UNIVERSAL MECHANISM 9

universal mechanism

User`s manual

UM Control / Block Editor Tool

2020

Contents

24. Functional Scheme Editor

24.1. Introduction

Block Editor tool from **UM Control** module is a standalone application designed to describe the structural schemes using basic functional elements. In fact, **Block Editor** is an analog of the **Simulink** tools from **Matlab / Simulink** software package. Both schemes developed in **Block Editor**, and in the **Matlab / Simulink** software are connected to Universal Mechanism exactly in the same way. In this sense, the **Block Editor** and **Matlab Import** are functional analogs with very similar methods of description of block diagrams and connection to the dynamic model in Universal Mechanism software.

Finished functional scheme includes some outside inputs and outputs. A structural scheme is fed with the input signals that are somehow converted into output signals that are sent to the UM model. A simple scheme with the summator with two inputs and one output is shown in Figure 24.1.

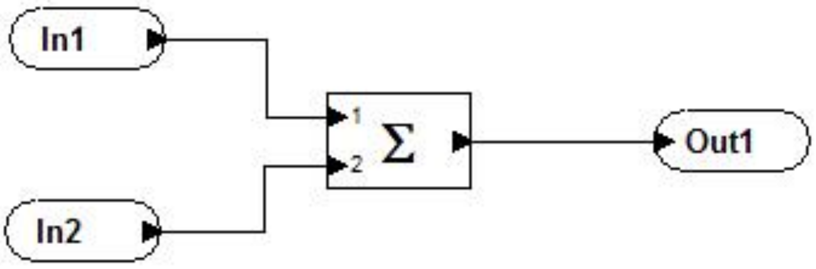


Figure 24.1. An example of simple functional scheme

Structural schemes, developed with the help of **Block Editor**, are connected with UM models with the help of **Wizard of external libraries** in **UM Simulation** program (see menu command **Tools/ Wizard of external libraries**).

This chapter of UM User's Manual is devoted to structural elements of the **Block Editor**. Principles, workflow and features of the interconnection of structural schemes and UM models are considered in the Chapter 5 of UM User's Manual "Programming in UM Environment", see Sect. 24.3. Creating and using external libraries; as well as you can find some information in the chapter "Getting started: UM Control".

Limitations

The current version of Block Editor supposes that the only *.be structural scheme should be connected to Universal Mechanism model. If you need to connect more than one scheme you should preliminarily incorporate all schemes into one *.be files manually or with the help of **Macroblock** element from the **Macros** tab sheet.

24.2. Sample

Sample model of the inverted pendulum with the stabilizing control system developed with the help of **UM Control / Block Editor** is placed here {UM Data}\SAMPLES\ TUTORIAL\blockeditor\inv_pend_blockeditor. Control system is placed in **inv_pend.be** file in the same folder.

To have a look how it works generally run **UM Simulation** program. Load the model from the folder {UM Data}\SAMPLES\TUTORIAL\blockeditor\inv_pend_blockeditor. General view of **UM Simulation** with the loaded model is shown in Figure 24.2.

