ControlLoopLibrary Library Documentation

Company:	3S - Smart Software Solutions GmbH
Title:	ControlLoopLibrary
Version:	1.0.0.0
Namespace:	Ctrl
Author:	3S - Smart Software Solutions GmbH
Placeholder:	CtrlLib

Description [1]

This library contains functionality for controlling algorithms and signal filtering.

Contents:

Indices and tables

[1] Based on ControlLoop.library, last modified 26.11.2019, 10:12:02.
The content file doc.clean.json was generated with CODESYS V3.5 SP14 on 26.11.2019, 10:12:10

Enums

Controller_Error (ENUM)

TYPE Controller_Error :

InOut:

Name	Initial
NO_ERROR	0
INPUT_OUT_OF_RANGE	1
OVERFLOW	2
WRONG_CONFIGURATION	3
PWM_CYCLE_SHORTER_THAN_TASK_CYCLE	4
SETTINGS_CHANGED_BUSY	5

FilterType (ENUM)

TYPE FilterType :

InOut:

Name	Initial
Lowpass	0
Highpass	
Bandpass	
Bandstop	

RoundingStrategy (ENUM)

TYPE RoundingStrategy :

InOut:

Name	Initial
round	0
ceil	
floor	

Function Blocks

This directory contains all function blocks of the library.

- AntiWindUp
  - AntiWindUp_BackCalculation (FB)
    - AntiWindUp_BackCalculation.Apply (METH)
  - AntiWindUp_Base (FB)
    - AntiWindUp_Base.Apply (METH)
    - AntiWindUp_Base.IsLimitExceeded (METH)
  - AntiWindUp_Clamping (FB)
    - AntiWindUp_Clamping.Apply (METH)
- Controller
  - BangBangController
    - BangBangController (FB)
      - BangBangController.CyclicAction (METH)
      - BangBangController.OutputValue (PROP)
      - BangBangController.ResetAction (METH)
    - BangBangControllerWithTimeHysteresis (FB)
      - BangBangControllerWithTimeHysteresis.CleanupAction (METH)
      - BangBangControllerWithTimeHysteresis.CyclicAction (METH)
      - BangBangControllerWithTimeHysteresis.StartAction (METH)
    - BangBangControllerWithValueHysteresis (FB)
      - BangBangControllerWithValueHysteresis.CyclicAction (METH)
      - BangBangControllerWithValueHysteresis.OutputValue (PROP)
      - BangBangControllerWithValueHysteresis.ResetAction (METH)
  - Controller_Base (FB)
    - Controller_Base.ActualValue (PROP)
    - Controller_Base.CleanupAction (METH)
    - Controller_Base.CycleInterval (PROP)
    - Controller_Base.CyclicAction (METH)
    - Controller_Base.ExecuteController (METH)
    - Controller_Base.GetCurrentError (METH)
    - Controller_Base.GetTaskCycleInterval (METH)
    - Controller_Base.HandleLimitsExceeded (METH)
    - Controller_Base.LimitsActive (PROP)
    - Controller_Base.ManualModeActive (PROP)
    - Controller_Base.MaxValue (PROP)
    - Controller_Base.MinValue (PROP)
    - Controller_Base.Offset (PROP)
    - Controller_Base.OutputValue (PROP)
    - Controller_Base.ResetAction (METH)
    - Controller_Base.SetManual (METH)
    - Controller_Base.SetPoint (PROP)
    - Controller_Base.StartAction (METH)
  - Controller_P (FB)
    - Controller_P.ExecuteController (METH)
    - Controller_P.KP (PROP)
  - Controller_PD (FB)
    - Controller_PD.CleanupAction (METH)
    - Controller_PD.ExecuteController (METH)
    - Controller_PD.KD (PROP)
    - Controller_PD.KP (PROP)
    - Controller_PD.ResetAction (METH)
    - Controller_PD.StartAction (METH)
  - Controller_PI (FB)
    - Controller_PI.CleanupAction (METH)
    - Controller_PI.ExecuteController (METH)
    - Controller_PI.KI (PROP)
    - Controller_PI.KP (PROP)
    - Controller_PI.ResetAction (METH)
    - Controller_PI.StartAction (METH)
  - Controller_PID (FB)
    - Controller_PID.CleanupAction (METH)
    - Controller_PID.ExecuteController (METH)
    - Controller_PID.KD (PROP)
    - Controller_PID.KI (PROP)

        - Controller_PID.KP (PROP)
        - Controller_PID.ResetAction (METH)
        - Controller_PID.StartAction (METH)
    - ThreePointController
        - ThreePointController (FB)
            - ThreePointController.CyclicAction (METH)
            - ThreePointController.OutputValue (PROP)
            - ThreePointController.ResetAction (METH)
        - ThreePointControllerWithValueHysteresis (FB)
            - ThreePointControllerWithValueHysteresis.CyclicAction (METH)
            - ThreePointControllerWithValueHysteresis.OutputValue (PROP)
            - ThreePointControllerWithValueHysteresis.ResetAction (METH)
- Differentiation
    - Differentiator_BackwardDifference (FB)
        - Differentiator_BackwardDifference.CleanupAction (METH)
        - Differentiator_BackwardDifference.DoDifferentiation (METH)
        - Differentiator_BackwardDifference.StartAction (METH)
    - Differentiator_Base (FB)
        - Differentiator_Base.CurrentDerivative (PROP)
        - Differentiator_Base.CyclicAction (METH)
        - Differentiator_Base.CyclicCall (METH)
        - Differentiator_Base.DoDifferentiation (METH)
        - Differentiator_Base.OutputValue (PROP)
        - Differentiator_Base.Reset (METH)
        - Differentiator_Base.ResetAction (METH)
    - Differentiator_LinearAverageApproximation (FB)
        - Differentiator_LinearAverageApproximation.CleanupAction (METH)
        - Differentiator_LinearAverageApproximation.DoDifferentiation (METH)
        - Differentiator_LinearAverageApproximation.StartAction (METH)
    - Differentiator_LinearFourPointApproximation (FB)
        - Differentiator_LinearFourPointApproximation.CleanupAction (METH)
        - Differentiator_LinearFourPointApproximation.DoDifferentiation (METH)
        - Differentiator_LinearFourPointApproximation.StartAction (METH)
    - Differentiator_ParabolicApproximation (FB)
        - Differentiator_ParabolicApproximation.CleanupAction (METH)
        - Differentiator_ParabolicApproximation.DoDifferentiation (METH)
        - Differentiator_ParabolicApproximation.StartAction (METH)
- Filter
    - Filter_Base (FB)
        - Filter_Base.CurrentValue (PROP)
        - Filter_Base.CyclicAction (METH)
        - Filter_Base.OutputValue (PROP)
        - Filter_Base.Reset (METH)
        - Filter_Base.ResetAction (METH)
    - Filter_FIR (FB)
        - Filter_FIR.CleanupAction (METH)
        - Filter_FIR.CyclicAction (METH)
        - Filter_FIR.FB_EXIT (METH)
        - Filter_FIR.StartAction (METH)
    - Filter_IIR (FB)
        - Filter_IIR.CleanupAction (METH)
        - Filter_IIR.CyclicAction (METH)
        - Filter_IIR.FB_EXIT (METH)
        - Filter_IIR.StartAction (METH)
    - Filter_SOS (FB)
        - Filter_SOS.CleanupAction (METH)
        - Filter_SOS.FB_EXIT (METH)
        - Filter_SOS.StartAction (METH)
- Integrator
    - Integrator_Base (FB)
        - Integrator_Base.ApplyAntiWindUpCorrection (METH)
        - Integrator_Base.CheckBoundaries (METH)
        - Integrator_Base.CleanupAction (METH)
        - Integrator_Base.CurrentIntegral (PROP)
        - Integrator_Base.CycleInterval (PROP)
        - Integrator_Base.CyclicAction (METH)
        - Integrator_Base.CyclicCall (METH)
        - Integrator_Base.DoIntegration (METH)
        - Integrator_Base.LastSegmentIntegralValue (PROP)

        - Integrator_Base.OutputValue (PROP)
        - Integrator_Base.Overflow (PROP)
        - Integrator_Base.Reset (METH)
        - Integrator_Base.ResetAction (METH)
    - Integrator_ParabolicApproximation (FB)
        - Integrator_ParabolicApproximation.CleanupAction (METH)
        - Integrator_ParabolicApproximation.DoIntegration (METH)
    - Integrator_RectangleApproximation (FB)
        - Integrator_RectangleApproximation.DoIntegration (METH)
    - Integrator_TrapezoidApproximation (FB)
        - Integrator_TrapezoidApproximation.CleanupAction (METH)
        - Integrator_TrapezoidApproximation.DoIntegration (METH)
- PWM_Creator
    - PWM_Creator (FB)
        - PWM_Creator.CleanupAction (METH)
        - PWM_Creator.CyclicAction (METH)
        - PWM_Creator.StartAction (METH)
    - PWM_Creator_FixedCycle (FB)
        - PWM_Creator_FixedCycle.CleanupAction (METH)
        - PWM_Creator_FixedCycle.CyclicAction (METH)
        - PWM_Creator_FixedCycle.StartAction (METH)

AntiWindUp

This directory contains all provided anti wind-up startegies.

- AntiWindUp_BackCalculation (FB)
  - AntiWindUp_BackCalculation.Apply (METH)
- AntiWindUp_Base (FB)
  - AntiWindUp_Base.Apply (METH)
  - AntiWindUp_Base.IsLimitExceeded (METH)
- AntiWindUp_Clamping (FB)
  - AntiWindUp_Clamping.Apply (METH)

AntiWindUp_BackCalculation (FB)

FUNCTION_BLOCK AntiWindUp_BackCalculation EXTENDS AntiWindUp_Base

Represents the back calculation anti-windup strategy

This strategy discharges the integrator over time depending on its overshoot compared to the specified limits.

> **Note**
>
> Integrators using this anti wind-up strategy must implement the interface IBackCalculationTaskIntervalProvider. All integrators contained in the library do so.

For more information refer to AntiWindUp_Base.

InOut:

Scope	Name	Type	Comment	Inherited from
Input	lrMinValue	LREAL	Represents the lower bound of `lrOutputValue` (`lrMinValue` has to smaller than `lrMaxValue`)	AntiWindUp_Base
	lrMaxValue	LREAL	Represents the upper bound of `lrOutputValue` (`lrMinValue` has to smaller than `lrMaxValue`)	AntiWindUp_Base
	lrResetTime	LREAL	Represents a time which determines how quickly the integrator is discharged [seconds] `lrResetTime` must not be smaller than the cycle interval time of the integrator	

AntiWindUp_BackCalculation.Apply (METH)

METHOD Apply

This method is called every cycle and applies an ati-windup strategy if the give integrator exceeds specified limits.

InOut:

Scope	Name	Type	Comment
Input	itfIntegrator	IIntegrator	Represents the integrator on which an anti wind-up strategy should be applied

AntiWindUp_Base (FB)

FUNCTION_BLOCK ABSTRACT AntiWindUp_Base IMPLEMENTS IAntiWindUp

Represents the base anti-windup strategy class

It implements the IAntiWindUp interface and contains common methods and properties for any pursuing anti-windup strategy.

> **Note**
>
> In order to built an individual anti wind-up strategy, the specific function block has to extend this function block or implement the IAntiWindUp interface.

Anti-windup strategies prevent integration wind-up of controllers which have an integral part. These strategies discharge or clamp the integrator of a controller whenever it exceeds specified output limits.

InOut:

Scope	Name	Type	Comment
Input	lrMinValue	LREAL	Represents the lower bound of `lrOutputValue` (`lrMinValue` has to smaller than `lrMaxValue`)
	lrMaxValue	LREAL	Represents the upper bound of `lrOutputValue` (`lrMinValue` has to smaller than `lrMaxValue`)

AntiWindUp_Base.Apply (METH)

METHOD Apply

This method is called every cycle and applies an ati-windup strategy if the give integrator exceeds specified limits.

InOut:

Scope	Name	Type	Comment
Input	itfIntegrator	IIntegrator	Represents the integrator on which an anti wind-up strategy should be applied

AntiWindUp_Base.IsLimitExceeded (METH)

METHOD PROTECTED IsLimitExceeded : BOOL

This method checks whether a given integrator exceeds the defined limits.

InOut:

Scope	Name	Type
Return	IsLimitExceeded	BOOL
Input	itfIntegrator	IIntegrator

AntiWindUp_Clamping (FB)

FUNCTION_BLOCK AntiWindUp_Clamping EXTENDS AntiWindUp_Base

Represents the clamping anti-windup strategy

This strategy clamps the integrator as long as the integrator exceeds the specificied output limits.

> **Note**
>
> Integrators using this anti wind-up strategy must implement the interface IClampingValueProvider. All integrators contained in the library do so.

For more information refer to AntiWindUp_Base.

InOut:

Scope	Name	Type	Comment	Inherited from
Input	lrMinValue	LREAL	Represents the lower bound of `lrOutputValue` (`lrMinValue` has to smaller than `lrMaxValue`)	AntiWindUp_Base
	lrMaxValue	LREAL	Represents the upper bound of `lrOutputValue` (`lrMinValue` has to smaller than `lrMaxValue`)	AntiWindUp_Base

AntiWindUp_Clamping.Apply (METH)

METHOD Apply

This method is called every cycle and applies an ati-windup strategy if the give integrator exceeds specified limits.

InOut:

Scope	Name	Type	Comment
Input	itfIntegrator	IIntegrator	Represents the integrator on which an anti wind-up strategy should be applied

Controller

This directory contains all provided controllers.

- BangBangController
  - BangBangController (FB)
    - BangBangController.CyclicAction (METH)
    - BangBangController.OutputValue (PROP)
    - BangBangController.ResetAction (METH)
  - BangBangControllerWithTimeHysteresis (FB)
    - BangBangControllerWithTimeHysteresis.CleanupAction (METH)
    - BangBangControllerWithTimeHysteresis.CyclicAction (METH)
    - BangBangControllerWithTimeHysteresis.StartAction (METH)
  - BangBangControllerWithValueHysteresis (FB)
    - BangBangControllerWithValueHysteresis.CyclicAction (METH)
    - BangBangControllerWithValueHysteresis.OutputValue (PROP)
    - BangBangControllerWithValueHysteresis.ResetAction (METH)
- Controller_Base (FB)
  - Controller_Base.ActualValue (PROP)
  - Controller_Base.CleanupAction (METH)
  - Controller_Base.CycleInterval (PROP)
  - Controller_Base.CyclicAction (METH)
  - Controller_Base.ExecuteController (METH)
  - Controller_Base.GetCurrentError (METH)
  - Controller_Base.GetTaskCycleInterval (METH)
  - Controller_Base.HandleLimitsExceeded (METH)
  - Controller_Base.LimitsActive (PROP)
  - Controller_Base.ManualModeActive (PROP)
  - Controller_Base.MaxValue (PROP)
  - Controller_Base.MinValue (PROP)
  - Controller_Base.Offset (PROP)
  - Controller_Base.OutputValue (PROP)
  - Controller_Base.ResetAction (METH)
  - Controller_Base.SetManual (METH)
  - Controller_Base.SetPoint (PROP)
  - Controller_Base.StartAction (METH)
- Controller_P (FB)
  - Controller_P.ExecuteController (METH)
  - Controller_P.KP (PROP)
- Controller_PD (FB)
  - Controller_PD.CleanupAction (METH)
  - Controller_PD.ExecuteController (METH)
  - Controller_PD.KD (PROP)
  - Controller_PD.KP (PROP)
  - Controller_PD.ResetAction (METH)
  - Controller_PD.StartAction (METH)
- Controller_PI (FB)
  - Controller_PI.CleanupAction (METH)
  - Controller_PI.ExecuteController (METH)
  - Controller_PI.KI (PROP)
  - Controller_PI.KP (PROP)
  - Controller_PI.ResetAction (METH)
  - Controller_PI.StartAction (METH)
- Controller_PID (FB)
  - Controller_PID.CleanupAction (METH)
  - Controller_PID.ExecuteController (METH)
  - Controller_PID.KD (PROP)
  - Controller_PID.KI (PROP)
  - Controller_PID.KP (PROP)
  - Controller_PID.ResetAction (METH)
  - Controller_PID.StartAction (METH)

- ThreePointController
  - ThreePointController (FB)
    - ThreePointController.CyclicAction (METH)
    - ThreePointController.OutputValue (PROP)
    - ThreePointController.ResetAction (METH)
  - ThreePointControllerWithValueHysteresis (FB)
    - ThreePointControllerWithValueHysteresis.CyclicAction (METH)
    - ThreePointControllerWithValueHysteresis.OutputValue (PROP)
    - ThreePointControllerWithValueHysteresis.ResetAction (METH)

BangBangController

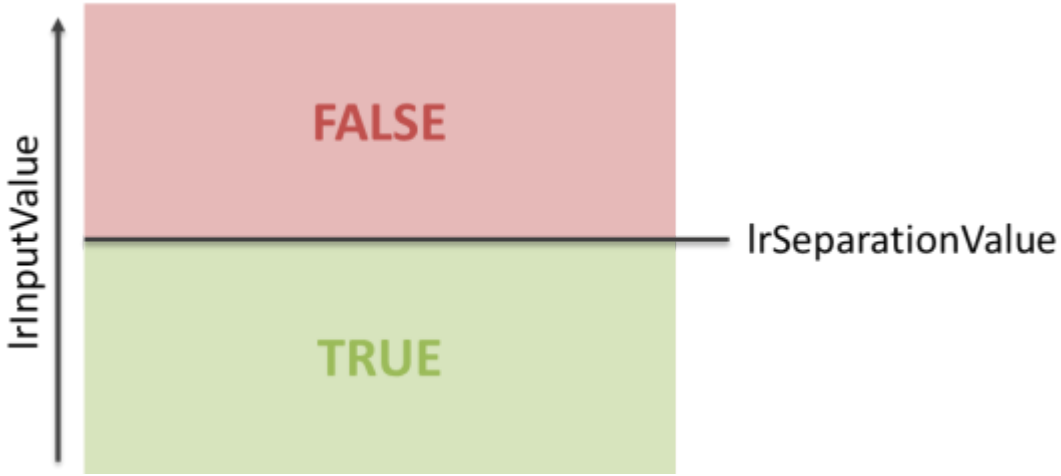
- BangBangController (FB)
  - BangBangController.CyclicAction (METH)
  - BangBangController.OutputValue (PROP)
  - BangBangController.ResetAction (METH)
- BangBangControllerWithTimeHysteresis (FB)
  - BangBangControllerWithTimeHysteresis.CleanupAction (METH)
  - BangBangControllerWithTimeHysteresis.CyclicAction (METH)
  - BangBangControllerWithTimeHysteresis.StartAction (METH)
- BangBangControllerWithValueHysteresis (FB)
  - BangBangControllerWithValueHysteresis.CyclicAction (METH)
  - BangBangControllerWithValueHysteresis.OutputValue (PROP)
  - BangBangControllerWithValueHysteresis.ResetAction (METH)

BangBangController (FB)

FUNCTION_BLOCK BangBangController EXTENDS CBML.LConC IMPLEMENTS IValueProvider

Represents a bang bang controller

A bang bang controller divides a one dimesional input space into two sections separated by `lrSeparationValue`. It maps a specific input (`lrInputValue`) to a boolean output indictating the section it belongs to.



