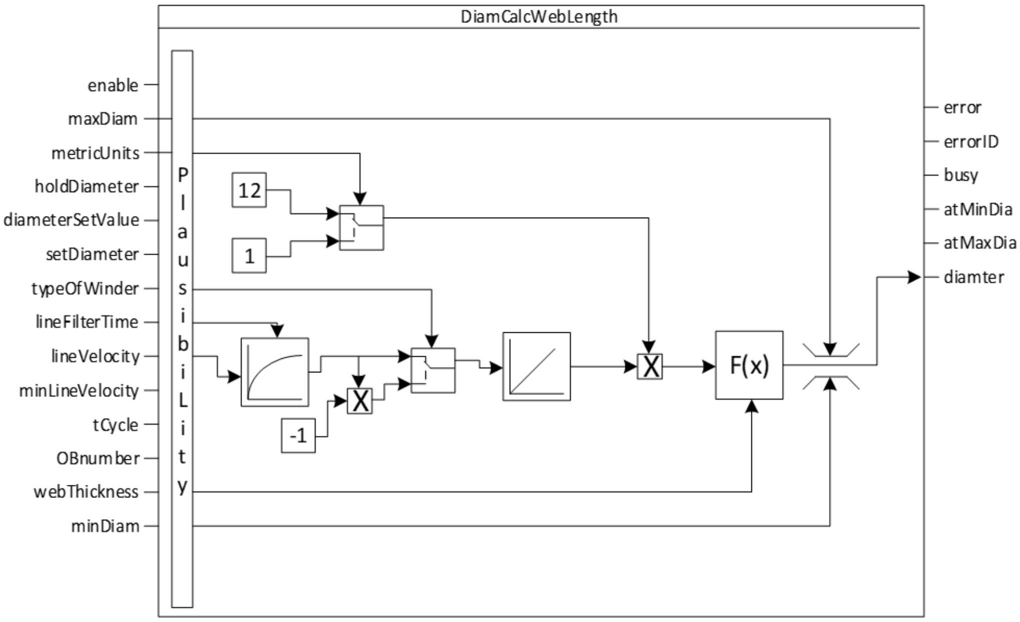
## Web length addition

The diameter calculator determines the actual diameter value of the roll using the following input quantities:

- Web Thickness - the actual web (material) thickness.
- The length of the winded material

With the help of the actual velocity of the leading axis the length of the rewinded or unwinded material on the core is calculated. In addition, with the material thickness the current diameter of the winder can be calculated.

**Note** This calculation method relies very much on the accuracy of the material thickness and the starting diameter value. Any deviation between the actual effective material thickness and the set material thickness will lead to an invalid diameter which error will get bigger after each revolution. An inaccurate starting value will remain as an offset between the actual and the calculated diameter. This calculation mode is mostly useful for materials where air entrapment and in wound tension does not influence the effective material thickness.




**Position mode**

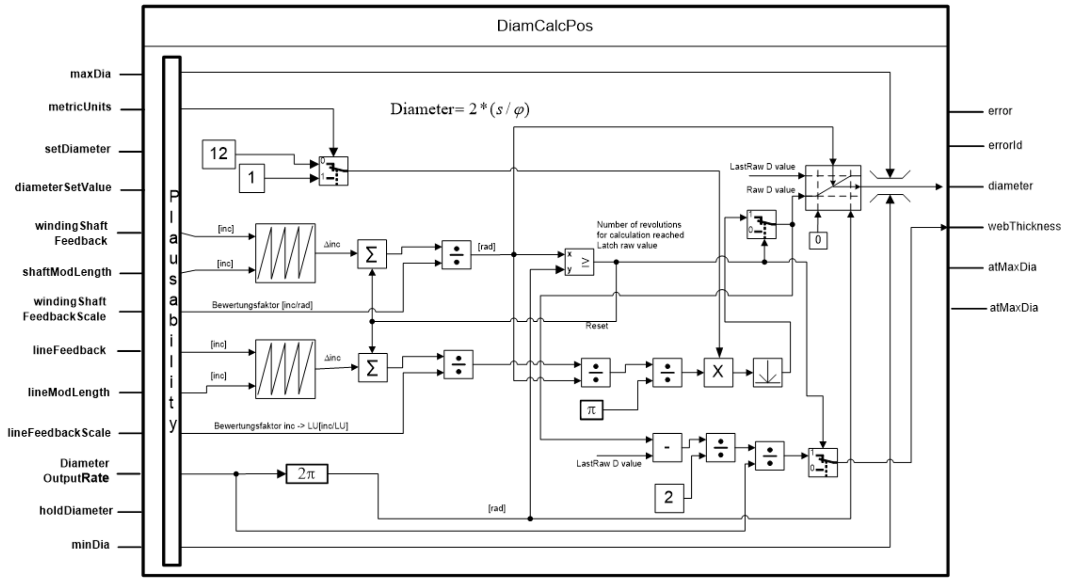
This diameter calculation technique directly intervenes at the position of the web and the roll being wound or unwound. Therefor the position is fed back into the FB via the input parameter lineFeedback and windingShaftFeedback and the difference to the previous cycle is measured. Overflows are intercepted using modulo limits that should be entered.

$$s = R \cdot \varphi, \qquad with\, d = 2 \cdot \frac{s}{\varphi}$$

If the number revolutions of the winding shaft – specified by parameter MeasuringRevolutionCount – are reached the differential value is reset and the calculated diameter value is further processed. In order to be able to identify overflows, the change of the feedback signal per cycle must be less than half of the difference of the modulo limits.

$$\Delta feedback < \frac{moduloLength}{2}$$

In addition, in every measuring interval the actual material thickness (CalculatedWebThickness) is calculated, e.g., for a rest length calculation.




### Floating diameter calculation (Only mode Integral and Position)

To optimize the diameter signal quality a floating averaging over a defined number of rotations (measuringRevolutionCount) is used internally. The floating diameter calculation can be configured by the following configuration parameter:

- MeasuringRevolutionCount
- FirstDiameterOutput
- DiameterOutputRate

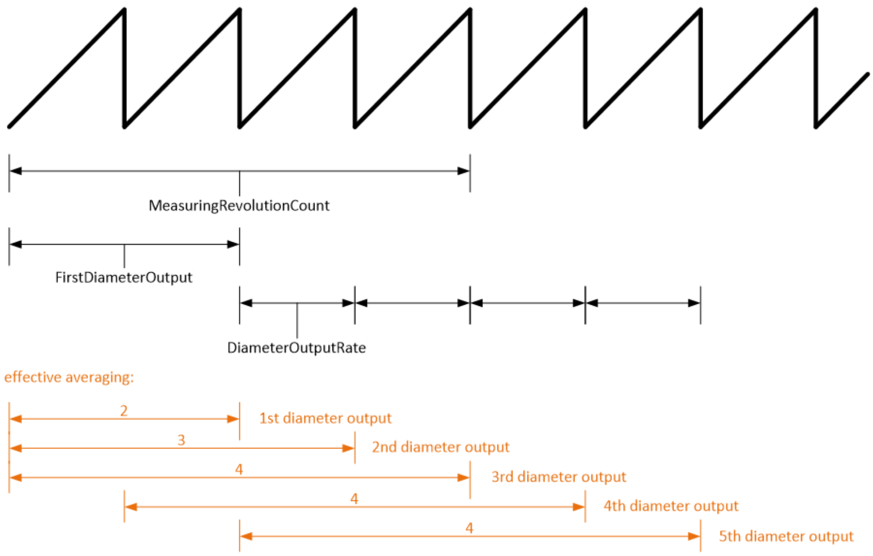
After rising edge of input “enable” the first calculated diameter will be output after the number of revolutions configured in “FirstDiameterOutput”. After the first diameter output the calculated diameter will be refreshed cylic after the number of revolutions configured in “DiameterOutputRate”. The calculated diameter will be averaged over the number of revolutions configured in “MeasuringRevolutionCount”, if the parameterized number of revolutions has been passed already. The averaging of a specific number or revolutions is determined by the depth (array length) of an internal ring buffer. Inside the ring buffer the position deltas of line and winder axis will be entered. The buffer depth is calculated based on the following formula:

$$bufferDepth = \frac{MeasuringRevolutionCount}{DiameterOutputRate}$$

Is the result of the calculation a non-natural number the value will be rounded to next natural number (N = {1, 2,3, …}). Thereby the effective averaging over a specific number of revolutions can differ to the configured value in “MeasuringRevolutionCount” (compare example 3). The following sketches show the interacting of the configuration parameter. In the function block a modulo cycle from 0° to 360° is recreated. The configuration of the technology object is independent of that.

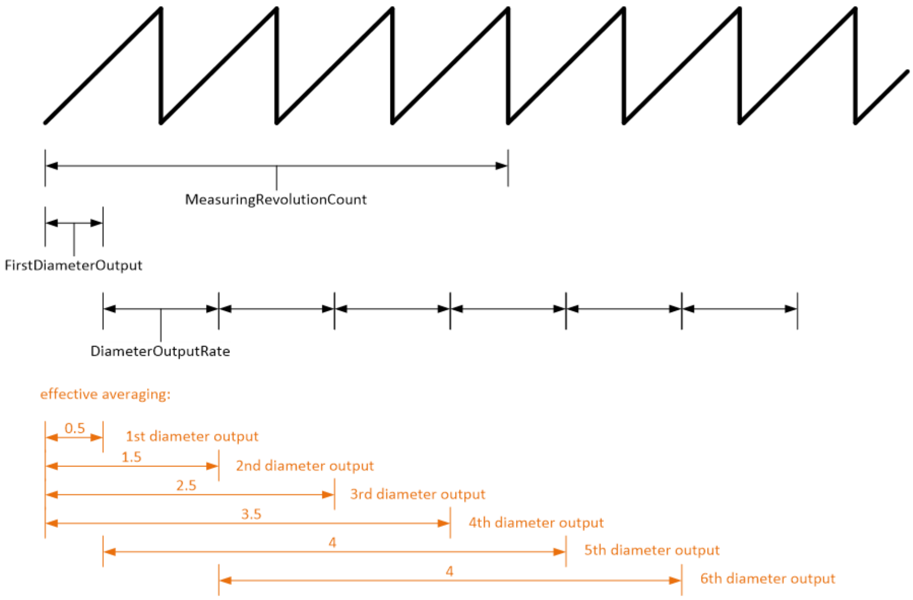
### Example 1

- MeasuringRevolutionCount := 4
- DiameterOutputRate := 1
- FirstDiameterOutput := 2



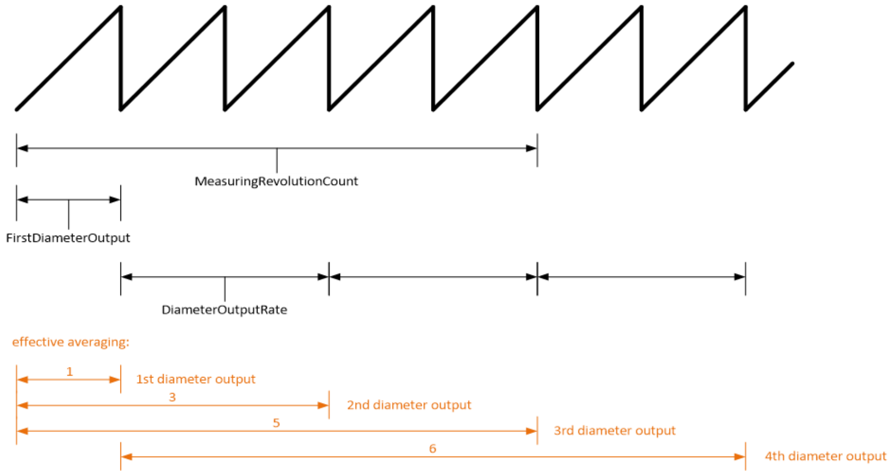

**Example 2**

- MeasuringRevolutionCount := 4
- DiameterOutputRate := 1
- FirstDiameterOutput := 0.5




**Example 3**

- MeasuringRevolutionCount := 5
- DiameterOutputRate := 2
- FirstDiameterOutput := 1




In addition, in every measuring interval based on the diameter change the actual material thickness is also calculated, e.g., for a rest length calculation.

$$Thickness = \frac{dimeterNew - diameterOld}{diameterOutputRate}$$

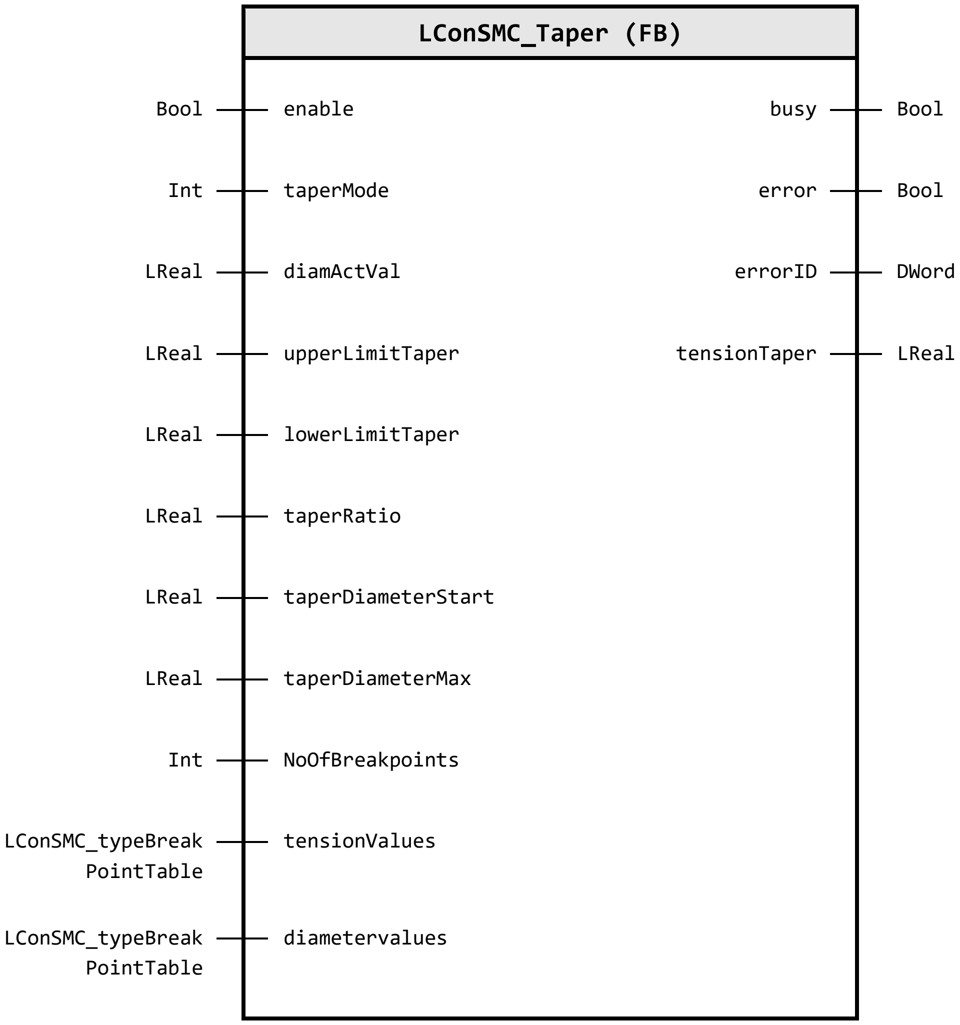
## 5.10.6 LConSMC_Taper (FB)