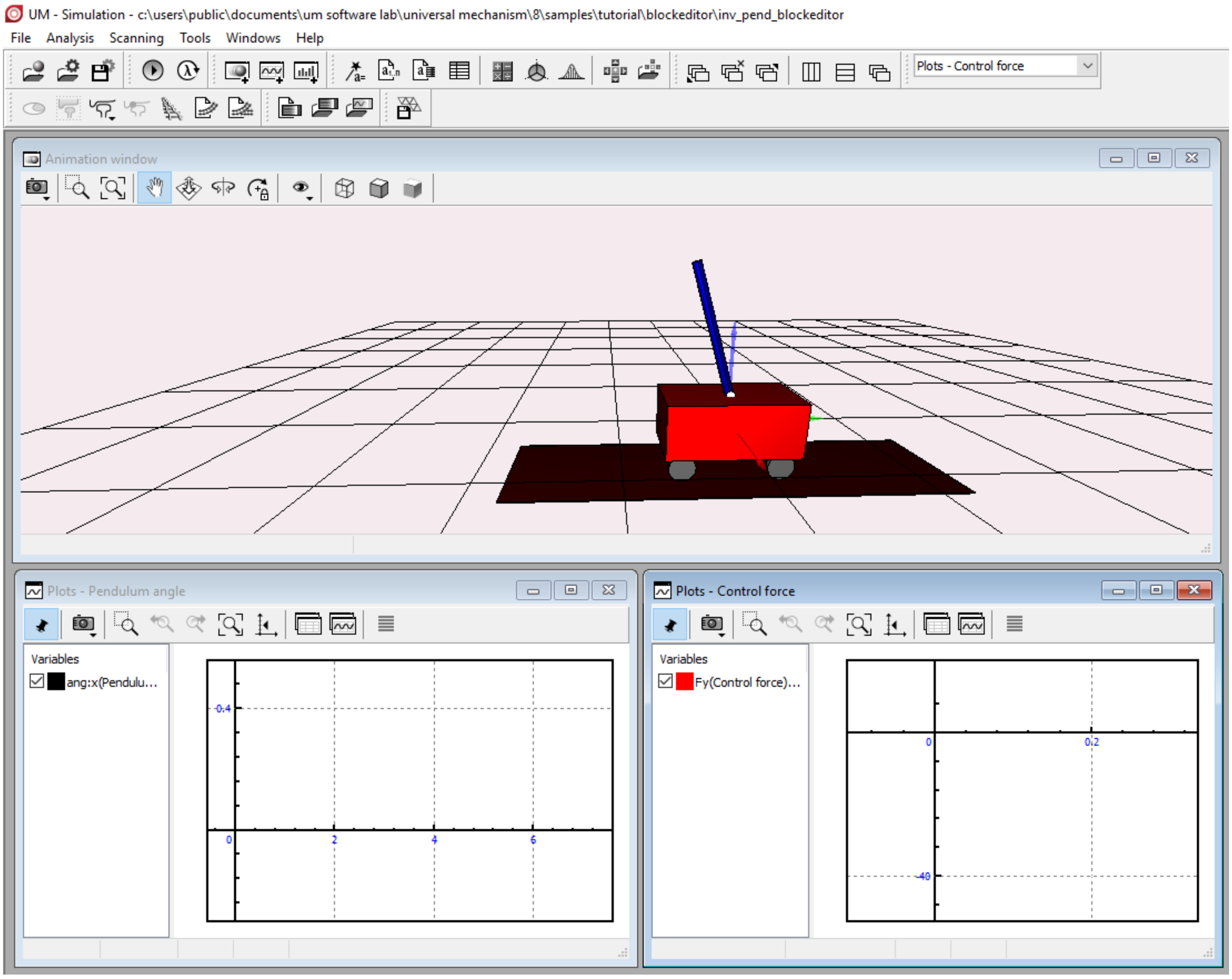
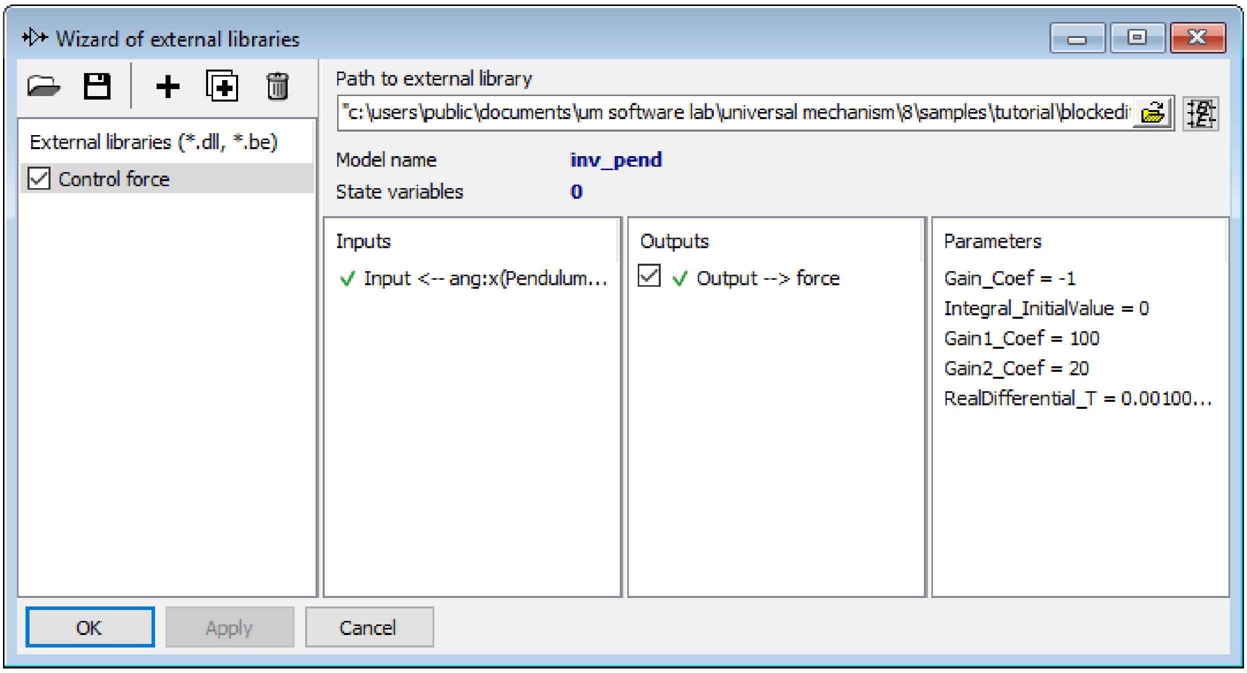

UM - Simulation - c:\users\public\documents\um software lab\universal mechanism\8\samples\tutorial\blockeditor\inv_pend_blockeditor
File Analysis Scanning Tools Windows Help
Plots - Control force
Animation window
Plots - Pendulum angle
Variables
ang:x(Pendulu...
0.4
0
2
4
6
Plots - Control force
Variables
Fy(Control force)...
0
0.2
-40

Figure 24.2. **UM Simulation**: general view

Click the menu command **Tools / External library interface**. The window, which is shown in Figure 24.3, appears. If the external library is the library prepared with the help of the **Block Editor**, button becomes available. Click this button to run **Block Editor** and automatically open the loaded control system.



