HollySys

Feedback Control Algorithm Command Manual for LX Series Programmable Logic Controllers

Versions：2026.03

Copyright Notice

The text, illustrations, charts, marks, trademarks, product models, programs, page layout and other contents included in this manual are under protection of “Copyright Law of the People’s Republic of China”, “Trademark Law of the People’s Republic of China” , “Patent Law of the People’s Republic of China” and the laws of applicable international conventions regarding copyright, trademark right, patent right or other property ownership, and they are owned or possessed exclusively by Beijing HollySys Intelligent Technologies Co., Ltd..

Since the equipment explained in this manual has a variety of uses, the user and those responsible for applying this equipment must satisfy themselves as to the acceptability of each application and use of the equipment. Under no circumstances will Beijing HollySys Intelligent Technologies Co., Ltd. be responsible or liable for any damage, including indirect or consequential losses resulting from the use, misuse, or application of this equipment.

Due to the many variables associated with specific uses or applications, Beijing HollySys Intelligent Technologies Co., Ltd. cannot assume responsibility or liability for actual use based upon the data provided in this manual.

This manual is provided only for commercial users to read. Without prior written permission of Beijing HollySys Intelligent Technologies Co., Ltd., no part of this manual should be reproduced and transmitted in any forms by any means, including electronic, mechanical or otherwise regardless of whatever reasons and purposes. We will investigate violator’s legal liability in accordance with the relevant laws.

The text HollySys, and the logos are registered trademarks of Beijing HollySys Intelligent Technologies Co., Ltd..

All other trademarks are the property of their respective holders.

All rights reserved for Beijing HollySys Intelligent Technologies Co., Ltd..

Address: Di Sheng Middle Road, No.2,
Economic-Technological Development Area, 100176, Beijing, China
Tel: +86 010-5898 1588
Consulting Hotline: 400-811-1999
Technology Request: +86 010-58981514
Fax: +86 010-5898 1558
Web: http://www.hollysys.com
FA Web: https://fa.hollysys.com/
Email: PLC@hollysys.com
Sina weibo: http://weibo.com/hollysysplc

Table of Contents

Chapter 1 Foreword

1.1 Purpose

This document will introduce the use of feedback control commang in the LX system, including product introduction, module components, technical parameters, indicators, wiring instructions, diagnostic information, etc. Please use it in conjunction with other supporting product materials to help you have a more comprehensive understanding of how to use the modules.

1.2 Object

This manual is intended for use by engineers or related professional technicians with expertise in automation and control systems.

1.3 Target Products

This manual is applicable to the following products:

- LX series PLC products

1.4 Statement of Use

- The contents of this manual have been tested according to regulations, and the diagrams are consistent with the software and hardware products described. However, errors are inevitable, and complete consistency cannot be guaranteed. If you have any suggestions for improving or correcting the contents of this manual, please let us know in time and we will make corrections in the next version.
- Due to product iteration updates, the relevant parameters and information in this manual may change without prior notice.
- Since the devices described in this manual can be used in a variety of ways, the user and the person responsible for the use of the devices must ensure that each method is admissible. HollySys will not be legally responsible for any direct or indirect losses resulting from the use or misuse of these devices.
- Due to uncertainties in the actual application, HollySys assumes no responsibility for the direct use of the data provided in this manual.

1.5 Safety Precautions

1.5.1 Safety Statement

(1) Please read and comply with these safety precautions before using this product.

(2) To ensure personal and device safety, please strictly abide by the safety precautions on the product label and in the manual when using this product.

(3) The words "Note", "Caution", "Warning" and "Danger" in this manual do not represent all the safety precautions that shall be observed, but only serve as a supplement to all precautions.

(4) This product shall be used in an environment that meets the design specifications. Otherwise, it may cause faults. Device damage caused by failure to comply with relevant regulations is not covered by the product quality guarantee.

(5) HollySys shall not be responsible for any personal safety accidents or property losses caused by any operation that does not comply with the provisions of this manual or does not meet the requirements.

1.5.2 Warning

For your safety and to avoid property losses, please pay attention to the warnings in this manual. The following warnings are indicated according to the hazard level from high to low. When there are multiple hazard levels, only the highest level is prompted. If the highest level is a warning that may cause personal injury, there may also be a warning that may cause property loss.

Warnings on personal safety: Indicated by a warning triangle .

Warnings on property loss: Without any warning triangle.

Danger
It indicates that death or serious personal injury may be caused if operations are not carried out as specified.

Warning
It indicates that death or serious personal injury may be caused if operations are not carried out as specified.

Caution
It indicates that minor personal injuries may be caused if operations are not carried out as specified.

Note
It indicates that property losses may be caused if operations are not carried out as specified.

Chapter 2 Document Guide

2.1 Related Manuals

The available manuals for this product are displayed in the table below. Please choose and refer to them based on your specific usage requirement.

Manual Name	Purpose	Content
Module Manual for LX Series Programmable Logic Controllers	Learn about the hardware modules of the LX system	Describes all hardware modules, including: Module function Hardware interface Wiring Indicator Technical parameters
Communication Manual for LX Series Programmable Logic Controllers	Guide users to build the LX system communication network	Describe the communication networks supported by the LX system, including: Introduction to communication protocols Function usage
Programming Manual for LX Series Programmable Logic Controllers	Learn about the use of FA-AutoThink programming software	Describe the interface, functions, and related operations of FA-AutoThink programming software, including: Software interface introduction Project creation Programming Online commissioning function
Basic Instruction Manual for LX Series Programmable Logic Controllers	Learn about the use of basic instructions of the LX system	Describe the use of basic instructions, including: Instruction function Parameter Example
Motion Instruction Manual for LX Series Programmable Logic Controllers	Learn about the use of motion control instructions of the LX system	Describe the use of motion instructions, including: Instruction function Parameter Example
PLCopen Instruction Manual for LX Series Programmable Logic Controllers	Learn about the use of PLCopen motion control instructions of the LX system	Describe the use of PLCopen instructions, including: Parameter Function Example Notes for Using See Synonyms at force

2.2 Usage Convention

The following symbols are used in this manual:

- Tip: This symbol indicates helpful tips or suggestions for using the product more efficiently.

2.3 How to Get the Manual

If you need the electronic PDF file of this manual, kindly access it through our website.

- The official website of HollySys (https://cn.hollysys.com/other/download.html) to download the file.

Chapter 3 Revision History

Version	Revision Date	Revision Content
V1.0	October 2025	Created

Chapter 4 Filters

4.1 IIR Filters

4.1.1 First_Order_Filter(First Order Filter)

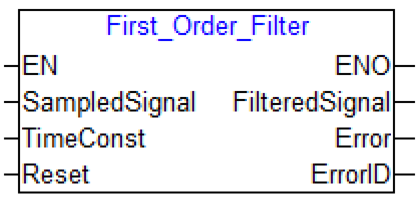
4.1.1.1 Version

- The software version supporting this function block is V3.2.3 and above. Controller firmware versions that do not support this instruction include LX-CU500-A01 (V1.0.0) and LX-CU500-A02 (V1.1.0).

4.1.1.2 Function

Using this function block allows for processing the input signal by filtering out high-frequency noise components.

The First_Order_Filter employs a transfer function containing a first-order inertial element, which is discretized using the Tustin method to derive the corresponding discrete difference equation. This difference equation is then applied to process the input signal, eliminating high-frequency components. If an error occurs during processing, the Error flag is set to TRUE, ErrorID indicates the specific cause of the error, and the FilteredSignal is reset to zero.



First_Order_Filter Functional Block

4.1.1.3 Parameter

Parameter	Statement	Data Type	Function Description	Initial Value
SampledSignal	Input	LREAL	Signal Input The collected signal is connected to this pin, the first-order filter will filter the signal.	0
TimeConst	Input	LREAL	Filter Time Constant (Unit: Seconds) The time constant for filtering the process value.	0

			When this value is set to 0, filtering of the process value is disabled, and the output value equals the input value itself. The setting of the TimeConst parameter directly affects the amplitude and phase of the signal. A larger value results in better noise reduction through filtering but leads to a smaller amplitude and greater phase lag. A smaller value provides poorer noise reduction but maintains an amplitude closer to the original signal and introduces less phase lag. The TimeConst parameter should be configured to 10–100 times the task cycle period in which the function block is called.	
Reset	Input	BOOL	Reset Configuration: TRUE: Reset FALSE: No Reset	FALSE
FilteredSignal	Output	LREAL	Filtered signal output.	0
Error	Output	BOOL	Input Parameter Invalid Flag. If the input parameter is invalid, the parameter will output TRUE.	FALSE
ErrorID	Output	CTRL_ALGO_LIB_ERR_LIST	VALID: No error Others: Refer to the error code	VALID

4.1.1.4 Error Code

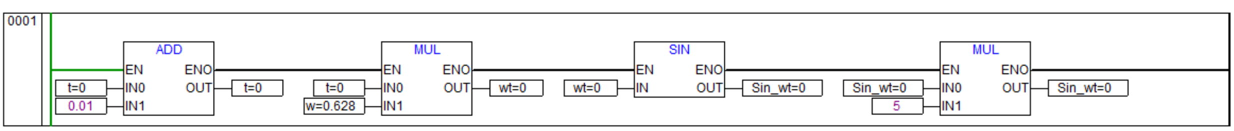
Error ID	Description
SampledSignal_REQUIRED_FINITE	Acquired signal range must be -1×10^{300} to 1×10^{300}
TimeConst_REQUIRED_FINITE	TimeConst must not exceed 1×10^{300}
TimeConst_REQUIRED_POSI	TimeConst must be a positive value
TimeConst_REQUIRED_NON_NEGA	TimeConst must be a non-negative value
TASK_IS_SINGLE	Task is Single execution
TASK_IS_FULLSPEED	Task is FullSpeed execution

4.1.1.5 Example

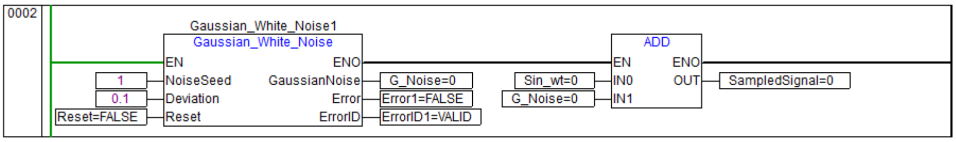
This example is only intended to further illustrate the use of First_Order_Filter, not for actual application scenarios.

This example uses a First_Order_Filter for signal filtering and noise reduction, with the following explanation:

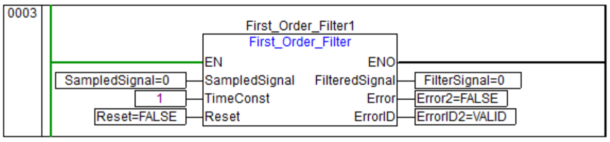
- Program Segment 0001 simulates a sine signal with an amplitude of 5, frequency of 0.628 rad/s, and initial phase of 0.



- Program Segment 0002 utilizes the custom function block Gaussian_Write_Noise to simulate Gaussian white noise with a mean square error of 0.1. This noise signal is then added to the sine signal from Program Segment 0001, resulting in the original signal SampledSignal.

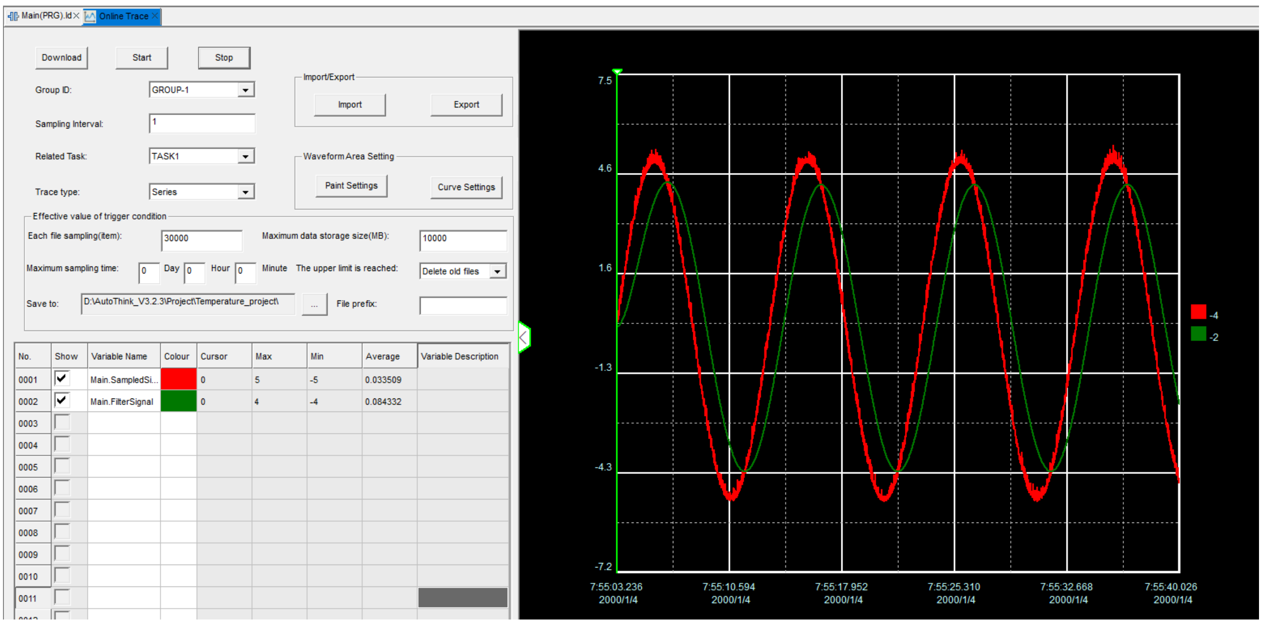


- Program Segment 0003 applies the First_Order_Filter to filter and reduce noise in the noisy SampledSignal. The task cycle is set to 0.01s, and TimeConst is set to 1.



The results of online tracking of SampledSignal and FilteredSignal are shown in the figure below. It can be observed that the First_Order_Filter has a noticeable noise reduction effect on the signal.

- Red line: The original signal before filtering (SampledSignal).
- Green line: The filtered signal (FilteredSignal).



4.1.1.6 Precautions

In practice, the signal code values from TC and AI modules require scaling to obtain the actual values (such as temperature and voltage) used as the SampledSignal raw data.

This function block needs to be called in a cyclic IEC task.

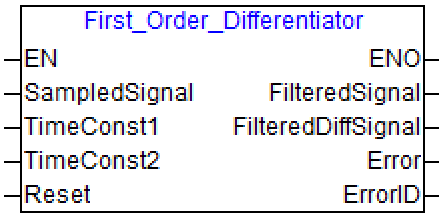
4.1.2 First_Order_Differentiator(First Order Differentiator)

4.1.2.1 Version

The software version supporting this function block is V3.2.3 and above. Controller firmware versions that do not support this instruction include LX-CU500-A01 (V1.0.0) and LX-CU500-A02 (V1.1.0).

4.1.2.2 Function

By connecting two first-order filters in series with a forward difference operation, high-frequency components in both the signal and its derivative are simultaneously filtered out.