InOut:

Scope	Name	Type	Initial	Comment	Inherited from
Input	xEnable	BOOL		`TRUE`: Activates the defined operation `FALSE`: Aborts/resets the defined operation	LConC
Output	xBusy	BOOL		`TRUE`: Operation is running	LConC
	xError	BOOL		`TRUE`: Error condition reached	LConC
Input	lrInputValue	LREAL		Represents the input value of the bang bang controller	
	lrSeparationValue	LREAL		Represents the value where the input space is separated	
Output	xOutput	BOOL		Represents the boolean output value	
	eErrorID	Controller_Error	Controller_Error.NO_ERROR	The current error. Only valid if `xError` is `TRUE`.	

- BangBangController.CyclicAction (METH)
- BangBangController.OutputValue (PROP)
- BangBangController.ResetAction (METH)

BangBangController.CyclicAction (METH)

METHOD CyclicAction

InOut:

Scope	Name	Type
Input	itfTimingController	CBML.ITimingController
Output	xComplete	BOOL
	iErrorID	INT

BangBangController.OutputValue (PROP)

PROPERTY OutputValue : LREAL

BangBangController.ResetAction (METH)

METHOD ResetAction

InOut:

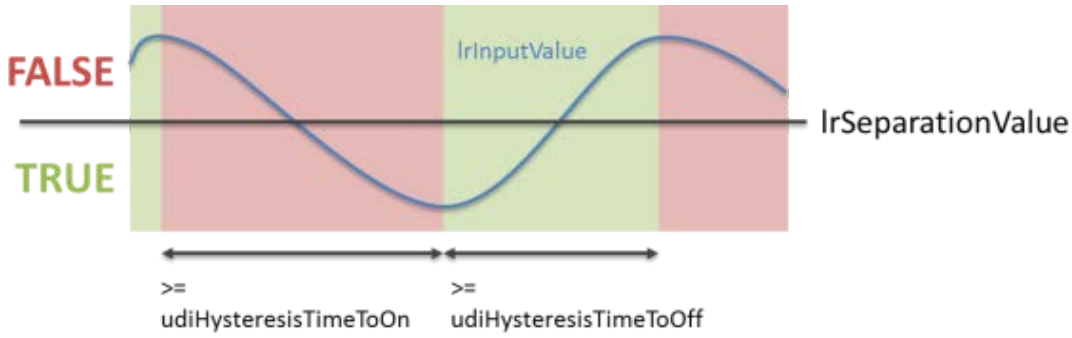
Scope	Name	Type
Output	xComplete	BOOL

BangBangControllerWithTimeHysteresis (FB)

FUNCTION_BLOCK BangBangControllerWithTimeHysteresis EXTENDS BangBangController

Represents a bang bang controller with time hysteresis

A bang bang controller with time hysteresis has basically the same behaviour as a BangBangController. An important feature of this function block (FB) is that an additional time hysteresis is taken into account. This means that the output of the FB does not only depend on the input, but also on the elapsed time since the last output change.



InOut:

Scope	Name	Type	Initial	Comment	Inherited from
Input	xEnable	BOOL		TRUE: Activates the defined operation FALSE: Aborts/resets the defined operation	LConC
Output	xBusy	BOOL		TRUE: Operation is running	LConC
	xError	BOOL		TRUE: Error condition reached	LConC
Input	lrInputValue	LREAL		Represents the input value of the bang bang controller	BangBangController
	lrSeparationValue	LREAL		Represents the value where the input space is separated	BangBangController
Output	xOutput	BOOL		Represents the boolean output value	BangBangController
	eErrorID	Controller_Error	Controller_Error.NO_ERROR	The current error. Only valid if xError is TRUE.	BangBangController
Input	udiHysteresisTimeToOn	UDINT		Represents the minimum time [microseconds] until the output value can again switch to TRUE after a switch to FALSE	
	udiHysteresisTimeToOff	UDINT		Represents the minimum time [microseconds] until the output	

Scope	Name	Type	Initial	Comment	Inherited from
				value can again switch to FALSE after a switch to TRUE	

BangBangControllerWithTimeHysteresis.CleanupAction (METH)

METHOD CleanupAction

InOut:

Scope	Name	Type
Input	xAbortProposed	BOOL
	iErrorIDProposed	INT
Output	xComplete	BOOL
	xAbort	BOOL
	iErrorID	INT

BangBangControllerWithTimeHysteresis.CyclicAction (METH)

METHOD CyclicAction

InOut:

Scope	Name	Type
Input	itfTimingController	CBML.ITimingController
Output	xComplete	BOOL
	iErrorID	INT

BangBangControllerWithTimeHysteresis.StartAction (METH)

METHOD StartAction

InOut:

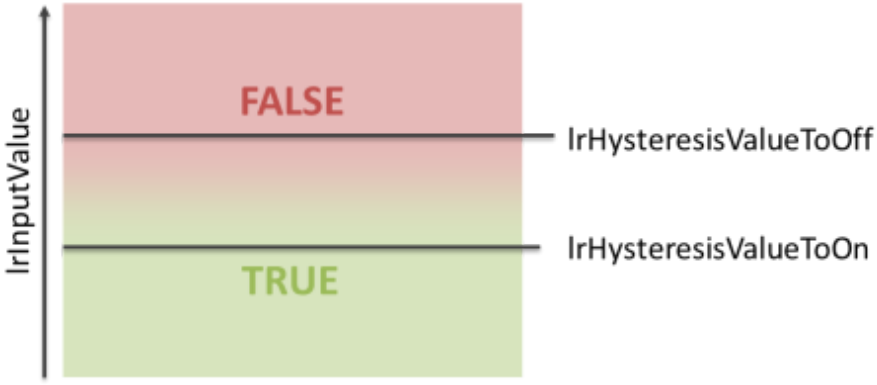
Scope	Name	Type
Output	xComplete	BOOL
	iErrorID	INT

BangBangControllerWithValueHysteresis (FB)

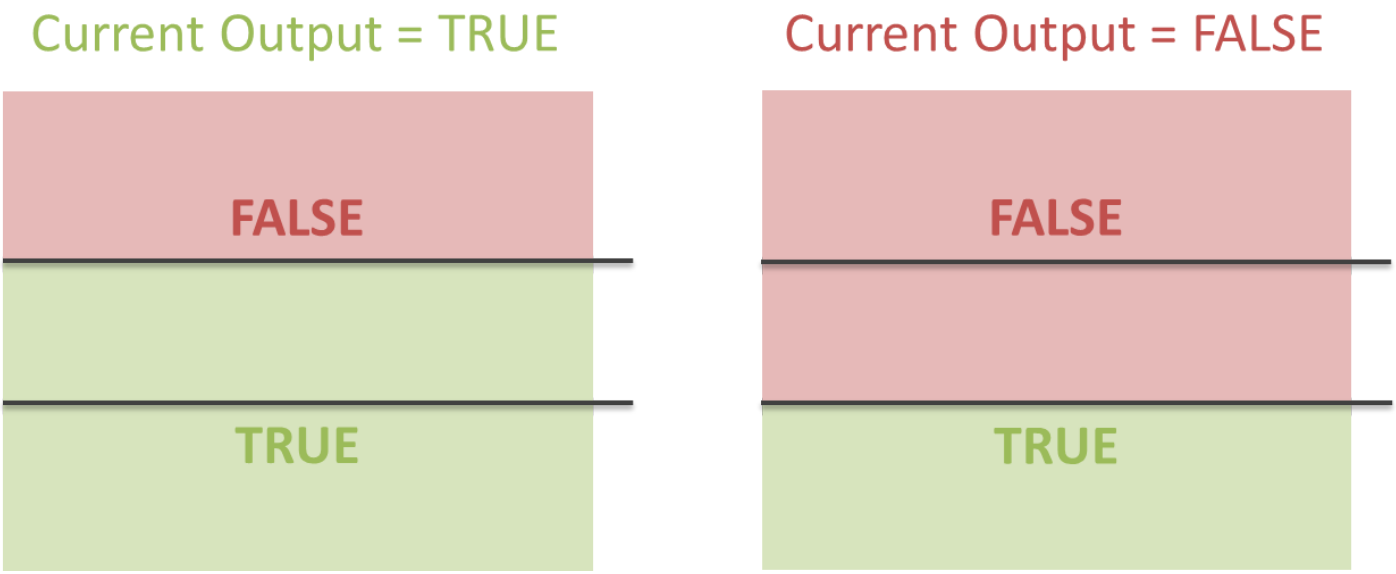
FUNCTION_BLOCK BangBangControllerWithValueHysteresis EXTENDS CBML.LConC

Represents a two point controller with value hysteresis

The two point controller with value hysteresis divides a one dimesional input space into two sections. The partitioning of the input space depends on the last output value and a hysteresis value. A specific input (`lrInputValue`) is mapped to a boolean output indicating the section it is belonging to.



The following figure shows the partitioning of the input space based on the current output value:



InOut:

Scope	Name	Type	Initial	Comment	Inherited from
Input	xEnable	BOOL		`TRUE`: Activates the defined operation `FALSE`: Aborts/resets the defined operation	LConC
Output	xBusy	BOOL		`TRUE`: Operation is running	LConC
	xError	BOOL		`TRUE`: Error condition reached	LConC
Input	lrInputValue	LREAL		Represents the input value of the two point controller	
	lrHysteresisValueToOn	LREAL		If `lrInputValue` falls below `lrHysteresisValueToOn` then `xOutput` is set to `TRUE`	

Scope	Name	Type	Initial	Comment	Inherited from
	lrHysteresisValueToOff	LREAL		If `lrInputValue` exceeds `lrHysteresisValueToOff` then `xOutput` is set to `FALSE`	
Output	xOutput	BOOL		Represents the boolean output value	
	eErrorID	Controller_Error	Controller_Error.NO_ERROR	The current error. Only valid if `xError` is `TRUE`.	

BangBangControllerWithValueHysteresis.CyclicAction (METH)

METHOD CyclicAction

InOut:

Scope	Name	Type
Input	itfTimingController	CBML.ITimingController
Output	xComplete	BOOL
	iErrorID	INT

BangBangControllerWithValueHysteresis.OutputValue (PROP)

PROPERTY OutputValue : LREAL

BangBangControllerWithValueHysteresis.ResetAction (METH)

METHOD ResetAction

InOut:

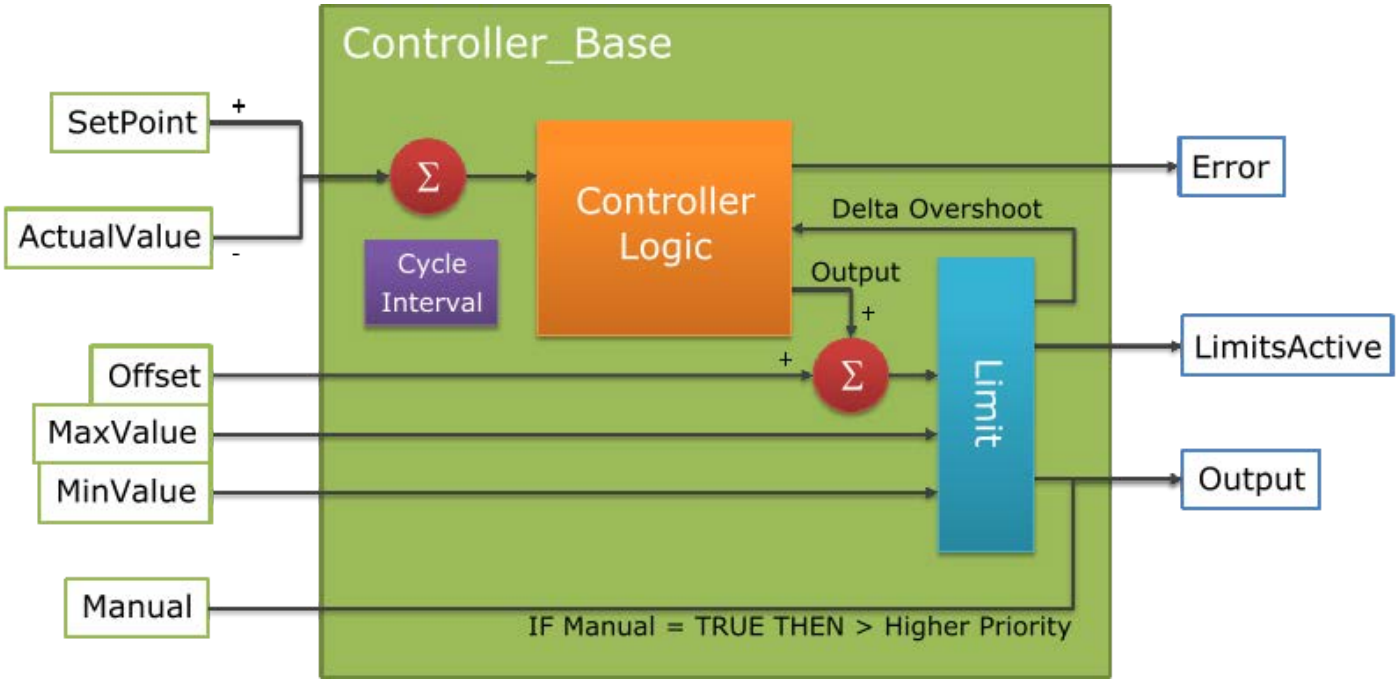
Scope	Name	Type
Output	xComplete	BOOL

Controller_Base (FB)

FUNCTION_BLOCK ABSTRACT Controller_Base EXTENDS CBML.LConC IMPLEMENTS IController

Represents the base controller function block

It extends the LConC of the Common Behaviour Model and includes common methods and properties for any pursuing controller. The structure of the base controller is presented in the following figure.