### Short description

Calculation of a reduction factor for the tension setpoint of a winder as a function of the diameter. The function block can be called in any cyclic task. Function block ID: 4101

### Interface description

### Block interface




### Input parameter

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| enable | Bool | FALSE | TRUE: Enable function block |
| taperMode | Int | 0 | Tension taper characteristik<br>0: No taper<br>1: Linear taper characteristic<br>2:<br>Hyperbolic taper characteristic (max. reduction at target diameter)<br>3: Hyperbolic taper characteristic (max. reduction at infinite diameter)<br>4: Breakpoint table |
| diamActVal | LReal | 0.0 | [m or inch] Actual diameter value |
| upperLimitTaper | LReal | 100.0 | [%] Maximum tension reduction |

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| lowerLimitTaper | LReal | 0.0 | [%] Minimum tension reduction |
| taperRatio | LReal | 0.0 | [%] Maximum tension reduction at "taperDiameterMax" (0-100%) Only used if "taperMode" = 1-3 |
| taperDiameter Start | LReal | 0.2 | [m or inch] Diameter starting value for the tension reduction |
| taperDiameterMax | LReal | 1.0 | [m or inch] Maximum diameter value. Diameter at which the maximum tension reduction is active for "taperMode" = 1 or 2 |
| NoOfBreakpoints | Int | 20 | Number of break points for breakpoint table if "taperMode" = 4 |
| tensionValues | LConSMC_typeBr eak PointTable | --- | Breakpoint table for tension reduction values if "taperMode" = 4.<br>NOTE: The values are interpreted as percent value and will be devided by 100 internally |
| diametervalues | LConSMC_typeBr eak PointTable | --- | Breakpoint table for diameter values if "taperMode" = 4 (Unit must be the same as the input “diamActValue”) |

### Output parameter

| Identifier | Data type | Description |
|---|---|---|
| busy | Bool | TRUE: The function block is active |
| error | Bool | TRUE: An error occured |
| errorID | DWord | Detailed error description |
| tensionTaper | LReal | Multiplicative value for tension reduction (0-1.0). The value must be multiplied with the tension setpoint |



### Functional description

The tension taper characteristic is used to define with which tension the material should be wound onto the roll as a function of the roll diameter. It only makes sense to use the tension taper characteristic while actually rewinding in order to wind the inner layers tightly. For dancer roll controls, the tension taper characteristic can be used to reduce the pressure setpoint for the dancer roll support. In this case, the output of the characteristic must be multiplied with the tension setpoint and made available for the dancer system interface, e.g. analog output. The block is activated using the enable input. Depending on the mode selected at input tensionTaperMode, the tension reduction factor is calculated as a function of the diameter present at diamActVal and output in tensionTaper. The way in which the reduction factor is processed depends on the mode that has been selected.

### Calculation modes

- No Taper (mode = 0)

No tension taper mode is defined, the function block outputs 1.0 (no reduction factor).

- Linear taper characteristic (mode = 1)

From a starting diameter that must be entered (taperDiameterStart), the reduction factor is linearly reduced to the maximum diameter (taperDiameterMax) with a reduction defined by the parameter taperRatio.

  - diamActVal < taperDiameterStart

$$tensionTaper = 1.0$$

  - taperDiameterStart < diamActVal < taperDiameterMax

$$tensionTaper = \left(1.0 - \frac{taperRatio}{100.0} \cdot \frac{diamActVal - taperDiameterStart}{taperDiameterMax - taperDiameterStart}\right)$$

- diamActVal > taperDiameterMax

$$tensionTaper = \left(1.0 - \frac{taperRatio}{100.0}\right)$$

- Hyperbolic taper characteristic 1 (mode = 2)

From a starting diameter that has to be entered (taperDiameterStart) the reduction factor is reduced as hyperbolic function with taperRatio to the maximum diameter (taperDiameterMax). In this case, the reduction is at a maximum when the maximum diameter has been reached.

- diamActVal < taperDiameterStart

$$tensionTaper = 1.0$$

- taperDiameterStart < diamActVal < taperDiameterMax

$$tensionTaper = 1.0 - \frac{taperRatio}{100.0} \cdot \frac{diamActVal - taperDiameterStart}{diamActVal} \cdot \frac{taperDiameterMax}{taperDiameterMax - taperDiameterStart}$$

- diamActVal > taperDiameterMax

$$tensionTaper = \left(1.0 - \frac{taperRatio}{100.0}\right)$$

- Hyperbolic taper characteristic 2 (mode = 3)

From a starting diameter that must be entered (taperDiameterStart) the reduction factor is reduced as hyperbolic function with taperRatio to the maximum diameter (taperDiameterMax). In this case, full reduction is never reached.

- diamActVal < taperDiameterStart

$$tensionTaper = 1.0$$

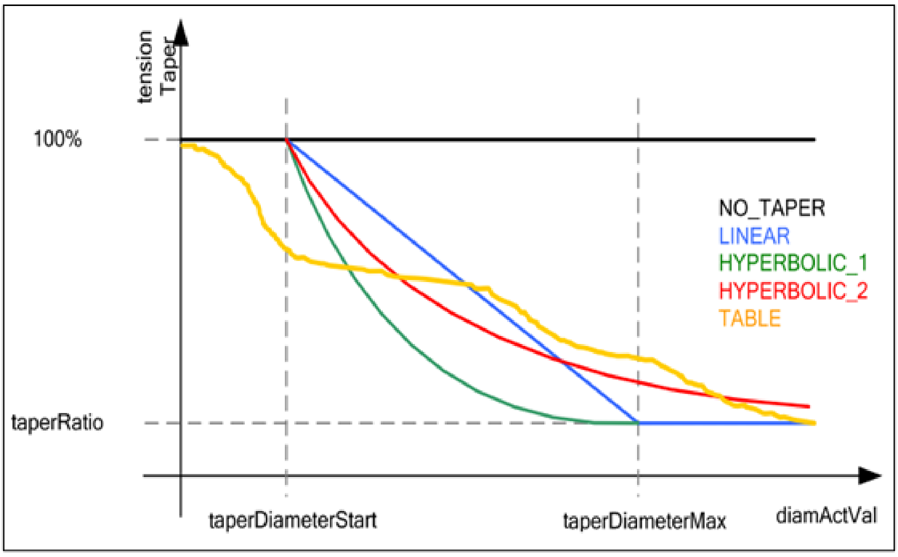
- taperDiameterStart < diamActVal

$$tensionTaper = 1 - \frac{taperRatio}{100} \cdot \left(1 - \frac{taperDiameterStart}{diamActVal}\right)$$

- Breakpoint table (mode = 4)

The reduction factor is determined by characteristic defined as breakpoint table. The breakpoint table consists of pairs of values diameter / reduction factor. The factor is linearly interpolated between the breakpoints of the characteristic.

**Graphical representation**



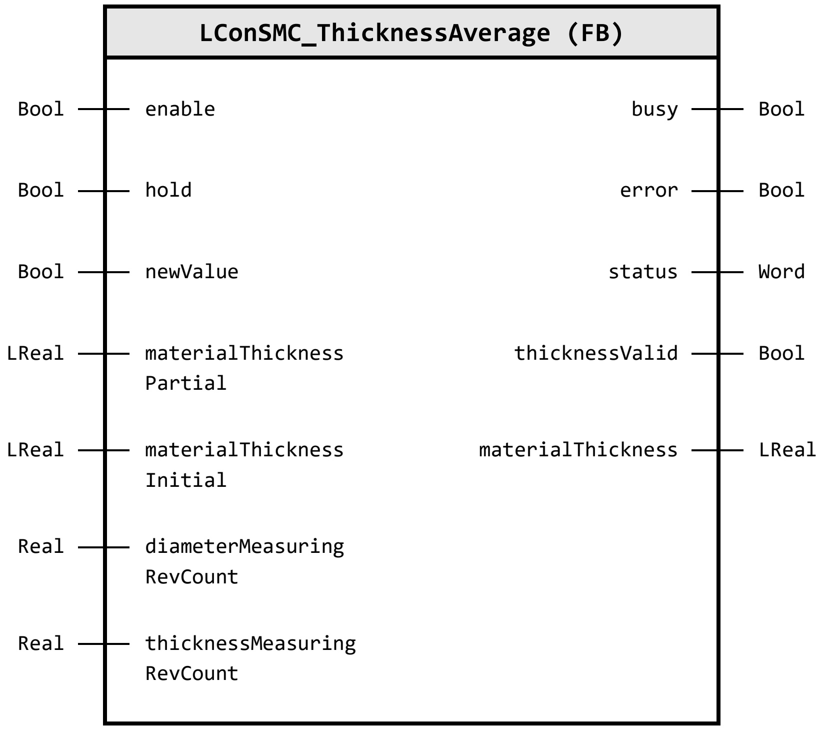
### 5.10.7 LConSMC_ThicknessAverage (FB)

#### Short description

The function block calculates a floating average value based on the cyclically calculated material thickness values of the diameter calculator, to increase the precision and stability of the value.

#### Interface description

#### Block interface




#### Input parameter

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| enable | Bool | FALSE | TRUE: Enable function block: A rising edge resets the block and fills the elements of the ring buffer with "materialThicknessInitial"<br>FALSE: Disable the function block |
| hold | Bool | FALSE | TRUE: Block does not accept any new values, e.g. if the winder is stopped or the machine velocity is changed. |
| newValue | Bool | FALSE | Rising edge: The value in materialThicknessPartial is written into the ring buffer |
| material Thickness Partial | LReal | 0.0 | Actual material thickness calculated by the diameter calculator |
| material Thickness Initial | LReal | 0.0 | Initial value of the material thickness calculation |
| diameter MeasuringRev Count | Real | 0.0 | Number of winder revolutions of the diameter calculation |

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| thickness MeasuringRev Count | Real | 0.0 | Number of winder revolutions of the material thickness calculation |

### Output parameter

| Identifier | Data type | Description |
|---|---|---|
| busy | Bool | Function block is active and no error is present |
| error | Bool | FALSE: No error<br>TRUE: An error has occurred, refer to the status description |
| status | Word | 16#0000 - 16#7FFF: Status of the FB, 16#8000 - 16#FFFF: Error identification |
| thicknessValid | Bool | TRUE: Ring buffer is filled with measured values |
| material Thickness | LReal | Material thickness measured under thicknessMeasuringRevCount revolutions |

### Status & Error codes

| Code / Value | Identifier / Description |
|---|---|
| 16#7000 | STATUS_NO_CALL<br>No job being currently processed |
| 16#7001 | STATUS_FIRST_CALL<br>First call after incoming new job (rising edge 'enable') |
| 16#7002 | STATUS_SUBSEQUENT_CALL<br>Subsequent call during active processing without further details |



### Functional description

The block is used as extension to the LConSMC_Winder block or the LConSMC_DiameterCalculation block.

Both of these blocks offer several calculation modes for calculating the diameter, whereby the DIAM_CALC_INTEGRAL (2) and DIAM_CALC_POSITION (5) position modes, not only calculate the diameter, but also the actual material thickness. This block improves the precision of the material thickness calculation by generating a floating average value (extension of the measurement interval), in order to make the value more suitable - for instance to calculate the remaining rest length.

The resulting material thickness is calculated according to the following formula:

$$materialThickness = \sum_{k} \frac{mt_i}{k}$$

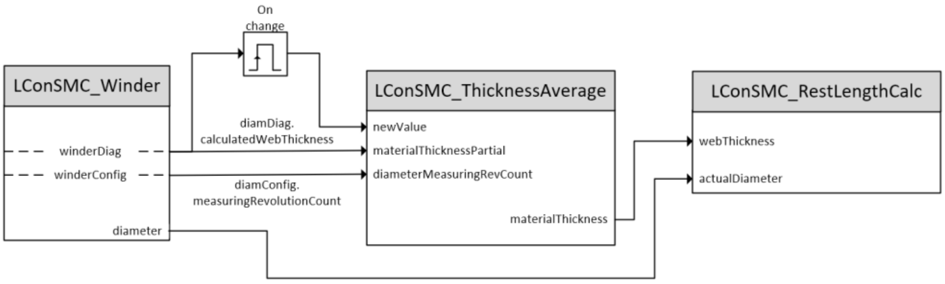
The resulting buffer depth “**k**” is calculated from the ratio between the inputs *diameterMesuringRevCount* (typically connected to winderConfig.diamConfig.measuringRevolutionCount) and *thicknessMeasuringRevCount*. The result is always rounded to the next whole number. **mt$_i$** is the calculated web thickness coming from the winder/diameter function block.