First_Order_Differentiator Functional Block

4.1.2.3 Parameter

Parameter	Statement	Data Type	Function Description	Initial Value
SampledSignal	Input	LREAL	Signal Input Connect the acquired signal to this pin, and the First_Order_Differentiator Functional will perform filtering on the signal.	0
TimeConst1	Input	LREAL	Filter Time Constant (Unit: Seconds) The time constant used for filtering the process value. When this value is set to 0, filtering for the process value is disabled, and the output value will be the same as the input value. The settings of TimeConst1 and TimeConst2 directly affect the amplitude and phase of the signal. Larger values of TimeConst1 and TimeConst2 result in better noise filtering and denoising effects but lead to reduced amplitude and greater phase lag. Smaller values of TimeConst1 and TimeConst2 result in poorer noise filtering and denoising effects but preserve amplitude closer to the original signal with less phase lag. The TimeConst1 and TimeConst2 parameters should be configured to be 10 to 100 times the task cycle.	0

TimeConst2	Input	LREAL	Differential Filter Time Constant (Unit: Seconds) This parameter sets the time constant used for filtering the derivative of the process value. When the value is set to 0, filtering for the derivative of the process value is disabled, and the output will be the forward first-order difference of the process value. The settings of TimeConst1 and TimeConst2 directly affect the amplitude and phase of the signal. Larger values of TimeConst1 and TimeConst2 result in better noise filtering and denoising effects but lead to reduced amplitude and greater phase lag. Smaller values of TimeConst1 and TimeConst2 result in poorer noise filtering and denoising effects but preserve amplitude closer to the original signal with less phase lag. The TimeConst1 and TimeConst2 parameters should be configured to be 10 to 100 times the task cycle.	0
Reset	Input	BOOL	Reset Configuration: TRUE: Reset FALSE: No Reset	FALSE
FilteredSignal	Output	LREAL	Filtered signal output.	0
FilteredDiffSignal	Output	LREAL	Filtered differential signal output.	0
Error	Output	BOOL	Input Parameter Invalid Flag. If the input parameter is invalid, the algorithm block will exit, and this parameter will output TRUE.	FALSE
ErrorID	Output	CTRL_ALGO_LIB_ERR_LIST	VALID: No error Others: Refer to the error code	VALID

4.1.2.4 Error Code

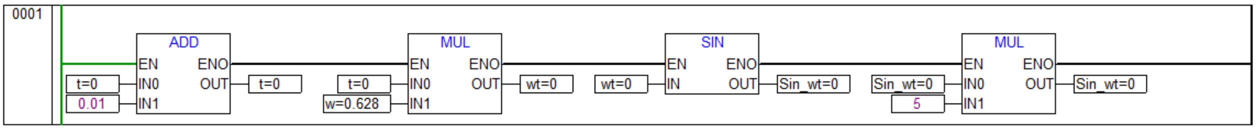
Error ID	Description
SampledSignal_REQUIRED_FINITE	Acquired signal range must be -1×10^{300}~1×10^{300}
TimeConst1_REQUIRED_NON_NEGA	TimeConst1 must be a non-negative number.
TimeConst1_REQUIRED_FINITE	TimeConst1 must not exceed 1×10^{300}.
TimeConst2_REQUIRED_NON_NEGA	TimeConst2 must be a non-negative number.
TimeConst2_REQUIRED_FINITE	TimeConst2 must not exceed 1×10^{300}.
TASK_IS_SINGLE	Task is Single execution
TASK_IS_FULLSPEED	Task is FullSpeed execution

4.1.2.5 Example

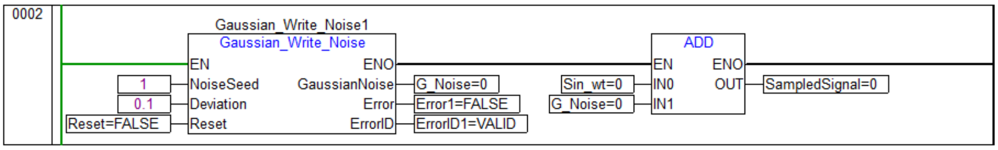
This example is only intended to further illustrate the use of First_Order_Differentiator, not for actual application scenarios.

The First_Order_Differentiator is used to filter and reduce noise in a signal while also extracting its differential signal with the following explanation:

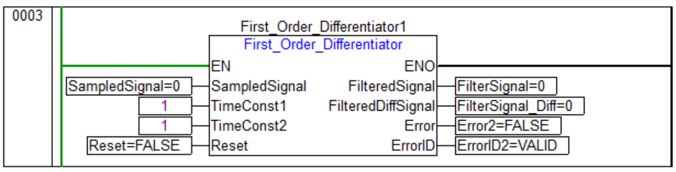
- Program Segment 0001 simulates a sine signal with an amplitude of 5, a frequency of 0.628 rad/s, and an initial phase of 0.



- Program Segment 0002 utilizes the custom function block Gaussian_Write_Noise to simulate Gaussian white noise with a mean square error of 0.1. This noise signal is then added to the sine signal from Program Segment 0001, resulting in the original signal SampledSignal.

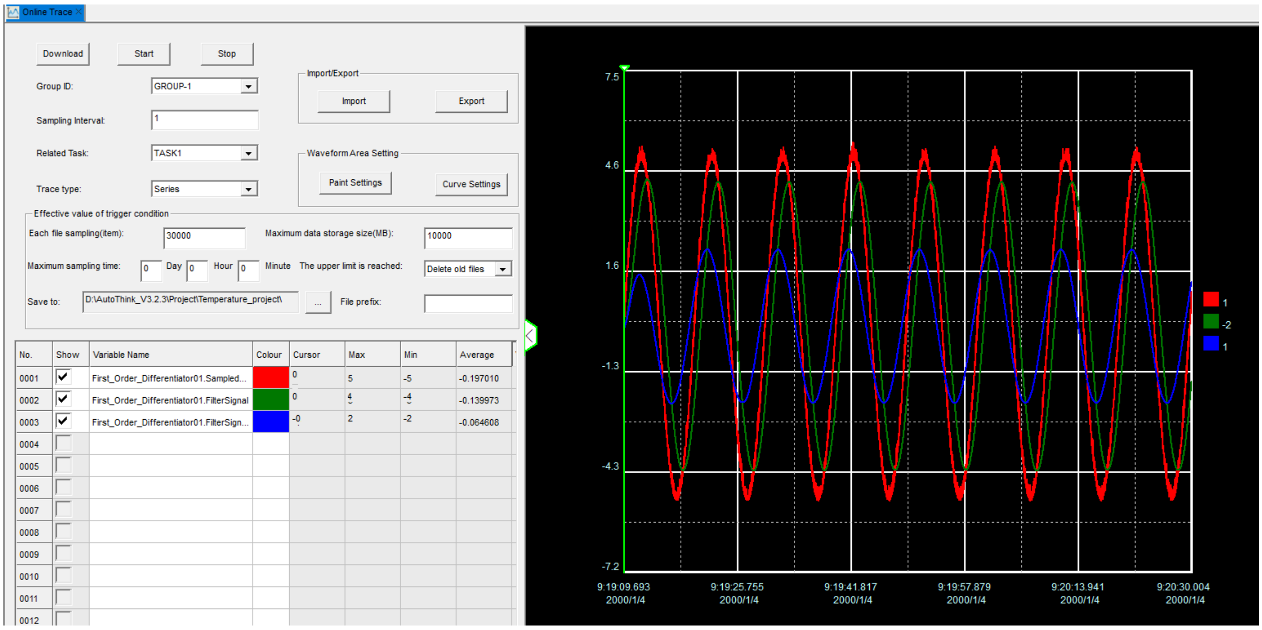


- Program Segment 0003 applies the First_Order_Differentiator to filter and reduce noise in the noisy SampledSignal and to extract its differential signal. The task cycle is set to 0.01s, and TimeConst1/TimeConst2 are both set to 1.



The online tracking results of SampledSignal, FilteredSignal, and FilteredDiffSignal are shown in the figure below. It can be observed that the First_Order_Differentiator effectively removes signal noise, and the extracted differential signal is essentially noise-free.

- Red line: the original signal before filtering (SampledSignal).
- Green line: the filtered signal (FilteredSignal).
- Blue line: the filtered differential signal (FilteredDiffSignal).



4.1.2.6 Precautions

In practice, the signal code values from TC and AI modules require scaling to obtain the actual values (such as temperature and voltage) used as the SampledSignal raw data.

This function block needs to be called in a cyclic IEC task.

4.2 FIR Filters

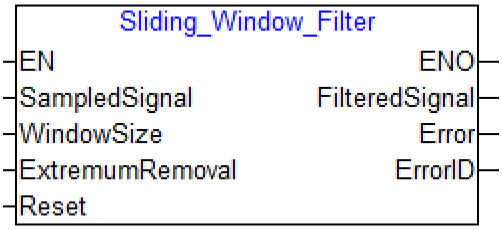
4.2.1 Sliding_Window_Filter(Sliding Window Filter)

4.2.1.1 Version

The software version supporting this function block is V3.2.3 and above. Controller firmware versions that do not support this instruction include LX-CU500-A01 (V1.0.0) and LX-CU500-A02 (V1.1.0).

4.2.1.2 Function

By averaging a fixed number of historical input signals to obtain the filtered signal, high-frequency noise in the signal can be effectively filtered out.



Sliding_Window_Filter Functional Block

4.2.1.3 Parameter

Parameter	Statement	Data Type	Function Description	Initial Value
SampledSignal	Input	LREAL	Feed the acquired temperature signal (or other process signals requiring monitoring) into this parameter, and the sliding window filter will process the signal for filtering.	0
WindowSize	Input	DINT	Window Size of the Sliding Window Filter When this value is set to 1 (unit: count), the sliding window filter will be disabled, and the output signal equals the input signal. The larger the WindowSize, the better the averaging effect but the greater the phase lag; the smaller the WindowSize, the poorer the averaging effect but the smaller the phase lag. Therefore, the selection of WindowSize requires careful consideration to choose parameters suitable for the signal.	0
ExtremumRemoval	Input	BOOL	Extreme Value Rejection Switch TRUE: Enables the function to reject maximum and minimum values in the sliding window filter (only effective when WindowSize is greater than or equal to 3). FALSE/WindowSize＜3: Disables the function. When individual values of the collected signal within the filtering window fluctuate significantly, it is recommended to remove the extreme values.	FALSE
Reset	Input	BOOL	Reset Configuration: TRUE: Reset FALSE: No Reset	FALSE
FilteredSignal	Output	LREAL	Filtered result of the acquired signal.	0
Error	Output	BOOL	If the input parameters are invalid, this parameter outputs TRUE.	FALSE
ErrorID	Output	CTRL_ALGO_LIB_ERR_LIST	VALID: No error Others: Refer to the error code	VALID

4.2.1.4 Error Code

Error ID	Description
SampledSignal_REQUIRED_FINITE	Acquired signal range must be -1×10^{300} to 1×10^{300}
WindowSize_REQUIRED_LE_1000	WindowSize must not exceed 1000.

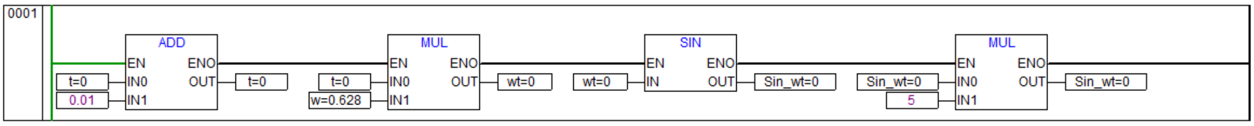
WindowSize_REQUIRED_POSI	WindowSize must be a positive value.
TASK_IS_SINGLE	Task is Single execution
TASK_IS_FULLSPEED	Task is FullSpeed execution

4.2.1.5 Example

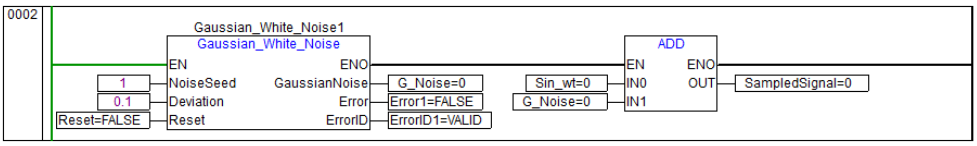
This example is only intended to further illustrate the use of Sliding_Window_Filter, not for actual application scenarios.

This example uses a Sliding_Window_Filter to perform moving average processing on a signal, with the following explanation:

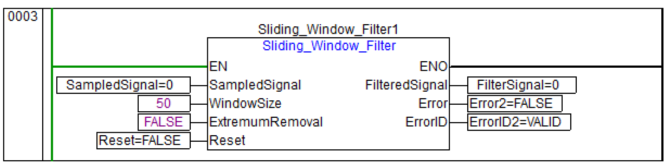
- Program Segment 0001 simulates a sine signal with an amplitude of 5, a frequency of 0.628 rad/s, and an initial phase of 0.



- Program Segment 0002 utilizes the custom function block Gaussian_Write_Noise to simulate Gaussian white noise with a mean square error of 0.1. This noise signal is then added to the sine signal from Program Segment 0001, resulting in the original signal SampledSignal.

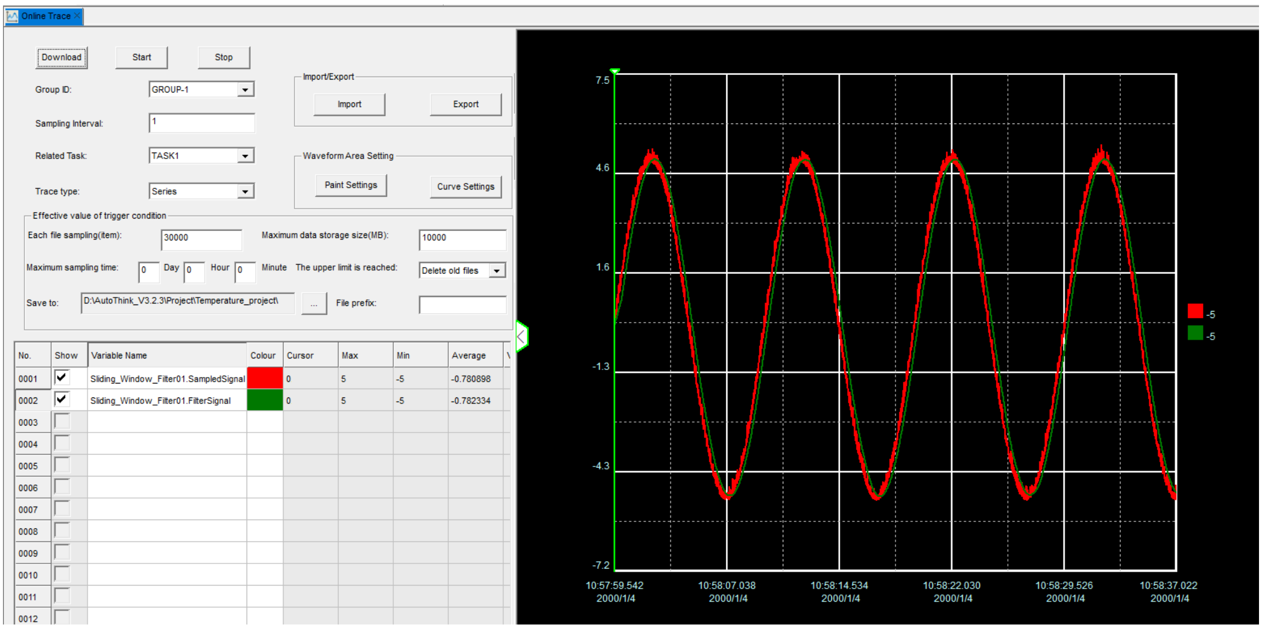


- Program Segment 0003 applies the Sliding_Window_Filter to filter and reduce noise in the noisy SampledSignal. The task cycle is set to 0.01s, and the window size is configured as 50.



The online tracking results of SampledSignal and FilteredSignal are shown in the figure below. It can be observed that the Sliding_Window_Filter effectively removes signal noise.

- Red line represents the original signal before filtering (SampledSignal).
- Green line represents the filtered signal (FilteredSignal).



4.2.1.6 Precautions

In practice, the signal code values from TC and AI modules require scaling to obtain the actual values (such as temperature and voltage) used as the SampledSignal raw data.