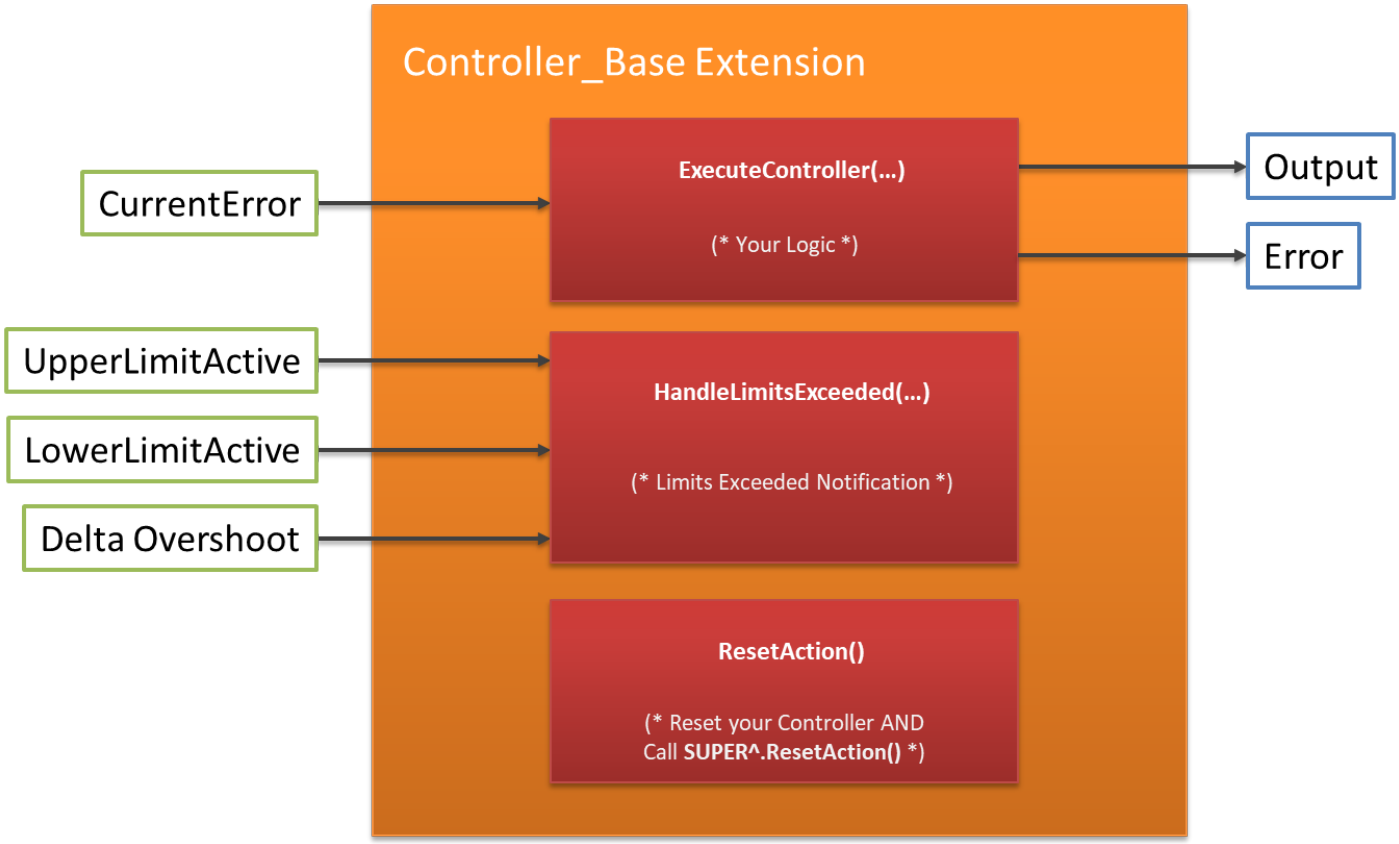


The base controller provides basic functionality, like adding an offset, limiting the output and handling the manual mode. Additionally, it offers the cycle interval time of the task it is working in.

> **Note**
>
> Individually designed controllers should extend this base function block.

In order to build your own controller, this function block must be extended. Afterwards, relevant functions of the base can be overwritten in order to obtain an individual controller behaviour. For more details refer to the figure below. It describes the `ControllerLogic` component of the figure above in more detail.

> **Note**
>
> Individually designed controllers have to call `SUPER^();` during their function block call.



Note

Limitation is done by the base controller function block. The function `HandleLimitsExceeded(...)` is only a notification.

The library provides tools that facilitate the construction of individual controllers. These include techniques for integral and derivative approximations. For more details refer to:

- Differentiators: Differentiator_Base
- Integrators: Integrator_Base.

InOut:

Scope	Name	Type	Comment	Inherited from
Input	xEnable	BOOL	`TRUE`: Activates the defined operation `FALSE`: Aborts/resets the defined operation	LConC
Output	xBusy	BOOL	`TRUE`: Operation is running	LConC
	xError	BOOL	`TRUE`: Error condition reached	LConC
Input	lrActualValue	LREAL	Represents the measured actual value	
	lrSetPoint	LREAL	Represents the desired set point	
	xManual	BOOL	`TRUE`: Activates the manual setting of `lrOutput` `FALSE`: `lrOutput` is calculated by the controller	
	lrManualValue	LREAL	Represents the value which is forwarded to `lrOutput` if `xManual` is `TRUE`	
	lrOffset	LREAL	Represents an offset that is added to `lrOutput` (no influence if `xManual` is `TRUE`)	
	lrMinValue	LREAL	Represents the lower bound of `lrOutput` (No Limitation: Set `lrMinValue == lrMaxValue == 0`)	
	lrMaxValue	LREAL	Represents the upper bound of `lrOutput` (No Limitation: Set `lrMinValue == lrMaxValue == 0`)	
Output	lrOutput	LREAL	Represents the correction value of the controller (`lrOutput` is always between `lrMinValue` and `lrMaxValue`)	
	xLimitsActive	BOOL	`TRUE`: Indetifies that the calculated `lrOutput` has exceeded the boundaries `FALSE`: `lrOutput` lies between the boundaries	
	eErrorID	Controller_Error		

Controller_Base.ActualValue (PROP)

PROPERTY ActualValue : LREAL

Returns or sets the actually measured process variable

Controller_Base.CleanupAction (METH)

METHOD CleanupAction

InOut:

Scope	Name	Type
Input	xAbortProposed	BOOL
	iErrorIDProposed	INT
Output	xComplete	BOOL
	xAbort	BOOL
	iErrorID	INT

Controller_Base.CycleInterval (PROP)

PROPERTY PROTECTED CycleInterval : LREAL

Controller_Base.CyclicAction (METH)

METHOD CyclicAction

InOut:

Scope	Name	Type
Input	itfTimingController	CBML.ITimingController
Output	xComplete	BOOL
	iErrorID	INT

Controller_Base.ExecuteController (METH)

METHOD PROTECTED ABSTRACT ExecuteController

This method is called when the manual mode is deactivated. It contains the controller logic and outputs the results.

InOut:

Scope	Name	Type	Comment
Input	lrCurrentError	LREAL	Represents the current difference between SetPoint and ActualValue
Output	lrControllerOutput	LREAL	Represents the result of the controller computation
	iControllerErrorID	INT	Indicates if an error occured during the computation (e.g Overflow)

Controller_Base.GetCurrentError (METH)

METHOD PROTECTED FINAL GetCurrentError

This method calculates the error between the SetPoint and the current value

Controller_Base.GetTaskCycleInterval (METH)

METHOD FINAL PROTECTED GetTaskCycleInterval : BOOL

InOut:

Scope	Name	Type
Return	GetTaskCycleInterval	BOOL

Controller_Base.HandleLimitsExceeded (METH)

METHOD PROTECTED HandleLimitsExceeded

This method is called when the calculated output value exceeds the specified limits.

InOut:

Scope	Name	Type	Comment
Input	xUpperLimitActive	BOOL	Indicates whether the upper bound is exceeded
	xLowerLimitActive	BOOL	Indicates whether the lower bound is exceeded
	lrDeltaOvershoot	LREAL	Represents the difference between the output limit and the calculated output.

Controller_Base.LimitsActive (PROP)

PROPERTY LimitsActive : BOOL

Controller_Base.ManualModeActive (PROP)

PROPERTY ManualModeActive : BOOL

Indicates if the manual mode is active

Controller_Base.MaxValue (PROP)

PROPERTY MaxValue : LREAL

Returns or sets the upper bound of the controller output

Controller_Base.MinValue (PROP)

PROPERTY MinValue : LREAL

Returns or sets the lower bound of the controller output

Controller_Base.Offset (PROP)

PROPERTY Offset : LREAL

Returns or sets an offset

Controller_Base.OutputValue (PROP)

PROPERTY OutputValue : LREAL

Controller_Base.ResetAction (METH)

METHOD ResetAction

The `ResetAction` is running until the output `xComplete` is `TRUE`.

InOut:

Scope	Name	Type
Output	xComplete	BOOL

Controller_Base.SetManual (METH)

METHOD SetManual

This method activates or deactivates the manual mode of the controller For more details see: Controller_Base

InOut:

Scope	Name	Type	Comment
Input	xManual	BOOL	Indicates if the manual mode should be activated or deactivated
	lrValue	LREAL	Represents the value which is used in the manual mode

Controller_Base.SetPoint (PROP)

PROPERTY SetPoint : LREAL

Returns or sets the desired set point

Controller_Base.StartAction (METH)

METHOD StartAction

This method computes the cycle interval of the task in which the controller is working.

InOut:

Scope	Name	Type
Output	xComplete	BOOL
	iErrorID	INT

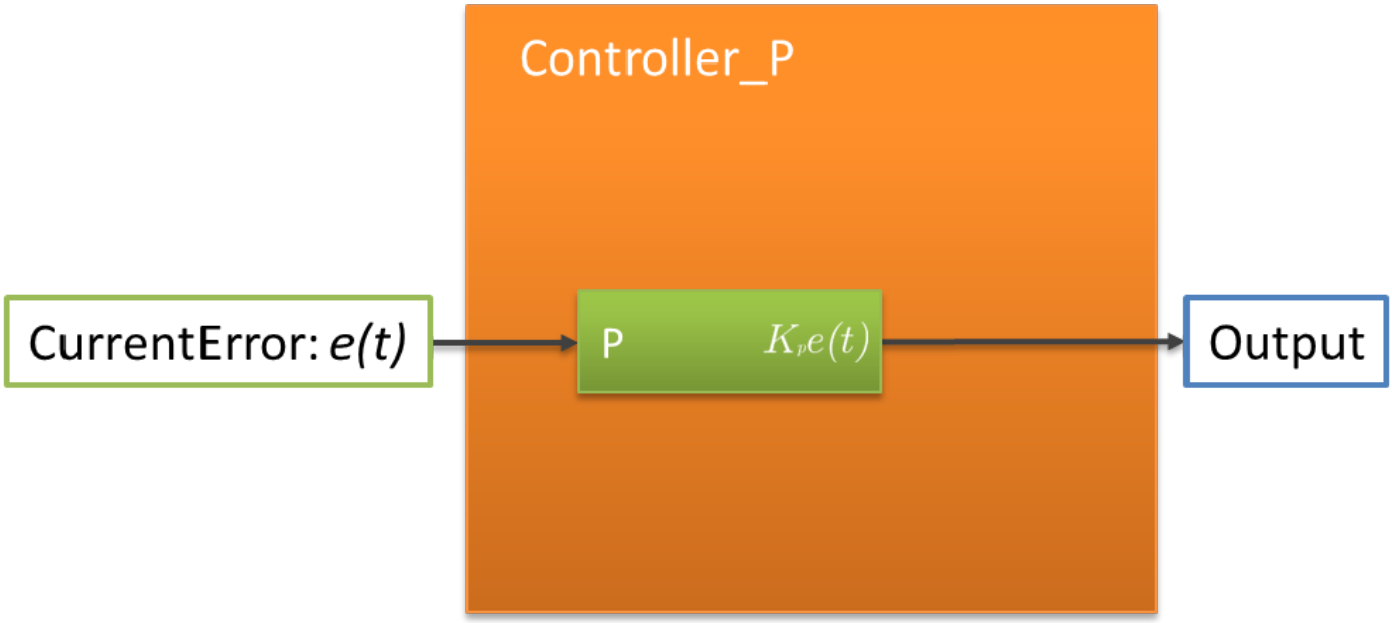
Controller_P (FB)

FUNCTION_BLOCK Controller_P EXTENDS Controller_Base IMPLEMENTS IController_P

Represents a P controller

The P controller uses an error value $e(t)$ which is continuously calculated by the Controller_Base function block. It represents the difference between a desired set point and a measured process variable. The P controller applies a correction based on a proportional term (sometimes denoted P respectively) which gives its name to the controller type.

For more information refer to: Controller_PID



InOut:

Scope	Name	Type	Comment	Inherited from
Input	xEnable	BOOL	TRUE: Activates the defined operation FALSE: Aborts/resets the defined operation	LConC
Output	xBusy	BOOL	TRUE: Operation is running	LConC
	xError	BOOL	TRUE: Error condition reached	LConC
Input	lrActualValue	LREAL	Represents the measured actual value	Controller_Base
	lrSetPoint	LREAL	Represents the desired set point	Controller_Base
	xManual	BOOL	TRUE: Activates the manual setting of `lrOutput` FALSE: `lrOutput` is calculated by the controller	Controller_Base
	lrManualValue	LREAL	Represents the value which is forwarded to `lrOutput` if `xManual` is TRUE	Controller_Base
	lrOffset	LREAL	Represents an offset that is added to `lrOutput` (no influence if `xManual` is TRUE)	Controller_Base
	lrMinValue	LREAL	Represents the lower bound of `lrOutput` (No Limitation: Set `lrMinValue == lrMaxValue == 0`)	Controller_Base
	lrMaxValue	LREAL	Represents the upper bound of `lrOutput` (No Limitation: Set `lrMinValue == lrMaxValue == 0`)	Controller_Base
Output	lrOutput	LREAL	Represents the correction value of the controller (`lrOutput` is always between `lrMinValue` and `lrMaxValue`)	Controller_Base
	xLimitsActive	BOOL	TRUE: Indetifies that the calculated `lrOutput` has exceeded the boundaries FALSE: `lrOutput` lies between the boundaries	Controller_Base
	eErrorID	Controller_Error		Controller_Base
Input	lrKP	LREAL	Represents the parameter of the proportional term	

Controller_P.ExecuteController (METH)

METHOD PROTECTED ExecuteController

This method is called when the manual mode is deactivated. It contains the controller logic and outputs the results.

InOut:

Scope	Name	Type	Comment
Input	lrCurrentError	LREAL	Represents the current difference between SetPoint and ActualValue
Output	lrControllerOutput	LREAL	Represents the result of the controller computation
	iControllerErrorID	INT	Indicates if an error occured during the computation (e.g Overflow)

Controller_P.KP (PROP)

PROPERTY KP : LREAL

Returns or sets the proportional parameter

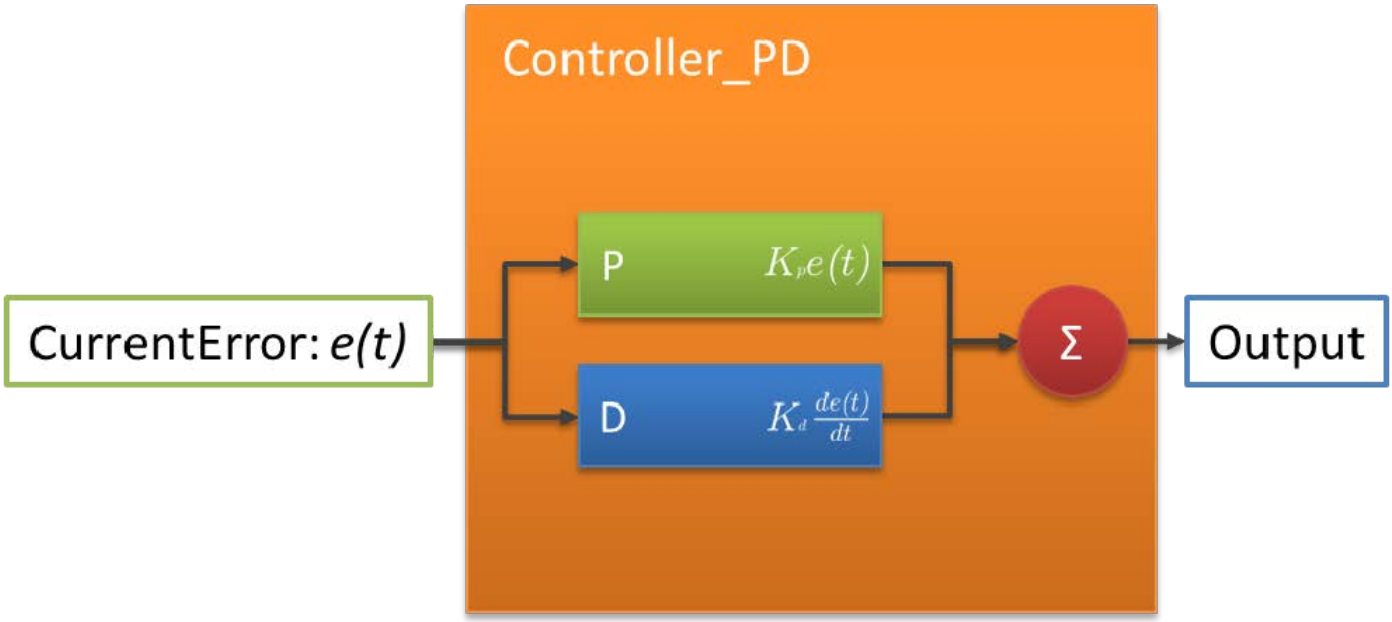
Controller_PD (FB)

FUNCTION_BLOCK Controller_PD EXTENDS Controller_Base IMPLEMENTS IController_P, IController_D

Represents a PD controller

The PD controller uses an error value $e(t)$ which is continuously calculated by the Controller_Base function block. It represents the difference between a desired set point and a measured process variable. The PD controller applies a correction based on proportional and derivative terms (sometimes denoted P and D respectively) which give their name to the controller type.