---

Example:

- diameterMesuringRevCount = **3** (The material thickness which is calculated from the winder function block is already averaged over three winder rotations)
- thicknessMeasuringRevCount = **10** (The calculated material thickness should be averaged over 10 rotations)
- buffer depth = 10/3 = 3,33 = **4** (Resulting averaging = 12 winder rotations)

Example connection to calculate the rest length using the averaged material thickness:


On change
LConSMC_Winder
winderDiag
winderConfig
diameter
diamDiag.
calculatedWebThickness
diamConfig.
measuringRevolutionCount
LConSMC_ThicknessAverage
newValue
materialThicknessPartial
diameterMeasuringRevCount
materialThickness
LConSMC_RestLengthCalc
webThickness
actualDiameter

## 5.10.8 LConSMC_WITorquePrecontrol (FB)

### Short description

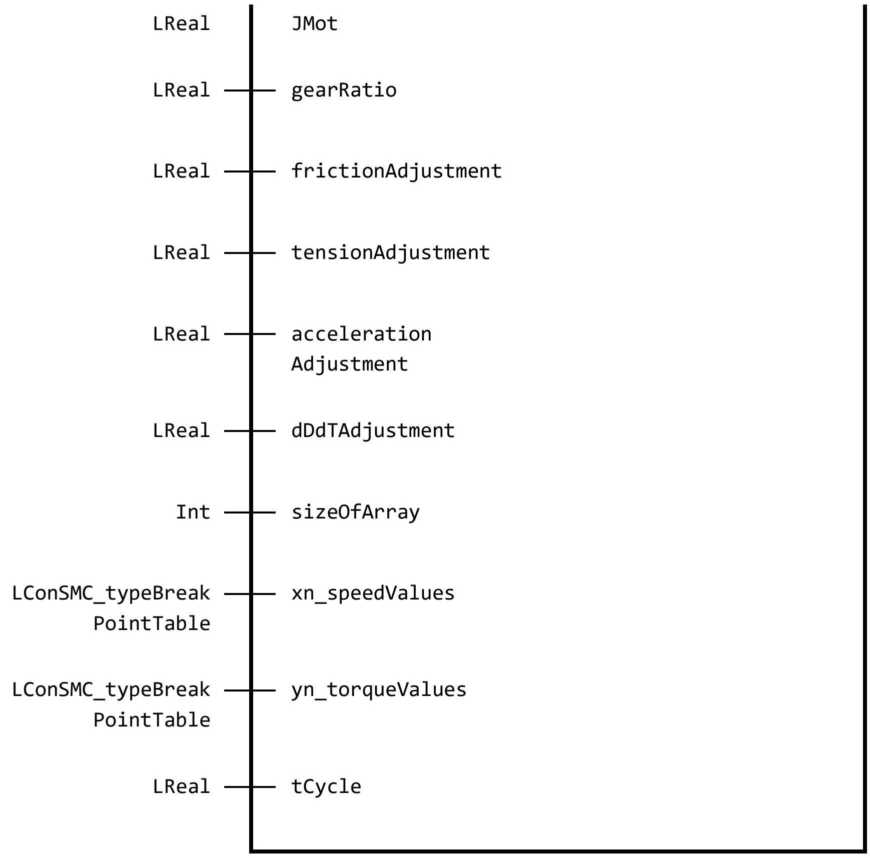
Torque pre-control calculation for winder. The function block has to be called in a cyclic task.
Function block ID: 4102

### Interface description

### Block interface

**LConSMC_WITorquePrecontrol ( FB)**

| Type | Input | Output | Type |
|---|---|---|---|
| Bool | enable | busy | Bool |
| Bool | torqueOn | error | Bool |
| Bool | metricUnits | errorID | DWord |
| Int | typeOfWinder | totalinertia | LReal |
| LReal | drivespeed | mass | LReal |
| LReal | lineSpeed | preCtrlMotor | LReal |
| LReal | lineAcceleration | totalTorquePreCtrl | LReal |
| LReal | diamActVal | variableInertia | LReal |
| LReal | tensionSetpoint | maxInertia | LReal |
| LReal | additiveTorque | tensionTorque | LReal |
| LReal | density | | |
| LReal | diamMin | | |
| LReal | diamMax | | |
| LReal | width | | |
| LReal | fillFactor | | |
| LReal | JFix | | |
| LReal | JMandrel | | |




### Input parameter

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| enable | Bool | FALSE | TRUE: Enable function block |
| torqueOn | Bool | FALSE | TRUE: Calculation of torque and intertia values<br>FALSE: Calculation of intertia values |
| metricUnits | Bool | TRUE | TRUE: Use metric system [m] / [m/min] / [m/s^2] / [N]<br>FALSE: Use angloamerican system [inch] / [ft/min] / [ft/s^2] / [Lbf] |
| typeOfWinder | Int | 0 | Definition of winding direction<br>0: Rewind from above<br>1: Rewind from below<br>2: Unwind from above<br>3: Unwind from below |
| drivespeed | LReal | 0.0 | [RPM] Drive speed for the friction characteristic |
| lineSpeed | LReal | 0.0 | [m/min or ft/min] Web velocity |
| lineAcceleration | LReal | 0.0 | [m/s^2]/[ft/s^2] Web acceleration |
| diamActVal | LReal | 0.0 | [m] / [inch] Diameter of sectional drive |
| tensionSetpoint | LReal | 0.0 | [N] / [lbf] Tension setpoint |
| additiveTorque | LReal | 0.0 | [Nm or Lbf ft] Additive torque value on drive side for rewinder (.e.g. compensation of breakawy torque). The torque is in rewinding directio and only active of "typeOfWinder" = 0/1 |
| density | LReal | 0.0 | [kg/m^3 or lb / ft^3 ] Material density |
| diamMin | LReal | 0.0 | [m or inch] Minimum diameter |
| diamMax | LReal | 0.0 | [m or inch] Maximum diameter |
| width | LReal | 0.0 | [m or inch] Material width |

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| fillFactor | LReal | 0.0 | Material fill factor (1.0-100%)<br>Scaling factor for calculation of the material intertia (e.g. no solid material is winded) |
| JFix | LReal | 0.0 | [kgm^2] / [lbft^2] Inertia on drive side (without drive, e.g. gear box) |
| JMandrel | LReal | 0.0 | [kgm^2] / [lbft^2] Inertia on load side (e.g. mandrel) |
| JMot | LReal | 0.0 | [kgm^2] / [lbft^2] Drive inertia |
| gearRatio | LReal | 0.0 | Gear ratio |
| friction Adjustment | LReal | 0.0 | Multiplicative adaption factor for friction torque (1.0 = 100%) |
| tension Adjustment | LReal | 0.0 | Multiplicative adaption factor for tension torque (1.0 = 100%) |
| acceleration Adjustment | LReal | 1.0 | Multiplicative adaption factor for acceleration torque from web acceleration (1.0 = 100%) |
| dDdTAdjustment | LReal | 0.0 | Multiplicative adaption factor for acceleration torque based on diameter change (1.0 = 100%) |
| sizeOfArray | Int | 20 | Number of breakpoints that should be used for the friction characteristic |
| xn_speedValues | LConSMC_typeBreak PointTable | --- | Numer of breakpoints that should be used from the friction characteristic |
| yn_torqueValues | LConSMC_typeBreak PointTable | --- | Speed breakpoints for friction characteristic |
| tCycle | LReal | 0.0 | [ms] Cycle time the function block is called |



**Output parameter**

| Identifier | Data type | Description |
|---|---|---|
| busy | Bool | TRUE : Function block is executed |
| error | Bool | TRUE: Function block has detected an error |
| errorID | DWord | Detailed error description |
| totalinertia | LReal | [kgm^2] / [lbft^2] Total inertia on the drive side |
| mass | LReal | [kg or lbft] Mass of the material |
| preCtrlMotor | LReal | [Nm] / [Lbf ft] Drive torque from friction and acceleration |
| totalTorquePre Ctrl | LReal | [Nm] / [Lbf ft] Total torque precontrol value on drive side |
| variableInertia | LReal | [kgm^2 or lbft /s^2] Inertia of the material on drive side |
| maxInertia | LReal | [kgm^2 or lbft /s^2] Maximum interia (at maximum diameter "diamMax") |
| tensionTorque | LReal | [Nm] / [Lbf ft] Tension torque on drive side |

**Functional description**

The block determines the following:

- The variable moment of inertia for Kp adaptation of the speed and tension controller
- The pre-controlled torque for Δv/Δt and ΔD/Δt

The torque pre-control outputs the following torque setpoint to pre-control the winder drive:

- A torque setpoint as a function of the acceleration setpoint of the machine/system ramp-function generator and the inertia
- A torque setpoint as a function of the changing diameter and the current line speed. This pre-control is for example needed if thick material wound on a thin sleeve after a flying roll

change. As a result of the fast-changing diameter at constant web velocity high acceleration values can occur.

- Friction torque setpoint is generated depending on the shaft speed of the winder using a characteristic.
- Tension torque based on the tension setpoint and the diameter
- Constant torque offset for rewinders (e.g. to compensate breakaway torque)

To quickly compensate for disturbances and to implement the closed-loop control in the first place (for a replacing speed control), the torque to be provided from the drive must be specified as accurately as possible. Depending on the control type, this is either realized additively after the speed controller or by dynamically adjusting the torque limits. In both cases, starting from the weight of the wound material, the appropriate torque must be calculated using its moment of inertia (wound material) and transferred to the drive.

**Note** The precise setting of the compensation is especially important for indirect tension control. This is so that the torque-generating current emulates, as far as possible, the material tension.



**Calculations**

- Mass of the wound material

$$Mass = \frac{\pi \cdot Density \cdot width \cdot (diamActVal^2 \cdot diamMin^2)}{4}$$

- Moment of inertia of the wound material

$$VariableInertia = Mass \cdot \frac{diamActVal^2 + diamMinVal^2}{8 \cdot GearRatio^2}$$

- Total moment of inertia

$$TotalInertia = \frac{J_{Mandrel}}{GearRatio^2} + VariableInertia + J_{Mot} + J_{Fix}$$

- Maximum mass of the wound material

$$Mass_{Max} = \frac{\pi \cdot Density \cdot width \cdot (diamMax^2 - diamMin^2)}{4}$$

- Maximum moment of inertia of the wound material

$$VariableInertia_{Max} = Mass_{Max} \cdot \frac{diamMax^2 + diamMin^2}{8 \cdot GearRatio^2}$$

- Maximum moment of inertia

$$TotalInertia = \frac{J_{Mandrel}}{GearRatio^2} + VariableInertia + J_{Mot} + J_{Fix}$$

- Angular acceleration of the roll

$$\alpha = 2 \cdot \frac{lineAcceleration \cdot GearRatio}{diamActVal}$$

- Acceleration torque (based on diameter change)

$$M_{dDdT} = \frac{2 \cdot TotalInertia \cdot GearRatio \cdot lineSpeed}{\frac{tCycle}{1000}} \cdot \left(\frac{1}{diamAct_n} - \frac{1}{diamAct_{n-1}}\right) \cdot dDdtAdjustment$$

- Acceleration torque

$$M_{accel} = TotalInertia \cdot \alpha \cdot accelerationAdjustment$$

- Friction torque

$$M_{fric} = f(n) \cdot frictionAdjustment$$

- Tension torque

$$M_{tension} = \frac{tensionSetpoint \cdot tensionAdjustment \cdot diamActVal}{2 \cdot GearRatio}$$

This friction torque is internally determined using the function block LConSMC_TableChar. The user is responsible for providing valid and sensible values in the appropriate data fields. If the factor FrictionAdjustment is set to 0.0, LConSMC_TableChar is not called and therefore a frictional torque is not added. The pre-control torque of the motor, the sum of all pre-control torques and the tension torque are calculated depending on the winder type. The total torque is always the sum of the preCtrlMotor and tensionTorque. The winding mode is always defined assuming a positive line speed. If the leading value becomes negative the winding mode will be inverted automatically (a rewinder becomes an unwinder and vice versa). The same applies for winding from above and from below. This inversion is automatic.