This function block needs to be called in a cyclic IEC task.

Chapter 5 Signal Generators and Processors

5.1 Pulse_Width_Modulation_ZOH(Zero Order Hold Pulse Width Modulation)

5.1.1 Version

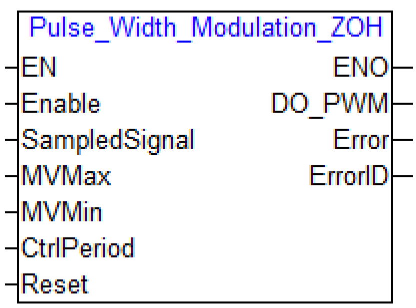
The software version supporting this function block is V3.2.3 and above. Controller firmware versions that do not support this instruction include LX-CU500-A01 (V1.0.0) and LX-CU500-A02 (V1.1.0).

5.1.2 Function

This function block implements the generation of Pulse Width Modulation (abbreviated as PWM) signals. PWM is a digital technique used to control analog signals by adjusting the duty cycle of the signal.

In this function block, the input signal undergoes linear interpolation between sampling points via a zero-order hold to generate the PWM signal. The control system converts discrete control signals into continuous step signals using the zero-order hold. Combined with PWM technology, it achieves digital closed-loop control, which suppresses high-frequency interference and optimizes system stability.

This function block is suitable for application scenarios that require switching outputs for control purposes, such as relay control.



Pulse_Width_Modulation_ZOH Functional Block

5.1.3 Parameter

Parameter	Statement	Data Type	Function Description	Initial Value

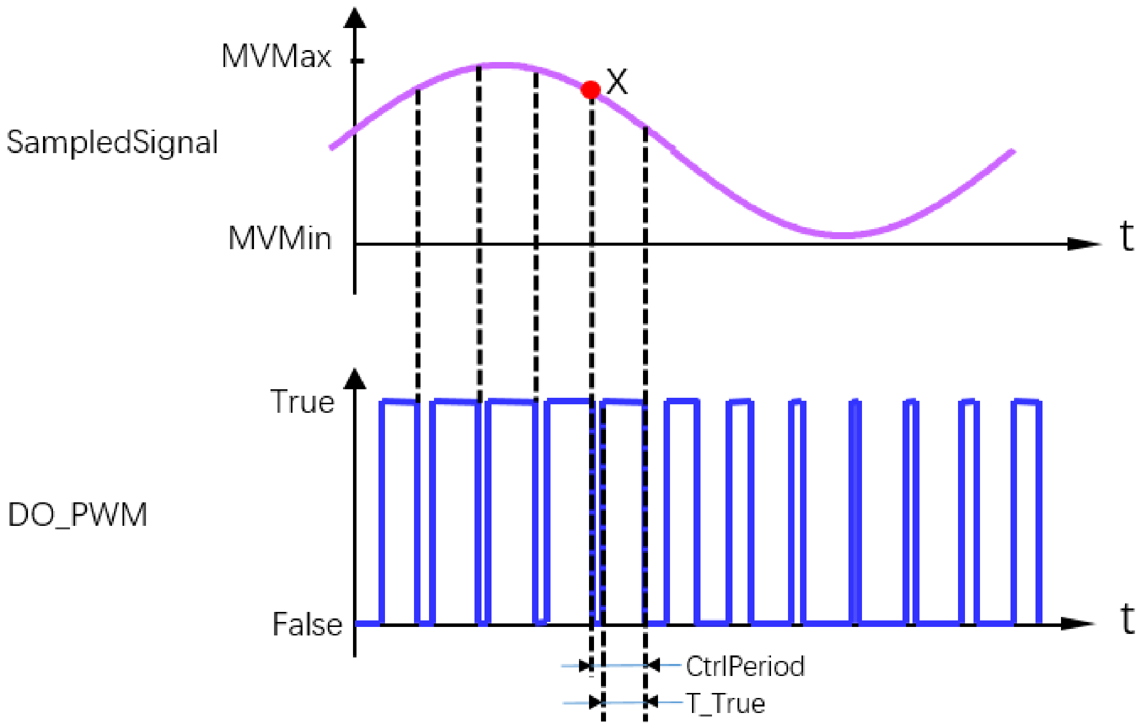
Enable	Input	BOOL	High-level enable switch.	FALSE
SampledSignal	Input	LREAL	Input control variable, usually a certain control output value.	0
MVMax	Input	LREAL	Maximum controllable output of the channel (must use the same unit as the control quantity in control mode).	100
MVMin	Input	LREAL	Minimum controllable output of the channel (must use the same unit as the control quantity in control mode).	0
CtrlPeriod	Input	LREAL	The period of the PWM output signal (unit: seconds) This parameter sets the control period for pulse width modulation. After the task cycle is set, a larger CtrlPeriod value results in higher duty cycle output precision but a lower sampling frequency. Conversely, a smaller CtrlPeriod value leads to a higher sampling frequency but reduced duty cycle output precision. The task cycle for calling this function block is generally set to 10~100ms, and CtrlPeriod is recommended to be set to 10~100 times the task cycle of calling this function block.	0
Reset	Input	BOOL	Reset Configuration: TRUE: Reset FALSE: No Reset	FALSE
DO_PWM	Output	BOOL	Pulse width modulation digital output point, i.e., the DO point used for control.	FALSE
Error	Output	BOOL	If the input parameters are invalid, this parameter outputs TRUE.	FALSE
ErrorID	Output	CTRL_ALGO_LIB_ERR_LIST	VALID: No error Others: Refer to the error code	VALID

5.1.4 Error Code

Error ID	Description
SampledSignal_REQUIRED_FINITE	Acquired signal range must be -1×10^{300} to 1×10^{300}
MVMax_REQUIRED_FINITE	MVMax must be within the range of -1×10^{300} to 1×10^{300}.
MVMax_REQUIRED_GT_MVMin	MVMax must be greater than MVMin.
MVMin_REQUIRED_FINITE	MVMin must be within the range of -1×10^{300} to 1×10^{300}.
ControlPeriod_REQUIRED_FINITE	CtrlPeriod must be within the range of 1 × task cycle time to 2×10^{9} × task cycle time.
TASK_IS_SINGLE	Task is Single execution
TASK_IS_FULLSPEED	Task is FullSpeed execution

5.1.5 Example

As shown in the figure, the Pulse_Width_Modulation_ZOH converts the acquired analog signal into a digital signal. Within one control cycle, it first outputs a low level (FALSE), followed by a high level (TRUE). The duty cycle of the high level (TRUE) is: $\frac{\text{X-MVMin}}{\text{MVMax-MVMin}}$ or $\frac{\text{T\_True}}{\text{CtrlPeriod}}$, while the duty cycle of the low level (FALSE) is: 1-High Level Duty Cycle.



Note:

- **X**: Actual value of SampledSignal.
- **T_True**: Duration of the high level within the cycle.

5.1.6 Precautions

MVMax and MVMin represent the upper and lower limits of the output clamping. Since physical systems inherently have output constraints, output limiting is a mandatory configuration item. Please ensure appropriate settings for the upper and lower limits (MVMax and MVMin).

This function block needs to be called in a cyclic IEC task.

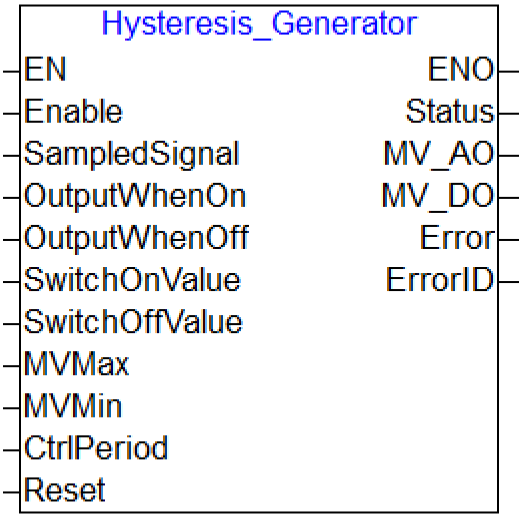
5.2 Hysteresis_Generator(Hysteresis Generator)

5.2.1 Version

The software version supporting this function block is V3.2.3 and above. Controller firmware versions that do not support this instruction include LX-CU500-A01 (V1.0.0) and LX-CU500-A02 (V1.1.0).

5.2.2 Function

This function block is used to handle scenarios where the input signal causes frequent switching of relay logic around a critical point due to dead zone hysteresis. After setting the activation value and deactivation value of the hysteresis loop, the output signal switches to the ON state when the input signal exceeds the set activation value, and switches to the OFF state when the input signal falls below the set deactivation value.



Hysteresis_Generator Functional Block

5.2.3 Parameter

Parameter	Statement	Data Type	Function Description	Initial Value
Enable	Input	BOOL	High-level enable switch.	FALSE
SampledSignal	Input	LREAL	Input signal.	0
OutputWhenOn	Input	LREAL	Output value when relay is ON.	0
OutputWhenOff	Input	LREAL	Output value when relay is OFF.	0
SwitchOnValue	Input	LREAL	Hysteresis Loop Deactivation Value: If the input signal is greater than or equal to this value, the output signal will equal the input value of the OutputWhenOn pin.	0
SwitchOffValue	Input	LREAL	Hysteresis Loop Activation Value: If the input signal is less than or equal to this value, the output signal will equal the input value of the OutputWhenOff pin.	0
MVMax	Input	LREAL	Maximum controllable output of the channel (must use the same unit as the control quantity in control mode).	0
MVMin	Input	LREAL	Minimum controllable output of the channel (must use the same unit as the control quantity in control mode)	0

CtrlPeriod	Input	LREAL	Control cycle of DO output signal (unit: seconds). The task cycle for calling this function block is generally set to 10~100ms, and CtrlPeriod is recommended to be set to 10~100 times the task cycle of calling this function block.	0
Reset	Input	BOOL	Reset Configuration: TRUE: Reset FALSE: No Reset	FALSE
Status	Output	BOOL	Current State of Hysteresis Loop TRUE: Hysteresis loop is in the activated state. FALSE: Hysteresis loop is in the deactivated state.	FALSE
MV_AO	Output	LREAL	Connected to analog output point.	0
MV_DO	Output	BOOL	Connected to relay output point.	FALSE
Error	Output	BOOL	If the input parameters are invalid, this parameter outputs TRUE.	FALSE
ErrorID	Output	CTRL_ALGO_LIB_ERR_LIST	VALID: No error Others: Refer to the error code	VALID

5.2.4 Error Code

Error ID	Description
SampledSignal_REQUIRED_FINITE	SampledSignal must be within the range of -1×10^{300} to 1×10^{300}.
MVMax_REQUIRED_FINITE	MVMax must be within the range of -1×10^{300} to 1×10^{300}.
MVMax_REQUIRED_GT_MVMin	MVMax must be greater than MVMin.
MVMin_REQUIRED_FINITE	MVMin must be within the range of -1×10^{300} to 1×10^{300}.
SwitchOffValue_REQUIRED_FINITE	SwitchOffValue must be within the range of -1×10^{300} to 1×10^{300}.
SwitchOnValue_REQUIRED_GE_SwitchOffValue	SwitchOnValue must be greater than or equal to SwitchOffValue.
SwitchOnValue_REQUIRED_FINITE	SwitchOnValue must be within the range of -1×10^{300} to 1×10^{300}.
ControlPeriod_REQUIRED_FINITE	ControlPeriod must be within the range of 1 × task cycle time to 2×10^{9} × task cycle time.
ControlPeriod_REQUIRED_GE_TimeStep	ControlPeriod must be greater than or equal to the time step.
OutputWhenOn_REQUIRED_GE_MVMin	OutputWhenOn must be greater than or equal to MVMin.
OutputWhenOn_REQUIRED_LE_MVMax	OutputWhenOn must be less than or equal to MVMax.
OutputWhenOff_REQUIRED_LE_MVMax	OutputWhenOff must be less than or equal to MVMax.
OutputWhenOff_REQUIRED_GE_MVMin	OutputWhenOff must be greater than or equal to MVMin.
TASK_IS_SINGLE	Task is Single execution
TASK_IS_FULLSPEED	Task is FullSpeed execution

5.2.5 Example

- The AO output is described as follows:

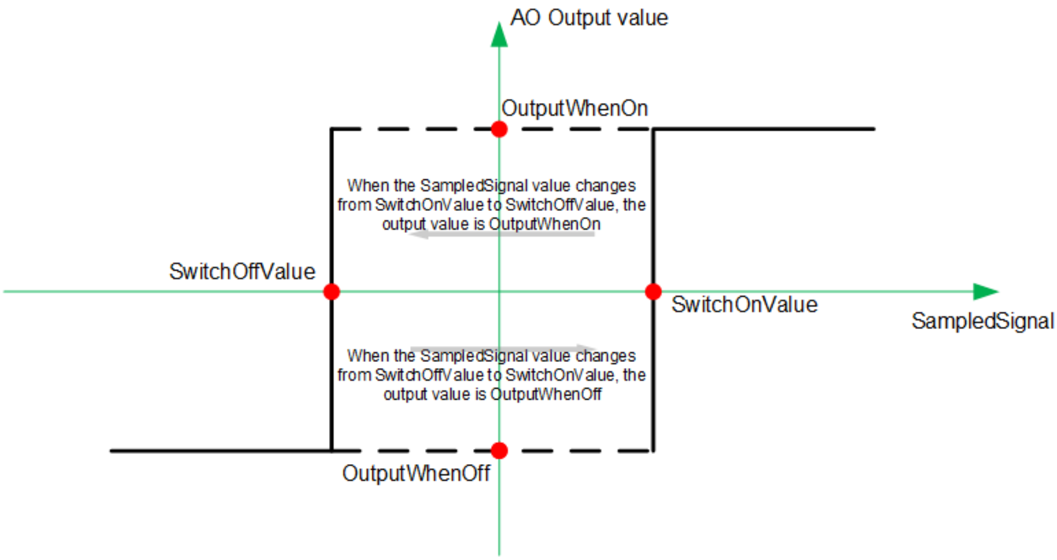
  Basic parameter settings are described as follows:

  - SwitchOnValue (Activation Point): When the input signal is greater than or equal to this value, the output switches to the "On" value.
  - SwitchOffValue (Deactivation Point): When the input signal is less than or equal to this value, the output switches to the "Off" value.

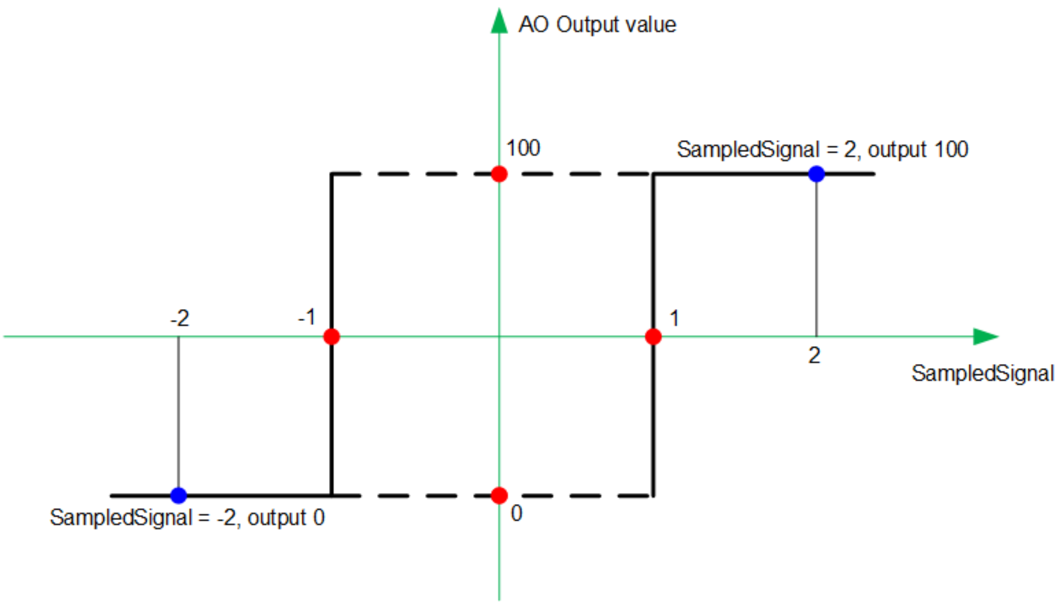
- OutputWhenOn (Output When Activated): Output value when the relay is activated.
- OutputWhenOff (Output When Deactivated): Output value when the relay is deactivated.