For more information refer to: Controller_PID



InOut:

Scope	Name	Type	Comment	Inherited from
Input	xEnable	BOOL	TRUE: Activates the defined operation FALSE: Aborts/resets the defined operation	LConC
Output	xBusy	BOOL	TRUE: Operation is running	LConC
	xError	BOOL	TRUE: Error condition reached	LConC
Input	lrActualValue	LREAL	Represents the measured actual value	Controller_Base
	lrSetPoint	LREAL	Represents the desired set point	Controller_Base
	xManual	BOOL	TRUE: Activates the manual setting of `lrOutput` FALSE: `lrOutput` is calculated by the controller	Controller_Base
	lrManualValue	LREAL	Represents the value which is forwarded to `lrOutput` if `xManual` is TRUE	Controller_Base
	lrOffset	LREAL	Represents an offset that is added to `lrOutput` (no influence if `xManual` is TRUE)	Controller_Base
	lrMinValue	LREAL	Represents the lower bound of `lrOutput` (No Limitation: Set lrMinValue == lrMaxValue == 0)	Controller_Base
	lrMaxValue	LREAL	Represents the upper bound of `lrOutput` (No Limitation: Set lrMinValue == lrMaxValue == 0)	Controller_Base
Output	lrOutput	LREAL	Represents the correction value of the controller (`lrOutput` is always between `lrMinValue` and `lrMaxValue`)	Controller_Base
	xLimitsActive	BOOL	TRUE: Indetifies that the calculated `lrOutput` has exceeded the boundaries FALSE: `lrOutput` lies between the boundaries	Controller_Base
	eErrorID	Controller_Error		Controller_Base
Input	itfDifferentiator	IDifferentiator	Represents the desired derivative approximation	
	lrKP	LREAL	Represents the parameter of the proportional term	
	lrKD	LREAL	Represents the parameter of the derivative term	

Controller_PD.CleanupAction (METH)

METHOD CleanupAction

InOut:

Scope	Name	Type
Input	xAbortProposed	BOOL
	iErrorIDProposed	INT
Output	xComplete	BOOL
	xAbort	BOOL
	iErrorID	INT

Controller_PD.ExecuteController (METH)

METHOD PROTECTED ExecuteController

This method is called when the manual mode is deactivated. It contains the controller logic and outputs the results.

InOut:

Scope	Name	Type	Comment
Input	lrCurrentError	LREAL	Represents the current difference between SetPoint and ActualValue
Output	lrControllerOutput	LREAL	Represents the result of the controller computation
	iControllerErrorID	INT	Indicates if an error occured during the computation (e.g Overflow)

Controller_PD.KD (PROP)

PROPERTY KD : LREAL

Returns or sets the derivative parameter

Controller_PD.KP (PROP)

PROPERTY KP : LREAL

Returns or sets the proportional parameter

Controller_PD.ResetAction (METH)

METHOD ResetAction

The `ResetAction` is running until the output `xComplete` is `TRUE`.

InOut:

Scope	Name	Type
Output	xComplete	BOOL

Controller_PD.StartAction (METH)

METHOD StartAction

This method computes the cycle interval of the task in which the controller is working.

InOut:

Scope	Name	Type
Output	xComplete	BOOL
	iErrorID	INT

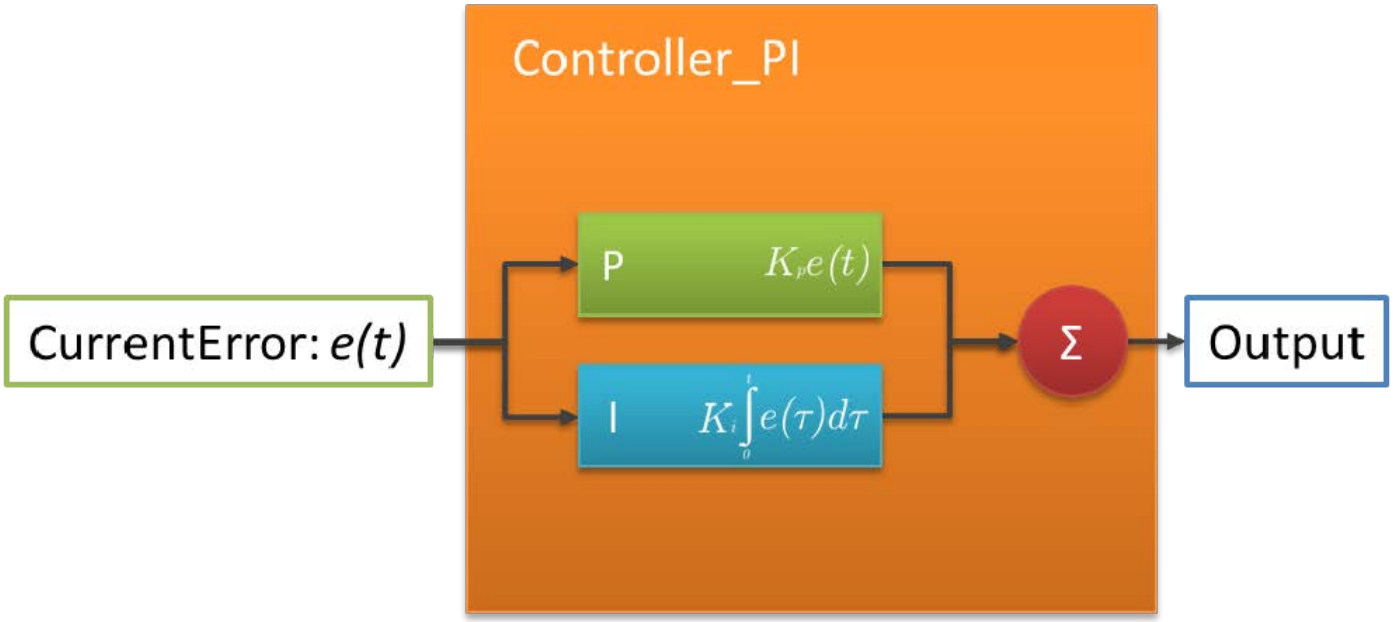
Controller_PI (FB)

FUNCTION_BLOCK Controller_PI EXTENDS Controller_Base IMPLEMENTS IController_P, IController_I

Represents a PI controller

The PI controller uses an error value $e(t)$ which is continuously calculated by the Controller_Base function block. It represents the difference between a desired set point and a measured process variable. The PI controller applies a correction based on proportional and integral terms (sometimes denoted P and I) which give their name to the controller type.

For more information refer to: Controller_PID



InOut:

Scope	Name	Type	Comment	Inherited from
Input	xEnable	BOOL	TRUE: Activates the defined operation FALSE: Aborts/resets the defined operation	LConC
Output	xBusy	BOOL	TRUE: Operation is running	LConC
	xError	BOOL	TRUE: Error condition reached	LConC
Input	lrActualValue	LREAL	Represents the measured actual value	Controller_Base
	lrSetPoint	LREAL	Represents the desired set point	Controller_Base
	xManual	BOOL	TRUE: Activates the manual setting of lrOutput FALSE: lrOutput is calculated by the controller	Controller_Base
	lrManualValue	LREAL	Represents the value which is forwarded to lrOutput if xManual is TRUE	Controller_Base
	lrOffset	LREAL	Represents an offset that is added to lrOutput (no influence if xManual is TRUE)	Controller_Base
	lrMinValue	LREAL	Represents the lower bound of lrOutput (No Limitation: Set lrMinValue == lrMaxValue == 0)	Controller_Base
	lrMaxValue	LREAL	Represents the upper bound of lrOutput (No Limitation: Set lrMinValue == lrMaxValue == 0)	Controller_Base
Output	lrOutput	LREAL	Represents the correction value of the controller (lrOutput is always between lrMinValue and lrMaxValue)	Controller_Base
	xLimitsActive	BOOL	TRUE: Indetifies that the calculated lrOutput has exceeded the boundaries FALSE: lrOutput lies between the boundaries	Controller_Base
	eErrorID	Controller_Error		Controller_Base
Input	itfIntegrator	IIntegrator	Represents the desired integral approximation	
	lrKP	LREAL	Represents the parameter of the proportional term	
	lrKI	LREAL	Represents the parameter of the integral term	

Controller_PI.CleanupAction (METH)

METHOD CleanupAction

InOut:

Scope	Name	Type
Input	xAbortProposed	BOOL
	iErrorIDProposed	INT
Output	xComplete	BOOL
	xAbort	BOOL
	iErrorID	INT

Controller_PI.ExecuteController (METH)

METHOD PROTECTED ExecuteController

This method is called when the manual mode is deactivated. It contains the controller logic and outputs the results.

InOut:

Scope	Name	Type	Comment
Input	lrCurrentError	LREAL	Represents the current difference between SetPoint and ActualValue
Output	lrControllerOutput	LREAL	Represents the result of the controller computation
	iControllerErrorID	INT	Indicates if an error occured during the computation (e.g Overflow)

Controller_PI.KI (PROP)

PROPERTY KI : LREAL

Returns or sets the integral parameter

Controller_PI.KP (PROP)

PROPERTY KP : LREAL

Returns or sets the proportional parameter

Controller_PI.ResetAction (METH)

METHOD ResetAction

The `ResetAction` is running until the output `xComplete` is `TRUE`.

InOut:

Scope	Name	Type
Output	xComplete	BOOL

Controller_PI.StartAction (METH)

METHOD StartAction

This method computes the cycle interval of the task in which the controller is working.

InOut:

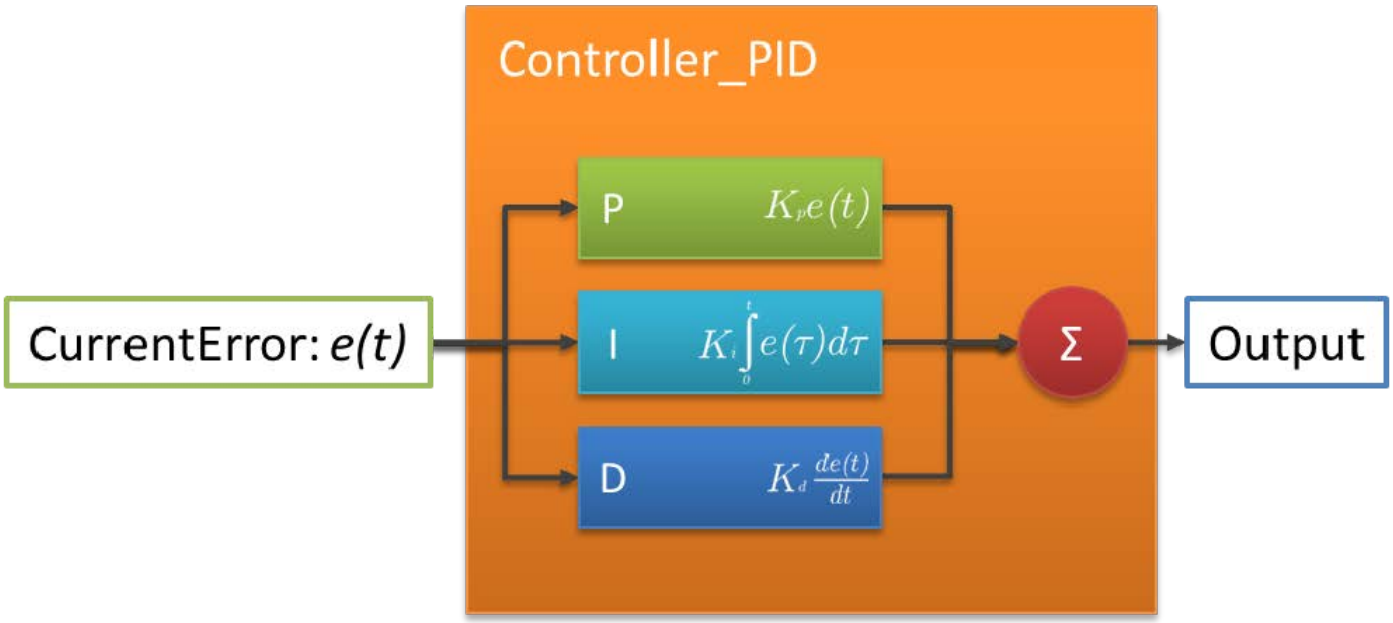
Scope	Name	Type
Output	xComplete	BOOL
	iErrorID	INT

Controller_PID (FB)

FUNCTION_BLOCK Controller_PID EXTENDS Controller_Base IMPLEMENTS IController_P, IController_I, IController_D

Represents a PID controller

The PID controller uses an error value $e(t)$ which is continuously calculated by the Controller_Base function block. It represents the difference between a desired set point and a measured process variable. The PID controller applies a correction based on proportional, integral, and derivative terms (sometimes denoted P, I, and D respectively) which give their name to the controller type.



- P accounts for present values of the error. For example, if the error is large and positive, the control output will also be large and positive.
- I accounts for past values of the error. For example, if the current output is not sufficiently strong, the integral of the error will accumulate over time, and the controller will respond by applying a stronger action.
- D accounts for possible future trends of the error, based on its current rate of change.

As a PID controller relies only on the actually measured process variable, not on knowledge of the underlying process, it is broadly applicable. By tuning the three parameters of the model, a PID controller can deal with specific process requirements. The response of the controller can be described in terms of its responsiveness to an error, of the degree to which the system overshoots a setpoint, and of the degree of any system oscillation. The use of the PID algorithm does not guarantee optimal control of the system or even its stability.

InOut:

Scope	Name	Type	Comment	Inherited from
Input	xEnable	BOOL	`TRUE`: Activates the defined operation `FALSE`: Aborts/resets the defined operation	LConC
Output	xBusy	BOOL	`TRUE`: Operation is running	LConC
	xError	BOOL	`TRUE`: Error condition reached	LConC
Input	lrActualValue	LREAL	Represents the measured actual value	Controller_Base
	lrSetPoint	LREAL	Represents the desired set point	Controller_Base
	xManual	BOOL	`TRUE`: Activates the manual setting of `lrOutput` `FALSE`: `lrOutput` is calculated by the controller	Controller_Base
	lrManualValue	LREAL	Represents the value which is forwarded to `lrOutput` if `xManual` is `TRUE`	Controller_Base
	lrOffset	LREAL	Represents an offset that is added to `lrOutput` (no influence if `xManual` is `TRUE`)	Controller_Base
	lrMinValue	LREAL	Represents the lower bound of `lrOutput` (No Limitation: Set `lrMinValue == lrMaxValue == 0`)	Controller_Base
	lrMaxValue	LREAL	Represents the upper bound of `lrOutput` (No Limitation: Set `lrMinValue == lrMaxValue == 0`)	Controller_Base
Output	lrOutput	LREAL	Represents the correction value of the controller (`lrOutput` is always between `lrMinValue` and `lrMaxValue`)	Controller_Base

Scope	Name	Type	Comment	Inherited from
	xLimitsActive	BOOL	`TRUE`: Indetifies that the calculated `lrOutput` has exceeded the boundaries `FALSE`: `lrOutput` lies between the boundaries	Controller_Base
	eErrorID	Controller_Error		Controller_Base
Input	itfIntegrator	IIntegrator	Represents the desired integral approximation	
	itfDifferentiator	IDifferentiator	Represents the desired derivative approximation	
	lrKP	LREAL	Represents the parameter of the proportional term	
	lrKI	LREAL	Represents the parameter of the integral term	
	lrKD	LREAL	Represents the parameter of the derivative term	

Controller_PID.CleanupAction (METH)

METHOD CleanupAction

InOut:

Scope	Name	Type
Input	xAbortProposed	BOOL
	iErrorIDProposed	INT
Output	xComplete	BOOL
	xAbort	BOOL
	iErrorID	INT

Controller_PID.ExecuteController (METH)

METHOD PROTECTED ExecuteController

This method is called when the manual mode is deactivated. It contains the controller logic and outputs the results.

InOut:

Scope	Name	Type	Comment
Input	lrCurrentError	LREAL	Represents the current difference between SetPoint and ActualValue
Output	lrControllerOutput	LREAL	Represents the result of the controller computation
	iControllerErrorID	INT	Indicates if an error occured during the computation (e.g Overflow)

Controller_PID.KD (PROP)

PROPERTY KD : LREAL

Returns or sets the derivative parameter

Controller_PID.KI (PROP)

PROPERTY KI : LREAL

Returns or sets the integral parameter

Controller_PID.KP (PROP)

PROPERTY KP : LREAL

Returns or sets the proportional parameter

Controller_PID.ResetAction (METH)

METHOD ResetAction

The `ResetAction` is running until the output `xComplete` is `TRUE`.

InOut:

Scope	Name	Type
Output	xComplete	BOOL

Controller_PID.StartAction (METH)

METHOD StartAction

This method computes the cycle interval of the task in which the controller is working.