External libraries (*.dll, *.be)

☑ Control force

Path to external library

"c:\users\public\documents\um software lab\universal mechanism\8\samples\tutorial\blockedi

Model name inv_pend

State variables 0

Inputs

✓ Input <-- ang:x(Pendulum...

Outputs

☑ ✓ Output --> force

Parameters

Gain_Coef = -1

Integral_InitialValue = 0

Gain1_Coef = 100

Gain2_Coef = 20

RealDifferential_T = 0.00100...

Figure 24.3. Wizard of external libraries

24.3. Functional Scheme Editor

The functional scheme editor is run by the executable file **BlockEditor.exe**. Ready functional schemes are saved in files with the **be** extension.

The general view of the editor window is below in Figure 24.4.

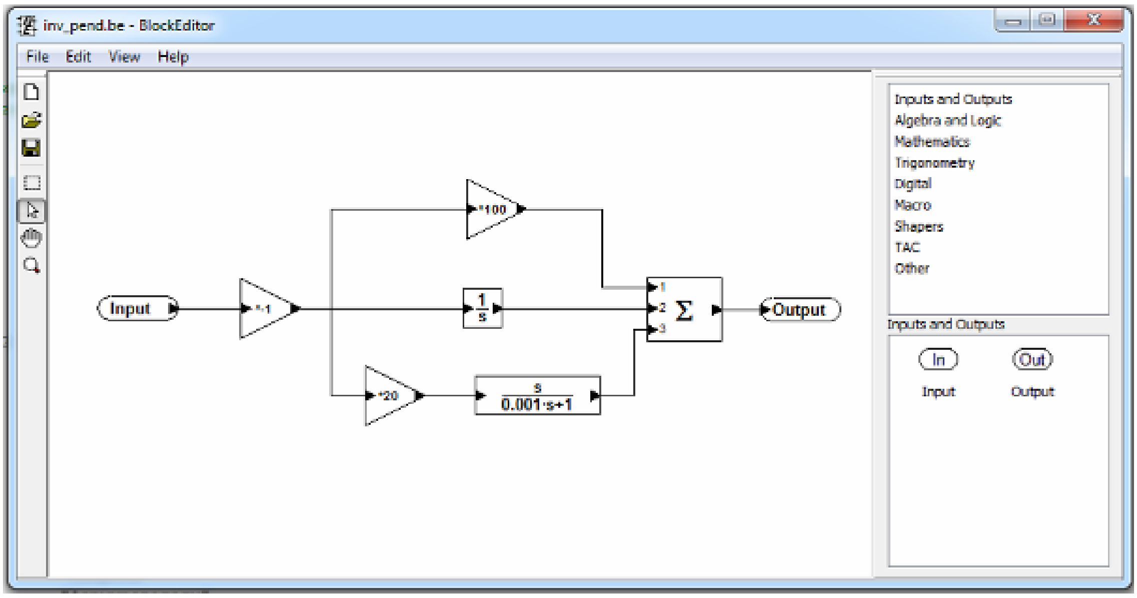


Figure 24.4. General Editor View

The program window is divided on some areas. At the top of the window there is main menu. Tools panel is on the left and in the center is the working area of the program. On the right there is functional blocks library, divided on the list of categories (top part) and the list of blocks (bottom part).

24.3.1. Main Menu

The main menu gives access to the main program operations such as open and save schemes, exit from the program and so on. It has menu items File, Edit, View and Help.

Menu item **File** includes submenu *New*, *Open*, *Save*, *Save as* and *Exit*.

The command **New** is to create new functional scheme. If the command is executed during the work with scheme, then the query to save the current scheme is generated, after which the working area is cleared.

The command **Open** is to load the file with functional scheme. When it is performed the standard open file dialog is displayed.

The command **Save** is to save in file previously opened or just created functional scheme. With this, if scheme was open from some file, it will be saved in the same file. But if it’s a new scheme then the command *Save as* is executed.

The command Save as is to save functional scheme in the selected file. When it is performed the standard open file dialog is displayed.

The command **Exit** is to stop the working with the program. If before the Exit functional scheme was changed, then the query to save the current scheme is displayed.

Menu item **Edit** includes the commands: *Copy, Cut, Paste, Delete, Select All*.

The command **Copy** is to copy one or more blocks into Clipboard. The command is available only if some blocks are selected in the editor.