Function block logic is described as follows:

- When the input signal ≥ SwitchOnValue, the output becomes OutputWhenOn.
- When the input signal ≤ SwitchOffValue, the output becomes OutputWhenOff.
- When the input signal is between SwitchOffValue and SwitchOnValue, the output remains unchanged.



For example, set SwitchOnValue, SwitchOffValue, OutputWhenOn, and OutputWhenOff to 1, -1, 100, and 0 respectively, as shown in the figure below.



Program example description is as follows:

When SampledSignal is 2 (greater than SwitchOnValue), the output becomes 100.

When SampledSignal is -2 (less than SwitchOffValue), the output becomes 0.

When SampledSignal is between SwitchOnValue and SwitchOffValue, two scenarios occur:

- If SampledSignal gradually decreases from 2 to -1, the output remains 100 until SampledSignal reaches SwitchOffValue (-1), at which point the output switches to 0.
- If SampledSignal gradually increases from -2 to 1 the output remains 0 until SampledSignal reaches SwitchOnValue (1), at which point the output switches to 100.

- The DO output delivers a high or low level in the form of a duty cycle. For a detailed explanation, please refer to the Pulse_Width_Modulation_ZOH(Zero Order Hold Pulse Width Modulation) example.

5.2.6 Precautions

In practice, the signal code values from TC, AI and AO modules require scaling to obtain the actual values (such as temperature and voltage) used as the SampledSignal raw data.

MVMax and MVMin represent the upper and lower limits of the output clamping. Since physical systems inherently have output constraints, output clamping is mandatorily enabled and cannot be disabled. Please ensure appropriate settings for the upper and lower limits (MVMax and MVMin).

This function block needs to be called in a cyclic IEC task.

Chapter 6 PID Controller and Related Functions

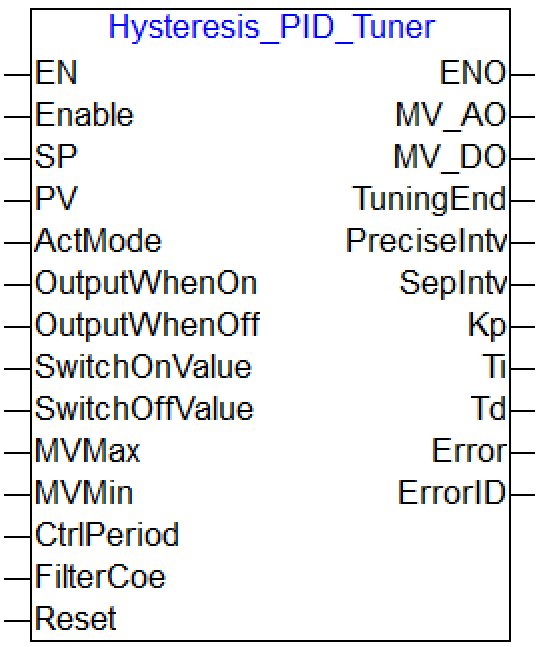
6.1 Hysteresis_PID_Tuner(Hysteresis PID Tuner)

6.1.1 Version

The software version supporting this function block is V3.2.3 and above. Controller firmware versions that do not support this instruction include LX-CU500-A01 (V1.0.0) and LX-CU500-A02 (V1.1.0).

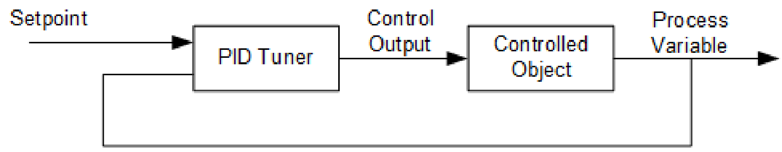
6.1.2 Function

This function block utilizes relay oscillation to excite the controlled object, causing its process variable to oscillate around the setpoint for two cycles. By collecting the relevant data, key frequency-domain parameters of the controlled object are identified, and the corresponding parameters for the PID controller are obtained.



Hysteresis_PID_Tuner Functional Block

The relationship and workflow diagram among the controlled object, setpoint, process variable, and control output of the Hysteresis PID Tuner are as follows.



6.1.3 Parameter

Parameter	Statement	Data Type	Function Description	Initial Value
Enable	Input	BOOL	High-level enable switch.	FALSE
SP	Input	LREAL	Control target setpoint.	0
PV	Input	LREAL	Process variable signal of the controlled object.	0
ActMode	Input	BOOL	Set to TRUE when an increase in the control output amplifies the error (direct action, e.g., cooling control). Set to FALSE when an increase in the control output reduces the error (reverse action, e.g., heating control).	FALSE
OutputWhenOn	Input	LREAL	Output value when relay is ON.	0
OutputWhenOff	Input	LREAL	Output value when relay is OFF.	0
SwitchOnValue	Input	LREAL	If the input signal is greater than this value, the output signal equals the input value of the OutputWhenOn pin.	0
SwitchOffValue	Input	LREAL	If the input signal is less than this value, the output signal equals the input value of the OutputWhenOff pin.	0
MVMax	Input	LREAL	Maximum controllable output of the channel (must use the same unit as the control quantity in control mode).	0
MVMin	Input	LREAL	Minimum controllable output of the channel (must use the same unit as the control quantity in control mode).	0
CtrlPeriod	Input	LREAL	Control cycle of DO output signal (unit: seconds). The task cycle for calling this function block is generally set to 10~100ms, and CtrlPeriod is recommended to be set to 10~100 times the task cycle of calling this function block.	0
FileCoe	Input	LREAL	Time Constant of the First-Order Filter (embedded in the auto-tuning module)	0
Reset	Input	BOOL	Reset Configuration: TRUE: Reset FALSE: No Reset	FALSE
MV_AO	Output	LREAL	Connected to analog output.	0
MV_DO	Output	BOOL	Connected to relay output.	FALSE
TuningEnd	Output	BOOL	Auto-Tuning Completion Flag: Auto-tuning ends when this parameter transitions from FALSE to TRUE.	FALSE

PreciseIntv	Output	LREAL	System Inertia Zone. Serves as a reference for setting parameters in the PID controller's imprecise control mode.	0
SepIntv	Output	LREAL	System Amplitude. Serves as a reference for setting parameters related to the integral separation interval of the PID controller.	0
KP	Output	LREAL	Proportional coefficient of the PID controller.	0
TI	Output	LREAL	Integral time of the PID controller (unit: seconds).	0
TD	Output	LREAL	Derivative time of the PID controller (unit: seconds).	0
Error	Output	BOOL	If the input parameters are invalid, this parameter outputs TRUE.	FALSE
ErrorID	Output	CTRL_ALGO_LIB_ERR_LIST	VALID: No error Others: Refer to the error code	VALID

6.1.4 Error Code

Error ID	Description
MVMax_REQUIRED_FINITE	MVMax must be within the range of -1×10^{300} to 1×10^{300}.
MVMin_REQUIRED_FINITE	MVMin must be within the range of -1×10^{300} to 1×10^{300}.
MVMax_REQUIRED_GT_MVMin	MVMax must be greater than MVMin.
SwitchOffValue_REQUIRED_FINITE	SwitchOffValue must be within the range of -1×10^{300} to 1×10^{300}.
SwitchOnValue_REQUIRED_GE_SwitchOffValue	SwitchOnValue must be greater than or equal to SwitchOffValue.
SwitchOnValue_REQUIRED_FINITE	SwitchOnValue must be within the range of -1×10^{300} to 1×10^{300}.
ControlPeriod_REQUIRED_FINITE	ControlPeriod must be within the range of 1 × task cycle time to 2×10^{9} × task cycle time.
ControlPeriod_REQUIRED_GE_TimeStep	ControlPeriod must be greater than or equal to the time step.
OutputWhenOn_REQUIRED_GE_MVMin	OutputWhenOn must be greater than or equal to MVMin.
OutputWhenOn_REQUIRED_LE_MVMax	OutputWhenOn must be less than or equal to MVMax.
OutputWhenOff_REQUIRED_LE_MVMax	OutputWhenOff must be less than or equal to MVMax.
OutputWhenOff_REQUIRED_GE_MVMin	OutputWhenOff must be greater than or equal to MVMin.
FilterCoe_REQUIRED_NON_NEGA	FilterCoe must be a non-negative value.
FilterCoe_REQUIRED_FINITE	FilterCoe must be within the range of 0 to 1×10^{300}.
TASK_IS_SINGLE	Task is Single execution
TASK_IS_FULLSPEED	Task is FullSpeed execution

6.1.5 Example

This example is only intended to further illustrate the use of Hysteresis_PID_Tuner, not for actual application scenarios.

This example demonstrates the use of PID auto-tuning by controlling a heating device. For simplicity and practical relevance, the input to the controlled object can be directly regarded as power, and the output as temperature.

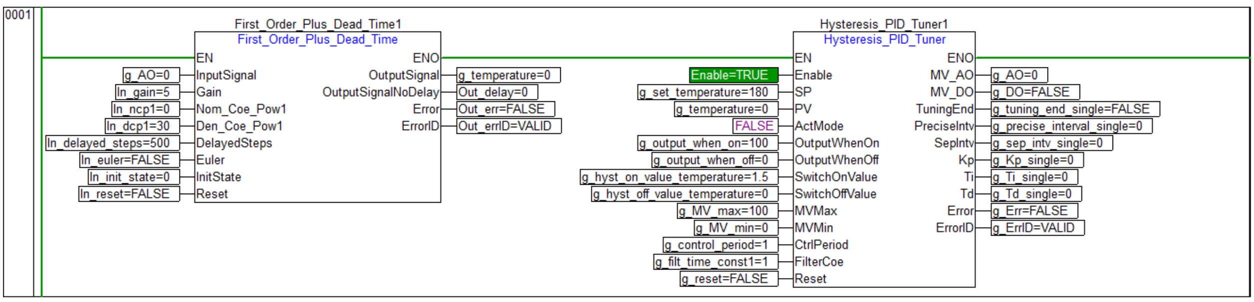
In the program editing area, call the function block First_Order_Plus_Dead_Time, for simulating process signals. In this example, this corresponds to the transfer function $\frac{5}{30s+1}e^{-5s}$ of the controlled object's output, with a Gain of 5, Den_Coe_Pow1 of 30, and DelayedSteps of 500.

The figure below shows an example program before tuning has started.

Description of Hysteresis_PID_Tuner Parameter Settings:

Parameter Configuration Description:

- For the input pins: Enable is triggered by a high level. SP is the setpoint. PV is the process value (in actual engineering applications, this corresponds to the controlled variable acquired from channels such as TC or AI; in this example, it represents the output of the heating device as the controlled object).
- ActMode is set to FALSE (set to TRUE for cooling mode).
- OutputWhenOn, OutputWhenOff, SwitchOnValue, and SwitchOffValue are the four hysteresis loop variables. It is generally recommended to set these parameters to 100, 0, 1.5, and 0, respectively. For details on hysteresis loop usage, refer to the Hysteresis_Generator(Hysteresis Generator) documentation.
- MVMax and MVMin are the maximum and minimum output values, typically set to 100 and 0. In some engineering scenarios, MVMax may be 1000, requiring adjustments to OutputWhenOn and SwitchOnValue (e.g., set to 1000 and 15, respectively). Parameters can also be customized based on actual field conditions.
- CtrlPeriod is the control cycle for DO output, set to 10-100 times the task cycle (typically 1). FilterCoe is the filtering coefficient for PV, generally set to 1.



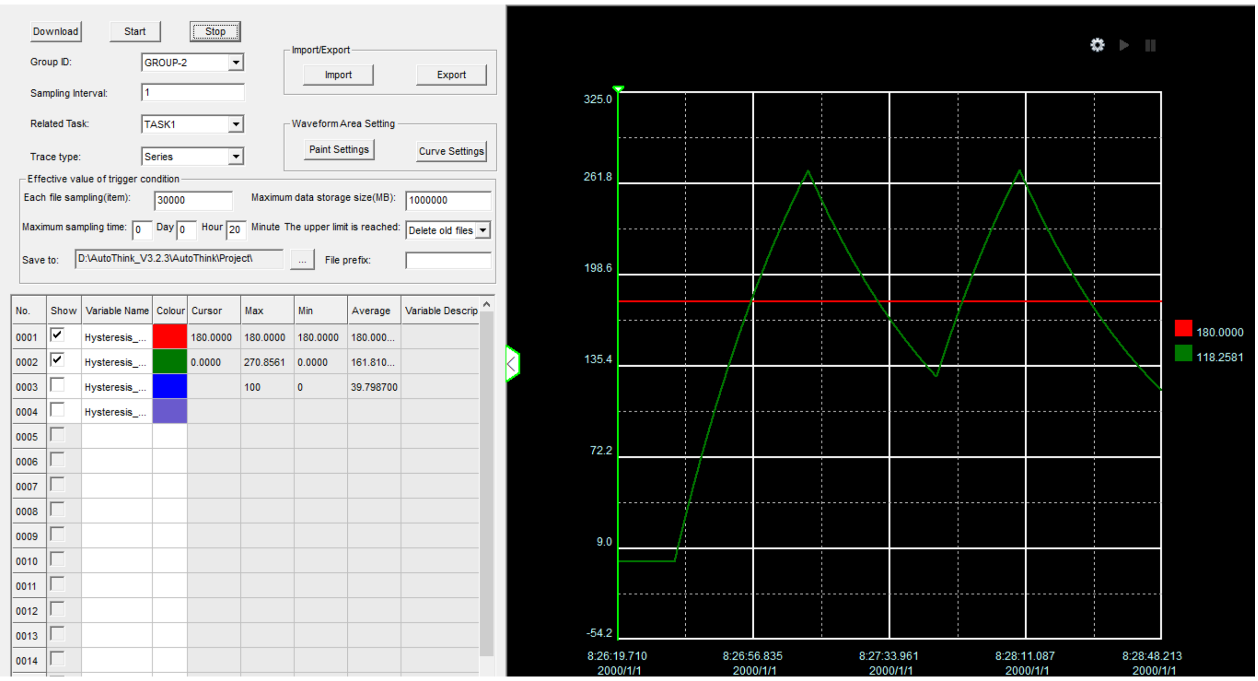
In the parameter settings described above:

- OutputWhenOn is set to 100, meaning the output is 100 when turned on. OutputWhenOff is set to 0, meaning the output is 0 when turned off.
- SwitchOnValue is set to 1.5, indicating that when the filtered value of SP minus PV is less than 1.5, the output becomes OutputWhenOn (100). This occurs when the process temperature is relatively low, requiring heating at full power.
- SwitchOffValue is set to 0, indicating that when the filtered value of SP minus PV is greater than 0, the output becomes OutputWhenOff (0). This occurs when the process temperature reaches the setpoint, and heating power is reduced to 0.
- MVMax and MVMin clamp the output value (MV). Their parameter range must include the interval between OutputWhenOn and OutputWhenOff; otherwise, the program will trigger errors (Error and ErrorID) during operation.
- For output pins, if using analog control, pass MV_AO to the output channel. If using digital control, use

MV_DO. For detailed instructions, please refer to Pulse_Width_Modulation_ZOH(Zero Order Hold Pulse Width Modulation) and Hysteresis_Generator(Hysteresis Generator) . Please ensure proper conversion of analog values and configuration of digital output channels.