**Note** The optional additive torque value (e.g. to compensate a breakaway torque for rewinders) is directly effective on the drive side always in rewinding direction

**Winding modes**

- Rewind from above

$$M_{preCtrlMotor} = M_{acc} - M_{dDdt} + M_{fric} + M_{add} \qquad M_{TensionTorque} = M_{TensionTorque}$$

- Rewind from below

$$M_{preCtrlMotor} = -M_{acc} + M_{dDdt} + M_{fric} - M_{add} \qquad M_{TensionTorque} = -M_{TensionTorque}$$

- Unwind from above

$$M_{preCtrlMotor} = M_{acc} + M_{dDdt} + M_{fric} \qquad M_{TensionTorque} = -M_{TensionTorque}$$

- Unwind from below

$$M_{preCtrlMotor} = -M_{acc} - M_{dDdt} + M_{fric} \qquad M_{TensionTorque} = M_{TensionTorque}$$

# 6 PLC data types

## 6.1 LConSMC_typeFrictionMeasurement_V2 (UDT)

### Description

In-/Outputs of the friction measurement

### Parameter description

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| in | LConSMC_type Friction MeasurementV2In | --- | Inputs of the FB FrictionMeasurementV2 |
| out | LConSMC_type Friction MeasurementV2O ut | --- | Outputs of the FB FrictionMeasurementV2 |

## 6.2 LConSMC_typeKpAdaption (UDT)

### Description

In-/Outputs of the speed controller Kp-Adaption

### Parameter description

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| in | LConSMC_typeKp AdaptionInput | --- | Inputs for FB KpAdaption |
| out | LConSMC_typeKp AdaptionOutput | --- | Outputs for FB KpAdaption |

## 6.3 LConSMC_typeSectionalDriveConfig (UDT)

### Description

Configuration data of the sectional drive

### Parameter description

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| position Controlled | Bool | FALSE | TRUE: Enable MoveVelocity position controlled in case of positioning axis |
| axisCtrl | Bool | TRUE | FALSE: No axis control just calculated setpoints<br>TRUE: Use MoveVelocity to forward setpoints to axis |
| jogVelocity | Real | 0.0 | [m/min oder ft/min] Web velocity to jog the drive via the inputs "jogPos" and "jogNeg" |
| diameter | LReal | 1.0 | [m or inch] Diameter of sectional drive axis |
| overrideVelocity | Real | 10.0 | [m/min or ft/min] Additive velocity offset for speed controller<br>This value is only needed for the torque controlled modes (controlMode = 1-3) |
| maxVelocity | Real | 0.0 | [m/min or ft/min] Maximum possible web velocity. Maximum speed setpoint of the axis is limited to maxVelocity (+ overrideVelocity) and minimum diameter |

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| getAxisConfigAt Start | Bool | TRUE | TRUE: Readout gear,reference torque and units from axis with rising edge. FALSE: Take config values |
| noOfSamplesLine Velocity | UInt | 1 | Number of samples for moving average filter of line velocity setpoint for synchronizing to a moving web |
| unit Configuration | LConSMC_typeU nit Configuration | --- | Configuration of the units |
| SectionalDrive Dynamics | LConSMC_typeAx is Dynamics | --- | Configuration of the axis dynamics |
| technology Controller Config | LConSMC_type Technology ControllerConfig | --- | Configuration parameters for technology controller (Dancer or load cell) |
| torque Precontrol Config | LConSMC_type SectionalDrive TorquePrecontrol Config | --- | Configuration parameters for torque precontrol |
| webbreak DetectionConfig | LConSMC_type WebbreakDetect Config | --- | Configuration parameters for webbreak detection |

## 6.4 LConSMC_typeSectionalDriveDiag (UDT)



### Description

Diagnostic data of the sectional drive

### Parameter description

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| typeOfAxis | Int | 0 | Type of axis<br>0: Invalid axis<br>1: Speed axis<br>2: Positioning axis<br>3: Synchronous axis |
| reference Velocity | LReal | 0.0 | [TO unit] Reference velocity which can be used for an external axis control (sectionalDriveConfig.axisCtrl = FALSE). Represents the maximum axis velocity at maximum line velocity |
| diameterFactor | LReal | 0.0 | [%] Scaling factor for parameter "sectionalDriveDiag.referenceVelocity" which can be used for an external axis control (sectionalDriveConfig.axisCtrl = FALSE). Velocity setpoint = referenceVelocity * diameterFactor/100.0. The diameterFactor can directly be copied to the velocityOverride parameter of the TO (<TO>.Override.Velocity) |
| torquePre ControlDiag | LConSMC_type TorquePrecontrol Diag | --- | Diagnostic structure for torque precontrol |
| technology ControllerDiag | LConSMC_type Technollogy ControllerDiag | --- | Diagnostic structure for technology controller |
| webbreakDiag | LConSMC_type WebbreakDetect Diag | --- | Diagnostic structre for webbreak detection |

## 6.5 LConSMC_typeTaper (UDT)

### Description

In-/Outputs of the tension taper

### Parameter description

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| in | LConSMC_typeTaper Input | --- | Inputs for FB Taper |
| out | LConSMC_typeTaper Output | --- | Outputs for FB Taper |

## 6.6 LConSMC_typeWinderConfig (UDT)

### Description

Configuration data of the winder

### Parameter description

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| position Controlled | Bool | FALSE | TRUE: Enable MoveVelocity position controlled in case of positioning axis |
| axisCtrl | Bool | TRUE | FALSE: Setpoints (Velocity, Torque) are not forwarded to the axis. An external axis control must be realized (see diagnostic outputs "referenceVelocity" and "diameterFactor")<br>TRUE: Setpoints (Velocity, Torque) are forwarded to the axis internally |
| jogVelocity | Real | 0.0 | [m/min oder ft/min] Web velocity to jog the drive via the inputs "jogPos" and "jogNeg" |
| overrideVelocity | Real | 0.0 | [m/min or ft/min] Additive velocity offset for speed controller<br>This value is only needed for the torque controlled modes (controlMode = 2&3) |
| maxVelocity | Real | 0.0 | [m/min or ft/min] Maximum possible web velocity. Maximum speed setpoint of the axis is limited to maxVelocity (+ overrideVelocity) and minimum diameter |
| getAxisConfigAt Start | Bool | TRUE | TRUE: Readout gear,reference torque and units from axis with rising edge. FALSE: Take config values |
| noOfSamplesLine Velocity | UInt | 1 | Number of samples for moving average filter of line velocity setpoint for synchronizing to a moving web |
| unit Configuration | LConSMC_typeUnit Configuration | --- | Configuration of the units |
| winderDynamics | LConSMC_typeAxis Dynamics | --- | Configuration of the axis dynamics |
| torque Precontrol Config | LConSMC_type TorquePrecontrol Config | --- | Configuration parameters for torque precontrol |
| diamConfig | LConSMC_typeDiam Config | --- | Configuration parameters for diameter calculation |

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| technology Controller Config | LConSMC_type Technology ControllerConfig | --- | Configuration parameters for technology controller (Dancer or load cell) |
| webbreak DetectionConfig | LConSMC_type WebbreakDetect Config | --- | Configuration parameters for webbreak detection |

## 6.7 LConSMC_typeWinderDiag (UDT)

### Description

Diagnostic data of the winder

### Parameter description

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| typeOfAxis | Int | 0 | Type of axis<br>0: Invalid axis<br>1: Speed axis<br>2: Positioning axis<br>3: Synchronous axis |
| reference Velocity | LReal | 0.0 | [TO unit] Reference velocity which can be used for an external axis control (winderConfig.axisCtrl = FALSE). Represents the maximum axis velocity at maximum line velocity and minimum diameter |
| diameterFactor | LReal | 0.0 | [%] Scaling factor for parameter "winderDiag.referenceVelocity" which can be used for an external axis control (winderConfig.axisCtrl = FALSE). Velocity setpoint = referenceVelocity * diameterFactor/100.0. The diameterFactor can directly be copied to the velocityOverride parameter of the TO (<TO>.Override.Velocity) |
| diamCalcDiag | LConSMC_typeDiam CalcDiag | --- | Diagnostic structure for diameter calculation |
| torquePre ControlDiag | LConSMC_type TorquePrecontrol Diag | --- | Diagnostic structure for torque precontrol |
| technology ControllerDiag | LConSMC_type Technollogy ControllerDiag | --- | Diagnostic structure for technology controller |
| webbreakDiag | LConSMC_type WebbreakDetect Diag | --- | Diagnostic structre for webbreak detection |

## 6.8 Additional

### 6.8.1 LConSMC_typeAxisDynamics (UDT)

#### Description

Dynamic seetings for different operation modes

#### Parameter description

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| stopDynamics | LConSMC_typeSt op Dynamics | --- | [Unit of the axis] Dynamics for stopping the axis<br>Active if "enable" = FALSE or "speedMatch" = FLASE and running web |
| syncDynamics | LConSMC_typeSy nc Dynamics | --- | [Unit of the axis] Dynamics for synchronizing to line speed<br>Active if "enable" & "speedMatch" = TRUE |
| operation Dynamics | LConSMC_type OperationDynamic s | --- | [Unit of the axis] Dynamics during normal operation<br>Active if "enable" & "speedMatch" & "ctrlEnable" = TRUE |
| jogDynamics | LConSMC_typeJo g Dynamics | --- | [Unit of the axis] Dynamics for jogging the axis<br>Active if "JogPos" or "jogNeg" = TRUE |



### 6.8.2 LConSMC_typeDiamCalcDiag (UDT)

#### Description

Diagnostic strukture of diameter calculation

#### Parameter description

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| atMinDia | Bool | FALSE | TRUE: Diameter calculation has reached minimum diameter |
| atMaxDia | Bool | FALSE | TRUE: Diameter calculation has reached maximum diameter |
| diamIsHold | Bool | FALSE | TRUE: Diameter calculation is hold:<br>line velocity < diamConfig.minLineVelo<br>shaftSpeed < diamConfig.minShaftSpeed<br>input "holdDiameter" = TRUE |
| calculatedWeb Thickness | LReal | 0.0 | Calculated web thickness<br>Only relevant for calculation modes integral (2) and position (5) |

### 6.8.3 LConSMC_typeDiamConfig (UDT)

#### Description

Configuration structure of diameter calculation

#### Parameter description

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| minLineVelocity | LReal | 0.0 | [m/min or ft/min] Minimum web speed to activate the diametet calculation |
| minShaftSpeed | LReal | 0.0 | [min-1] Minimum winder shaft speed to activate diameter calculation |
| minDia | LReal | 0.0 | [m or inch] Minimum diameter value |

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| maxDia | LReal | 0.0 | [m or inch] Maximum diameter value |
| webThickness | LReal | 0.0 | [m or inch] Web thickness<br>Necessary for the calculation modes 1&4 |
| useLineAxis ActualValues | Bool | TRUE | TRUE: Use line actual values (s,v) for the diameter calculation from winder input "lineAxisActualValues"<br>FALSE: Use values from the winder input "lineAxisMotionVector" |
| diameterSetValue | LReal | 0.0 | [m or inch] For diameter calculation mode 0,1,2,4,5: Diameter set value (If winder input "setDiameter" = TRUE)<br>For diameter calculation mode 3 (External): External diameter sensor value (input "setDiameter" is not relevant) |
| lineFilterTime | Real | 0.0 | [ms] PT1 filter time for line velocity |
| shaftFilterTime | Real | 0.0 | [ms] PT1 filter time for winder speed |
| measuring RevolutionCount | Real | 4.0 | Number of shaft revolutions used for averaging the diameter<br>Only relevant for calculation modes integral (2) and position (5) |
| firstDiameter Output | Real | 0.5 | Number of shaft revolutions after which the first diameter value is calculated<br>Only relevant for calculation modes integral (2) and position (5) |
| diameterOutput Rate | Real | 1.0 | Number of shaft revolutions after the first diameter after which a new diameter value is calculated<br>Only relevant for calculation modes integral (2) and position (5) |
| lineModLength | Real | 0.0 | Modulo length of the line axis. 0 = No modulo axis<br>Only relevant for calculation mode position (5) |
| shaftModLength | Real | 0.0 | Modulo length of the winder axis. 0 = No modulo axis<br>Only relevant for calculation mode position (5) |
| diameterFilter Time | Real | 0.0 | [ms] PT1 filter time for diameter output value |
| integration Direction | UInt | 0 | Limit integration direction of diameter calculation for modes Integral(2) and Position(5):<br>0: Both directions<br>1: Only increasing diameter values<br>2: Only decreasing diameter values |

### 6.8.4 LConSMC_typeJogDynamics (UDT)

**Description**

Jog dynamics

**Parameter description**

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| acceleration | Real | 0.2 | Acceleration value: Units are taken from the TO |
| deceleration | Real | 0.2 | Deceleration value: Units are taken from the TO |
| jerk | Real | 0.4 | Jerk value : Units are taken from the TO |

### 6.8.5 LConSMC_typeLineAxisActValues (UDT)

#### Description

Actuak values of the web

#### Parameter description

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| s | LReal | 0.0 | [m] / [ft] Position |
| v | LReal | 0.0 | [m/min] / [ft/min] Velocity |

### 6.8.6 LConSMC_typeOperationDynamics (UDT)

#### Description

[Unit of the axis] Dynamics during normal operation  Active if “enable” & “speedMatch” & “ctrlEnable” = TRUE

#### Parameter description

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| acceleration | Real | -1.0 | Acceleration value: Units are taken from the TO |
| deceleration | Real | -1.0 | Deceleration value: Units are taken from the TO |
| jerk | Real | -1.0 | Jerk value : Units are taken from the TO |



### 6.8.7 LConSMC_typeSectionalDriveTorquePrecontrolConfig (UDT)

#### Description

Configuration data of torque precontrol for sectional drives

#### Parameter description

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| useSetValFor FrictionComp | Bool | FALSE | TRUE: Use speed setpoint for friction compensation FALSE: Use speed actual value |
| friction Adjustment | Real | 0.0 | Multiplicative adaption factor for friction torque (1.0 = 100%) |
| xn_speedValues | LConSMC_typeBr eak PointTable | --- | [1/min] Speed values for fiction characteristic |
| yn_torqueValues | LConSMC_typeBr eak PointTable | --- | [Nm or lbf ft] Torque values for friction characteristic |
| sizeOfArray | Int | 20 | Size of friction characteristic array |
| loadGear | LConSMC_typeLo ad Gear | --- | Load gear settings |
| JFixed | LReal | 0.0 | [kgm^2] / [lbft^2] Inertia on drive side (without drive, e.g. gear box) |
| JMandrel | LReal | 0.0 | [kgm^2] / [lbft^2] Inertia on load side (e.g. mandrel) |
| JMotor | LReal | 0.0 | [kgm^2] / [lbft^2] Drive inertia |
| tension Adjustment | Real | 0.0 | Multiplicative adaption factor for tension torque (1.0 = 100%) |
| acceleration Adjustment | Real | 1.0 | Multiplicative adaption factor for acceleration torque from web acceleration (1.0 = 100%) |

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| upperTorqueLimit | LReal | 1.0 | [%] 1.0 = 100%<br>Upper torque limit<br>0..200% of nominal torque |
| lowerTorqueLimit | LReal | -1.0 | [%] 1.0 = 100%<br>Lower torque limit (0..-200% of nominal torque) |
| referenceTorque | LReal | 0.0 | Reference torque (SINAMICS p2003)<br>!Only necessary if "getAxisConfigAtStart" = FALSE! |

### 6.8.8 LConSMC_typeStopDynamics (UDT)

**Description**

[Unit of the axis] Dynamics for stopping the axis  Active if “enable” = FALSE or “speedMatch” = FLASE and running web

**Parameter description**

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| deceleration | Real | -1.0 | Deceleration value: Units are taken from the TO |
| jerk | Real | -1.0 | Jerk value : Units are taken from the TO |
| abort<br>Acceleration | Bool | FALSE | TRUE: Abort acceleration immediately, FALSE:<br>Reduce current acceleration via Jerk |