The command **Cut** is to copy one or more blocks into Clipboard with simultaneous removal original ones. The command is available only if some blocks are selected in the editor.

The command **Paste** is to add some blocks from Clipboard to scheme. The command is available only if there some blocks in the *Clipboard*.

The command **Delete** is to delete one or more selected blocks from the scheme.

The command **Select All** is to select all the blocks in the current scheme.

Menu item **View** includes the commands: Scale *100%*, Scale *200%*, Scale *400%*, Fit to working area.

The command **Scale 100%** is to display the scheme in its original size.

The command **Fit to working area** is to display the whole scheme (if no one block is selected) or the group of selected blocks in the program working area.

Menu item **Help** includes the unique command – **About the program**, which opens dialog window with the information about the version of the program and its authors. To close the dialog window one should put the button «Ok» or «Close», and also one can push the keyboard button Enter or Esc.

24.3.2. Context Menu

Context menu is opened on mouse right button click in the editor working area (Figure 24.5). That menu is used to get quick access to commands of the main menu item «Edit»: *Copy, Cut, Paste, Delete*. It also has the point «Properties», which opens properties window for block being clicked by mouse. Menu items are active only if it is ran for one or more selected blocks.

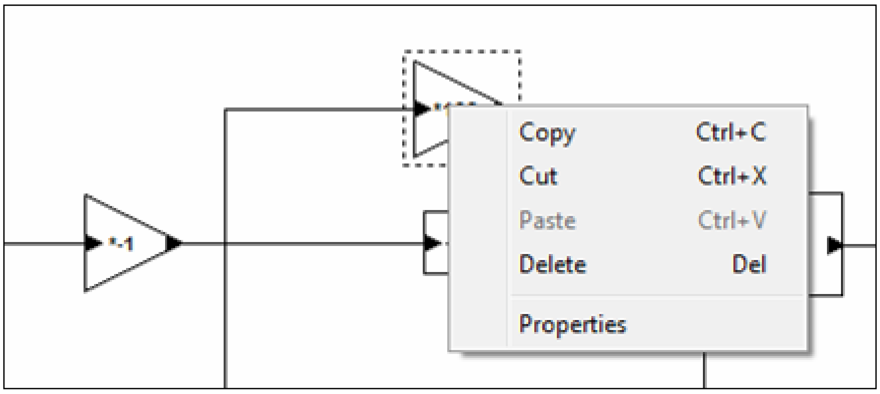


Figure 24.5. Block Context Menu

24.3.3. Toolbar

The toolbar includes buttons for quick access to frequently used commands. It has the following commands: *New, Open, Save, Group of Blocks Selection, Selection Mode, Navigation*

Mode, Scale. The commands *New, Open* and *Save* duplicate the similar commands from menu File.

The command **Group of Blocks Selection** is to select one or more blocks in the working area. If to push this button, the editor switches to the mode of working with block groups and the cursor takes the form of cross. In that mode it's possible to pick out and move the groups of blocks. To pick out the block group it's necessary to place mouse cursor in free space of working area, push the left mouse button and without releasing it to cover the group of desired blocks with dash frame. To move the group it's necessary to pick out by mouse left button any selected block and without releasing it drag the group to the new working area place.

Selection Mode is the main working mode with the Block editor. In this mode the cursor has the form of ordinary arrow. In such a mode one can select one scheme block or one connected line between blocks, drag the selected block to the new place of working area, connect the inputs of some blocks with the outputs of another ones, edit segments of the connecting lines between blocks and open the window of parameters for any block to change them further.

To select a single block one should make single click on it with the mouse.

To drag a single block one should click on it with the mouse and without releasing it drag it on the new place of the working area,

To select the connecting line one should make single click on it with the mouse.

To edit segments of the connecting line one should click on the desired segment with the mouse and without releasing it drag it to a new place.

To open the dialog window of block parameters one should make twice click on that block with the mouse. Then to save new parameters one should press the button «Ok» in the dialog window or the button Enter on the keyboard.

Navigation Mode is to move the whole working area. In this mode the cursor has the form of a hand. To move the working area one should make a single click in any place of it with the left button of the mouse and without releasing it move the area in the desired direction.

The command **Scale** is to pass to the scale mode. At this case the cursor has a form of lens. For an arbitrary change of scale one should click with the left button of the mouse in any place of working area and without releasing it move the pointer to the left (decrease) or to the right (increase).

24.3.4. Block Library

The block library is subdivided on two lists by the movable horizontal line. In the top list are block categories and in the bottom one the block icons from the selected category are displayed. The category list includes eight sections: Inputs and Outputs, Algebra and Logic, Mathematics, Trigonometry, Digital, Macros, Shapers, TAC.

To display the blocks of desired category one should click on that category with the mouse in the top list – in the bottom list all the blocks of this category are displayed immediately. Then one can click with the left button of the mouse on the desired block and without releasing it drag the block to the editor working area.