The figure below shows the complete tuning process captured through data tracking.

- Red line: Setpoint (SP).
- Green line: Process Value (PV).

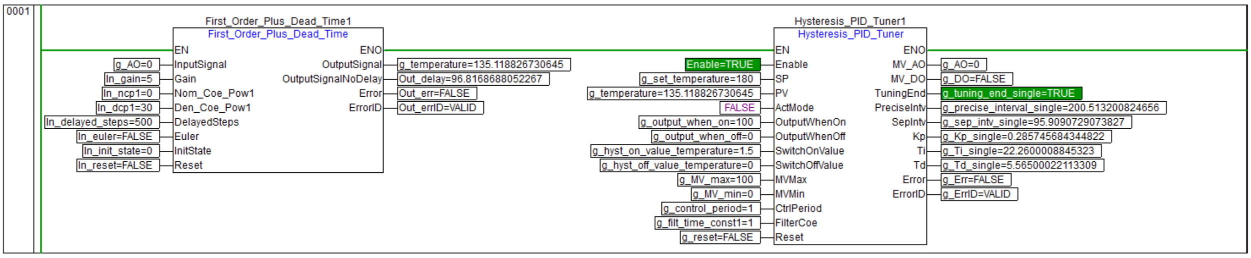


The Hysteresis_PID_Tuner after completing tuning is shown in the figure below.

TuningEnd is the tuning completion flag. If it is TRUE, it indicates that tuning has ended, and parameters such as PreciseIntv, SepIntv, Kp, Ti, and Td will be updated. The output parameters are described as follows:

- PreciseIntv: Recommended fine-tuning interval (Outside the range of setpoint ± PreciseIntv, bang-bang control is applied, i.e., full power or zero power adjustment; within this range, PID control is used).
- SepIntv: Recommended integral separation interval.
- Kp/Ti/Td: Proportional coefficient, integral time, and derivative time, respectively.

- To avoid manually recording the tuning results, please set the above 5 output parameters as global variables.



If an error occurs, Error will be TRUE, and ErrorID will display the error message.

6.1.6 Precautions

In practice, the signal code values from TC, AI and AO modules require scaling to obtain the actual values (such as temperature and voltage) used as the PV raw data.

Please distinguish between the output parameters OutputWhenOn (output when enabled) and OutputWhenOff (output when disabled), and the actual hardware channel states (on/off), to avoid confusion.

MVMax and MVMin represent the upper and lower limits of the output clamping. Since physical systems inherently have output constraints, output clamping is mandatorily enabled and cannot be disabled. Ensure appropriate settings for these limits (MVMax and MVMin).

It is recommended to configure PreciseIntv, SepIntv, Kp, Ti, and Td as global variables and set to a retentive type. Additionally, ensure that the output parameters of the Hysteresis_PID_Tuner (PreciseIntv, SepIntv, Kp, Ti, Td) match the input parameters of the PID_Controller (PreciseIntv, SepIntv, Kp, Ti, Td). This avoids the need to manually record tuning results from the Hysteresis_PID_Tuner and manually input parameters into the PID_Controller.

The tuning process is time-consuming. It is advisable to save the tuning results promptly to prevent loss due to operational errors, which may require repeating the tuning process.

This function block needs to be called in a cyclic IEC task.

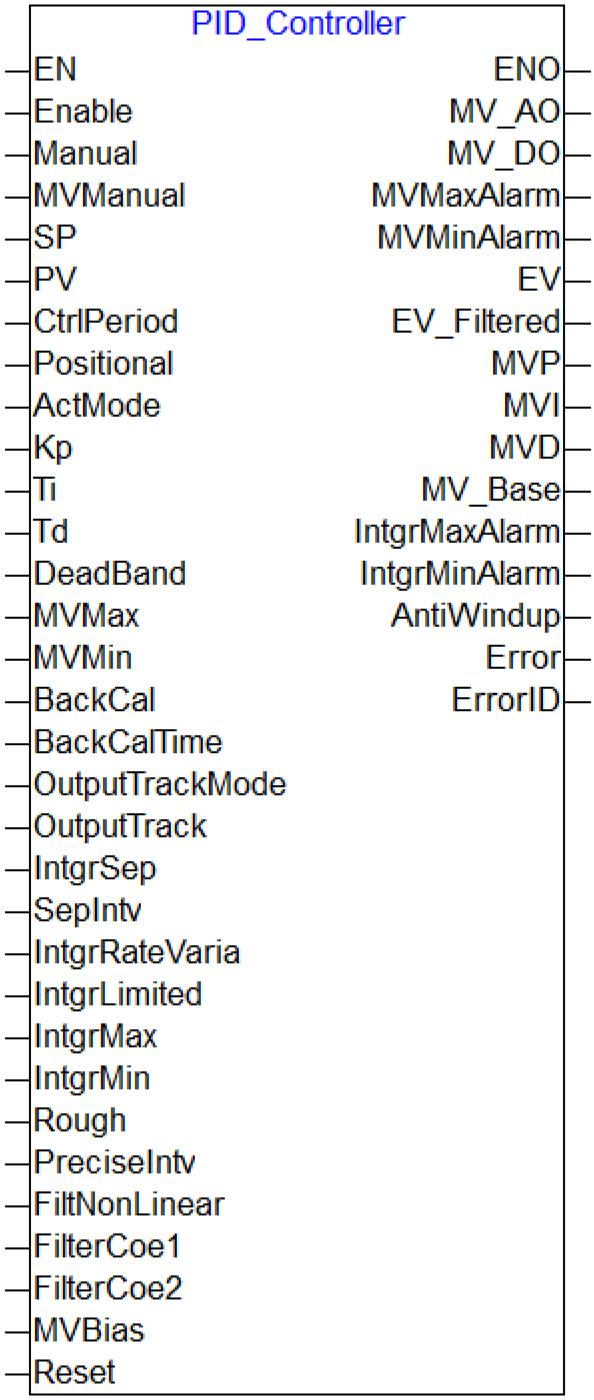
6.2 PID_Controller(PID Controller)

6.2.1 Version

The software version supporting this function block is V3.2.3 and above. Controller firmware versions that do not support this instruction include LX-CU500-A01 (V1.0.0) and LX-CU500-A02 (V1.1.0).

6.2.2 Function

Designed for process control applications, the PID controller autonomously adjusts the control output by monitoring error signals, ensuring the process value of the controlled object stabilizes near the setpoint.



PID_Controller Functional Block

PID_Controller Functional Description:

- The seamless manual/automatic switching is implemented through incremental mode:

$$action\_pid = manual\_mv + \Delta action$$

- Both incremental and positional PID calculations are performed simultaneously. The Positional pin ultimately determines which calculated control output (incremental or positional) is selected.

- In direct action mode, by taking the negative value of the error, a directly controlled object can be transformed into an equivalent reverse-acting controlled object.
- The process value undergoes derivative-first processing to prevent sudden changes in the derivative component caused by setpoint jumps.
- The process value is filtered, and the filter can be selected based on the FiltNonLinear pin.
- In dead zone operation, only the integral component continues calculating, while the proportional and derivative components are reset to zero.
- By calculating the difference between the unconstrained MV and constrained MV, integral feedback is generated to achieve anti-windup compensation.
- Output tracking mode is suitable for control scenarios where there is a numerical discrepancy between the MV command and its execution. By reading the actually executed MV value through a sensor and feeding it back to the PID module, the PID module can perform incremental calculations, anti-windup operations, and bumpless manual/automatic switching operations more accurately.
- Variable integration rate only comes into effect when integral separation is enabled. The formula for calculating its variable integration coefficient r is as follows:

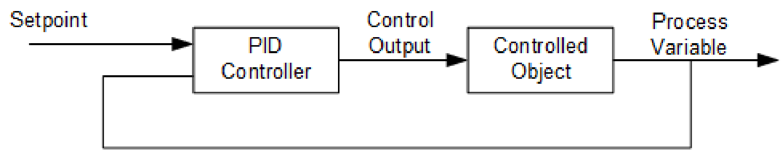
$$r = \frac{error - dead\_zone}{sep\_intv - dead\_zone}$$

- The PID module adopts the following standard form for calculations:

$$\frac{Y(s)}{X(s)} = K_p\left(1 + \frac{1}{T_i s} + T_d s\right)$$

Before using this function block, it is generally necessary to configure the channel acquisition for temperature or process values, as well as analog or digital output control. Please pay attention to the conversion of code values during use. For such external configurations, refer to the user manuals of modules such as TC, AO, and DO.

Regarding the controlled object, setpoint, process value, and control output of the PID controller, their relationships and flowchart are as follows.



6.2.3 Parameter

Parameter	Statement	Data Type	Function Description	Initial Value
Enable	Input	BOOL	High-level enable switch.	FALSE
Manual	Input	BOOL	Mode Switching TRUE: Manual Mode	FALSE

			FALSE: Automatic Mode In manual mode, the input value of MVManual is directly assigned to the output parameter.	
MVManual	Input	LREAL	The control output value in manual mode. In this mode, the input value of this parameter will be used as the PID control output, and PID control will no longer be performed.	0
SP	Input	LREAL	The set control target value.	0
PV	Input	LREAL	The process variable that requires control.	0
CtrlPeriod	Input	LREAL	Control cycle of DO output signal (unit: seconds). It is recommended to set the parameter to 10–100 times the task cycle that calls this function block.	0
Positional	Input	BOOL	Position/Incremental Mode Switch TRUE: Positional PID FALSE: Incremental PID For general temperature control scenarios, it is recommended to use the default incremental control mode, as it provides better anti-windup handling.	FALSE
ActMode	Input	BOOL	Cooling/Heating Mode Switch FALSE: Heating mode. The control output and process variable change in the same direction—this mode should be used when the control output and error change in opposite directions (e.g., the heating power of a heater causes temperature increase). TRUE: Cooling mode. The control output and process variable change in opposite directions—this mode should be used when the control output and error change in the same direction (e.g., the cooling power of a chiller causes temperature decrease).	FALSE
Kp	Input	LREAL	Proportional coefficient of the PID controller.	0
Ti	Input	LREAL	Integral time of the PID controller (unit: seconds).	0
Td	Input	LREAL	Derivative time of the PID controller (unit: seconds).	0
DeadBand	Input	LREAL	Dead zone of the PID controller. Within the range of setpoint ± DeadBand, only integral control is performed.	0
MVMax	Input	LREAL	Maximum controllable output of the channel (must use the same unit as the control quantity in control mode).	0
MVMin	Input	LREAL	Minimum controllable output of the channel (must use the same unit as the control quantity in control mode).	0
BackCal	Input	BOOL	Anti-windup Method Switch FALSE: Use clamping method for anti-windup TRUE: Use back-calculation method for anti-windup	FALSE
BackCalTime	Input	LREAL	When using back-calculation anti-windup, this parameter sets the integral time for the anti-windup calculation. Setting this value to 0 disables the anti-windup function.	0
OutputTrackMode	Input	BOOL	Output Tracking Mode TRUE: Enabled FALSE: Disabled	FALSE

			This pin should be enabled when there is a discrepancy between the actual control output and the calculated PID output, provided that the control mechanism has corresponding sensors. If the actuator's input shows significant changes while the output remains largely unchanged (possibly indicating integral saturation within the actuator), using the actuator's output as an integral reference can prevent integral saturation.	
OutputTrack	Input	LREAL	Used to input the actual control output value executed by the actuator. When both BackCal and OutputTrackMode are set to TRUE and the BackCalTime parameter is configured, OutputTrack represents the converted value of the actuator's output (conversion enables comparison with MV).	0
IntgrSep	Input	BOOL	Integral Separation Switch TRUE: The integral component will only accumulate when the error enters the integral separation range FALSE: The integral component accumulates throughout the entire process	FALSE
SepIntv	Input	LREAL	Integral separation value. When the integral separation switch is set to TRUE, integral control will only be active within the SP±SepInt range; outside this range, only PD control will be performed.	0
IntgrRateVaria	Input	BOOL	Variable Integration Rate Mode FALSE: Disabled TRUE: Enabled When enabled (requires integral separation to be activated first), within the integral separation range, the integral component will be multiplied by a coefficient (greater than 0 and less than 1). This mode further weakens the integral effect on the basis of integral separation. It is suitable for controlled objects that are extremely prone to overshoot.	FALSE
IntgrLimited	Input	BOOL	Integral Limiting Switch TRUE: Upper limit for integral component can be set FALSE: No upper limit for integral component When IntgrLimited is TRUE, the maximum value of the integral component is IntgrMax, and the minimum value is IntgrMin. If FALSE, there are no restrictions.	FALSE
IntgrMax	Input	LREAL	Manually set upper limit value for the integral component.	0
IntgrMin	Input	LREAL	.Manually set lower limit value for the integral component	0
Rough	Input	BOOL	Coarse/Fine Adjustment Switch TRUE: Fine adjustment (PID control). When set to TRUE, PID control is performed within the fine adjustment zone (setpoint ± fine adjustment range), while full-power control is applied outside this zone. FALSE: Coarse adjustment (full-power control: either maximum or zero power). When set to FALSE, PID control is applied throughout the entire process.	FALSE
PreciseIntv	Input	LREAL	Fine adjustment zone value setting.	0

FiltNonLinear	Input	BOOL	Tracking Derivative/First-Order Filtering Derivative Switch TRUE: The PID controller module uses a tracking-differentiator method to filter and differentiate the process variable FALSE: The PID controller module uses a traditional first-order filter method to filter and differentiate the process variable	FALSE
FilterCoe1	Input	LREAL	When the FiltNonLinear pin input is TRUE: This pin inputs the filtering factor of the built-in tracking-differentiator in the PID controller module. For settings, refer to the relevant instructions of the Tracker_Differentiator(Tracker Differentiator). When the FiltNonLinear pin input is FALSE: This pin inputs the time constant of the built-in first-order filter used for processing the process value in the PID controller module. For settings, refer to the relevant instructions of the First_Order_Differentiator(First Order Differentiator).	0
FilterCoe2	Input	LREAL	When the FiltNonLinear pin input is TRUE: This pin inputs the tracking speed factor of the built-in tracking-differentiator in the PID controller module. For settings, refer to the relevant instructions of the Tracker_Differentiator(Tracker Differentiator). When the FiltNonLinear pin input is FALSE: This pin inputs the time constant of the built-in first-order filter in the PID controller module. For settings, refer to the relevant instructions of the First_Order_Differentiator(First Order Differentiator).	0
MVBias	Input	LREAL	Feedforward control value: The output of the PID controller is the sum of the control value calculated by the controller itself and the control value input by this parameter.	0
Reset	Input	BOOL	Reset Configuration: TRUE: Reset FALSE: No Reset	FALSE
MV_AO	Output	LREAL	Connected to analog output.	0
MV_DO	Output	BOOL	Connected to switch duty cycle output.	FALSE
MV_MaxAlarm	Output	BOOL	When this flag is TRUE, the control value calculated by the PID controller module exceeds the maximum control value.	FALSE
MV_MinAlarm	Output	BOOL	When this flag is TRUE, the control value calculated by the PID controller module is below the minimum control value.	FALSE
EV	Output	LREAL	Error value.	0
EV_Filtered	Output	LREAL	Filtered error value.	0
MVP	Output	LREAL	Proportional component's contribution to the total control output.	0
MVI	Output	LREAL	Integral component's contribution to the total control output.	0
MVD	Output	LREAL	Derivative component's contribution to the total control output.	0
MV_Base	Output	LREAL	When PID mode is set to incremental, this pin displays the controller output in internal automatic mode (activated when switching from manual mode to automatic mode).	0