InOut:

Scope	Name	Type
Output	xComplete	BOOL
	iErrorID	INT

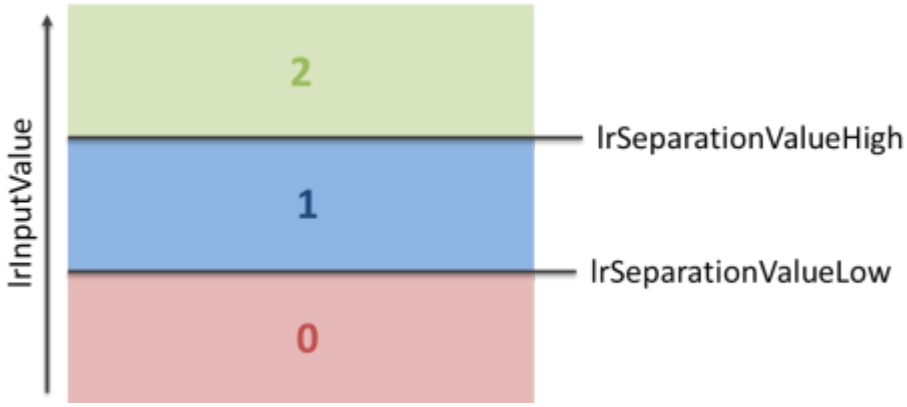
ThreePointController

ThreePointController (FB)

FUNCTION_BLOCK ThreePointController EXTENDS CBML.LConC IMPLEMENTS IValueProvider

Represents a three point controller

The three point controller divides a one dimesional input space into three sections separated by `lrSeparationValueLow` and `lrSeparationValueHigh`. It maps a specific input (`lrInputValue`) to an integer output [0,1,2] indictating the section it is belonging to.



InOut:

Scope	Name	Type	Initial	Comment	Inherited from
Input	xEnable	BOOL		`TRUE`: Activates the defined operation `FALSE`: Aborts/resets the defined operation	LConC
Output	xBusy	BOOL		`TRUE`: Operation is running	LConC
	xError	BOOL		`TRUE`: Error condition reached	LConC
Input	lrInputValue	LREAL		Represents the input value of the three point controller	
	lrSeparationValueHigh	LREAL		Represents the upper separation value (above: `xOutput` = 2, below: `xOutput` = 1)	
	lrSeparationValueLow	LREAL		Represents the lower separation value (above: `xOutput` = 1, below: `xOutput` = 0)	
Output	iOutput	INT		Represents the integer output value (possible values [0,1,2])	
	eErrorID	Controller_Error	Controller_Error.NO_ERROR	The current error. Only valid if `xError` is `TRUE`.	

ThreePointController.CyclicAction (METH)

METHOD CyclicAction

InOut:

Scope	Name	Type
Input	itfTimingController	CBML.ITimingController
Output	xComplete	BOOL
	iErrorID	INT

ThreePointController.OutputValue (PROP)

PROPERTY OutputValue : LREAL

ThreePointController.ResetAction (METH)

METHOD ResetAction

InOut:

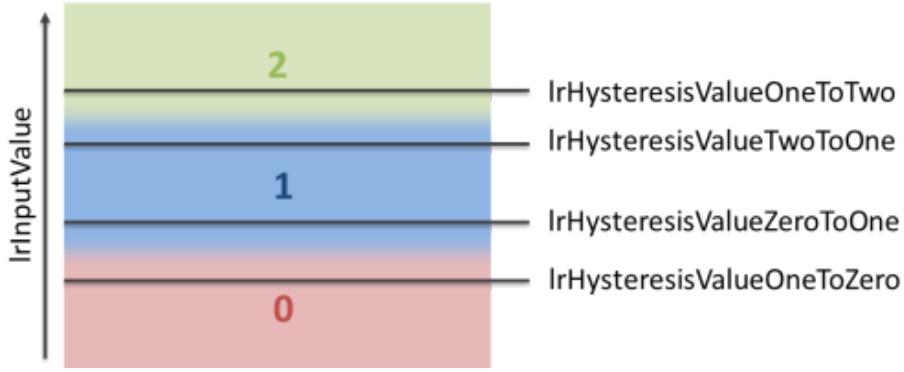
Scope	Name	Type
Output	xComplete	BOOL

ThreePointControllerWithValueHysteresis (FB)

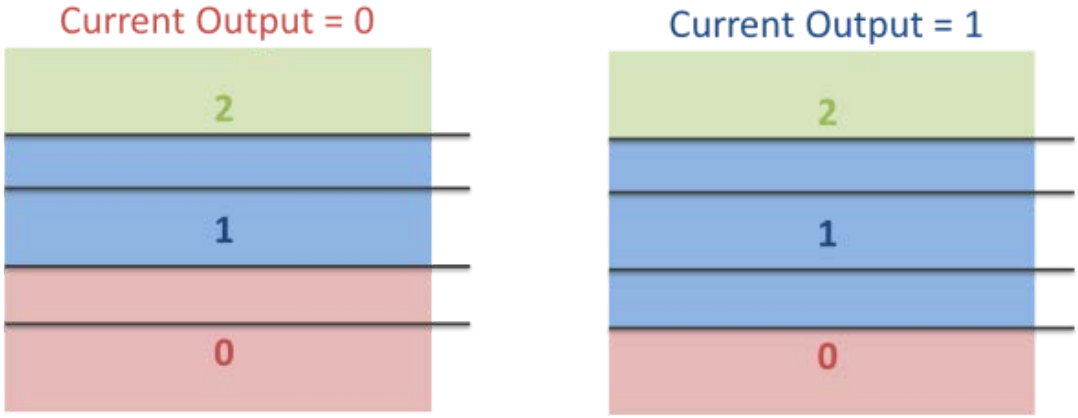
FUNCTION_BLOCK ThreePointControllerWithValueHysteresis EXTENDS CBML.LConC

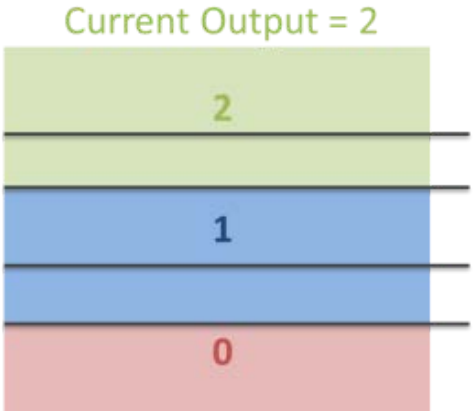
Represents a three point controller with value hysteresis

The three point controller with value hysteresis divides a one dimesional input space into three sections. The partitioning depends on the last output value and an associated hysteresis. A specific input (`lrInputValue`) is mapepd to an integer output [0,1,2] indictating the section it is belonging to.



The following figure shows the partitioning of the input space based on the current output value:





InOut:

Scope	Name	Type	Initial	Comment	Inherited from
Input	xEnable	BOOL		`TRUE`: Activates the defined operation `FALSE`: Aborts/resets the defined operation	LConC
Output	xBusy	BOOL		`TRUE`: Operation is running	LConC
	xError	BOOL		`TRUE`: Error condition reached	LConC
Input	lrInputValue	LREAL		Represents the input value of the three point controller	

Scope	Name	Type	Initial	Comment	Inherited from
	lrHysteresisValueOneToTwo	LREAL		If `lrInputValue` exceeds `lrHysteresisValueOneToTwo` AND `iOutput` = 0|1 then `iOutput` is set to 2	
	lrHysteresisValueTwoToOne	LREAL		If `lrInputValue` falls below `lrHysteresisValueTwoToOne` AND `iOutput` = 2 then `iOutput` is set to `1`	
	lrHysteresisValueZeroToOne	LREAL		If `lrInputValue` exceeds `lrHysteresisValueZeroToOne` AND `iOutput` = 0 then `iOutput` is set to 1	
	lrHysteresisValueOneToZero	LREAL		If `lrInputValue` falls below `lrHysteresisValueOneToZero` AND `iOutput` = 1|2 then `iOutput` is set to `0`	
Output	iOutput	INT		Represents the integer output value (possible values [0,1,2])	
	eErrorID	Controller_Error	Controller_Error.NO_ERROR	The current error. Only valid if `xError` is `TRUE`.	

ThreePointControllerWithValueHysteresis.CyclicAction (METH)

METHOD CyclicAction

InOut:

Scope	Name	Type
Input	itfTimingController	CBML.ITimingController
Output	xComplete	BOOL
	iErrorID	INT

ThreePointControllerWithValueHysteresis.OutputValue (PROP)

PROPERTY OutputValue : LREAL

ThreePointControllerWithValueHysteresis.ResetAction (METH)

METHOD ResetAction

InOut:

Scope	Name	Type
Output	xComplete	BOOL

Differentiation

This directory contains all provided derivative approximations.

- Differentiator_BackwardDifference (FB)
  - Differentiator_BackwardDifference.CleanupAction (METH)
  - Differentiator_BackwardDifference.DoDifferentiation (METH)
  - Differentiator_BackwardDifference.StartAction (METH)
- Differentiator_Base (FB)
  - Differentiator_Base.CurrentDerivative (PROP)
  - Differentiator_Base.CyclicAction (METH)
  - Differentiator_Base.CyclicCall (METH)
  - Differentiator_Base.DoDifferentiation (METH)
  - Differentiator_Base.OutputValue (PROP)
  - Differentiator_Base.Reset (METH)
  - Differentiator_Base.ResetAction (METH)
- Differentiator_LinearAverageApproximation (FB)
  - Differentiator_LinearAverageApproximation.CleanupAction (METH)
  - Differentiator_LinearAverageApproximation.DoDifferentiation (METH)
  - Differentiator_LinearAverageApproximation.StartAction (METH)
- Differentiator_LinearFourPointApproximation (FB)
  - Differentiator_LinearFourPointApproximation.CleanupAction (METH)
  - Differentiator_LinearFourPointApproximation.DoDifferentiation (METH)
  - Differentiator_LinearFourPointApproximation.StartAction (METH)
- Differentiator_ParabolicApproximation (FB)
  - Differentiator_ParabolicApproximation.CleanupAction (METH)
  - Differentiator_ParabolicApproximation.DoDifferentiation (METH)
  - Differentiator_ParabolicApproximation.StartAction (METH)

Differentiator_BackwardDifference (FB)

FUNCTION_BLOCK Differentiator_BackwardDifference EXTENDS Differentiator_Base

The function block uses recent values to approximate the derivation of a value with respect to time.

The last value is recorded and used together with the current value and the elapsed time to estimate the derivation.

InOut:

Scope	Name	Type	Initial	Comment	Inherited from
Input	xEnable	BOOL		`TRUE`: Activates the defined operation `FALSE`: Aborts/resets the defined operation	LConC
Output	xBusy	BOOL		`TRUE`: Operation is running	LConC
	xError	BOOL		`TRUE`: Error condition reached	LConC
Input	lrValue	LREAL		Represents the current value that should be differentiated	Differentiator_Base
	lrCycleInterval	LREAL		Represents the elapsed time in [second]	Differentiator_Base
Output	lrOutputValue	LREAL		Represents the current derivative of the differentiator	Differentiator_Base
	eErrorID	Controller_Error	Controller_Error.NO_ERROR	The current error. Only valid if `xError` is `TRUE`.	Differentiator_Base

Differentiator_BackwardDifference.CleanupAction (METH)

METHOD CleanupAction

InOut:

Scope	Name	Type
Input	xAbortProposed	BOOL
	iErrorIDProposed	INT
Output	xComplete	BOOL
	xAbort	BOOL
	iErrorID	INT

Differentiator_BackwardDifference.DoDifferentiation (METH)

METHOD PROTECTED DoDifferentiation

This method approximates the current derivative. It is called by the Differentiator_Base function block.

InOut:

Scope	Name	Type	Comment
Input	lrValue	LREAL	Represents the current value that should be differentiated
	lrCycleInterval	LREAL	Represents the elapsed time in [seconds]
Output	lrDerivative	LREAL	Represents the value of the current derivative

Differentiator_BackwardDifference.StartAction (METH)

METHOD StartAction

InOut:

Scope	Name	Type
Output	xComplete	BOOL
	iErrorID	INT

Differentiator_Base (FB)

FUNCTION_BLOCK ABSTRACT Differentiator_Base EXTENDS CBML.LConC IMPLEMENTS IDifferentiator, IValueProvider

Represents the base derivator class

It implements the IDifferentiator interface and contains common methods and properties for any pursuing differentiator.

The Differentiator_Base function block can be extended in order to build an individual differentiator. Afterwards, the `DoDifferentiation()` method has to be overridden to include individual logic.

The library proposes some derivative approximations:
- Differentiator_BackwardDifference
- Differentiator_LinearAverageApproximation
- Differentiator_LinearFourPointApproximation
- Differentiator_ParabolicApproximation

> **Note**
>
> In order to built an individual derivator, the specific function block has to implement the IDifferentiator interface or extend the derivator base function block.

InOut:

Scope	Name	Type	Initial	Comment	Inherited from
Input	xEnable	BOOL		`TRUE`: Activates the defined operation `FALSE`: Aborts/resets the defined operation	LConC
Output	xBusy	BOOL		`TRUE`: Operation is running	LConC
	xError	BOOL		`TRUE`: Error condition reached	LConC
Input	lrValue	LREAL		Represents the current value that should be differentiated	
	lrCycleInterval	LREAL		Represents the elapsed time in [second]	
Output	lrOutputValue	LREAL		Represents the current derivative of the differentiator	
	eErrorID	Controller_Error	Controller_Error.NO_ERROR	The current error. Only valid if `xError` is `TRUE`.	

Differentiator_Base.CurrentDerivative (PROP)

PROPERTY FINAL CurrentDerivative : LREAL

Returns the current derivative

Differentiator_Base.CyclicAction (METH)

METHOD CyclicAction

InOut:

Scope	Name	Type
Input	itfTimingController	CBML.ITimingController
Output	xComplete	BOOL
	iErrorID	INT

Differentiator_Base.CyclicCall (METH)

METHOD FINAL CyclicCall

This method approximates the derivative of a given value.
Either call this method each cycle or use the main mathod instead. Never both!

InOut:

Scope	Name	Type	Comment
Input	lrValue	LREAL	Represents the current value that should be differentiated
	lrCycleInterval	LREAL	Represents the elapsed time since the last call in [second]

Differentiator_Base.DoDifferentiation (METH)

METHOD PROTECTED ABSTRACT DoDifferentiation

This method approximates the current derivative. It is called by the Differentiator_Base function block.

InOut:

Scope	Name	Type	Comment
Input	lrValue	LREAL	Represents the current value that should be differentiated
	lrCycleInterval	LREAL	Represents the elapsed time in [seconds]
Output	lrDerivative	LREAL	Represents the value of the current derivative

Differentiator_Base.OutputValue (PROP)

PROPERTY FINAL OutputValue : LREAL

Differentiator_Base.Reset (METH)

METHOD FINAL Reset

Differentiator_Base.ResetAction (METH)

METHOD ResetAction

InOut:

Scope	Name	Type
Output	xComplete	BOOL

Differentiator_LinearAverageApproximation (FB)

FUNCTION_BLOCK Differentiator_LinearAverageApproximation EXTENDS Differentiator_Base

The function block uses recent values to approximate the derivation of a value with respect to time.

Four consecutive values are recorded and utilized in the calculation so that the resulting derivative is as accurate as possible.

InOut:

Scope	Name	Type	Initial	Comment	Inherited from
Input	xEnable	BOOL		TRUE: Activates the defined operation FALSE: Aborts/resets the defined operation	LConC
Output	xBusy	BOOL		TRUE: Operation is running	LConC
	xError	BOOL		TRUE: Error condition reached	LConC
Input	lrValue	LREAL		Represents the current value that should be differentiated	Differentiator_Base
	lrCycleInterval	LREAL		Represents the elapsed time in [second]	Differentiator_Base
Output	lrOutputValue	LREAL		Represents the current derivative of the differentiator	Differentiator_Base
	eErrorID	Controller_Error	Controller_Error.NO_ERROR	The current error. Only valid if xError is TRUE.	Differentiator_Base

Differentiator_LinearAverageApproximation.CleanupAction (METH)

METHOD CleanupAction

InOut:

Scope	Name	Type
Input	xAbortProposed	BOOL
	iErrorIDProposed	INT
Output	xComplete	BOOL
	xAbort	BOOL
	iErrorID	INT

Differentiator_LinearAverageApproximation.DoDifferentiation (METH)

METHOD PROTECTED DoDifferentiation

This method approximates the current derivative. It is called by the Differentiator_Base function block.

InOut:

Scope	Name	Type	Comment
Input	lrValue	LREAL	Represents the current value that should be differentiated
	lrCycleInterval	LREAL	Represents the elapsed time in [seconds]
Output	lrDerivative	LREAL	Represents the value of the current derivative

Differentiator_LinearAverageApproximation.StartAction (METH)

METHOD StartAction

InOut:

Scope	Name	Type
Output	xComplete	BOOL
	iErrorID	INT

Differentiator_LinearFourPointApproximation (FB)

FUNCTION_BLOCK Differentiator_LinearFourPointApproximation EXTENDS Differentiator_Base

The function block uses recent values to approximate the derivation of a value with respect to time.

The last three values are recorded and used together with the current value and the elapsed time to estimate the derivation. The approximation is based on a linear approximation. This is extended to consider the last two and the current derivation. A weighted sum of these three derivatives is output.

> **Note**
>
> The sum of the weights should give 1.
>
> By default, this function block mimics the behavior of Differentiator_BackwardDifference.