24.3.5. Working Area

The main work with the functional scheme is carried in the working area. There the blocks and the connections between them are created. To create the block one should select that block in the block library and drag it to the working area. To connect blocks one should click with the left button of the mouse on the output of one block and, without releasing the button, move the cursor to the input of other block.

24.4. Development of the Control System Functional Scheme

Control System Functional Scheme defines the algorithm of control panel work. Functional Scheme consists of blocks, each of which makes his own function in processing its input signals and forming output signals. The functional scheme developer places blocks from library on working area and connects the inputs and outputs of these blocks.

So some blocks of functional scheme produce initial signals, another ones transform them (add, multiply and so on) and some blocks are targets of that information – these are blocks, connected with executive mechanisms of the construction. Signals on their outputs form voltage on the engines, turn on or off lights and so on.

All the blocks of the functional scheme work in pseudo parallel, i.e. although the blocks computation is made consequentially, the designer may think that signals are processed by all the blocks simultaneously.

It’s recommended while use the editor to make attention on pop-up tips. The tips are appeared with blocks on the block library palette, and also with blocks on the working area. There is the tip for the whole block as well as the tip for each its input and output.

There are two kinds of signals, transferred from block to block in functional scheme: analog signal and logic (digit) signal.

Analog signal – is a signal, which changes in range from -∞ до +∞. The value of that signal characterizes some physical parameter, e.g. velocity, rotation angle and so on.

Digital signal – is a signal, which takes only one of two values: the value Boolean zero «0», or the value Boolean unit «1».

If analog signal is set to the digital input, and if the value of that signal is more than 0.5, then the block detects that signal as logical «1». If it is less or equal 0.5, then the block detects it as logical «0».

During the time digital signal can be changed, i.e. to transfer from logical «0» to logical «1» and back. Such change is named impulse.

The transition from the state «0» to the state «1» is named front edge of the impulse. The transition from the state «1» to the state «0» is named back edge of the impulse.

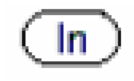
When are said that «event occurs on the front edge of the impulse», that means that the event occurs only when the input signal transfers from a state «0» to a state «1».

24.5. Block Description for Functional Scheme Editor

24.5.1. Inputs and Outputs

Input

The block, connected with outer input signal of the functional scheme.



Inputs:
None

Outputs:
Outer Input signal of the scheme

Output

The block, connected with outer output signal of the functional scheme.

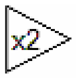
Inputs:
Output signal of the scheme

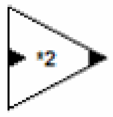
Outputs:
None

24.5.2. Algebra and Logic

Gain

The block multiplies the input value by specified constant value (gain) and sends the result to the output.





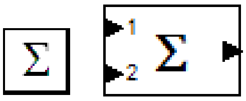
Inputs:
The original signal

Outputs:
Original signal, multiplied by constant K, specified as parameter.

Parameters:
Constant K

Summator

The block performs addition on its inputs, so the output is the resulting sum.



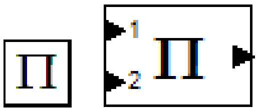
Inputs:
Input 1 ... input n.

Outputs:
The sum of the input signals.

Parameters:
The number of inputs.

Multiplier

The block multiplies two or more input signals, so the output is the resulting multiplication.



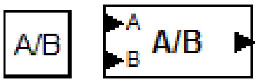
Inputs:
Input 1: ... input n.

Outputs:
The product of the input signals.

Parameters:
The number of inputs.

Analog Divider

It divides signal A by signal B. The result is an output. If signal B equals zero, then output is zero too.



Inputs:
A is dividend.
B is divisor.

Outputs:

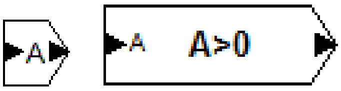
The result of the division.

Parameters:

None

Comparison with the Constant

The block compares Input signal and the constant, defined as a block parameter. Comparison operation (more, less, equal etc.) is specified as a block parameter. If the result of comparison is true, then the output is 1, otherwise it is 0.



Inputs:

Compared signal.

Outputs:

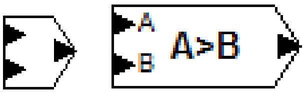
Comparison result (0 or 1).

Parameters:

1. The constant with which the signal is compared
2. Comparison operation: <, =, >, ≤, ≥

Two Signals Comparison

The block compares two its input signals. The operation for comparing (greater, less, equal etc.) is specified as a block parameter. If the comparison result is true, then the output is 1, otherwise the output is 0.



Inputs:

Signal A

Signal B

Outputs:

Comparison result (0 или 1).

Parameters:

The operation for comparing: <, =, >, ≤, ≥

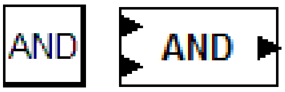
Logical Element «AND»

The block computes logical function «AND» on its input signals. Output value will be «1» only when all the input signals are «1», otherwise output value equals «0».