IntgrMaxAlarm	Output	BOOL	This flag is TRUE when the integral component calculated by the PID controller module exceeds the set upper limit.	FALSE
IntgrMinAlarm	Output	BOOL	This flag is TRUE when the integral component calculated by the PID controller module falls below the set lower limit.	FALSE
AntiWindup	Output	BOOL	This flag is TRUE when the PID controller module is in anti-windup state.	FALSE
Error	Output	BOOL	If the input parameters are invalid, this parameter outputs TRUE.	FALSE
ErrorID	Output	CTRL_ALGO_LIB_ERR_LIST	VALID: No error Others: Refer to the error code	VALID

6.2.4 Error Code

Error ID	Description
MVManual_REQUIRED_FINITE	The control output in manual mode must be within specified limits.
SampledSignal_REQUIRED_FINITE	SampledSignal range requirement: -1×10^{300} to 1×10^{300}.
Kp_REQUIRED_FINITE	Kp range requirement: 0 to 1×10^{300}.
Kp_REQUIRED_NON_NEGA	Kp must be a non-negative number.
RECIPROCAL_OF_Ti_REQUIRED_FINITE	The allowable range for the reciprocal of Ti is 0 to 1×10^{300}.
Ti_REQUIRED_NON_NEGA	Ti must be a non-negative number.
TD_REQUIRED_FINITE	Td range requirement: 0 to 1×10^{300}.
TD_REQUIRED_NON_NEGA	Td must be a non-negative number.
CtrlPeriod_REQUIRED_FINITE	CtrlPeriod requirement: 1 * task cycle time ~ 2×10^{9} * task cycles.
CtrlPeriod_REQUIRED_GE_TimeStep	CtrlPeriod must be greater than or equal to TimeStep.
DeadBand_REQUIRED_FINITE	DeadBand range must be between 0 and 1×10^{300}.
DeadBand_REQUIRED_NON_NEGA	DeadBand must be a non-negative number.
MVMax_REQUIRED_FINITE	MVMax range must be between -1×10^{300} and 1×10^{300}.
MVMin_REQUIRED_FINITE	MVMin range must be between -1×10^{300} and 1×10^{300}.
MVMax_REQUIRED_GT_MVMin	MVMax must be greater than MVMin.
BackCalTime_REQUIRED_FINITE	BackCalTime range must be between $1/1\times10^{300}$ and 1×10^{300}.
BackCalTime_REQUIRED_NON_NEGA	BackCalTime must be a non-negative number.
SepIntv_REQUIRED_FINITE	SepIntv range must be greater than the dead zone and less than 1×10^{300}.
SepIntv_REQUIRED_GT_DeadBand	SepIntv must be greater than the dead zone.
IntgrMax_REQUIRED_FINITE	IntgrMax range must be between $1/1\times10^{300}$ and 1×10^{300}.
IntgrMax_REQUIRED_GT_IntgrMin	IntgrMax must be greater than the IntgrMin.
IntgrMin_REQUIRED_FINITE	IntgrMin range must be between $1/1\times10^{300}$ and 1×10^{300}.
PreciseIntv_REQUIRED_FINITE	PreciseIntv must be greater than or equal to SepIntv and less than 1×10^{300}.
PreciseIntv_REQUIRED_GE_Sepintv	PreciseIntv must be greater than or equal to SepIntv.
FilterCoe1_REQUIRED_FINITE	FilterCoe1 must be between 0 and 1×10^{300}.
FilterCoe1_REQUIRED_NON_NEGA	FilterCoe1 is a non-negative value.
FilterCoe2_REQUIRED_FINITE	FilterCoe2 must be between 0 and 1×10^{300}.
FilterCoe2_REQUIRED_NON_NEGA	FilterCoe2 is a non-negative value.
TASK_IS_SINGLE	Task is Single execution
TASK_IS_FULLSPEED	Task is FullSpeed execution

6.2.5 Example

This example is only intended to further illustrate the use of PID_Controller, not for actual application scenarios.

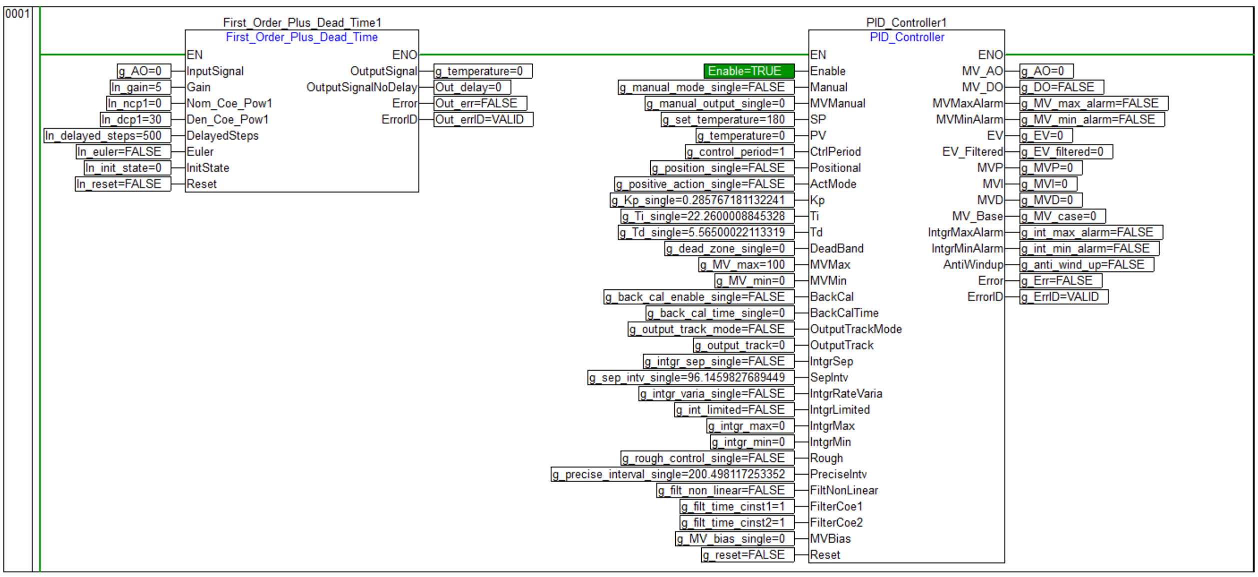
In the program editing area, call the function block First_Order_Plus_Dead_Time, for simulating process signals. In this example, this corresponds to the transfer function $\frac{5}{30s+1}e^{-5s}$ output of the controlled object, with a Gain of 5, Den_Coe_Pow1 of 30, and DelayedSteps of 500.

The figure below shows an example program before control begins.

The input parameters PreciseIntv, SepIntv, Kp, Ti, and Td are the results of PID auto-tuning. Please set them as global variables.

For an explanation of the AO output (MV_AO), refer to [Hysteresis_Generator(Hysteresis Generator)].

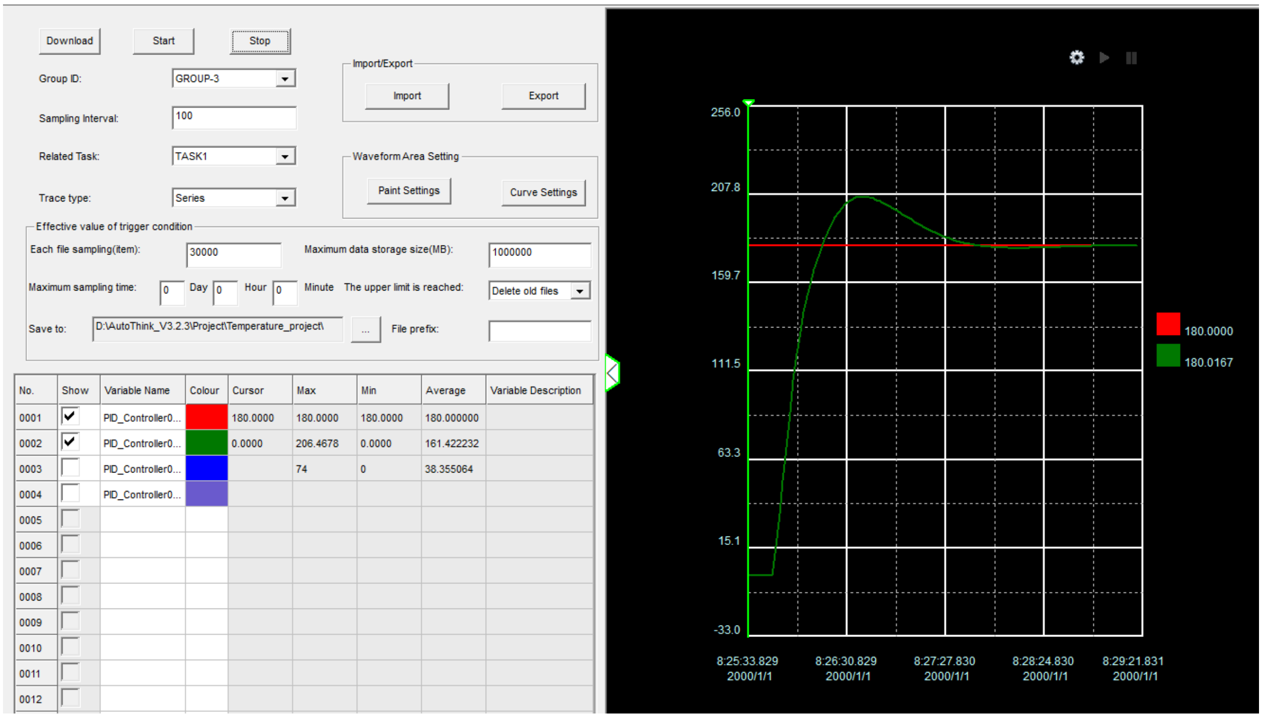
For an explanation of the DO output (MV_DO), refer to [Pulse_Width_Modulation_ZOH(Zero Order Hold Pulse Width Modulation)].



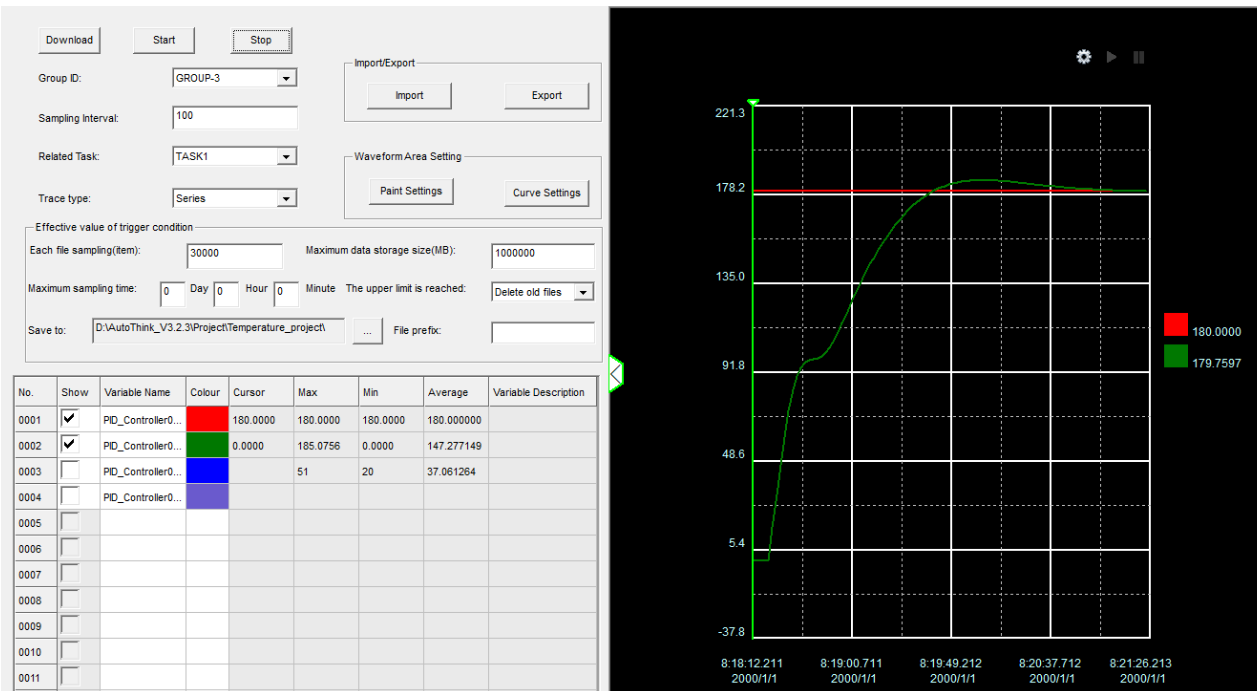
The following figure shows the setpoint and process values for a heating equipment controlled object under three conditions: using only PID control, adding integral separation (IntgrSep set to TRUE), and further adding variable integral rate (IntgrSep and IntgrRateVaria set to TRUE).

- Red line: The setpoint SP.
- Green line: The process value PV.

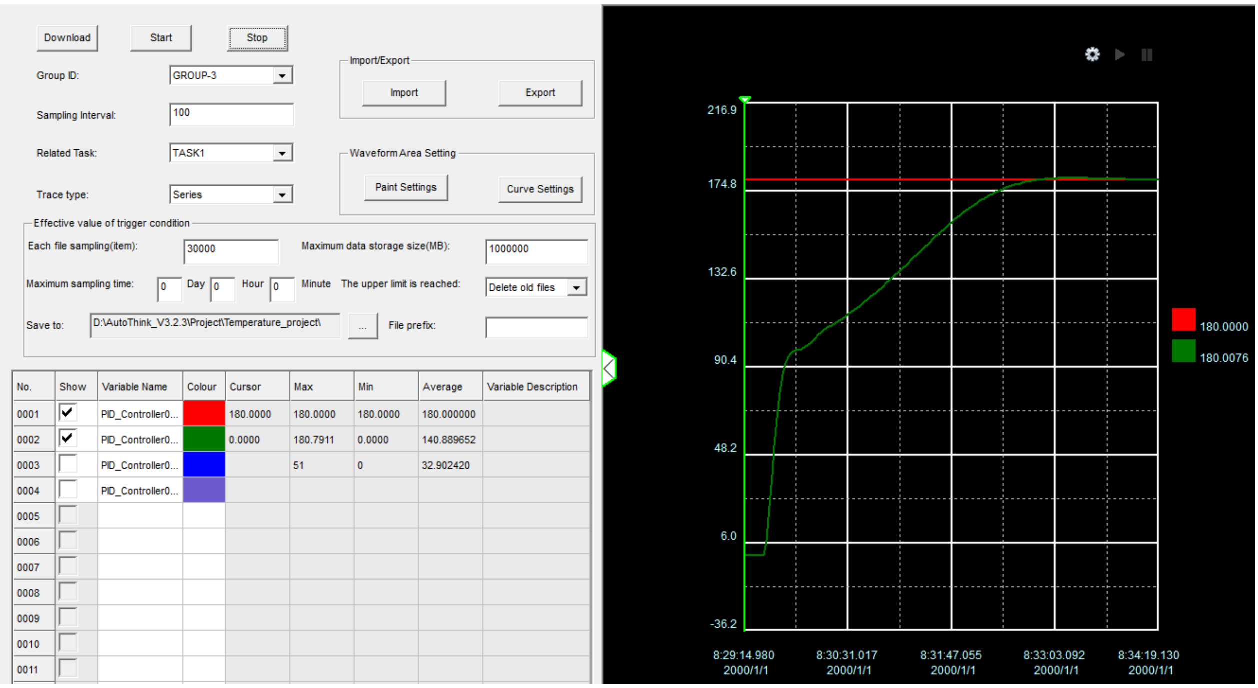
The results of online tracking of SP and PV under pure PID control are shown in the following figure:



The results of online tracking of SP and PV after incorporating integral separation are shown in the following figure:



The results of online tracking of SP and PV after incorporating both integral separation and variable integral gain are shown in the following figure:



If cascade control is used, simply connect two PID modules in series, noting that the AO (analog output) of the outer-loop PID should be linked to the PV (process value) of the inner-loop PID.