### 6.8.9 LConSMC_typeSyncDynamics (UDT)

**Description**

[Unit of the axis] Dynamics for synchronizing to line speed  Active if “enable” & “speedMatch” = TRUE

**Parameter description**

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| acceleration | Real | -1.0 | Acceleration value: Units are taken from the TO |
| deceleration | Real | -1.0 | Deceleration value: Units are taken from the TO |
| jerk | Real | -1.0 | Jerk value : Units are taken from the TO |
| abort<br>Acceleration | Bool | FALSE | TRUE: Abort acceleration immediately, FALSE:<br>Reduce current acceleration via Jerk |
| syncTolerance<br>Factor | Real | 1.0 | Factor to adjust synchronization tolerance |

### 6.8.10 LConSMC_typeTechnollogyControllerDiag (UDT)

**Description**

Diagnostic data of the technology controller

**Parameter description**

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| pidoutValue | LReal | 0.0 | PID-controller output value:<br>Torque control [N or Lbf]<br>Velocity adaption [m/min or ft/min] |
| pidEffectiveKp | LReal | 0.0 | Effective P-gain if the gain adaption is used |
| pidUpperLimit<br>Reached | Bool | FALSE | TRUE: PID output<br>has reached the upper limit |

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| pidLowerLimit Reached | Bool | FALSE | TRUE: PID output<br>has reached the upper limit |
| pidRFGSetpoint Reached | Bool | FALSE | TRUE: The internal ramp generator has reached the setpoint |

### 6.8.11 LConSMC_typeTechnologyControllerConfig (UDT)

#### Description

Configuration parameters for technology controller (Dancer or load cell)

#### Parameter description

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| x1 | LReal | 0.0 | Breakpoint 1 x-axis for P-adaption (if x1=x2 the adaption is deactivated) |
| Kp1 | LReal | 1.0 | Breakpoint 1 P-gain for P-adaption (if x1=x2 the adaption is deactivated) |
| x2 | LReal | 0.0 | Breakpoint 2 x-axis for P-adaption (if x1=x2 the adaption is deactivated) |
| Kp2 | LReal | 1.0 | Breakpoint 2 P-gain for P-adaption (if x1=x2 the adaption is deactivated) |
| KpAdaptionValue | LReal | 0.0 | Adaption value for P-Gain adaption of the controller |
| integTime | LReal | 2000.0 | [ms] Time constant for integrator. A value of 0 disables the I-part |
| derivTime | LReal | 0.0 | [ms] Time constant for differentiator. A value of 0 disables the D-part |
| derivControlMode | Bool | FALSE | TRUE: D component active in actual value branch<br>FALSE: D component in controller deviation |
| setNominalValue | LReal | 100.0 | Reference value for ramp time of setpoint channel (Ramp from 0 to "setNominalValue" in "setRampTime") |
| setRampTime | LReal | 1000.0 | [ms] Ramp time for setpoint ramp generator |
| filterTime | LReal | 10.0 | [ms] PT1- Filter for actual value. A value of 0 disables the filter |
| intSetValue | LReal | 0.0 | Integrator set value (if "setIntChannel" = TRUE) |
| setIntChannel | Bool | FALSE | TRUE: The output of the integrator is set to "intSetValue" |
| holdIntChannel | Bool | FALSE | TRUE: The output of the integrator is hold<br>FALSE: The output is integrated |
| deadBand | LReal | 0.0 | Value of the dead band (symmetrical). A value of 0 disables the deadband |
| outValueFactor | LReal | 1.0 | Multiplicative scaling of the controller output |
| upperLimit | LReal | 20.0 | Upper limit of the controller output |
| lowerLimit | LReal | -20.0 | Lower limit of the controller output |
| limRampTime | LReal | 0.0 | [ms] Ramp time for output value limitation |
| tensionReference | LReal | 0.0 | [N or lbf] Reference value for tension setpoint<br>If the value is 0 the setpoint is interpreted as an abolute value and not percentage |

### 6.8.12 LConSMC_typeTorquePrecontrolConfig (UDT)

**Description**

Configuration parameters for technology controller (Dancer or load cell)

**Parameter description**

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| useSetValFor FrictionComp | Bool | FALSE | TRUE: Use speed setpoint for friction compensation<br>FALSE: Use speed actual value |
| friction Adjustment | Real | 0.0 | Multiplicative adaption factor for friction torque (1.0 = 100%) |
| xn_speedValues | LConSMC_typeBreak PointTable | --- | [1/min] Speed values for fiction characteristic |
| yn_torqueValues | LConSMC_typeBreak PointTable | --- | [Nm or lbf ft] Torque values for friction characteristic |
| sizeOfArray | Int | 20 | Size of friction characteristic array |
| loadGear | LConSMC_typeLoad Gear | --- | Load gear settings |
| MaterialWidth | LReal | 0.0 | [m or inch] Width of the web |
| MaterialDensity | LReal | 0.0 | [kg/m^3 or lb / ft^3 ] Material density |
| MaterialFill Factor | Real | 1.0 | Material fill factor (1.0-100%)<br>Scaling factor for calculation of the material intertia (e.g. no solid material is winded) |
| JFixed | LReal | 0.0 | [kgm^2] / [lbft^2] Inertia on drive side (without drive, e.g. gear box) |
| JMandrel | LReal | 0.0 | [kgm^2] / [lbft^2] Inertia on load side (e.g. mandrel) |
| JMotor | LReal | 0.0 | [kgm^2] / [lbft^2] Drive inertia |
| tension Adjustment | Real | 0.0 | Multiplicative adaption factor for tension torque (1.0 = 100%) |
| acceleration Adjustment | Real | 1.0 | Multiplicative adaption factor for acceleration torque from web acceleration (1.0 = 100%) |
| dDdTAdjustment | Real | 0.0 | Multiplicative adaption factor for acceleration torque based on diameter change (1.0 = 100%) |
| upperTorqueLimit | LReal | 1.0 | [%] 1.0 = 100%<br>Upper torque limit<br>0..200% of nominal torque |
| lowerTorqueLimit | LReal | -1.0 | [%] 1.0 = 100%<br>Lower torque limit (0..-200% of nominal torque) |
| referenceTorque | LReal | 0.0 | Reference torque (SINAMICS p2003)<br>!Only necessary if "getAxisConfigAtStart" = FALSE! |
| additiveTorque | LReal | 0.0 | [Nm or Lbf ft] Additive torque value on drive side for rewinder (.e.g. compensation of breakawy torque). The torque is in rewinding directio and only active of "typeOfWinder" = 0/1 |

### 6.8.13 LConSMC_typeTorquePrecontrolDiag (UDT)

**Description**

Diagnostic structure for torque precontrol

**Parameter description**

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| upperTorqueLimit | LReal | 0.0 | [Unit of the axis] Upper torque limit |
| lowerTorqueLimit | LReal | 0.0 | [Unit of the axis] Lower torque limit |
| totalTorquePre Control | LReal | 0.0 | [Unit of the axis] Total torque precontrol value |
| totalInertia | LReal | 0.0 | [kgm^2 or lbft^2] Current intertia of the winder |
| maxInertia | LReal | 0.0 | [kgm^2 or lbft^2] Maximum inertia at maximum diameter |
| referenceTorque | LReal | 0.0 | Used reference torque value |
| loadGear | LConSMC_typeLo ad Gear | --- | Gear ratio read out |

### 6.8.14 LConSMC_typeUnitConfiguration (UDT)

**Description**

Configuration of the units

**Parameter description**

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| metricUnits | Bool | TRUE | TRUE: Use metric system [m] / [m/min] / [m/s^2] / [N]<br>FALSE: Use angloamerican system [inch] / [ft/min] / [ft/s^2] / [Lbf] |
| axisVelocityUnit | UDInt | 1083 | Velocity unit of TO (see TO.Units.VelocityUnit)<br>!Only necessary if "getAxisConfigAtStart" = FALSE! |
| axisTorqueUnit | UDInt | 1126 | Torque unit of TO<br>(see TO.Units.TorqueUnit) !Only necessary if "getAxisConfigAtStart" = FALSE! |

### 6.8.15 LConSMC_typeWebbreakDetectConfig (UDT)

**Description**

Configuration parameters for webbreak detection

**Parameter description**

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| enableDetection | Bool | FALSE | TRUE: Enable webbreak detection |
| minLineVelocity | LReal | 0.0 | [m/min or ft/min] Minimum line speed for detection to be active |
| upperLimitError | LReal | 0.0 | [Unit of tension/dancer feedback] Upper limit for detecting web break<br>!In the control mode "Indirect tension control" the torque deviation in relation to the reference torque in precent is monitored! |

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| lowerLimitError | LReal | 0.0 | [Unit of tension/dancer feedback] Lower limit for detecting web break<br>!In the control mode "Indirect tension control" the torque deviation in relation to the reference torque in precent is monitored! |
| delayTime | Time | T#0s | Delay time before diagnostic output "webBreak" is set to TRUE |

### 6.8.16 LConSMC_typeWebbreakDetectDiag (UDT)

#### Description

Diagnostic structre for webbreak detection

#### Parameter description

| Identifier | Data type | Default value | Description |
|---|---|---|---|
| webbreak DetectionActive | Bool | FALSE | TRUE: Web break detection is active |
| webBreak | Bool | FALSE | TRUE: A web break has been detected. The PID controlled will be disabled |

# 7 PLC tags & constants

## 7.1 LConSMC_Configuration

Constant identifier, values and description

| Identifier & Value | Description |
|---|---|
| **LCON_THICKNESS_MAX_SAMPLES**<br>50 | [-] Maximum number of samples used for averaging the material thickness |
| **LCON_VELO_MONITORING_ADJUSTMENT_PERCENT**<br>20.0 | [%] Maximum velocity deviation to detect drive runaway in torque controlled modes. Value is a percentage value of the active velocity override |

## 7.2 LConSMC_Errors

Status & Error codes

| Code / Value | Identifier / Description |
|---|---|
| 16#0005_8000 | LCON_ERROR_CALC_STOP_DISTANCE_INVALID_CONFIGURATION<br>Invalid configuration data for CalcStopDistance:<br>* maxVelocity = 0.0<br>* rampDownTime = 0.0<br>* (roundingTime <> 0.0 and rampUpTime = 0.0) |
| 16#0105_8000 | LCON_ERROR_AVARAGE_INVALID_CONFIGURATION<br>Invalid configuration data for AverageValue calculation:<br>* sizeOfArray < 2<br>* sizeOfArray > 20 |
| 16#0106_8000 | LCON_ERROR_TABLE_CHAR_INVALID_CONFIGURATION<br>Invalid configuration data for TableChar:<br>* sizeOfArray < 2<br>* sizeOfArray > 20 |
| 16#0106_8007 | LCON_ERROR_TABLE_CHAR_ENTRIES_NOT_MONOTONIC<br>The table entries are not monotonic increasing/decreasing |
| 16#1101_8001 | LCON_ERROR_PT1_INVALID_CYCLE_TIME<br>Function block is not called in a cylic OB (input tCycle <= 0.0) |
| 16#1102_8001 | LCON_ERROR_INTEGRATOR_INVALID_CYCLE_TIME<br>Function block is not called in a cylic OB (input tCycle <= 0.0) |
| 16#1103_8001 | LCON_ERROR_DIFFERENTIATOR_INVALID_CYCLE_TIME<br>Function block is not called in a cylic OB (input tCycle <= 0.0) |
| 16#1104_8001 | LCON_ERROR_DT1_INVALID_CYCLE_TIME<br>Function block is not called in a cylic OB (input tCycle <= 0.0) |
| 16#1105_8000 | LCON_ERROR_PID_INVALID_CONFIGURATION<br>Invalid configuration data for PID controller:<br>* lowerLimit > upperLimit |
| 16#1105_8001 | LCON_ERROR_PID_INVALID_CYCLE_TIME<br>Function block is not called in a cylic OB (input tCycle <= 0.0) |
| 16#1106_8000 | LCON_ERROR_RFG_INVALID_CONFIGURATION<br>Invalid configuration data for RFG:<br>* rampTimeMode < 0<br>* rampTimeMode > 1 |
| 16#1106_8001 | LCON_ERROR_RFG_INVALID_CYCLE_TIME<br>Function block is not called in a cylic OB (input tCycle <= 0.0) |
| 16#1107_8000 | LCON_ERROR_RFGJ_INVALID_CONFIGURATION<br>Invalid configuration data for RFGJ:<br>* rampTimeMode < 0<br>* rampTimeMode > 1<br>* nominalValue = 0.0 |
| 16#1107_8001 | LCON_ERROR_RFGJ_INVALID_CYCLE_TIME<br>Function block is not called in a cylic OB (input tCycle <= 0.0) |
| 16#1108_8000 | LCON_ERROR_TPID_INVALID_CONFIGURATION<br>Invalid configuration data for TPID controller:<br>* setNominalValue <= 0.0 |
| 16#1109_8000 | LCON_ERROR_ANALOG_INPUT_INVALID_CONFIGURATION<br>Invalid configuration data for AnalogInput:<br>* base = 0.0 |
| 16#1109_8001 | LCON_ERROR_ANALOG_INPUT_INVALID_CYCLE_TIME<br>Function block is not called in a cylic OB (input tCycle <= 0.0) |
| 16#3103_8000 | LCON_ERROR_FRICTION_INVALID_CONFIGURATION<br>Invalid configuration data for friction measurement:<br>* sizeOfArray < 1 |