InOut:

Scope	Name	Type	Initial	Comment	Inherited from
Input	xEnable	BOOL		`TRUE`: Activates the defined operation `FALSE`: Aborts/resets the defined operation	LConC
Output	xBusy	BOOL		`TRUE`: Operation is running	LConC
	xError	BOOL		`TRUE`: Error condition reached	LConC
Input	lrValue	LREAL		Represents the current value that should be differentiated	Differentiator_Base
	lrCycleInterval	LREAL		Represents the elapsed time in [second]	Differentiator_Base
Output	lrOutputValue	LREAL		Represents the current derivative of the differentiator	Differentiator_Base
	eErrorID	Controller_Error	Controller_Error.NO_ERROR	The current error. Only valid if `xError` is `TRUE`.	Differentiator_Base
Input	lrWeightD1	LREAL	1		
	lrWeightD2	LREAL	0		
	lrWeightD3	LREAL	0		

Differentiator_LinearFourPointApproximation.CleanupAction (METH)

METHOD CleanupAction

InOut:

Scope	Name	Type
Input	xAbortProposed	BOOL
	iErrorIDProposed	INT
Output	xComplete	BOOL
	xAbort	BOOL
	iErrorID	INT

Differentiator_LinearFourPointApproximation.DoDifferentiation (METH)

METHOD PROTECTED DoDifferentiation

This method approximates the current derivative. It is called by the Differentiator_Base function block.

InOut:

Scope	Name	Type	Comment
Input	lrValue	LREAL	Represents the current value that should be differentiated
	lrCycleInterval	LREAL	Represents the elapsed time in [seconds]
Output	lrDerivative	LREAL	Represents the value of the current derivative

Differentiator_LinearFourPointApproximation.StartAction (METH)

METHOD StartAction

InOut:

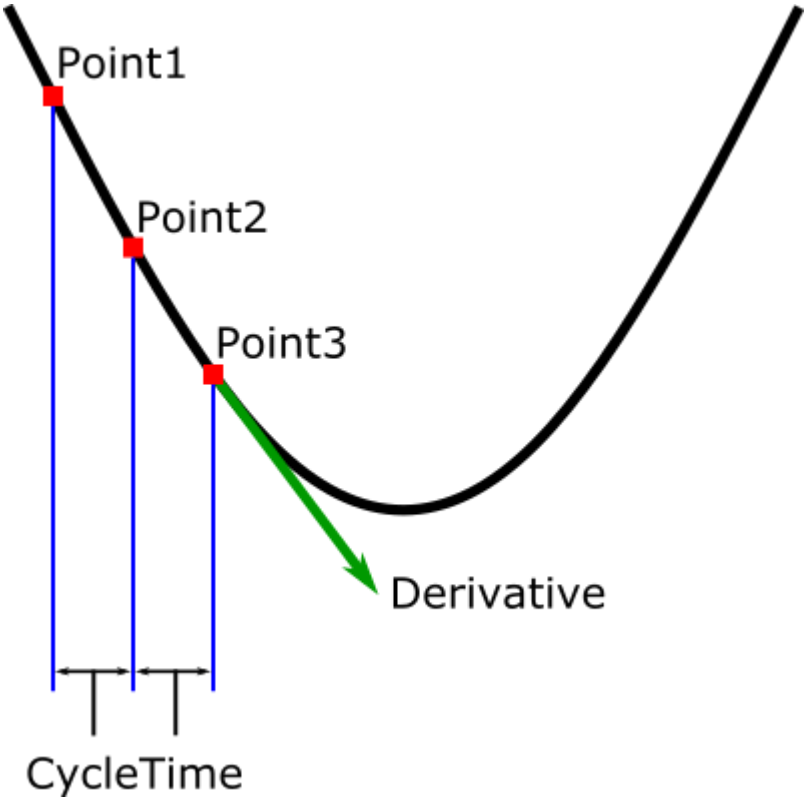
Scope	Name	Type
Output	xComplete	BOOL
	iErrorID	INT

Differentiator_ParabolicApproximation (FB)

FUNCTION_BLOCK Differentiator_ParabolicApproximation EXTENDS Differentiator_Base

The function block uses recent values to approximate the derivation of a value with respect to time.

The last two values are recorded and used together with the current value and the elapsed time to estimate the derivation.



`Point1 = ( 0, Y2 ), Point2 = ( cycleTime, Y1 ), Point3 = ( 2*cycleTime, CurrentValue )`

A parabola is calculated that contains these three points. The approximate derivative is assumed to be the slope of the parabola at `Point3`.

InOut:

Scope	Name	Type	Initial	Comment	Inherited from
Input	xEnable	BOOL		`TRUE`: Activates the defined operation `FALSE`: Aborts/resets the defined operation	LConC
Output	xBusy	BOOL		`TRUE`: Operation is running	LConC
	xError	BOOL		`TRUE`: Error condition reached	LConC
Input	lrValue	LREAL		Represents the current value that should be differentiated	Differentiator_Base
	lrCycleInterval	LREAL		Represents the elapsed time in [second]	Differentiator_Base
Output	lrOutputValue	LREAL		Represents the current derivative of the differentiator	Differentiator_Base
	eErrorID	Controller_Error	Controller_Error.NO_ERROR	The current error. Only valid if `xError` is `TRUE`.	Differentiator_Base

Differentiator_ParabolicApproximation.CleanupAction (METH)

METHOD CleanupAction

InOut:

Scope	Name	Type
Input	xAbortProposed	BOOL
	iErrorIDProposed	INT
Output	xComplete	BOOL
	xAbort	BOOL
	iErrorID	INT

Differentiator_ParabolicApproximation.DoDifferentiation (METH)

METHOD PROTECTED DoDifferentiation

This method approximates the current derivative. It is called by the Differentiator_Base function block.

InOut:

Scope	Name	Type	Comment
Input	lrValue	LREAL	Represents the current value that should be differentiated
	lrCycleInterval	LREAL	Represents the elapsed time in [seconds]
Output	lrDerivative	LREAL	Represents the value of the current derivative

Differentiator_ParabolicApproximation.StartAction (METH)

METHOD StartAction

InOut:

Scope	Name	Type
Output	xComplete	BOOL
	iErrorID	INT

Filter

This directory contains all provided filters.

- Filter_Base (FB)
  - Filter_Base.CurrentValue (PROP)
  - Filter_Base.CyclicAction (METH)
  - Filter_Base.OutputValue (PROP)
  - Filter_Base.Reset (METH)
  - Filter_Base.ResetAction (METH)
- Filter_FIR (FB)
  - Filter_FIR.CleanupAction (METH)
  - Filter_FIR.CyclicAction (METH)
  - Filter_FIR.FB_EXIT (METH)
  - Filter_FIR.StartAction (METH)
- Filter_IIR (FB)
  - Filter_IIR.CleanupAction (METH)
  - Filter_IIR.CyclicAction (METH)
  - Filter_IIR.FB_EXIT (METH)
  - Filter_IIR.StartAction (METH)
- Filter_SOS (FB)
  - Filter_SOS.CleanupAction (METH)
  - Filter_SOS.FB_EXIT (METH)
  - Filter_SOS.StartAction (METH)

Filter_Base (FB)

FUNCTION_BLOCK ABSTRACT Filter_Base EXTENDS CBML.LConC IMPLEMENTS IFilter, IValueProvider
Represents the base filter class

It implements the IFilter interface and contains common methods and properties for any pursuing filter.

The library provides the following filters:

- Filter_FIR
- Filter_IIR
- Filter_SOS

InOut:

Scope	Name	Type	Initial	Comment	Inherited from
Input	xEnable	BOOL		TRUE: Activates the defined operation FALSE: Aborts/resets the defined operation	LConC
Output	xBusy	BOOL		TRUE: Operation is running	LConC
	xError	BOOL		TRUE: Error condition reached	LConC
Input	lrValue	LREAL		The current value that shall be filtered	
Output	lrFilteredValue	LREAL		Represents the current value of the filter	
	eErrorID	Controller_Error	Controller_Error.NO_ERROR	The current error	

Filter_Base.CurrentValue (PROP)

PROPERTY FINAL CurrentValue : LREAL

Filter_Base.CyclicAction (METH)

METHOD CyclicAction

InOut:

Scope	Name	Type
Input	itfTimingController	CBML.ITimingController
Output	xComplete	BOOL
	iErrorID	INT

Filter_Base.OutputValue (PROP)

PROPERTY FINAL OutputValue: LREAL

Filter_Base.Reset (METH)

METHOD FINAL Reset

Filter_Base.ResetAction (METH)

METHOD ResetAction

InOut:

Scope	Name	Type
Output	xComplete	BOOL

Filter_FIR (FB)

FUNCTION_BLOCK Filter_FIR EXTENDS Filter_Base

Represents a Finite-Impulse-Response (FIR) filter

The FIR filter relies on the last M input values and calculates the filtered value as a weighted sum of them.

$$y(i) = \sum_{m=0}^{M} b_m x(i-m)$$

- $y(i)$ represents the current filtered value
- $x(i)$ represents the current input value ($x(i-1)$ depicts the last input value, $x(i-2)$ the second last...)
- b denotes a weight vector consisting of M elements

InOut:

Scope	Name	Type	Initial	Comment	Inherited from
Input	xEnable	BOOL		TRUE: Activates the defined operation FALSE: Aborts/resets the defined operation	LConC
Output	xBusy	BOOL		TRUE: Operation is running	LConC
	xError	BOOL		TRUE: Error condition reached	LConC
Input	lrValue	LREAL		The current value that shall be filtered	Filter_Base
Output	lrFilteredValue	LREAL		Represents the current value of the filter	Filter_Base
	eErrorID	Controller_Error	Controller_Error.NO_ERROR	The current error	Filter_Base
Input	palrCoefficientsB	POINTER TO ARRAY [0..0] OF LREAL		Represent the b coefficients of the filter Only change this setting when the filter is not busy (xBusy = FALSE).	
	udiSizeCoefficientsB	UDINT		Represents the size of the coefficient array Only change this setting when the filter is not busy (xBusy = FALSE).	

Filter_FIR.CleanupAction (METH)

METHOD CleanupAction

InOut:

Scope	Name	Type
Input	xAbortProposed	BOOL
	iErrorIDProposed	INT
Output	xComplete	BOOL
	xAbort	BOOL
	iErrorID	INT

Filter_FIR.CyclicAction (METH)

METHOD CyclicAction

InOut:

Scope	Name	Type
Input	itfTimingController	CBML.ITimingController
Output	xComplete	BOOL
	iErrorID	INT

Filter_FIR.FB_EXIT (METH)

METHOD FB_EXIT : BOOL

InOut:

Scope	Name	Type
Return	FB_EXIT	BOOL
Input	bInCopyCode	BOOL

Filter_FIR.StartAction (METH)

METHOD StartAction

InOut:

Scope	Name	Type
Output	xComplete	BOOL
	iErrorID	INT

Filter_IIR (FB)

FUNCTION_BLOCK Filter_IIR EXTENDS Filter_Base

Represents an Infinite-Impulse-Response (IIR) filter

In contrast to the Filter_FIR the IIR filter does not only rely on the last M input values but also on the last N filter output values. The filter output value is calculated as a difference of two weighted sums of input and output values

$$y(i) = \sum_{m=0}^{M} b_m x(i-m) - \sum_{n=1}^{N} a_n y(i-n)$$

- $y(i)$ represents the current filtered value ($y(i-1)$ depicts the last output value, $y(i-1)$, $y(i-2)$ the second last...)
- $x(i)$ represents the current input value ($x(i-1)$ depicts the last input value, $x(i-2)$ the second last...)
- a denotes a weight vector consisting of N elements ($a_0 = 1$)
- b denotes a weight vector consisting of M elements

InOut:

Scope	Name	Type	Initial	Comment	Inherited from
Input	xEnable	BOOL		TRUE: Activates the defined operation FALSE: Aborts/resets the defined operation	LConC
Output	xBusy	BOOL		TRUE: Operation is running	LConC
	xError	BOOL		TRUE: Error condition reached	LConC
Input	lrValue	LREAL		The current value that shall be filtered	Filter_Base
Output	lrFilteredValue	LREAL		Represents the current value of the filter	Filter_Base
	eErrorID	Controller_Error	Controller_Error.NO_ERROR	The current error	Filter_Base
Input	palrCoefficientsA	POINTER TO ARRAY [0..0] OF LREAL		Represents the a coefficients of the filter (has to be normed -> divide each entry by a[0] if a[0] <> 1) Only change this setting when the filter is not busy (xBusy = FALSE).	
	udiSizeCoefficientsA	UDINT		Represents the size of the a coefficient array Only change this setting when the filter is not busy (xBusy = FALSE).	
	palrCoefficientsB	POINTER TO ARRAY [0..0] OF LREAL		Represents the b coefficients of the filter Only change this setting when the filter is not busy (xBusy = FALSE).	
	udiSizeCoefficientsB	UDINT		Represents the size of the b	

			Initial	Comment	Inherited from
				coefficient array Only change this setting when the filter is not busy (xBusy = FALSE).	

Filter_IIR.CleanupAction (METH)

METHOD CleanupAction

InOut:

Scope	Name	Type
Input	xAbortProposed	BOOL
	iErrorIDProposed	INT
Output	xComplete	BOOL
	xAbort	BOOL
	iErrorID	INT

Filter_IIR.CyclicAction (METH)

METHOD CyclicAction

InOut:

Scope	Name	Type
Input	itfTimingController	CBML.ITimingController
Output	xComplete	BOOL
	iErrorID	INT

Filter_IIR.FB_EXIT (METH)

METHOD FB_EXIT : BOOL

InOut:

Scope	Name	Type
Return	FB_EXIT	BOOL
Input	bInCopyCode	BOOL

Filter_IIR.StartAction (METH)

METHOD StartAction

InOut:

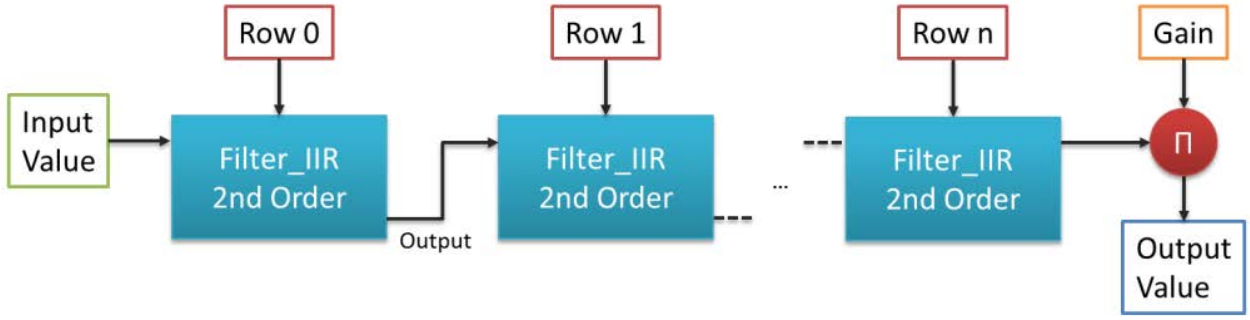
Scope	Name	Type
Output	xComplete	BOOL
	iErrorID	INT

Filter_SOS (FB)

FUNCTION_BLOCK Filter_SOS EXTENDS Filter_Base

Represents a Second-Order Section (SOS) filter

Cascaded Second-Order Section filters fragment the system function into subsystems of second order. These are more robust to quantization errors and have less stability problems compared to direct form filters like Filter_IIR.



InOut:

Scope	Name	Type	Initial	Comment	Inherited from
Input	xEnable	BOOL		TRUE: Activates the defined operation FALSE: Aborts/resets the defined operation	LConC
Output	xBusy	BOOL		TRUE: Operation is running	LConC
	xError	BOOL		TRUE: Error condition reached	LConC
Input	lrValue	LREAL		The current value that shall be filtered	Filter_Base
Output	lrFilteredValue	LREAL		Represents the current value of the filter	Filter_Base
	eErrorID	Controller_Error	Controller_Error.NO_ERROR	The current error	Filter_Base
Input	paCoefficientMatrix	POINTER TO ARRAY [0..0] OF FilterCoefficients_SOS		Represents the coefficient matrix in second-order section form Only change this setting while this filter is not working.	
	udiSizeCoefficientMatrix	UDINT		Represents the size of the coefficient matrix Only change this setting while this filter is not working.	

Scope	Name	Type	Initial	Comment	Inherited from
	lrGain	LREAL		Represents the gain of the second-order section representation	

Filter_SOS.CleanupAction (METH)

METHOD CleanupAction

InOut:

Scope	Name	Type
Input	xAbortProposed	BOOL
	iErrorIDProposed	INT
Output	xComplete	BOOL
	xAbort	BOOL
	iErrorID	INT

Filter_SOS.FB_EXIT (METH)

METHOD FB_EXIT : BOOL

InOut:

Scope	Name	Type
Return	FB_EXIT	BOOL
Input	bInCopyCode	BOOL

Filter_SOS.StartAction (METH)

METHOD StartAction

InOut:

Scope	Name	Type
Output	xComplete	BOOL
	iErrorID	INT

Integrator

This directory contains all provided integral approximations.