The number of block inputs is specified as a parameter.

The block works with logical (digital) signals. Such signals have only two states: the state of Boolean TRUE («1»), or the state of Boolean FALSE («0»).

The input signal is assumed to be «1», if its value exceeds 0.5.



Inputs:

Input 1 ... input *n*.

Outputs:

Conjunction of the input signals

Parameters:

The number of inputs.

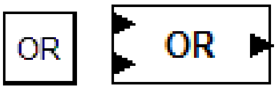
Logical Element «OR»

The block computes logical function «OR» on its input signals. Output value will be «1» only when at least one input signal is «1». If all the inputs are «0», the output value equals «0» too.

The number of block inputs is specified as a parameter.

The block works with logical (digital) signals. Such signals have only two states: the state of Boolean TRUE («1»), or the state of Boolean FALSE («0»).

The input signal is assumed to be «1», if its value exceeds 0.5.



Inputs:

Input 1 ... input *n*.

Outputs:

Disjunction of the input signals

Parameters:

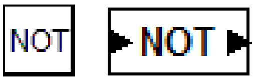
The number of inputs.

Logical Element «NOT»

The block computes logical function «NOT» on its input signal. Output value will be «1» if the input signal is «0». Otherwise, if the input is «1», then the output value will be «0».

The block works with logical (digital) signals. Such signals have only two states: the state of Boolean TRUE («1»), or the state of Boolean FALSE («0»).

The input signal is assumed to be «1», if its value exceeds 0.5.



Inputs:

A

Outputs:

NOT(A) (0 / 1)

Parameters:

None

Sum with sign

Summing up the sign ("+" or "-") two or more signals applied to the input of this block. The output generates the result of the summation.

Inputs:

Input 1: ... input n.

Outputs:

Sum of input signals.

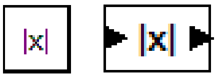
Parameters:

The sequence of characters for the inputs "+" or "-".

24.5.3. Mathematics

Absolute Value

The block outputs the absolute value of its input.



Inputs:

x

Outputs:

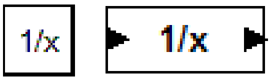
Abs(x)

Parameters:

None

Function 1/x Calculation

The block calculates inverse input signal and sets the result on the output. If input signal is zero, then the output is 0 too.



Inputs:

x

Outputs:

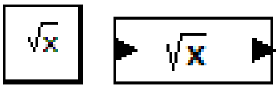
1/x

Parameters:

None

Square Root

The block computes square root of the input, If Input signal is negative then the output is zero.



Inputs:

X

Outputs:

Square root of x.

Parameters:

None

Sign of the Input

The block returns the sign of the input. If input signal is positive then output is 1, if it’s negative the output is –1, if it is zero the output is zero too.

Inputs:

X

Outputs:

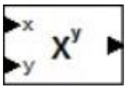
Sign(x)   (-1,0,1)

Parameters:

None

Exponentiation

Raises signal x to the power of signal y. The output generates the result of exponentiation. If the signal x is less than zero, then this signal is taken modulo.



Inputs:

Signal x – base.

Signal y – indicator.

Outputs:

Exponentiation result |x|^y.

Parameters:

None

Min/Max signal

Depending on the type of function selected, the block computes the maximum, minimum, or average value of the input signals. Sets the result on the output.

Inputs:

Input 1: ... input n.

Outputs:

Result of function computation.

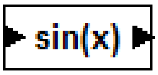
Parameters:

1. The number of inputs.
2. Type of function: Min, Max, Avg

24.5.4. Trigonometry

Sine

The block computes the sine of the input.



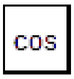
Inputs:
x (in radians)

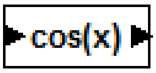
Outputs:
sin(x)

Parameters:
None

Cosine

The block computes the cosine of the input.





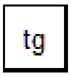
Inputs:
x (in radians)

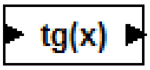
Outputs:
cos(x)

Parameters:
None

Tangent

The block computes the tangent of the input.





Inputs:
x (in radians)

Outputs:
tg(x)

Parameters:
None

Cotangent

The block computes the cotangent of the input.

Inputs:
x (in radians)

Outputs:
ctg(x)

Parameters:
None

Arc Sine

The block computes the arc sine of the input.

Inputs:
X

Outputs:
arcsin(x) (in radians)

Parameters:
None

Arc Cosine

The block computes the arc cosine of the input.

Inputs:
x,

Outputs:
arccos(x) (in radians)

Parameters:
None

Arc Tangent

The block computes the arc tangent of the input.

arctg
arctg(x)

Inputs:
x

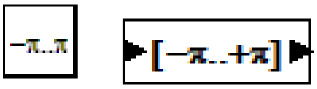
Outputs:
arctan(x) (in radians)

Parameters:
None

Reduction Input Angle to the Range from –π to +π

Input signal of this block is an angle, given in radians. It can have any values, including values which exceed π. It means that input angle includes some rotations.