| Code / Value | Identifier / Description |
|---|---|
| 16#3107_8004 | LCON_ERROR_ENABLE_INTERFACE_ADDITIVE_TORQUE<br>Error MC command MC_AdditiveTorque.<br>ErrorID of MC command is displayed at output "subfunctionState" of FB LConSMC_EnableInterface |
| 16#3107_8005 | LCON_ERROR_ENABLE_INTERFACE_TORQUE_RANGE<br>Error MC command MC_TorqueRange.<br>ErrorID of MC command is displayed at output "subfunctionState" of FB LConSMC_EnableInterface |
| 16#3107_8006 | LCON_ERROR_ENABLE_INTERFACE_INVALID_AXIS<br>Invalid axis connected to function block. Only TO_SpeedAxis, TO_PositioningAxis or TO_SynchronousAxis are supported |
| 16#3107_8007 | LCON_ERROR_ENABLE_INTERFACE_MOVE_ABORTED<br>Error MC command MC_MoveVelocity.<br>The internal move command has been aborted from an external command. The axis will be stopped |
| 16#3107_800C | LCON_ERROR_ENABLE_INTERFACE_MOVE_VELOCITY<br>Error MC command MC_MoveVelocity.<br>ErrorID of MC command is displayed at output "subfunctionState" of FB LConSMC_EnableInterface |
| 16#3107_800E | LCON_ERROR_ENABLE_INTERFACE_HALT<br>Error MC command MC_Halt.<br>ErrorID of MC command is displayed at output "subfunctionState" of FB LConSMC_EnableInterface |
| 16#3107_8600 | LCON_ERROR_ENABLE_INTERFACE_INVALID_STATE<br>Internal error: Invalid state |
| 16#3109_8000 | LCON_ERROR_KP_ADAPTION_INVALID_CONFIGURATION<br>Invalid configuration data for Kp Adaption:<br>* diamConfig.maxDia = 0.0<br>* diamConfig.maxDia <= diamConfig.minDia<br>* diameter <= 0.0<br>* torquePreControl.maxInertia <= 0.0<br>* torquePreControlDiag.totalInertia <= 0.0 |
| 16#3109_8002 | LCON_ERROR_KP_ADAPTION_TELEGRAM_EXTENSION<br>Invalid telegram address:<br>* No valid HW-Address connected<br>* Invalid telegram length (Must be at least two Bytes) |
| 16#3113_8000 | LCON_ERROR_FRICTION_V2_INVALID_CONFIGURATION<br>Invalid configuration data for friction measurement:<br>* sizeOfArray < 1<br>* No telegram 750 active at TO<br>* The speed setpoints (speedValues) are not increasing<br>* A linear axis has been connected |
| 16#3113_8601 | LCON_ERROR_FRICTION_V2_MOVE_VELOCITY<br>Error MC command MC_MoveVelocity.<br>ErrorID of MC command is displayed at FB output "subfunctionState" |
| 16#4004_8000 | LCON_ERROR_CALC_TARGET_DIAM_STOP_COND_INVALID_CONFIGURATION<br>Invalid configuration data:<br>* windingMode < 0<br>* windinMode > 1<br>* minDiameter >= maxDiameter<br>* rampUpTime <= 0.0<br>* ramDownTime <= 0.0<br>* roundingTIme <= 0.0 |
| 16#4004_8001 | LCON_ERROR_CALC_TARGET_DIAM_STOP_COND_INVALID_CYCLE_TIME<br>Function block is not called in a cylic OB (input tCycle <= 0.0) |

| Code / Value | Identifier / Description |
|---|---|
| 16#4101_8000 | LCON_ERROR_TAPER_INVALID_CONFIGURATION<br>Invalid configuration data for tension taper:<br>* taperMode < 0 or > 4<br>* upperLimitTaper < lowerLimitTaper<br>* lowerLimitTaper < 0.0<br>Additionally for taperMode 1-3:<br>* taperDiameterStart < 0.0<br>* taperDiameterMax <= taperDiameterStart<br>* taperRatio < 0.0<br>* taperRatio > 100.0 |
| 16#4101_8008 | LCON_ERROR_TAPER_INVALID_INPUT<br>Invalid input data:<br>* diamActVal <= 0 for taperMode <> 0<br>* taperMode < 0<br>* taperMode > 4 |
| 16#4102_8000 | LCON_ERROR_WI_TORQUE_PRECONTROL_INVALID_CONFIGURATION<br>Invalid configuration data for winder torque precontrol:<br>* gearRatio/diamMin/fillFactor <= 0.0<br>* density/width/JFix/JMandrel/JMot < 0.0<br>* diamMax <= diamMin<br>* typeOfWinder < 0 or > 3 |
| 16#4102_8001 | LCON_ERROR_WI_TORQUE_PRECONTROL_INVALID_CYCLE_TIME<br>Function block is not called in a cylic OB (input tCycle <= 0.0) |
| 16#4102_8008 | LCON_ERROR_WI_TORQUE_PRECONTROL_INVALID_INPUT<br>Invalid input data:<br>* diamActVal <= 0<br>* diamActVal < diamMin |
| 16#5104_8000 | LCON_ERROR_DIAM_CALC_INVALID_CONFIGURATION<br>Invalid configuration data for diameter calculation:<br>* minDia/maxDia <= 0.0<br>* minDia > maxDia<br>* diameterFilterTime < 0.0<br>* webThickness <= 0.0 (if mode 1/4 selected)<br>* typeOfWinder < 0 or > 3<br>* measuringRevolutionCount <= 0.0<br>* measuringRevolutionCount < diameterOutputRate |
| 16#5104_8001 | LCON_ERROR_DIAM_CALC_INVALID_CYCLE_TIME<br>Function block is not called in a cylic OB (input tCycle <= 0.0) |
| 16#5104_8006 | LCON_ERROR_DIAM_CALC_MAX_BUFFER_SIZE<br>The maximum buffer size for the floating diameter calculation has been exceeded. Reduce the "measuringRevolutionCount" or increase the "diameterOutputRate" |
| 16#5104_800B | LCON_ERROR_DIAM_CALC_INVALID_MODE<br>Invalid diameter calculation mode selected ( < 0 or > 5) |
| 16#5105_8000 | LCON_ERROR_WINDER_INVALID_CONFIGURATION<br>Invalid configuration data for winder:<br>* Invalid axis unit<br>* controlMode < 0 or > 5<br>* typeOfWinder < 0 or > 3<br>* diameter calculation mode < 0 or > 5<br>* gearRatio <= 0<br>* No telegram 750 connected for torque controlled modes<br>* diamConfig.minDia/maxDia = 0.0<br>* diamConfig.minDia > diamConfig.maxDia |
| 16#5105_8001 | LCON_ERROR_WINDER_INVALID_CYCLE_TIME<br>Function block is not called in a cylic OB (input tCycle <= 0.0) |
| 16#5105_8009 | LCON_ERROR_WINDER_DATA_ADAPTION<br>Error during data adaption of the TO |

| Code / Value | Identifier / Description |
|---|---|
| 16#5105_800A | LCON_ERROR_WINDER_INVALID_TYPE_OF_AXIS<br>Invalid axis type selected.<br>The input "winderAxis" is either 0 or not of the type Speed axis, Positioning axis or Synchronous axis. |
| 16#5105_800B | LCON_ERROR_WINDER_INVALID_DIAMETER_MODE<br>Position based diameter calculation mode can't be used for speed axes |
| 16#5105_8010 | LCON_ERROR_WINDER_DRIVE_NOT_ENABLED<br>The drive must be enabled when enabling the function block |
| 16#6101_8000 | LCON_ERROR_TORQUE_PRECONTROL_INVALID_CONFIGURATION<br>Invalid configuration data for sectional drive torque precontrol:<br>* JFix/JMandrel/JMot < 0.0<br>* 0 < typeOfDrive > 1 |
| 16#6101_8008 | LCON_ERROR_TORQUE_PRECONTROL_INVALID_INPUT<br>diameter <= 0.0 |
| 16#7102_8000 | LCON_ERROR_SECTIONAL_DRIVE_INVALID_CONFIGURATION<br>Invalid configuration data for sectional drive:<br>* Invalid axis unit<br>* controlMode < 0 or > 5<br>* typeOfDrive < 0 or > 1<br>* sectionalConfig.diameter <= 0.0<br>* gearRatio <= 0<br>* No telegram 750 connected for torque controlled modes |
| 16#7102_8001 | LCON_ERROR_SECTIONAL_DRIVE_INVALID_CYCLE_TIME<br>Function block is not called in a cylic OB (input tCycle <= 0.0) |
| 16#7102_8009 | LCON_ERROR_SECTIONAL_DRIVE_DATA_ADAPTION<br>Error during data adaption of the TO |
| 16#7102_800A | LCON_ERROR_SECTIONAL_DRIVE_INVALID_TYPE_OF_AXIS<br>Invalid axis type selected |
| 16#7102_8010 | LCON_ERROR_SECTIONAL_DRIVE_DRIVE_NOT_ENABLED<br>The drive must be enabled when enabling the function block |
| 16#8200 | LCON_ERROR_CALC_TENSION_INVALID_INPUTS<br>Invalid input data:<br>* gearRatio <= 0.0<br>* diameter <= 0.0 |
| 16#8201 | LCON_ERROR_CALC_TENSION_INVALID_STATE<br>Internal error: Invalid state |
| 16#8202 | LCON_ERROR_THICKNESS_AVG_INVALID_INPUT_DATA<br>Invalid input data:<br>* thicknessMeasuringRevCount <= diameterMeasuringRevCount<br>* diameterMeasuringRevCount <= 0.0 |
| 16#8203 | LCON_ERROR_INVALID_DANCER_POSITION_LIMITS<br>The upperLimit is smaller or equal than the lower dancer limit |
| 16#8204 | LCON_ERROR_INVALID_TO_UNIT<br>Only unit of degree is allowed for the axis |
| 16#8205 | LCON_ERROR_INVALID_CONVERSION_FACTOR<br>The input conversionFactor must not be zero |
| 16#8401 | LCON_ERROR_AXIS_NOT_IN_STANDSTILL<br>The axis must be in standstill when starting the measurement |
| 16#8402 | LCON_ERROR_AXIS_NOT_ENABLED<br>The axis must be enabled before starting the measurement |
| 16#8403 | LCON_ERROR_INVALID_DANCER_START_POSITION<br>The dancer actual position must be under the upper limit when starting the measurement |
| 16#8404 | LCON_ERROR_INVALID_DANCER_POS_AFTER_LOWER_LIMIT<br>After stopping at lower limit the dancer has already exceeded the upper limit (e.g. range between lower and upper limit is too small or the used dynamics for stopping the axis are too small) |
| 16#8600 | LCON_ERR_UNDEFINED_STATE<br>Error: due to an undefined state in state machine |

| Code / Value | Identifier / Description |
|---|---|
| 16#8601 | LCON_ERROR_MOVING_THE_AXIS<br>An error processing MC_MoveVelocity occurred. See output "diagnostics.subfunctionState" for further information. |
| 16#8602 | LCON_ERROR_STOPPING_THE_AXIS<br>An error processing MC_Halt occurred. See output "diagnostics.subfunctionState" for further information. |
| 16#8603 | LCON_ERROR_ABORTING_MEASUREMENT<br>An error processing MC_Halt occurred during aborting of the measurement. See output "diagnostics.subfunctionState" for further information. |

## 7.3 LConSMC_Warnings

Status & Error codes

| Code / Value | Identifier / Description |
|---|---|
| 16#0106_4001 | LCON_WARNING_TABLE_CHAR_NO_VALUE_FOUND<br>Warning that no value is found |
| 16#3113_4000 | LCON_WARNING_FRICTION_V2_OUTPUT_LIMITED<br>Warning that the output is limited |
| 16#4001 | LCON_WARNING_MEASUREMENT_STARTS_ABOVE_LOWER_LIMIT<br>Warning: When the measurement is started the dancer position is already above the lower limit. This could affect the accuracy of the measurement. |
| 16#4004_4000 | LCON_WARNING_CALC_TARGET_DIAM_STOP_COND_OUTPUT_LIMITED<br>Warning that the output is limited |
| 16#4201 | LCON_WARNING_THICKNESS_AVG_LIMIT_SAMPLES<br>Warning: The maximum number of elements used for averaging is limited to LCON_THICKNESS_MAX_SAMPLES. Reduce input thicknessMeasuringRevCount |
| 16#5105_4000 | LCON_WARNING_WINDER_OUTPUT_LIMITED<br>Warning that the output is limited. The torque limits are limited between 0-200% |
| 16#5105_4002 | LCON_WARNING_WINDER_TORQUE_PRECONTROL_ACTIVE<br>No telegram 750 is connected but torque precontrol is selected at the FB (adjustment <> 0.0). Torque precontrol is deactivated internally. |
| 16#5105_4003 | LCON_WARNING_WINDER_NO_WEB_BREAK_DETECTION_POSSIBLE<br>Web break detection not possible in this control mode |
| 16#5105_4004 | LCON_WARNING_WINDER_MAX_VELOCITY_EXCEEDED<br>Warning that the output is limited. The input line velocity is limited to config.maxVelocity |
| 16#5105_4005 | LCON_WARNING_WINDER_TORQUE_PRECONTROL_DISABLED<br>The torque precontrol has been disabled internally if the sign of the torque limit changes for torque controlled modes |
| 16#5105_4006 | LCON_WARNING_WINDER_RUN_AWAY_DETECTED<br>In a torque controlled tension mode the deviation between line velocity setpoint and actual circumference velocity has exceeded a limit. The friction and/or acceleration torque pre-control values might be invalid and have been disabled. |
| 16#5105_4007 | LCON_WARNING_WINDER_VELOCITY_DEVIATION<br>For torque controlled modes if the actual speed and setpoint speed start to differ by more than a defined percentage value ("LCON_VELO_MONITORING_ADJUSTMENT_PERCENT") of the override. Drive can't follow it's setpoint anymore. See if acceleration and friction torque-precontrol is valid. |
| 16#7102_4000 | LCON_WARNING_SECTIONAL_DRIVE_OUTPUT_LIMITED<br>Warning that the output is limited. The torque limits are limited between 0-200% |
| 16#7102_4004 | LCON_WARNING_SECTIONAL_DRIVE_MAX_VELOCITY_EXCEEDED<br>Warning that the output is limited. The input line velocity is limited to config.maxVelocity |
| 16#7102_4005 | LCON_WARNING_SECTIONAL_DRIVE_TORQUE_PRECONTROL_DISABLED<br>The torque precontrol has been disabled internally if the sign of the torque limit changes for torque controlled modes |
| 16#7102_4006 | LCON_WARNING_SECTIONAL_DRIVE_RUN_AWAY_DETECTED<br>In a torque controlled tension mode the deviation between line velocity setpoint and actual circumference velocity has exceeded a limit. The friction and/or acceleration torque pre-control values might be invalid and have been disabled. |
| 16#7102_4007 | LCON_WARNING_SECTIONAL_DRIVE_VELOCITY_DEVIATION<br>For torque controlled modes if the actual speed and setpoint speed start to differ by more than a defined percentage value ("LCON_VELO_MONITORING_ADJUSTMENT_PERCENT") of the override. Drive can't follow it's setpoint anymore. See if acceleration and friction torque-precontrol is valid. |

# 8 Appendix

## 8.1 Service and support

### Industry Online Support

Do you have any questions or need assistance?
Siemens Industry Online Support offers round the clock access to our entire service and support know-how and portfolio.