- Integrator_Base (FB)
  - Integrator_Base.ApplyAntiWindUpCorrection (METH)
  - Integrator_Base.CheckBoundaries (METH)
  - Integrator_Base.CleanupAction (METH)
  - Integrator_Base.CurrentIntegral (PROP)
  - Integrator_Base.CycleInterval (PROP)
  - Integrator_Base.CyclicAction (METH)
  - Integrator_Base.CyclicCall (METH)
  - Integrator_Base.DoIntegration (METH)
  - Integrator_Base.LastSegmentIntegralValue (PROP)
  - Integrator_Base.OutputValue (PROP)
  - Integrator_Base.Overflow (PROP)
  - Integrator_Base.Reset (METH)
  - Integrator_Base.ResetAction (METH)
- Integrator_ParabolicApproximation (FB)
  - Integrator_ParabolicApproximation.CleanupAction (METH)
  - Integrator_ParabolicApproximation.DoIntegration (METH)
- Integrator_RectangleApproximation (FB)
  - Integrator_RectangleApproximation.DoIntegration (METH)
- Integrator_TrapezoidApproximation (FB)
  - Integrator_TrapezoidApproximation.CleanupAction (METH)
  - Integrator_TrapezoidApproximation.DoIntegration (METH)

Integrator_Base (FB)

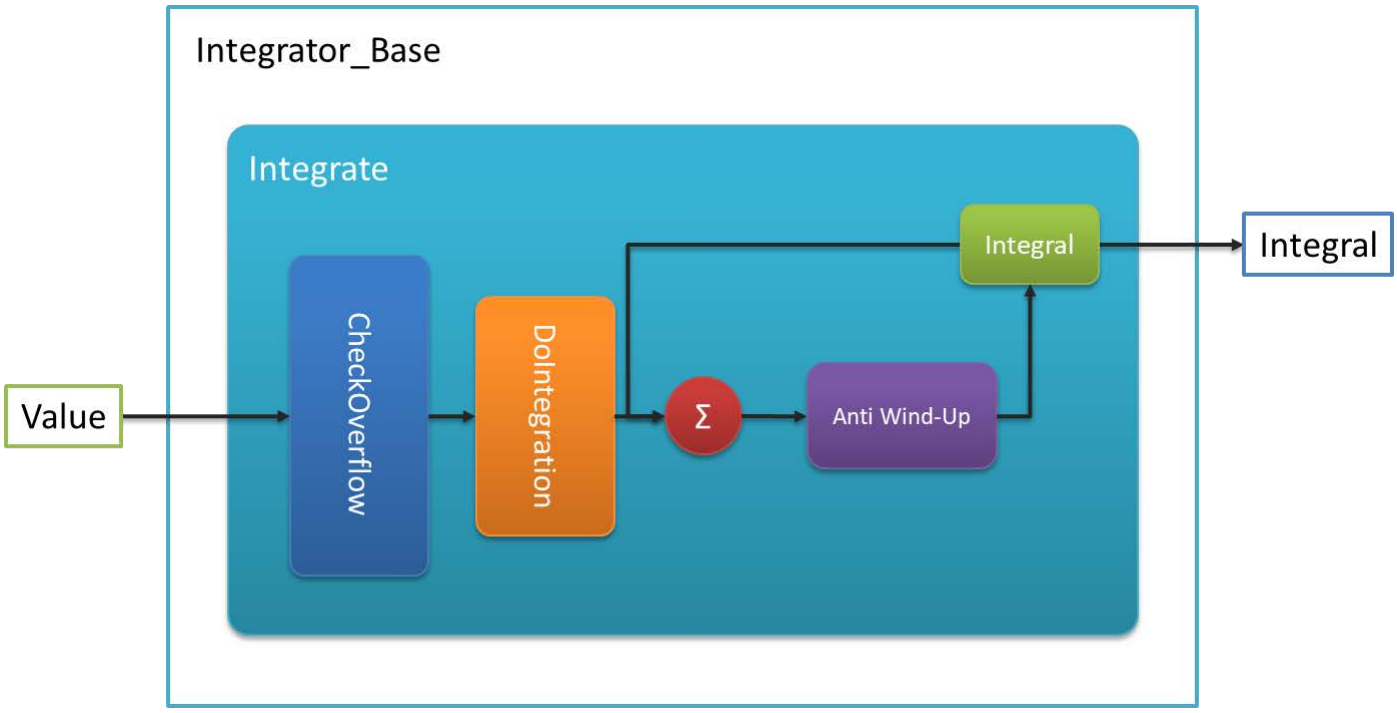
FUNCTION_BLOCK ABSTRACT Integrator_Base EXTENDS CBML.LConC IMPLEMENTS IIntegrator, IClampingValueProvider, IValueProvider, IBackCalculationTaskIntervalProvider

Represents the base integrator class

It implements the IIntegrator interface and contains common methods and properties for any pursuing integrator.

The base integrator provides basic functionality, like limiting the output and detecting overflows. If the limits are exceeded it applies an anti wind-up strategy to avoid integrator wind up.

For more details refer to the figure below.



The Integrator_Base function block can be extended in order to build an individual integrator. Afterwards, the `DoIntegration()` method has to be overridden to include individual integrator logic.

> **Note**
>
> If the Reset() method is overridden: Call `Super^.Reset();`

> **Note**
>
> Individual integrators can implement the IIntegrator interface if no anti-windup strategy shall be offered.

The library proposes some integral approximations:

- Integrator_TrapezoidApproximation
- Integrator_RectangleApproximation
- Integrator_ParabolicApproximation

Integrators that extend this function block may include an anti-windup strategy to avoid integrator windup due to output limitations.

The library proposes the following anti windup strategies:

- AntiWindUp_Clamping
- AntiWindUp_BackCalculation

InOut:

Scope	Name	Type	Initial	Comment	Inherited from
Input	xEnable	BOOL		TRUE: Activates the defined operation FALSE: Aborts/resets the defined operation	LConC
Output	xBusy	BOOL		TRUE: Operation is running	LConC
	xError	BOOL		TRUE: Error condition reached	LConC
Input	itfAntiWindUp	IAntiWindUp		Represents the desired anti-windup strategy (If not set: no anti-windup strategy is applied)	
	lrValue	LREAL		Represents the current value that shall be integrated	
	lrCycleInterval	LREAL		Represents the elapsed time since the last call in [second]	
Output	lrOutputValue	LREAL		Represents the current value of the integral	
	xOverflow	BOOL	FALSE	TRUE: Indicates that an overflow occured in the integrator FALSE: No overflow occured	
	eErrorID	Controller_Error	Controller_Error.NO_ERROR	The current error. Only valid if xError is TRUE.	

Integrator_Base.ApplyAntiWindUpCorrection (METH)

METHOD ApplyAntiWindUpCorrection

This method is called when a controller exceeds its output limits. For more information see: IAntiWindUp

InOut:

Scope	Name	Type	Comment
Input	lrCorrectionValue	LREAL	Represents the computed correction value of the anti-windup strategy.

Integrator_Base.CheckBoundaries (METH)

METHOD CheckBoundaries

This method checks whether the current input value will cause an overflow

Integrator_Base.CleanupAction (METH)

METHOD CleanupAction

InOut:

Scope	Name	Type
Input	xAbortProposed	BOOL
	iErrorIDProposed	INT
Output	xComplete	BOOL
	xAbort	BOOL
	iErrorID	INT

Integrator_Base.CurrentIntegral (PROP)

PROPERTY FINAL CurrentIntegral : LREAL

Returns the current value of the integrator

Integrator_Base.CycleInterval (PROP)

PROPERTY CycleInterval : LREAL

Integrator_Base.CyclicAction (METH)

METHOD CyclicAction

InOut:

Scope	Name	Type
Input	itfTimingController	CBML.ITimingController
Output	xComplete	BOOL
	iErrorID	INT

Integrator_Base.CyclicCall (METH)

METHOD CyclicCall

Use either this method or the main method to call the FB once in every cycle. Never both!

InOut:

Scope	Name	Type	Comment
Input	lrValue	LREAL	Represents the current value that shall be integrated
	lrCycleInterval	LREAL	Represents the elapsed time since the last call in [second]

Integrator_Base.DoIntegration (METH)

METHOD PROTECTED ABSTRACT DoIntegration

This method approximates the change of the integral between the last and current time step. It is called by the Integrator_Base function block.

InOut:

Scope	Name	Type	Comment
Input	lrValue	LREAL	Represents the current value that should be integrated
	lrCycleInterval	LREAL	Represents the elapsed time in [seconds]
Output	lrSegmentIntegralValue	LREAL	Represents the value of the integral between the last call and the current call

Integrator_Base.LastSegmentIntegralValue (PROP)

PROPERTY LastSegmentIntegralValue : LREAL

Integrator_Base.OutputValue (PROP)

PROPERTY FINAL OutputValue : LREAL

Integrator_Base.Overflow (PROP)

PROPERTY FINAL Overflow : BOOL

Integrator_Base.Reset (METH)

METHOD FINAL Reset

Integrator_Base.ResetAction (METH)

METHOD ResetAction

InOut:

Scope	Name	Type
Output	xComplete	BOOL

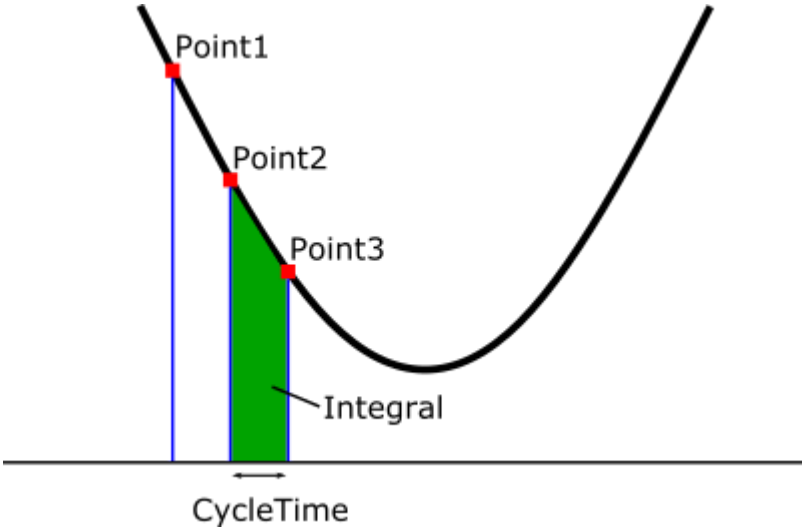
Integrator_ParabolicApproximation (FB)

FUNCTION_BLOCK Integrator_ParabolicApproximation EXTENDS Integrator_Base

The function block approximates an integral over time.

> **Note**
>
> This integrator has similar results compared to the trapezoid integrator, but is computationally more intensive!

The last two values are recorded and used together with the current value to approximate the current integral within the elapsed time.



`Point1 = ( 0, Y2 ), Point2 = ( cycleTime, Y1 ), Point3 = ( 2*cycleTime, CurrentValue )`

A parabola is computed which contains these three points. The approximated integral is assumed to be the integral of the parabola between `Point2` and `Point3`.

InOut:

Scope	Name	Type	Initial	Comment	Inherited from
Input	xEnable	BOOL		`TRUE`: Activates the defined operation `FALSE`: Aborts/resets the defined operation	LConC
Output	xBusy	BOOL		`TRUE`: Operation is running	LConC
	xError	BOOL		`TRUE`: Error condition reached	LConC
Input	itfAntiWindUp	IAntiWindUp		Represents the desired anti-windup strategy (If not set: no anti-windup strategy is applied)	Integrator_Base
	lrValue	LREAL		Represents the current value that shall be integrated	Integrator_Base
	lrCycleInterval	LREAL		Represents the elapsed time since the last call in [second]	Integrator_Base
Output	lrOutputValue	LREAL		Represents the current value of the integral	Integrator_Base
	xOverflow	BOOL	FALSE	`TRUE`: Indicates that an overflow occured	Integrator_Base

Scope	Name	Type	Initial	Comment	Inherited from
				in the integrator `FALSE`: No overflow occured	
	eErrorID	Controller_Error	Controller_Error.NO_ERROR	The current error. Only valid if `xError` is `TRUE`.	Integrator_Base

Integrator_ParabolicApproximation.CleanupAction (METH)

METHOD CleanupAction

InOut:

Scope	Name	Type
Input	xAbortProposed	BOOL
	iErrorIDProposed	INT
Output	xComplete	BOOL
	xAbort	BOOL
	iErrorID	INT

Integrator_ParabolicApproximation.DoIntegration (METH)

METHOD PROTECTED DoIntegration

This method approximates the change of the integral between the last and current time step. It is called by the Integrator_Base function block.

InOut:

Scope	Name	Type	Comment
Input	lrValue	LREAL	Represents the current value that should be integrated
	lrCycleInterval	LREAL	Represents the elapsed time in [seconds]
Output	lrSegmentIntegralValue	LREAL	Represents the value of the integral between the last call and the current call

Integrator_RectangleApproximation (FB)

FUNCTION_BLOCK Integrator_RectangleApproximation EXTENDS Integrator_Base

The function block approximates an integral over time.

The current value is used to approximate the integral within the elapsed time.

InOut:

Scope	Name	Type	Initial	Comment	Inherited from
Input	xEnable	BOOL		`TRUE`: Activates the defined operation `FALSE`: Aborts/resets the defined operation	LConC
Output	xBusy	BOOL		`TRUE`: Operation is running	LConC
	xError	BOOL		`TRUE`: Error condition reached	LConC
Input	itfAntiWindUp	IAntiWindUp		Represents the desired anti-windup strategy (If not set: no anti-windup strategy is applied)	Integrator_Base
	lrValue	LREAL		Represents the current value that shall be integrated	Integrator_Base
	lrCycleInterval	LREAL		Represents the elapsed time since the last call in [second]	Integrator_Base
Output	lrOutputValue	LREAL		Represents the current value of the integral	Integrator_Base
	xOverflow	BOOL	FALSE	`TRUE`: Indicates that an overflow occured in the integrator `FALSE`: No overflow occured	Integrator_Base
	eErrorID	Controller_Error	Controller_Error.NO_ERROR	The current error. Only valid if `xError` is `TRUE`.	Integrator_Base

Integrator_RectangleApproximation.DoIntegration (METH)

METHOD PROTECTED DoIntegration

This method approximates the change of the integral between the last and current time step. It is called by the Integrator_Base function block.

InOut:

Scope	Name	Type	Comment
Input	lrValue	LREAL	Represents the current value that should be integrated
	lrCycleInterval	LREAL	Represents the elapsed time in [seconds]
Output	lrSegmentIntegralValue	LREAL	Represents the value of the integral between the last call and the current call

Integrator_TrapezoidApproximation (FB)

FUNCTION_BLOCK Integrator_TrapezoidApproximation EXTENDS Integrator_Base

The function block approximates an integral over time.

The arithmetic mean of the current and last value is used to approximate the integral within the elapsed time.

InOut:

Scope	Name	Type	Initial	Comment	Inherited from
Input	xEnable	BOOL		`TRUE`: Activates the defined operation `FALSE`: Aborts/resets the defined operation	LConC
Output	xBusy	BOOL		`TRUE`: Operation is running	LConC
	xError	BOOL		`TRUE`: Error condition reached	LConC
Input	itfAntiWindUp	IAntiWindUp		Represents the desired anti-windup strategy (If not set: no anti-windup strategy is applied)	Integrator_Base
	lrValue	LREAL		Represents the current value that shall be integrated	Integrator_Base
	lrCycleInterval	LREAL		Represents the elapsed time since the last call in [second]	Integrator_Base
Output	lrOutputValue	LREAL		Represents the current value of the integral	Integrator_Base
	xOverflow	BOOL	FALSE	`TRUE`: Indicates that an overflow occured in the integrator `FALSE`: No overflow occured	Integrator_Base
	eErrorID	Controller_Error	Controller_Error.NO_ERROR	The current error. Only valid if `xError` is `TRUE`.	Integrator_Base

Integrator_TrapezoidApproximation.CleanupAction (METH)

METHOD CleanupAction

InOut:

Scope	Name	Type
Input	xAbortProposed	BOOL
	iErrorIDProposed	INT
Output	xComplete	BOOL
	xAbort	BOOL
	iErrorID	INT

Integrator_TrapezoidApproximation.DoIntegration (METH)

METHOD PROTECTED DoIntegration

This method approximates the change of the integral between the last and current time step. It is called by the Integrator_Base function block.