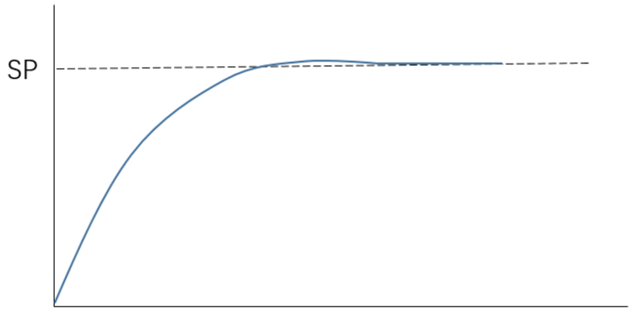
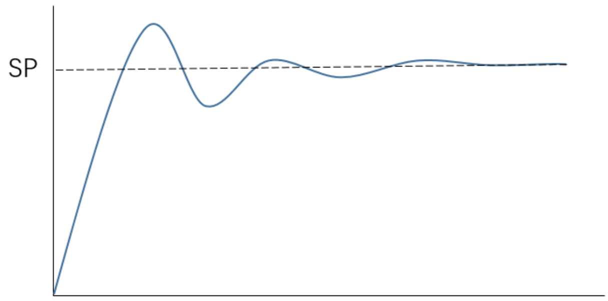
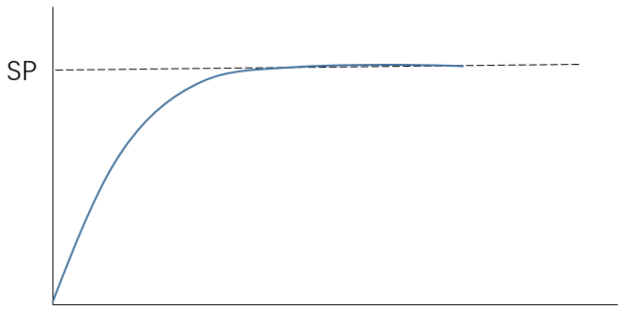
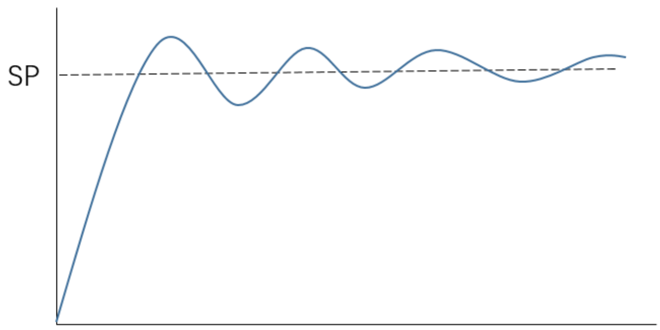
If slope tracking is required, it can be configured first. The code logic is as follows:

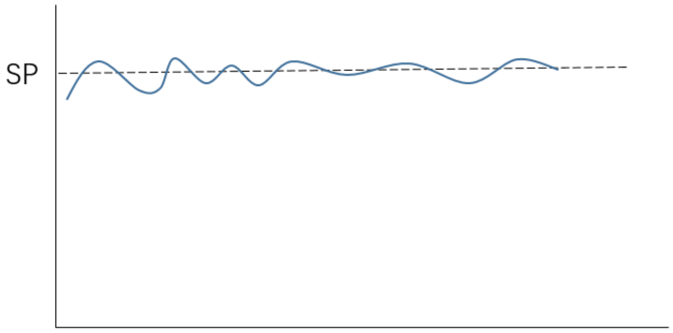
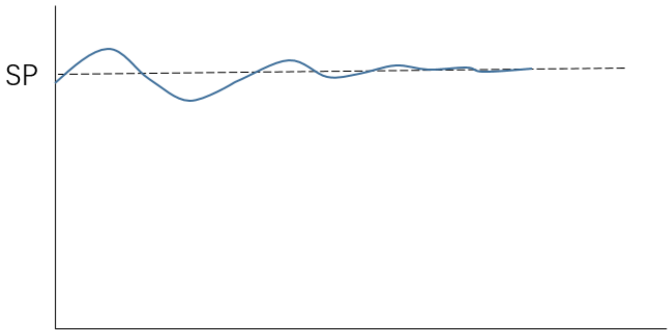
```
IF set_temperature < target_ temperature THEN
  set_temperature := set_temperature + set_ramp * time_step;
ELSE
   set_temperature := target_ temperature;
END_IF
```

Here, target_temperature refers to the final target temperature, set_temperature is the set temperature at that moment, set_ramp is the set ramp rate, and time_step is the time step, which is the task cycle. The overall logic of the code is that when the final target temperature has not been reached, the temperature changes according to the set ramp rate. Once the final target temperature is reached, the ramp tracking stops, and the set temperature becomes the final target temperature.

If the control performance of the auto-tuning parameters is unsatisfactory, the parameters can be adjusted manually. When manually adjusting the PID constants, the changes in the process value generally behave as follows:

	The change in the measured value

Decreasing the proportional coefficient (Kp)		The curve gradually rises with an extended settling time, which can help avoid overshoot.
Increasing the proportional coefficient (Kp)		Although overshoot and fluctuations may occur, the target value can be reached more quickly.
Increasing the integral time (Ti)		The time to reach the target value becomes longer, and the settling time increases, while fluctuations, overshoot, and undershoot are reduced.
Decreasing the integral time (Ti)		Fluctuations occur, but the system can start up rapidly.

Increasing the derivative time (Td)		Overshoot, undershoot, and settling time are all reduced, though slight fluctuations may occur near the set value.
Decreasing the derivative time (Td)		Overshoot and undershoot are more significant, and it takes longer to return to the set value.

6.2.6 Precautions

In practice, the signal code values from TC, AI and AO modules require scaling to obtain the actual values (such as temperature and voltage) used as the PV raw data.

MVMax and MVMin represent the upper and lower limits of the output clamping. Since physical systems inherently have output constraints, output clamping is mandatorily enabled and cannot be disabled. Ensure appropriate settings for these limits (MVMax and MVMin).

It is recommended to configure PreciseIntv, SepIntv, Kp, Ti, and Td as global variables. Additionally, ensure that the output parameters of the Hysteresis_PID_Tuner (PreciseIntv, SepIntv, Kp, Ti, Td) match the input parameters of the PID_Controller (PreciseIntv, SepIntv, Kp, Ti, Td). This avoids the need to manually record tuning results from the Hysteresis_PID_Tuner and manually input parameters into the PID_Controller.

This function block needs to be called in a cyclic IEC task.

Chapter 7 ADRC and Related Functions

Active Disturbance Rejection Control (ADRC) technology is a novel practical digital control technology developed by embracing the essence of PID control techniques and incorporating achievements from modern control theory. It was explored through the induction, summarization, and synthesis of computer simulation results. This technology does not rely on precise models of the controlled object and can serve as a replacement for PID control techniques.

The characteristics and advantages of ADRC technology are as follows:

- ADRC employs an "observation + compensation" approach to address nonlinearities and uncertainties in control systems, while utilizing nonlinear feedback methods to enhance the dynamic performance of the controller.
- The ADRC algorithm is simple, easy to implement, and offers high precision, fast response, and strong disturbance rejection capabilities.
- It provides a unified approach to control problems for both deterministic and uncertain systems; disturbance suppression does not require external disturbance models or the observability of disturbances; the control algorithm does not necessitate the identification of the controlled object; it uniformly handles nonlinear and linear systems; it can be applied to time-delay system control; and for decoupling control, only static coupling needs to be considered, without regard to dynamic coupling.

7.1 Components of ADRC

7.1.1 Tracker_Differentiator(Tracker Differentiator)

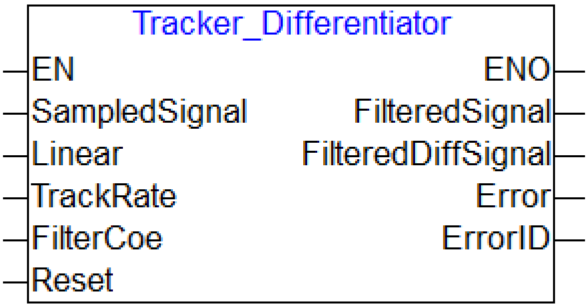
7.1.1.1 Version

The software version supporting this function block is V3.2.3 and above. Controller firmware versions that do not support this instruction include LX-CU500-A01 (V1.0.0) and LX-CU500-A02 (V1.1.0).

7.1.1.2 Function

The Tracking Differentiator is a core component of the Active Disturbance Rejection Controller, primarily used for the smooth tracking and differential extraction of noisy signals. By designing a transition process to track the input signal and combining it with a differential operator to extract the rate of change, it achieves noise suppression and signal shaping.

The function block performs filtering (tracking) and differentiation on the input signal. During data acquisition, this block can be used to preprocess the data, and it can also be employed in fixed heating rate control mode to monitor the heating rate.



Tracker_Differentiator Functional Block

7.1.1.3 Parameter

Parameter	Statement	Data Type	Function Description	Initial Value
SampledSignal	Input	LREAL	Feed the collected temperature signal (or other process signals requiring monitoring) into this pin, and the tracking differentiator will perform filtering and differentiation on the signal.	0
Linear	Input	BOOL	When the input is FALSE, the nonlinear mode will be adopted. When the input is TRUE, the linear mode will be adopted. It is recommended to use the nonlinear mode for better performance.	FALSE
TrackRate	Input	LREAL	This pin will only function if and only if the input to the Linear pin is TRUE. Input the tracking speed factor of the tracking differentiator; the larger this value, the smaller the phase lag, but the poorer the filtering effect.	0
FilterCoe	Input	LREAL	Input the filtering factor of the tracking differentiator; the larger this value, the better the filtering effect, but the greater the phase lag.	0
Reset	Input	BOOL	Reset Configuration: TRUE: Reset FALSE: No Reset	FALSE
FilteredSignal	Output	LREAL	The filtered result of the acquired signal.	0
FilteredDiffSignal	Output	LREAL	The differentiated result of the filtered acquired signal.	0
Error	Output	BOOL	If the input parameter is invalid, the parameter will output TRUE.	FALSE
ErrorID	Output	CTRL_ALGO_LIB_ERR_LIST	VALID: No error Others: Refer to the error code	VALID

7.1.1.4 Error Code

Error ID	Description

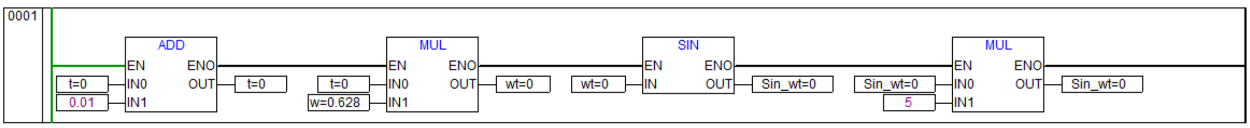
SampledSignal_REQUIRED_FINITE	SampledSignal range requirement -1×10^{300}~1×10^{300} .
FilterCoe_REQUIRED_POSI	FilterCoe must be positive.
FilterCoe_REQUIRED_FINITE	FilterCoe range requirement 0~1×10^{300}.
TrackRate_REQUIRED_POSI	TrackRate must be positive.
TrackRate_REQUIRED_FINITE	TrackRate range requirement 0~1×10^{300}.
TASK_IS_SINGLE	Task is Single execution.
TASK_IS_FULLSPEED	Task is FullSpeed execution..

7.1.1.5 Example

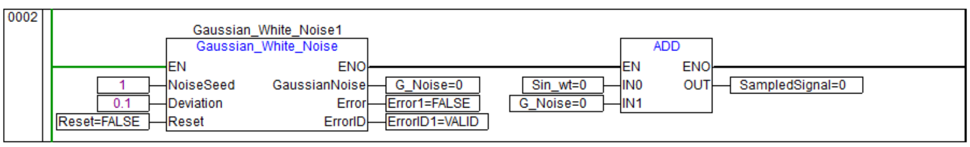
This example is only intended to further illustrate the use of Tracker_Differentiator, not for actual application scenarios.

This example uses a Tracker_Differentiator to perform sliding-window average processing on a signal, with the following explanation:

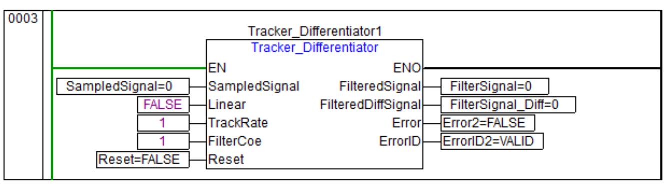
- Program Segment 0001 simulates a sine signal with an amplitude of 5, a frequency of 0.628 rad/s, and an initial phase of 0.



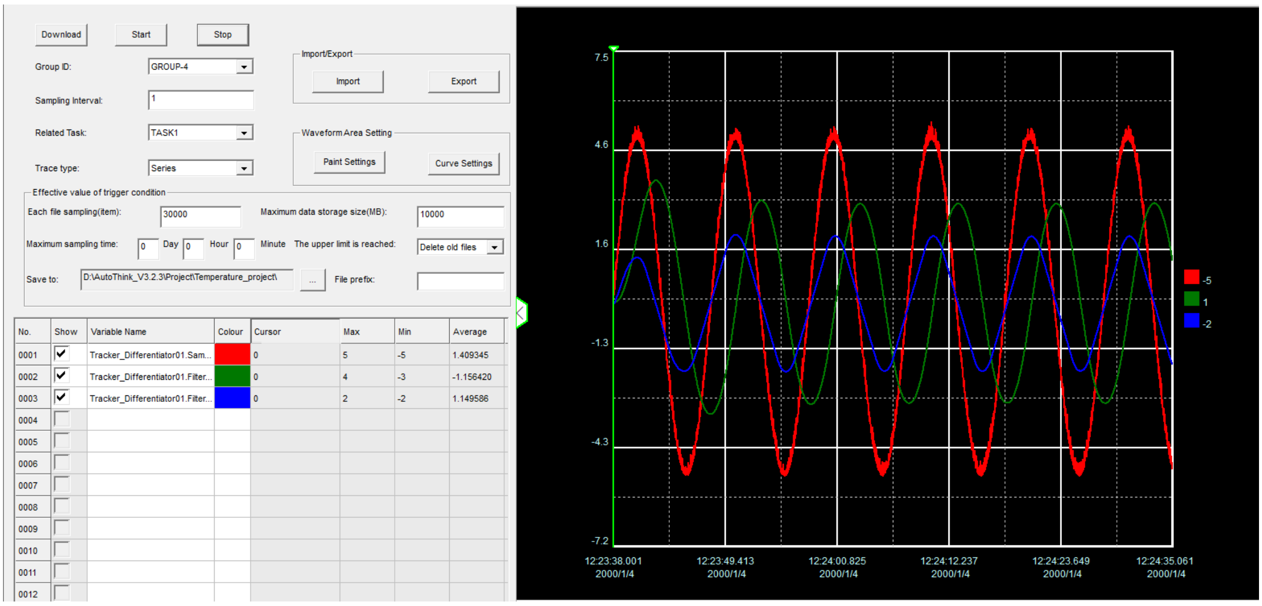
- Program Segment 0002 utilizes the custom function block Gaussian_Write_Noise to simulate Gaussian white noise with a mean square error of 0.1. This noise signal is then added to the sine signal from Program Segment 0001, resulting in the original signal SampledSignal.