The Industry Online Support is the central address for information about our products, solutions and services.

Product information, manuals, downloads, FAQs, application examples and videos - all information is accessible with just a few mouse clicks:

support.industry.siemens.com

### Technical Support

The Technical Support of Siemens Industry provides you fast and competent support regarding all technical queries with numerous tailor-made offers
- ranging from basic support to individual support contracts. Please send queries to Technical Support via Web form:

siemens.com/supportrequest

### SITRAIN - Digital Industry Academy

We support you with our globally available training courses for industry with practical experience, innovative learning methods and a concept that's tailored to the customer's specific needs.

For more information on our offered trainings and courses, as well as their locations and dates, refer to our web page:

siemens.de/sitrain

### Service offer

Our range of services includes the following:

- Plant data services
- Spare parts services
- Repair services
- On-site and maintenance services
- Retrofitting and modernization services
- Service programs and contract's

You can find detailed information on our range of services in the service catalog web page:

support.industry.siemens.com/cs/sc

### Industry Online Support app

You will receive optimum support wherever you are with the “Siemens Industry Online Support” app. The app is available for iOS and Android:

support.industry.siemens.com/cs/ww/de/sc/2067

## 8.2 Industry Mall


The Siemens Industry Mall is the platform on which the entire Siemens Industry product portfolio is accessible. From the selection of products to the order and the delivery tracking, the Industry Mall enables the complete purchasing processing – directly and independently of time and location:
mall.industry.siemens.com

## 8.3 Application Support

Siemens AG
Digital Industry Division
Factory Automation
DI FA PMA APC
Frauenauracher Str. 80
91056 Erlangen, Germany
mailto: printing.converting.team.automation@siemens.com

## 8.4 Links and references

| No. | Topic |
|---|---|
| \1\ | Siemens Industry Online Support<br>support.industry.siemens.com |
| \2\ | Link to the SIMATIC winder and tension control<br>support.industry.siemens.com/cs/ww/en/view/58565043 |
| \3\ | Link to the SIMOTION/SIMATIC/SINAMICS Converting Toolbox<br>support.industry.siemens.com/cs/ww/en/view/109744606 |
| \4\ | Link to the SIMIT Converting model for continuous web simulation<br>https://support.industry.siemens.com/cs/ww/en/view/109801697 |

## 8.5 Change log

| Version & Date | Change description |
|---|---|
| **V3.5.4**<br>07/2025 | **GLOBAL** / V3.5.4<br>• Compatible update containing updates and new functions<br><br>**NEW:**<br>LConSMC_CalcStartDiamDancerMeas / V1.0.0<br>• First released version<br><br>**UPDATED:**<br>LConSMC_LoopPitCtrl / V2.0.0<br>• updated UI, updated PID controller, mode 8 (open loop controller) added, bug fixes |
| **V3.5.3**<br>04/2025 | **GLOBAL** / V3.5.3<br>• Compatible update containing bugfixes and new functions<br><br>**NEW:**<br>LConSMC_LoopPitCtrl / V1.0.0<br>• First released version<br><br>**UPDATED:**<br>LConSMC_EnableInterface / V3.0.2<br>• c_00082: Performance optimization: Change call of MC-commands from cyclic to on trigger condition<br>LConSMC_SectionalDrive / V3.2.4<br>• Updated input parameter comments<br>• c_00080: Performance optimizations for not torque controlled modes<br>LConSMC_Winder / V3.2.4<br>• c_00081: Performance optimizations for not torque controlled modes |
| **V3.5.2**<br>01/2025 | **GLOBAL** / V3.5.2<br>• Compatible update containing bugfixes and new functions<br><br>**NEW:**<br>LConSMC_ThicknessAverage / V1.0.0<br>• First released version<br><br>**UPDATED:**<br>LConSMC_DiameterCalculation / V1.3.3<br>• c_00071: New configuration parameter “diamConfig.integrationDirection” to limit diameter change direction for mode Integral(2) and Position(5)<br>LConSMC_EnableInterface / V3.0.1<br>• c_00070: Wait until MC_TorqueRang command is disabled before disabling FB<br>LConSMC_FrictionMeasurement_V2 / V1.0.2<br>• Removed warning “Value of constant outside the permitted range” during compilation<br>LConSMC_SectionalDrive / V3.2.3<br>• c_00075: New warning 16#7102_4007 for torque controlled modes if the actual speed and setpoint speed start to differ by more than a defined percentage value (“LCON_VELO_MONITORING_ADJUSTMENT_PERCENT”) of the override.<br>• Bugfix: Under some conditions the 16#7102_4006 was not triggered correct<br>LConSMC_Winder / V3.2.3<br>• c_00074: New warning 16#5105_4007 for torque controlled modes if the actual speed and setpoint speed start to differ by more than a defined percentage value (“LCON_VELO_MONITORING_ADJUSTMENT_PERCENT”) of the override.<br>• Bugfix: Under some conditions the 16#5105_4006 was not triggered correct |
| **V3.5.1** | **GLOBAL** / V3.5.1 |

| Version & Date | Change description |
|---|---|
| 06/2024 | • Compatible update containing bugfixes<br><br>**UPDATED:**<br>LConSMC_DiameterCalculation / V1.3.2<br>• Bugfix: Initialize internal variables when changing parameters measuringRevolutionCount/diameterOutputRate on the fly (*c_00066*)<br>LConSMC_SectionalDrive / V3.2.2<br>• Bugfix: Under some conditions (e.g. after jogging function) the leading axis velocity was not considered anymore (*c_00068*)<br>LConSMC_TPID / V1.1.1<br>• Bugfix: Under some conditions during the disabling of the block an invalid output value (“outValue”) was calculated if the upper or lower limit has been set to zero (*c_00069*)<br>LConSMC_Winder / V3.2.2<br>• Bugfix: Under some conditions (e.g. after jogging function) the leading axis velocity was not considered anymore (*c_00067*) |
| **V3.5.0**<br>03/2024 | **GLOBAL** / V3.5.0<br>• Upgraded library to TIA V18<br>• PLC FW 3.0 with MC V7.0 or higher is mandatory<br><br>**UPDATED:**<br>LConSMC_DiameterCalculation / V1.3.1<br>• Bugfix: Initialize internal variables after setting the diameter without disabling the function block (*c_00065*)<br>LConSMC_EnableInterface / V3.0.0<br>• Runtime optimizations using TO references with MC V7.0<br>• Redesign of FB according to PLCopen<br>• New output *valid*<br>• Renamed output *errorID* to *status*<br>• Bugfixes: Velocity interface was not triggered correct at first start<br>• Bugfixes: Reset velocity override to 100% if function block is disabled and axis is stopped<br>LConSMC_GetAxisConfiguration / V1.1.0<br>• Runtime optimizations using TO references with MC V7.0<br>LConSMC_RFGJ / V1.3.2<br>• Bugfix: Under some conditions the output signal was not calculated correct when using a rounding time and changing the input values during acceleration/deceleration (*c_00064*)<br>• Updates: Internal optimizations for roundingTime = 0 (*c_00063*)<br>• Updates: Added REGIONS for structuring of the code<br>LConSMC_SectionalDrive / V3.2.1<br>• New function: Monitoring of drive runaway for torque-controlled modes. In case of a detection the torque pre-control of friction and acceleration is disabled and a warning 16#7102_4006 is generated (*c_00061*)<br>• Bugfix: Initializing internal variables (*c_00060*)<br>LConSMC_Winder / V3.2.1<br>• New function: Monitoring of drive runaway for torque-controlled modes. In case of a dectection the torque pre-control of friction and acceleration is disabled and a warning 16#5105_4006 is generated (*c_00061*)<br>• Bugfix: The external diameter value (typeOfDiameterCalc = 3), was only used if inputs speedMatch and ctrlEnable was TRUE but e.g., not for the jogging function (*c_00058*)<br>• Bugfix: Under some conditions an invalid diameter value (division by zero) was calculated when setting the diameter value (*c_00059*)<br><br>**DELETED:**<br>LConSMC_FrictionMeasurement / V1.5.0 |
| **V3.4.0**<br>10/2023 | **NEW:**<br>LConSMC_CalcActualTension / V1.0.0<br>• First released version |

| Version & Date | Change description |
|---|---|
| | **UPDATED:**<br>LConSMC_FrictionMeasurement / V1.5.0<br>• Updated input parameter description<br>LConSMC_FrictionMeasurement_V2 / V1.0.1<br>• Updated input parameter description<br>LConSMC_LineUnitConversion / V1.1.0<br>• Added units with “Higher resolution” option activated<br>LConSMC_PT1 / V2.0.1<br>• Setting the output to the setValue was not possible for input “smootingTime” <= 0.0 (*c_00054*)<br>LConSMC_RFG / V1.4.1<br>• Updated output parameter description<br>LConSMC_RFGJ / V1.3.1<br>• Updated output parameter description<br>LConSMC_SectionalDrive / V3.2.0<br>• Bugfix: Possible error during startup was overwritten (*c_00056*)<br>• Under some conditions the calculation of the output signal “inSync” was incorrect (*c_00058*)<br>• With this version linear axes are supported<br>• New control mode “Indirect tension control via speed setpoint adaption” (controlMode = 5)<br>LConSMC_TableChar / V1.3.0<br>• For input value “sizeOfArray” = 2 an invalid array access occured (*c00055*)<br>LConSMC_Winder / V3.2.0<br>• Bugfix: Possible error during startup was overwritten (*c_00056*)<br>• Under some conditions the calculation of the output signal “inSync” was incorrect (*c_00058*)<br>• New control mode “Indirect tension control via speed setpoint adaption” (controlMode = 5) |
| **V3.3.0**<br>03/2023 | **UPDATED:**<br>LConSMC_DiameterCalculation / V1.3.0<br>• Optimized configuration data (parameters “windingShaftFeedbackScal”/“lineFeedbackScal” are not required anymore)<br>• Bugfix validate configuration only if function block is enabled (*c_00048*)<br>• Avoid division by zero (*c_00049*)<br>• Initialize internal variable after changing the diameter calculation mode on the fly (*c_00050*)<br>LConSMC_SectionalDrive / V3.0.2<br>• Deleted output “sectionalDriveMotionVector”<br>• Added new parameters “diameterFactor” and “referenceVelocity” to diagnostic output structure (can be used for external axis control e.g. via the LPrint library)<br>• Bit 8 in STW2 (TravelToFixedEnstop) is set automatically for torque controlled modes to avoid SINAMICS fault F07900 Motor blocked/speed controller at its limit (*c_00052*)<br>• For torque controlled modes if the upper torque limit becomes negative or the lower torque limit becomes positive the torque precontrol had the wrong sign (*c_00046*)/(*c_00047*)<br>• Under some conditions the “inSync” output was not calculated correct (syncDynamics.jerk/acceleration < 0.0)<br>• Disabling sequence of tension control for torque controlled modes optimized (Reset speed controller integrator) (*c_00051*)<br>• Use ramped tension setpoint for torque controlled modes (*c_00053*)<br>• Bugfix calculating the output signal “inSync”<br>LConSMC_Winder / V3.0.2<br>• Deleted output “winderMotionVector”<br>• Added new parameters “diameterFactor” and “referenceVelocity” to diagnostic output structure (can be used for external axis control e.g. via the LPrint library)<br>• Bit 8 in STW2 (TravelToFixedEnstop) is set automatically for torque controlled modes to avoid SINAMICS fault F07900 Motor blocked/speed controller at its limit (*c_00052*)<br>• For torque controlled modes if the upper torque limit becomes negative or the lower torque limit becomes positive the torque precontrol had the wrong sign (*c_00046*)/(*c_00047*)<br>• Under some conditions the “inSync” output was not calculated correct (syncDynamics.jerk/acceleration < 0.0) |