InOut:

Scope	Name	Type	Comment
Input	lrValue	LREAL	Represents the current value that should be integrated
	lrCycleInterval	LREAL	Represents the elapsed time in [seconds]
Output	lrSegmentIntegralValue	LREAL	Represents the value of the integral between the last call and the current call

PWM_Creator

PWM_Creator (FB)

FUNCTION_BLOCK FINAL PWM_Creator EXTENDS PWM_CreatorBase

Represents a two point controller with pulse width modulation

InOut:

Scope	Name	Type	Initial	Comment	Inherited from
Input	xEnable	BOOL		TRUE: Activates the defined operation FALSE: Aborts/resets the defined operation	LConC
Output	xBusy	BOOL		TRUE: Operation is running	LConC
	xError	BOOL		TRUE: Error condition reached	LConC
Input	lrProportionValue	LREAL		Represents the demanded proportional value for the pwm controller between 0 and 1 If this value is out of range an error is produced.	PWM_CreatorBase
Output	xOutput	BOOL		Represents the boolean output value	PWM_CreatorBase
	eErrorID	Controller_Error	Controller_Error.NO_ERROR	The current error. Only valid if xError is TRUE.	PWM_CreatorBase
Input	udiMinSwitchingTime	UDINT		Represents the minimum time between an output switch of pwm controller in µs	

PWM_Creator.CleanupAction (METH)

METHOD CleanupAction

InOut:

Scope	Name	Type
Input	xAbortProposed	BOOL
	iErrorIDProposed	INT
Output	xComplete	BOOL
	xAbort	BOOL
	iErrorID	INT

PWM_Creator.CyclicAction (METH)

METHOD FINAL CyclicAction

InOut:

Scope	Name	Type
Input	itfTimingController	CBML.ITimingController
Output	xComplete	BOOL
	iErrorID	INT

PWM_Creator.StartAction (METH)

METHOD FINAL StartAction

InOut:

Scope	Name	Type
Output	xComplete	BOOL
	iErrorID	INT

PWM_Creator_FixedCycle (FB)

FUNCTION_BLOCK FINAL PWM_Creator_FixedCycle EXTENDS PWM_CreatorBase

Represents a two point controller with pulse width modulation using a fixed cycle width.

Note

If the cycle time given by *udiWidthCycleTime* can not be divided without remainder by the cycle time of the task calling this FB, the number of ON-cycles is calculated as defined by *eRoundingStrategy*

InOut:

Scope	Name	Type	Initial	Comment	Inherited from
Input	xEnable	BOOL		TRUE: Activates the defined operation FALSE: Aborts/resets the defined operation	LConC
Output	xBusy	BOOL		TRUE: Operation is running	LConC
	xError	BOOL		TRUE: Error condition reached	LConC
Input	lrProportionValue	LREAL		Represents the demanded proportional value for the pwm controller between 0 and 1 If this value is out of range an error is produced.	PWM_CreatorBase
Output	xOutput	BOOL		Represents the boolean output value	PWM_CreatorBase
	eErrorID	Controller_Error	Controller_Error.NO_ERROR	The current error. Only valid if xError is TRUE.	PWM_CreatorBase
Input	udiPWMCycleTime	UDINT		The cycle time in µs	
	eRoundingStrategy	RoundingStrategy	RoundingStrategy.round	The strategy being used to deal with the remainder of the PWM-CycleTime divided by the task cycle time calling this FB	

PWM_Creator_FixedCycle.CleanupAction (METH)

METHOD CleanupAction

InOut:

Scope	Name	Type
Input	xAbortProposed	BOOL
	iErrorIDProposed	INT
Output	xComplete	BOOL
	xAbort	BOOL
	iErrorID	INT

PWM_Creator_FixedCycle.CyclicAction (METH)

METHOD FINAL CyclicAction

InOut:

Scope	Name	Type
Input	itfTimingController	CBML.ITimingController
Output	xComplete	BOOL
	iErrorID	INT

PWM_Creator_FixedCycle.StartAction (METH)

METHOD FINAL StartAction

This method computes the cycle interval of the task in which the PWM_Creator is working.

InOut:

Scope	Name	Type
Output	xComplete	BOOL
	iErrorID	INT

Functions

- TransformParametersFromSeriesToParallel_PI (FUN)
- TransformParametersFromSeriesToParallel_PID (FUN)
- TransformParametersFromStandardToParallel_PI (FUN)
- TransformParametersFromStandardToParallel_PID (FUN)

TransformParametersFromSeriesToParallel_PI (FUN)

FUNCTION TransformParametersFromSeriesToParallel_PI : BOOL

This function converts paramameters given for series form into parallel form used in this library

InOut:

Scope	Name	Type
Return	TransformParametersFromSeriesToParallel_PI	BOOL
Input	lrK	LREAL
	lrTi	LREAL
Output	lrKp	LREAL
	lrKi	LREAL

TransformParametersFromSeriesToParallel_PID (FUN)

FUNCTION TransformParametersFromSeriesToParallel_PID : BOOL

This function converts paramameters given for series form into parallel form used in this library

InOut:

Scope	Name	Type
Return	TransformParametersFromSeriesToParallel_PID	BOOL
Input	lrK	LREAL
	lrTi	LREAL
	lrTd	LREAL
Output	lrKp	LREAL
	lrKi	LREAL
	lrKd	LREAL

TransformParametersFromStandardToParallel_PI (FUN)

FUNCTION TransformParametersFromStandardToParallel_PI : BOOL

This function converts paramameters given for ‘standard’ form into parallel form used in this library

InOut:

Scope	Name	Type
Return	TransformParametersFromStandardToParallel_PI	BOOL
Input	lrKp	LREAL
	lrTn	LREAL
Output	lrKi	LREAL

TransformParametersFromStandardToParallel_PID (FUN)

FUNCTION TransformParametersFromStandardToParallel_PID : BOOL

This function converts paramameters given for ‘standard’ form into parallel form used in this library

InOut:

Scope	Name	Type
Return	TransformParametersFromStandardToParallel_PID	BOOL
Input	lrKp	LREAL
	lrTn	LREAL
	lrTv	LREAL
Output	lrKi	LREAL
	lrKd	LREAL

GlobalConstants

This directory contains all global constants.

- Constants (GVL)

Constants (GVL)

InOut:

Scope	Name	Type	Initial
Constant	PI	LREAL	3.1415926535897931
	iMaxFilterOrder	INT	12

Interfaces

This directory contains all interfaces of the library.

- IAntiWindUp (ITF)
  - IAntiWindUp.Apply (METH)
- IBackCalculationTaskIntervalProvider (ITF)
  - IBackCalculationTaskIntervalProvider.CycleInterval (PROP)
- IClampingValueProvider (ITF)
  - IClampingValueProvider.LastSegmentIntegralValue (PROP)
- IController (ITF)
  - IController.ActualValue (PROP)
  - IController.LimitsActive (PROP)
  - IController.ManualModeActive (PROP)
  - IController.MaxValue (PROP)
  - IController.MinValue (PROP)
  - IController.Offset (PROP)
  - IController.SetManual (METH)
  - IController.SetPoint (PROP)
- IController_D (ITF)
  - IController_D.KD (PROP)
- IController_I (ITF)
  - IController_I.KI (PROP)
- IController_P (ITF)
  - IController_P.KP (PROP)
- IDifferentiator (ITF)
  - IDifferentiator.CurrentDerivative (PROP)
  - IDifferentiator.CyclicCall (METH)
- IFilter (ITF)
  - IFilter.CurrentValue (PROP)
- IIntegrator (ITF)
  - IIntegrator.ApplyAntiWindUpCorrection (METH)
  - IIntegrator.CurrentIntegral (PROP)
  - IIntegrator.CyclicCall (METH)
  - IIntegrator.Overflow (PROP)
- IPWM_Creator (ITF)
  - IPWM_Creator.Output (PROP)
- IValueProvider (ITF)
  - IValueProvider.OutputValue (PROP)

IAntiWindUp (ITF)

INTERFACE IAntiWindUp EXTENDS __SYSTEM.IQueryInterface

- IAntiWindUp.Apply (METH)

IAntiWindUp.Apply (METH)

METHOD Apply

This method is called every cycle and applies an ati-windup strategy if the give integrator exceeds specified limits.

InOut:

Scope	Name	Type	Comment
Input	itfIntegrator	IIntegrator	Represents the integrator on which an anti wind-up strategy should be applied

IBackCalculationTaskIntervalProvider (ITF)

INTERFACE IBackCalculationTaskIntervalProvider

- IBackCalculationTaskIntervalProvider.CycleInterval (PROP)

IBackCalculationTaskIntervalProvider.CycleInterval (PROP)

PROPERTY CycleInterval : LREAL

IClampingValueProvider (ITF)

INTERFACE IClampingValueProvider

- IClampingValueProvider.LastSegmentIntegralValue (PROP)

IClampingValueProvider.LastSegmentIntegralValue (PROP)

PROPERTY LastSegmentIntegralValue : LREAL

IController (ITF)

INTERFACE IController EXTENDS CBML.IBehaviourModel, IValueProvider

- IController.ActualValue (PROP)
- IController.LimitsActive (PROP)
- IController.ManualModeActive (PROP)
- IController.MaxValue (PROP)
- IController.MinValue (PROP)
- IController.Offset (PROP)
- IController.SetManual (METH)
- IController.SetPoint (PROP)

IController.ActualValue (PROP)

PROPERTY ActualValue : LREAL

Returns or sets the actually measured process variable

IController.LimitsActive (PROP)

PROPERTY LimitsActive : BOOL

IController.ManualModeActive (PROP)

PROPERTY ManualModeActive : BOOL

Indicates if the manual mode is active

IController.MaxValue (PROP)

PROPERTY MaxValue : LREAL

Returns or sets the upper bound of the controller output

IController.MinValue (PROP)

PROPERTY MinValue : LREAL

Returns or sets the lower bound of the controller output

IController.Offset (PROP)

PROPERTY Offset : LREAL

Returns or sets an offset

IController.SetManual (METH)

METHOD SetManual

This method activates or deactivates the manual mode of the controller For more details see: Controller_Base

InOut:

Scope	Name	Type	Comment
Input	xManual	BOOL	Indicates if the manual mode should be activated or deactivated
	lrValue	LREAL	Represents the value which is used in the manual mode

IController.SetPoint (PROP)

PROPERTY SetPoint : LREAL

Returns or sets the desired set point

IController_D (ITF)

INTERFACE IController_D EXTENDS __SYSTEM.IQueryInterface

- IController_D.KD (PROP)

IController_D.KD (PROP)

PROPERTY KD : LREAL

Returns or sets the derivative parameter

IController_I (ITF)

INTERFACE IController_I EXTENDS __SYSTEM.IQueryInterface

- IController_I.KI (PROP)

IController_I.KI (PROP)

PROPERTY KI : LREAL

Returns or sets the integral parameter

IController_P (ITF)

INTERFACE IController_P EXTENDS __SYSTEM.IQueryInterface

- IController_P.KP (PROP)

IController_P.KP (PROP)

PROPERTY KP : LREAL

Returns or sets the proportional parameter

IDifferentiator (ITF)

INTERFACE IDifferentiator EXTENDS CBML.IBehaviourModel

- IDifferentiator.CurrentDerivative (PROP)
- IDifferentiator.CyclicCall (METH)

IDifferentiator.CurrentDerivative (PROP)

PROPERTY CurrentDerivative : LREAL

Returns the current derivative

IDifferentiator.CyclicCall (METH)

METHOD CyclicCall

This method approximates the derivative of a given value. When using this method one must call it every cycle and make sure the implementing FB is not called.

InOut:

Scope	Name	Type	Comment
Input	lrValue	LREAL	Represents the current value that should be deviated
	lrCycleInterval	LREAL	Represents the elapsed time in [microseconds]

IFilter (ITF)

INTERFACE IFilter EXTENDS CBML.IBehaviourModel

- IFilter.CurrentValue (PROP)

IFilter.CurrentValue (PROP)

PROPERTY CurrentValue : LREAL

Represents the current filtered value

IIntegrator (ITF)

INTERFACE IIntegrator EXTENDS CBML.IBehaviourModel

- IIntegrator.ApplyAntiWindUpCorrection (METH)
- IIntegrator.CurrentIntegral (PROP)
- IIntegrator.CyclicCall (METH)
- IIntegrator.Overflow (PROP)

IIntegrator.ApplyAntiWindUpCorrection (METH)

METHOD ApplyAntiWindUpCorrection

This method is called when a controller exceeds its output limits. For more information see: IAntiWindUp

InOut:

Scope	Name	Type	Comment
Input	lrCorrectionValue	LREAL	Represents the computed correction value of the anti-windup strategy.

IIntegrator.CurrentIntegral (PROP)

PROPERTY CurrentIntegral : LREAL

Returns the current value of the integrator

IIntegrator.CyclicCall (METH)

METHOD CyclicCall

This method approximates the integral of a given value. When using this method one must call it every cycle and make sure that the implementing FB is not called.

InOut:

Scope	Name	Type	Comment
Input	lrValue	LREAL	Represents the current value that shall be integrated
	lrCycleInterval	LREAL	Represents the elapsed time since the last call in [second]

IIntegrator.Overflow (PROP)

PROPERTY Overflow : BOOL

Indicates if an overflow occured

IPWM_Creator (ITF)

INTERFACE IPWM_Creator EXTENDS CBML.IBehaviourModel

- IPWM_Creator.Output (PROP)

IPWM_Creator.Output (PROP)

PROPERTY Output : BOOL

IValueProvider (ITF)

INTERFACE IValueProvider EXTENDS __SYSTEM.IQueryInterface

- IValueProvider.OutputValue (PROP)

IValueProvider.OutputValue (PROP)

PROPERTY OutputValue : LREAL

Structs

This directory contains all structs of the library.

- FilterCoefficients_SOS (STRUCT)

FilterCoefficients_SOS (STRUCT)

TYPE FilterCoefficients_SOS : STRUCT

InOut:

Name	Type
b0	LREAL
b1	LREAL
b2	LREAL
a0	LREAL
a1	LREAL
a2	LREAL

File and Project Information

Scope	Name	Type	Content
FileHeader	companyName	string	3S-Smart Software Solutions GmbH
	contentFile		doc.clean.json
	libraryFile		ControlLoop.library
	productProfile		CODESYS V3.5 SP14
	productName		CODESYS
	version	version	2.0.0.0
	primaryProject	string	True
	creationDateTime	date	26.11.2019, 10:12:10
ProjectInformation	LastModificationDateTime		26.11.2019, 10:12:02
	Description	string	See: Description
	DocFormat		reStructuredText
	Author		3S - Smart Software Solutions GmbH
	DefaultNamespace		Ctrl
	LanguageModelAttribute		qualified-access-only
	Company		3S - Smart Software Solutions GmbH
	Placeholder		CtrlLib
	Project		ControlLoop
	Released	bool	False
	Version	version	1.0.0.0
	Title	string	ControlLoopLibrary
	LibraryCategories	library-category-list	
	IsEndUserLibrary	bool	False

Library Reference

This is a dictionary of all referenced libraries and their name spaces.

Analyzation

Library Identification

Placeholder: Analyzation
Default Resolution: Analyzation, 3.5.2.0 (System)
Namespace: Analyzation

Library Properties

- LinkAllContent: False
- QualifiedOnly: False
- Key: Analyzation
- SystemLibrary: True
- Optional: False

Library Parameter

Parameter: TABLE_UPPER_BOUND = 15

Parameter: STRING_LENGTH_ADDRESS = 20

Parameter: STRING_LENGTH_EXP = 255

Parameter: STRING_LENGTH_COMMENT = 255

Parameter: TABLE_SHOW_VALID_ITEMS = FALSE

Parameter: STRING_LENGTH_OUTSTRING = 255

CAA FB Factory

Library Identification

Placeholder: CAA FB Factory
Default Resolution: CAA FB Factory, * (CAA Technical Workgroup)
Namespace: FBF

Library Properties

- LinkAllContent: False
- QualifiedOnly: True
- Key: CAA FB Factory
- SystemLibrary: False
- Optional: False

CAA Memory Block Manager Extern

Library Identification

Placeholder: CAA MemBlockMan
Default Resolution: CAA Memory Block Manager Extern, * (CAA Technical Workgroup)
Namespace: MBM

Library Properties

- LinkAllContent: False
- QualifiedOnly: True
- Key: CAA MemBlockMan
- SystemLibrary: False
- Optional: False

CmpErrors2 Interfaces

Library Identification

Name: CmpErrors2 Interfaces
Version: **newest**
Company: System
Namespace: CmpErrors

Library Properties

- LinkAllContent: False
- QualifiedOnly: False
- Key: CmpErrors2 Interfaces, * (System)
- SystemLibrary: False
- Optional: False

Common Behaviour Model

Library Identification

Placeholder: CBML
Default Resolution: Common Behaviour Model, * (3S - Smart Software Solutions GmbH)
Namespace: CBML

Library Properties

- LinkAllContent: False
- QualifiedOnly: True
- Key: CBML
- SystemLibrary: False
- Optional: False

FloatingPointUtils

Library Identification

Placeholder: FloatingPointUtils
Default Resolution: FloatingPointUtils, * (System)
Namespace: FPU

Library Properties

- LinkAllContent: False
- QualifiedOnly: True
- Key: FloatingPointUtils
- SystemLibrary: False
- Optional: False

IecSfc

Library Identification

Placeholder: IecSfc
Default Resolution: IecSfc, 3.4.2.0 (System)
Namespace: IecSfc

Library Properties

- LinkAllContent: False
- QualifiedOnly: False
- Key: IecSfc
- SystemLibrary: True
- Optional: False

SysTask

Library Identification

Placeholder: SysTask
Default Resolution: SysTask, * (System)

Namespace: SysTask

Library Properties

- LinkAllContent: False
- QualifiedOnly: False
- Key: SysTask
- SystemLibrary: False
- Optional: False

SysTypes2 Interfaces

Library Identification

Name: SysTypes2 Interfaces
Version: **newest**
Company: System
Namespace: SysTypes

Library Properties

- LinkAllContent: False
- QualifiedOnly: False
- Key: SysTypes2 Interfaces, * (System)
- SystemLibrary: False
- Optional: False