The block reduces the input angle to the range of one period, i.e. to the range from –π до +π.



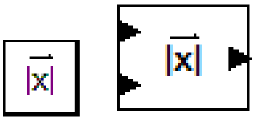
Inputs:
Origin signal

Outputs:
The signal, reduced to the range from –π to +π

Parameters:
None

Vector Length

Block inputs are the coordinates of n-dimensional vector. The block calculates vector length, i.e. square root from the sum of squares of inputs. The number of inputs is specified as a block parameter.



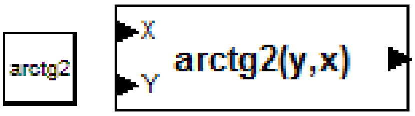
Inputs:
Input 1…input n
Outputs:
Vector length

Parameters:

Number of inputs

Arctangent Given Quarter

The block calculates polar arctangent (function arctan2(y, x)), i.e. arctangent of ratio y/x given quarter. Block input is a vector (x, y), the output is the angle (from –π to +π) between axis OX and radius vector (X, Y).



Inputs:
X is adjacent side.
Y is opposite side.

Outputs:
The angle between axis OX and radius vector (X, Y), in radians.

Parameters:
None

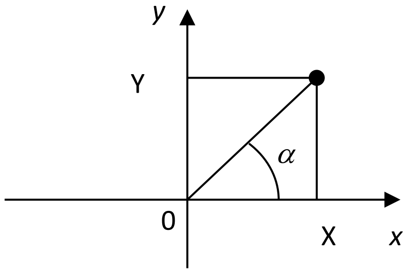
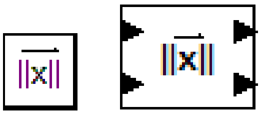


Illustration of the function arctg2

Vector Normalization (Reduction to Unit Length)

The block normalizes n-dimensional vector, i.e. reduces it to unit length. Block inputs are the vector coordinates, the outputs are coordinates of normalized vector.



Inputs:
X_1
X_2
…
X_n

Outputs:

X_1,norm

$X_{2,}$norm

…

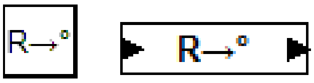
X_n,norm

Parameters:

None

Converting Radians to Degrees

The block converts input signal in radians to the output signal in degrees.



Inputs:

Radians.

Outputs:

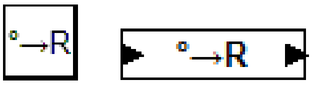
Degrees.

Parameters:

None

Converting Degrees to Radians

The block converts input signal in degrees to the output signal in radians.



Inputs:

Degrees.

Outputs:

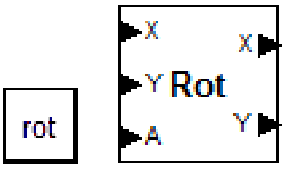
Radians.

Parameters:

None

Rotation of the Coordinate System at Angle A

The block input is the vector (X, Y) and the angle A of rotation (in radians). Block rotates that vector clockwise on the angle A and produces coordinates of rotated vector as block outputs.



Inputs:

X

Y

Angle of rotation

Outputs:

X after the rotation (X’).

Y after the rotation (Y’).

Parameters:

None

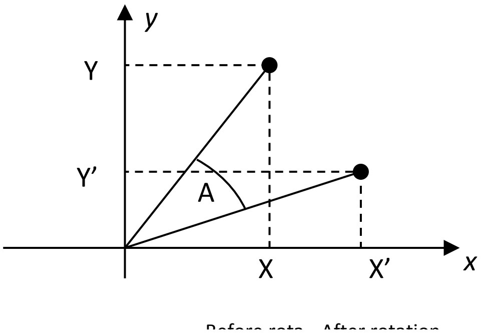
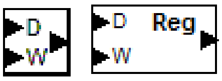


Illustration of the function Rot

24.5.5. Digital

Buffer Register

The block has two inputs: analog data input D and digital input W of record permission. If input W will be logical «1», then the block produces on its output the same analog signal, as on the input D. This value is saved in buffer register. If input W will be logical «0», then the block produces on its output the signal, saved in register.



Inputs:

D is data input.

W permits the recording permission. W equal to 1 permits to save signal D.

Outputs:

The last saved in the register value

Parameters:

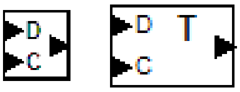
None

D-Trigger (trigger-latch)

D-trigger is a device with two states («0» or «1»). The trigger state is produced on its output.

D-trigger has two digital inputs: data input D and sync input C.

The value on input D doesn’t impact on trigger output until there is impulse on input C. On the front edge of the input C impulse (i.e. at the moment of signal C transition from state «0» to state «1»), trigger saves input D. Since that moment the trigger produces on its output this saved value until input C receives new impulse front edge.