| Version & Date | Change description |
|---|---|
| | • Disabling sequence of tension control for torque controlled modes optimized (Reset speed controller integrator) (*c_00051*)<br>• Use ramped tension setpoint for torque controlled modes (*c_00053*)<br>• Bugfix calculating the output signal “inSync” |
| **V3.2.1**<br>06/2022 | **UPDATED:**<br>LConSMC_EnableInterface / V2.0.1<br>• Bugfix resetting internal variables with rising edge of “enable” (*c_00045*)<br>• Renamed input “executeMoveVelocity” to “enableVelocityInterface” (Same behaviour as before)<br>LConSMC_RFGJ / V1.3.0<br>• Bugfix internal calculations for using the roundingTime<br>LConSMC_SectionalDrive / V3.0.1<br>• Bugfix: For some conditions the output “error” was not set correct<br>• Error occured when using the winder with the TO setting “Use position values with higher resolution” enabled (*c_00044*)<br>• New errorID if the FB is enabled but the drive is disabled (error = TRUE ; errorID = 5105_8010)<br>LConSMC_Winder / V3.0.1<br>• Bugfix: For some conditions the output “error” was not set correct<br>• Error occured when using the winder with the TO setting “Use position values with higher resolution” enabled (*c_00044*)<br>• New errorID if the FB is enabled but the drive is disabled (error = TRUE ; errorID = 5105_8010) |
| **V3.2.0**<br>02/2022 | **UPDATED:**<br>LConSMC_PID / V2.0.0<br>• The output “busy” was not reset correct<br>LConSMC_SectionalDrive / V3.0.0<br>• Adaption of FB structure (Deleted FB for each axis type)<br>LConSMC_Winder / V3.0.0<br>• Adaption of FB structure (Deleted FB for each axis type)<br>• It is possible to set a new diameter value also if the winder axis is disabled (enable = FALSE, setDiameter = TRUE) (*c_00043*)<br>LConSMC_WITorquePrecontrol / V2.1.0<br>• New parameter “winderConfig.torquePrecontrolConfig.rewinderTorqueAdditive” to add a constant torque value to the torque precontrol / torque limit (e.g. compensate static friction) (*c_00042*)<br><br>**DELETED:**<br>LConSMC_Scaling / V1.0.0<br>LConSMC_ScalingDWDRL / V1.0.0<br>LConSMC_ScalingRLDWD / V1.0.0<br>LConSMC_ScalingRLWD / V1.0.0<br>LConSMC_ScalingWDRL / V1.0.0<br>LConSMC_TRIG / V1.0.0 |
| **V3.1.2**<br>06/2021 | **UPDATED:**<br>LConSMC_CalcTargetDiamStopCondition / V1.2.0<br>• Bugfix: Busy is set TRUE with a rising edge of the input enable (*c_00041*)<br>LConSMC_DiameterCalculation / V1.2.0<br>• Bugfix calculating condition for outputs atMinDia/atMaxDia in mode Intragal and Position (*c_00040*)<br>LConSMC_EnableInterface / V1.0.4<br>• Bugfix setting the override to a value bigger 100% (*c_00037*)<br>LConSMC_SectionalDrive / V2.0.3<br>• see LConSMC_aVersion<br>LConSMC_Winder / V2.0.3<br>• In case of torque controlled modes at maximum line speed (winderConfig.maxVelocity) and |

| Version & Date | Change description |
|---|---|
| | minimum diameter, the speed setpoint was calculated incorrect (*c_00034*)<br>• Extended warning reason "5105_4002" if parameter "dDdtAdjustment" is > 0.0 but no telegram 750 is connected (*c_00038*)<br>LConSMC_WITorquePrecontrol / V2.0.0<br>• New inputs "tCycle" and "dDdTAdjustment"<br>• New functionality to precontrol acceleration torque based on diameter change (*c_00035*) |
| **V3.1.1**<br>01/2021 | **NEW:**<br>LConSMC_FrictionMeasurement_V2 / V1.0.0<br>• First released version<br>LConSMC_LineUnitConversion / V1.0.0<br>• Creation<br><br>**UPDATED:**<br>LConSMC_AverageValue / V1.1.1<br>• Removed warning "Parameter might not be initialized"<br>LConSMC_EnableInterface / V1.0.3<br>• Removed warning "Parameter might not be initialized"<br>LConSMC_FrictionMeasurement / V1.4.0<br>• Limited input variable "sizeOfArray" to global constant LCON_MAX_VALUE_BREAK_POINT_TABLE (*c_00032*)<br>LConSMC_KpAdaption / V1.0.1<br>• Removed warning "Parameter may not be initialized"<br>LConSMC_SectionalDrive / V2.0.2<br>• Changed input "axis" to from input paramter to InOut parameter<br>LConSMC_Winder / V2.0.2<br>• Changed input "axis" to from input paramter to InOut parameter |
| **V3.1.0**<br>09/2020 | **UPDATED:**<br>LConSMC_EnableInterface / V1.0.2<br>• Bugfix: Parametrizable deceleration possible via new datatype for motionVector (*c_00029*)<br>• Rename: moveVelocityActive -> MCInterfaceActive. MCInterfaceActive is TRUE if MC_MoveVelocity OR MC_Halt ist active! (*c_00026*)<br>• New ErrorMessage/Tag if MC_Halt has an error. (*c_00030*)<br>• Bugfix: Cycledelay between calculated and executed motion profile has been removed(*c_00028*)<br>LConSMC_SectionalDrive / V2.0.1<br>• Bugfix: Parametrizable deceleration possible via new datatype for winderMotionVector (*c_00025*)<br>• Bugfix: Webbreak: Relationfactor multiplied in Numerator not longer in Denumerator (*c_00027*)<br>• Enhancement: New Warning "LCON_WARNING_MAXVELOCITY_EXCEEDED" to determine if the<br>• lineVelocity + aditionalVelocity is higher then the configurated maxVelocity (*c_00031*)<br>LConSMC_Winder / V2.0.1<br>• Bugfix: Parametrizable deceleration possible via new datatype for winderMotionVector (*c_00025*)<br>• Bugfix: Webbreak: Relationfactor multiplied in Numerator not longer in Denumerator (*c_00027*)<br>• Enhancement: New Warning "LCON_WARNING_MAXVELOCITY_EXCEEDED" to determine if the<br>• lineVelocity + aditionalVelocity is higher then the configurated maxVelocity (*c_00031*) |
| **V3.0.1**<br>01/2020 | **UPDATED:**<br>LConSMC_EnableInterface / V1.0.1<br>• Bugfix: moveVelocity and stop will be resetted with enable (*c_00022*) |
| **V3.0.0**<br>10/2019 | **NEW:**<br>LConSMC_DiameterCalculation / V1.0.0<br>• Bugfix: After the first initialization the correct diameter value is used now. (*c_00019*) |

| Version & Date | Change description |
| --- | --- |
| | • Creation<br>LConSMC_EnableInterface / V1.0.0<br>• Creation<br>LConSMC_GetAxisConfiguration / V1.0.0<br>• Creation<br>LConSMC_KpAdaption / V1.0.0<br>• Creation<br><br>**UPDATED:**<br>LConSMC_SectionalDrive / V2.0.0<br>• Called via DB_Any<br>• new diag structure<br>• new in- and outputs<br>• communication of torque data via TO and not applicative<br>• Kp_Adaption as a seperate function block<br>• drive commands not via LAxisCtrl, now internal<br>• No UserMode. If needed, LAxisCtrl can be downloaded in SIOS<br>• new dynamic structures<br>• webbreak detection implemented<br>• Bugfix: the set web tension is maintained at maximum speed<br>LConSMC_Winder / V2.0.0<br>• Called via DB_Any<br>• new diag structure<br>• new in- and outputs<br>• communication of torque data via TO and not applicative<br>• Kp_Adaption as a seperate function block<br>• drive commands not via LAxisCtrl, now internal<br>• No UserMode. If needed, LAxisCtrl can be downloaded in SIOS<br>• new dynamic structures<br>• webbreak detection implemented<br>• Bugfix: the set web tension is maintained at maximum speed<br><br>**DELETED:**<br>LConSMC_PosAxisInterface / V1.0.2<br>LConSMC_SpeedAxisInterface / V1.0.2 |
| **V2.2.0**<br>09/2019 | **UPDATED:**<br>LConSMC_PosAxisInterface / V1.0.2<br>• bugfix (at Error Velocity Reached -> nothing happend, now: axis stops with stopdynamics, AdditiveTorque=0, open TorqueLimits) (*c_000018*)<br>LConSMC_PT1 / V2.0.0<br>• update variables also for smoothingTime <= 0.0 (*c_00015*)<br>LConSMC_SpeedAxisInterface / V1.0.2<br>• bugfix (at Error Velocity Reached -> nothing happend, now: axis stops with stopdynamics, AdditiveTorque=0, open TorqueLimits) (*c_000018*)<br>• bugfix (At Region: Write Torque Limits) at winder.enable = False, additiveTorque = 0, Torquelimits will be opend and axis will be ramped down.<br>LConSMC_WITorquePrecontrol / V1.3.0<br>• correct calculation of tension torque in anglo american units (*c_00017*) |
| **V2.1.1**<br>08/2018 | **UPDATED:**<br>LConSMC_CalcStopDistance / V1.2.0<br>• bugfix calc time to reduce acceleration with jerk limitation (*c_00009*)<br>LConSMC_PosAxisInterface / V1.0.1<br>• Bugfix reset error output (*c_00002*)<br>• Reset pending TO error at startup<br>• Adapted enable precontrol for USER_MODE (*c_00006*) |

| Version & Date | Change description |
|---|---|
| | • changed calculation of "inVelocity" signal for internal interface (*c_00007*)<br>LConSMC_RFG / V1.4.0<br>• Bugfix for mode increase&decrease if x = upper/lower limit (*c_00012*)<br>LConSMC_SpeedAxisInterface / V1.0.1<br>• Bugfix resetting the error output (*c_00001*)<br>• Adapted enable precontrol for USER_MODE (*c_00006*)<br>• changed calculation of "inVelocity" signal for internal interface (*c_00007*)<br>LConSMC_TableChar / V1.2.0<br>• Bugfix for x values smaller than first x coordinate *c_00004*<br>LConSMC_TorquePrecontrol / V1.2.0<br>• additional input "accelerationAdjustment"<br>LConSMC_Winder / V1.0.1<br>• New function: Use speed sepoint/actual speed value for friction torque precontrol (*c_00008*)<br>• Bugfix calculating actual velocity (*c_00010*)<br>• Calc output "setdriveSpeed" on drive side (*c_00011*)<br>• Deleted input "openBrake"<br>LConSMC_WITorquePrecontrol / V1.2.0<br>• additional input "accelerationAdjustment" |
| **V2.1.0**<br>11/2017 | **NEW:**<br>LConSMC_EvaluateModulo / V1.0.0<br>• Creation<br>LConSMC_ModuloIEC / V1.0.0<br>• Creation<br>LConSMC_PosAxisInterface / V1.0.0<br>• First released version<br>LConSMC_SectionalDrive / V1.0.0<br>• First released version<br>LConSMC_SpeedAxisInterface / V1.0.0<br>• First released version<br>LConSMC_Winder / V1.0.0<br>• Creation<br><br>**UPDATED:**<br>LConSMC_Integrator / V1.2.0<br>• Increas integration constant MAX_INT_TIME to 60000<br>LConSMC_RFGJ / V1.2.0<br>• Additional input "rampTimeMode" to define the use of rampup and rampdown time |
| **V2.0.0 and before**<br>11/2017 | **NEW:**<br>LConSMC_DeadBand / V1.0.0<br>• Creation<br>LConSMC_DeadZone / V1.0.0<br>• Creation<br>LConSMC_Differentiator / V1.0.0<br>• Removed warning "Parameter might not be initialized"<br>• Creation<br>LConSMC_DT1 / V1.0.0<br>• Removed warning "Parameter might not be initialized"<br>• Creation<br>LConSMC_LevelControl / V1.0.0<br>• adapted to styleguide; warning "variable might not be initialized" removed<br>• Creation<br>LConSMC_Limiter / V1.0.0<br>• Adapted to styleguide<br>• Creation |

| Version & Date | Change description |
|---|---|
| | LConSMC_Linear / V1.0.0<br>• adapted to styleguide<br>• Creation<br>LConSMC_ReadAnalogInput / V1.0.0<br>• adapted to styleguide; warning “variable might not be initialized” removed)<br>• replace “word_to_real” with convertion “word_to_int” and “int_to_real”<br>• Creation<br>LConSMC_ReadCycleTimeS71500 / V1.0.0<br>• adapted to styleguide<br>• extended to read cycle time from synchronous cycle OBs<br>• Creation<br>LConSMC_RestDiameterCalc / V1.0.0<br>• adapted to styleguide<br>• First released version<br>LConSMC_RestLengthCalc / V1.0.0<br>• Creation<br>LConSMC_Taper / V1.0.0<br>• adapted to styleguide; warning “variable might not be initialized” removed<br>• Creation<br>LConSMC_TPID / V1.0.0<br>• adapted to styleguide; warning “variable might not be initialized” removed<br>• Creation<br><br>**UPDATED:**<br>LConSMC_AverageValue / V1.0.0<br>• Adapted to styleguide<br>• Creation<br>LConSMC_CalcStopDistance / V1.0.0<br>• adapted to styleguide<br>• bugfix a < 0.0 (#TR 2016-04-11)<br>• Creation<br>LConSMC_CalcTargetDiamStopCondition / V1.0.0<br>• adapted to styleguide; warning “variable might not be initialized” removed<br>• Creation<br>LConSMC_EnableInterface / V2.0.0<br>• Renamed input “executeMoveVelocity” to “enableVelocityInterface” (Same behaviour as before)<br>LConSMC_FrictionMeasurement / V1.0.0<br>• Adapted to styleguide; warning “variable might not be initialized” removed<br>• Bugfix: error bit was not set in case #ENTRY_NUM_TOO_SMALL<br>• Bugfix: last speed value of speedValues table was not considered in case (#maxSpeed = 0.0)<br>• Bugfix value number<br>• Creation<br>LConSMC_Integrator / V1.0.0<br>• Removed warning “Parameter might not be initialized”<br>• Creation<br>LConSMC_PID / V1.0.0<br>• adapted to styleguide; warning “variable might not be initialized” removed<br>• Creation<br>LConSMC_PT1 / V1.0.0<br>• adapted to styleguide; warning “variable might not be initialized” removed<br>• Creation<br>LConSMC_RFG / V1.0.0 |

| Version & Date | Change description |
|---|---|
| | • Additional input “rampTimeMode” to define the use of rampup and rampdown Time<br>• adapted to styleguide; warning “variable might not be initialized” removed<br>• Bugfix respect of ramp while change of limit value<br>• Creation<br>LConSMC_RFGJ / V1.0.0<br>• adapted to styleguide; warning “variable might not be initialized” removed<br>• Creation<br>LConSMC_TableChar / V1.0.0<br>• adapted to styleguide; warning “variable might not be initialized” removed<br>• Creation<br>LConSMC_TorquePrecontrol / V1.0.0<br>• adapted to styleguide; warning “variable might not be initialized” removed<br>• First released version<br>LConSMC_WITorquePrecontrol / V1.0.0<br>• adapted to styleguide; warning “variable might not be initialized” removed<br>• Creation |