- Program Segment 0003 applies the Tracker_Differentiator to filter and reduce noise in the noisy SampledSignal. The task cycle is set to 0.01s, with both TrackRate and FilterCoe configured as 1.



The results of online tracking of SampledSignal, FilterSignal, and FilteredDiffSignal are shown in the figure below. In the figure, the red line represents the original signal, the green line represents the filtered signal, and the blue line represents the filtered differential signal. It can be observed that the Tracker_Differentiator has a significant noise filtering effect on the signal.



7.1.1.6 Precautions

In practice, the signal code values from TC and AI modules require scaling to obtain the actual values (such as temperature and voltage) used as the SampledSignal raw data.

The larger the TrackRate and FilterCoe, the better the noise reduction effect of the filter, but the smaller the amplitude reduction and the greater the phase lag. The smaller the TrackRate and FilterCoe, the worse the noise reduction effect of the filter, but the amplitude is closer to the original signal and the phase lag is smaller. Therefore, users must carefully select TrackRate and FilterCoe to choose parameters suitable for the signal.

This function block needs to be called in a cyclic IEC task.

Chapter 8 Long Dead Time

8.1 Smith_Predictor_TF(Smith Predictor)

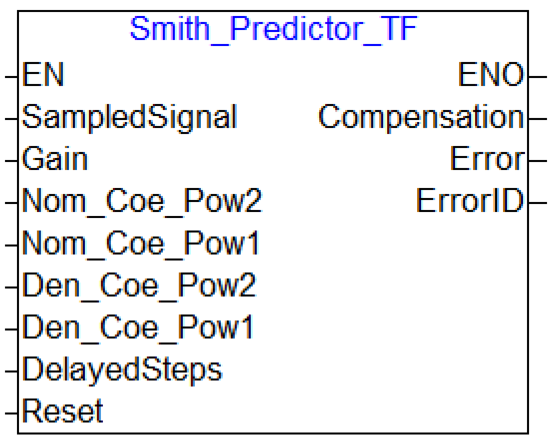
8.1.1 Version

The software version supporting this function block is V3.2.3 and above. Controller firmware versions that do not support this instruction include LX-CU500-A01 (V1.0.0) and LX-CU500-A02 (V1.1.0).

8.1.2 Function

The basic idea of the Smith predictor is to use the mathematical model of the system to predict future outputs, and then perform control in advance based on the prediction results. In this way, even if there is a delay in the actual output feedback, since the control action has already been taken in advance, the system performance will not be significantly affected.

With the aid of the Smith predictor, the negative impact of pure time delay on system control performance is significantly reduced, enabling the system to respond more quickly to input changes and allowing systems with long time delays to be controlled stably.



Smith_Predictor_TF Functional Block

8.1.3 Parameter

Parameter	Statement	Data Type	Function Description	Initial Value

SampledSignal	Input	LREAL	Collected controlled variable signal.	0
Gain	Input	LREAL	Gain of the controlled object's transfer function.	0
Nom_Coe_Pow2	Input	LREAL	Coefficient of order 2 in the numerator of the controlled object's transfer function.	0
Nom_Coe_Pow1	Input	LREAL	Coefficient of order 1 in the numerator of the controlled object's transfer function.	0
Den_Coe_Pow2	Input	LREAL	Coefficient of order 2 in the denominator of the controlled object's transfer function.	0
Den_Coe_Pow1	Input	LREAL	Coefficient of order 1 in the denominator of the controlled object's transfer function.	0
DelayedSteps	Input	DINT	Time delay steps of the controlled object's transfer function (time delay steps = delay time / task period).	0
Reset	Input	BOOL	Reset Configuration: TRUE: Reset FALSE: No Reset	FALSE
Compensation	Output	LREAL	Output of the Smith predictor, which compensates for system time delay.	0
Error	Output	BOOL	If the input parameter is invalid, the parameter will output TRUE.	FALSE
ErrorID	Output	CTRL_ALGO_LIB_ERR_LIST	VALID: No error Others: Refer to the error code	VALID

8.1.4 Error Code

Error ID	Description
SampledSignal_REQUIRED_FINITE	The range requirement for SampledSignal is -1×10^{300} to 1×10^{300}.
ARITHMETIC_OVERFLOW	Numerical calculation overflow in output parameters.
Gain_REQUIRED_FINITE	The range requirement for Gain is -1×10^{300} to 1×10^{300}.
Nom_Coe_Pow1_REQUIRED_FINITE	The range requirement for Nom_Coe_Pow1 is -1×10^{300} to 1×10^{300}.
Nom_Coe_Pow2_REQUIRED_FINITE	The range requirement for Nom_Coe_Pow2 is -1×10^{300} to 1×10^{300}.
Den_Coe_Pow1_REQUIRED_FINITE	The range requirement for Den_Coe_Pow1 is 0 to 1×10^{300}.
Den_Coe_Pow2_REQUIRED_FINITE	The range requirement for Den_Coe_Pow2 is 0 to 1×10^{300}.
DelayedSteps_REQUIRED_LE_2E4	DelayedSteps must be less than or equal to 2×10^{4}.
DelayedSteps_REQUIRED_NON_NEGA	DelayedSteps must be non-negative.
TimeStep_REQUIRED_POSI	TimeStep must be positive.
TimeConstant_REQUIRED_FINITE	TimeConstant must be less than 30000.
NatureFreqency_REQUIRED_FINITE	NaturalFrequency must be less than 1000.
DampingRatio_REQUIRED_FINITE	DampingRatio must be less than 1000.
TASK_IS_SINGLE	Task is Single execution.
TASK_IS_FULLSPEED	Task is FullSpeed execution.

8.1.5 Example

This example is only intended to further illustrate the use of Smith_Predictor_TF, not for actual application scenarios.

Usage steps:

(1) Before implementing control using a Smith predictor, it is necessary to first identify the system using the identification function to obtain its transfer function model.

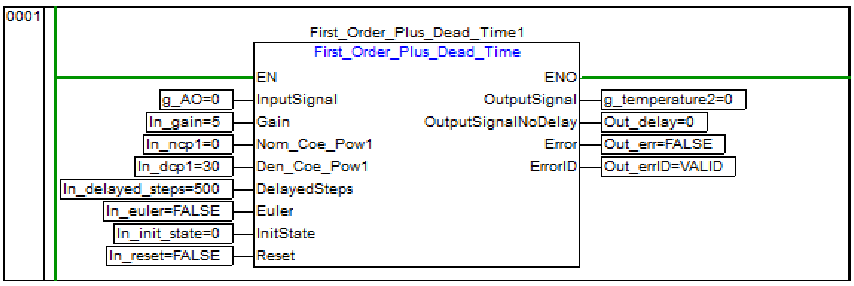
(2) Use the Hysteresis_PID_Tuner to adjust the parameters of the transfer function model without time delay to obtain the PID parameters.

(3) Control the plant using the Smith_Predictor_TF and PID_Controller.

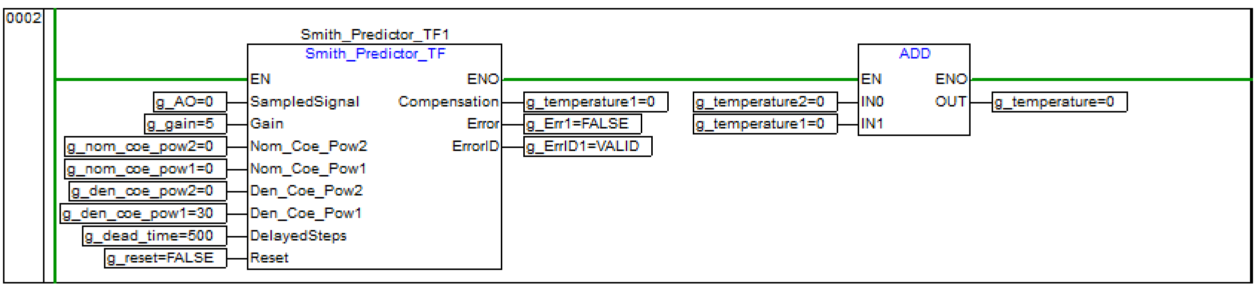
This example uses a heating device as the plant, which has a time delay of 100 seconds, 3.3 times the time constant, making it a typical long time-delay system. This example only describes the control using the Smith predictor.

The system configuration is shown in the figure below. For the inputs to the Smith predictor, SampledSignal is the sampled signal, Reset is the reset, and the remaining parameters are all transfer function model parameters of the heating device.

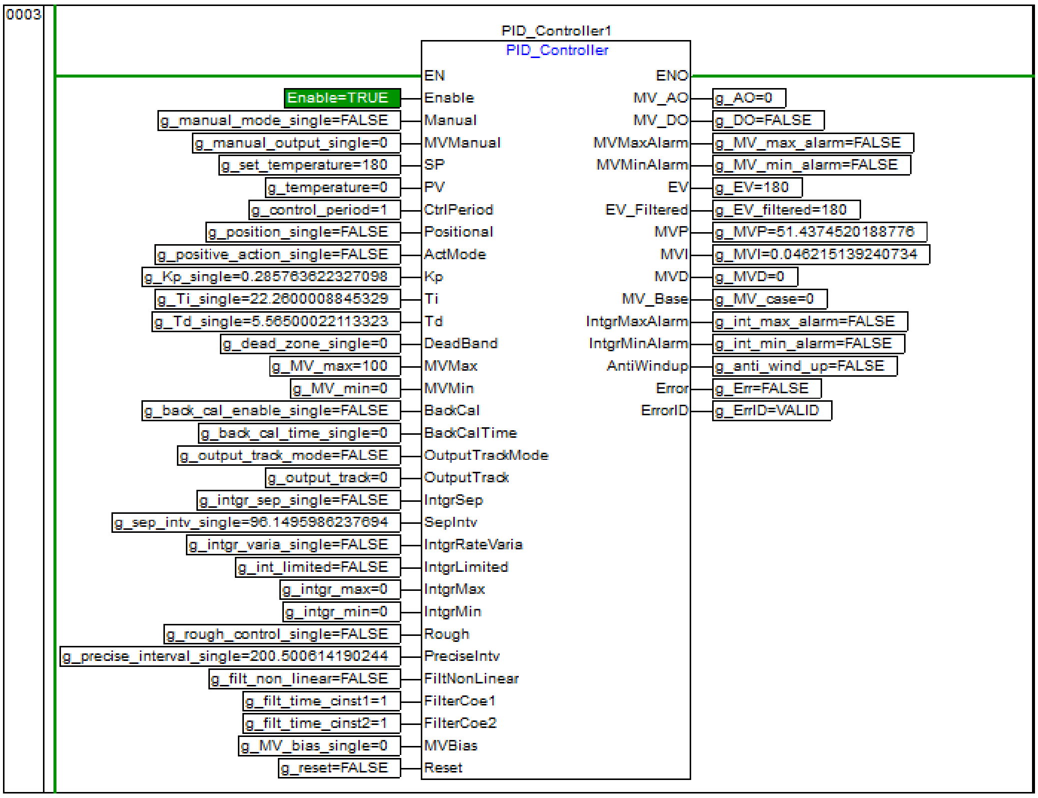
- Program Segment 0001 simulates the input and output of the system through the function block First_Order_Plus_Dead_Time. In this example, it corresponds to the transfer function of the controlled object's output, with a Gain of 5, Den_Coe_Pow1 of 30, and DelayedSteps of 500.



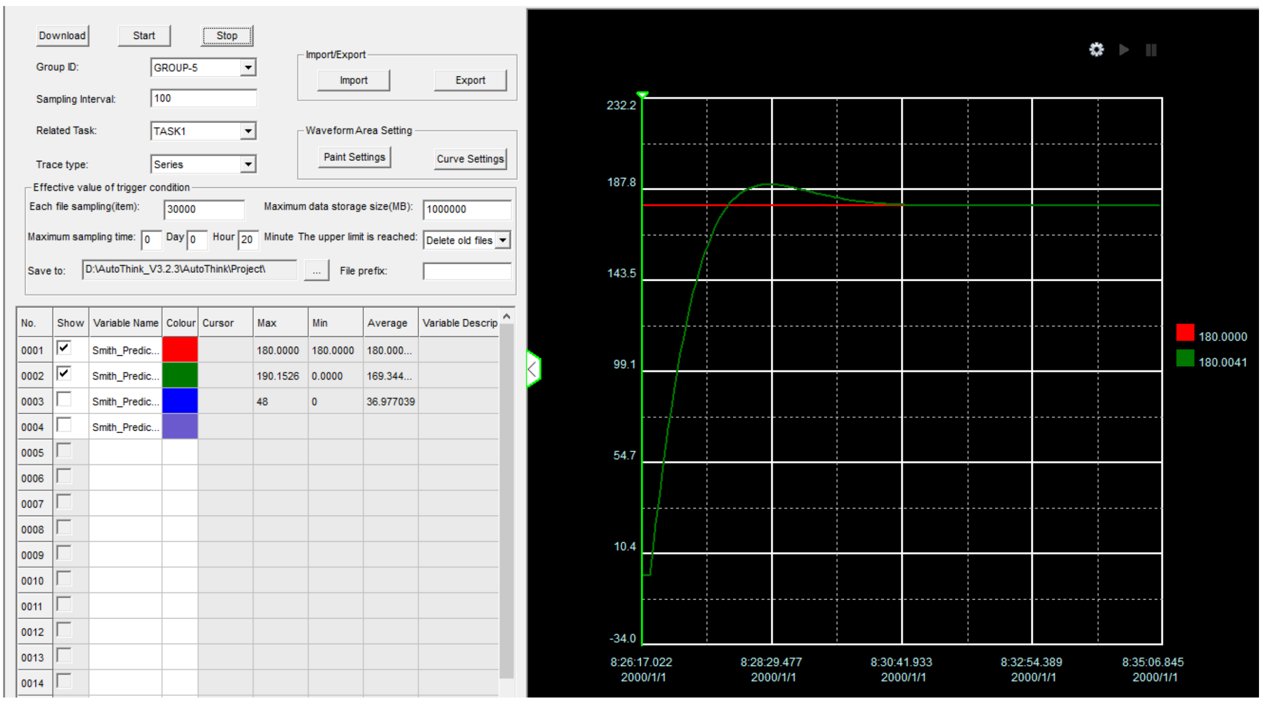
- In Program Segment 0002, Smith_Predictor_TF is used to perform dead-time compensation for the system, and the compensated signal is added to the OutputSignal in Program Segment 0001 to obtain the process variable (PV).



- Program Segment 0003 calls the PID to control the controlled object and stabilizes the process value around the setpoint of 180.



The final control performance is illustrated in the figure below.



For general field applications, it is recommended to incorporate auxiliary temperature control measures such as setpoint ramp tracking.

The code logic is as follows:

```
IF set_temperature < target_ temperature THEN
```

```
  set_temperature := set_temperature + set_ramp * time_step;
ELSE
   set_temperature := target_ temperature;
END_IF
```

Here, target_temperature refers to the final target temperature, set_temperature is the set temperature at that moment, set_ramp is the set ramp rate, and time_step is the time step, which is the task cycle. The overall logic of the code is that when the final target temperature has not been reached, the temperature changes according to the set ramp rate. Once the final target temperature is reached, the ramp tracking stops, and the set temperature becomes the final target temperature.

8.1.6 Precautions

This function block needs to be called in a cyclic IEC task.

Beijing HollySys Intelligent Technologies Co., Ltd..
Di Sheng Middle Road, No.2
Economic-Technological Development Area
100176 Beijing, China
Tel: 010-5898 1588
Hotline: 400-811-1999
Fax: 010-5898 1558
Web: http://www.hollysys.com
FA Web: https://fa.hollysys.com/