Inputs:

D is the input signal.

C is sync input: on the front edge of that signal trigger “latches” data from input D and produces them on the output.

Outputs:

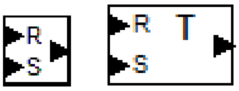
«Latched» value.

Parameters:

None

RS-Trigger

RS-trigger is a device with two states («0» or «1»). The trigger state is produced on its output. D-trigger has two digital inputs: R (Reset) and S (Set). When impulse of logical «1» is at input S the trigger is switched to state «1». When impulse of logical «1» is at input R the trigger is switched to state «0». In case both inputs will have logic «1», the priority has the signal R.



Inputs:
R sets output to «0» on logical «1».
S sets output to «1» on logical «1».

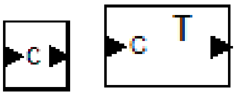
Outputs:
Trigger state.

Parameters:
None

Trigger with Countable Input

Trigger with countable input – is a device with two states («0» or «1»). The trigger state is produced on its output.

The trigger has unique digital input C. On the front edge of the impulse at input C trigger switches into opposite state. It means that on front edge of first impulse at the input C the trigger moves from state «0» to state «1». On front edge of second impulse it moves from state «1» to state «0». On front edge of third impulse it moves from state «0» to state «1» and so on.



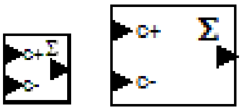
Inputs:
Countable input. Changes trigger state on front edge of the impulse.

Outputs:
Trigger state.

Parameters:
None

Impulse Counter

Counter is a device, which counts the number of entered impulses. The output is the number of counted impulses. Initially that value is 0. The counter has two inputs: C+ и C-. On the front edge of the impulse at input C+ the value of counter output increases by 1. On the front edge of the impulse at input C- the value of counter output decreases by 1.



Inputs:
Signal 1 increases the sum on front edge of each impulse.
Signal 2 decreases the sum on front edge of each impulse.

Outputs:
The sum.

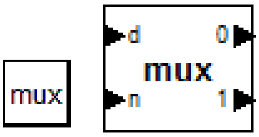
Parameters:
None

Multiplexer

Analog multiplexer. It transfers the input analog signal to the one of block outputs. Other outputs have value 0.

The output number, on which the signal will be set, is selected depending on value n of the input signal. If n = 0, zero output is selected, if n = 1, first output is selected, if n = 2 the second one and so on.

The number n is rounded to the nearest integer and is reduced to the range from 0 to m-1 (where m is the number of outputs).



Inputs:
Data.
n is the output number, on which the signal will be set.

Outputs:
Output 0 ... output m-1.

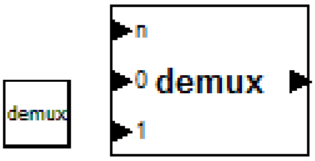
Parameters:
The number of outputs.

Demultiplexer

Analog demultiplexer. It transfers the input analog signal from one of the block inputs to the block output. The input number, from which the data will be transfer, is selected depending on the input signal n. If n = 0, zero input is selected, if n = 1, first input is selected, if n = 2 the second one and so on.

The number of inputs is a block parameter.

The number n is rounded to the nearest integer and is reduced to the range from 0 to m-1 (where m is the number of analog inputs).



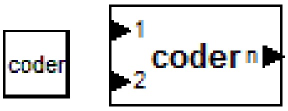
Inputs:
n is the input number, from which the data will be taken.
0..m-1 are data input 0 ... data input m-1

Outputs:
Signal value from the input, pointed by n.

Parameters:
The number of inputs.

Priority Encoder

It outputs the number of the first input where the signal is logic unit. If the signal of logic unit is on several inputs, the priority has the input with least number. If all the inputs are logical zero, then the output is zero too.



Inputs:
Input 1 ... input n

Outputs:
Signal number.

Parameters:
The number of inputs.

24.5.6. Macros

Macroblock

This block makes it possible to create nested schemes. To do this one should set that block, enter to its properties and push the button «open content». So the editor window of inner block scheme will be open.

M
Macroblock

Inputs:

They are defined by the number and kind of block «macroinput from main scheme» in the inner block scheme.

Outputs:

They are defined by the number and kind of block «macrooutput to main scheme» in the inner block scheme.

Parameters:

1. The name
2. Inner scheme editor.
3. Inputs and outputs sequence editor.

Macroinput from the Main Scheme

The block creates additional input to macroblock. That block can't be set on main system scheme and is used only inside macroblocks. While it is set, macroblock of upper level scheme gets one more input. In the properties of block «Macroinput from main scheme» one can define the input name and prompt, which will be pop up for that input.

Outputs:

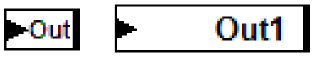
Input from main scheme.

Parameters:

1. Input name.
2. Prompt text.

Macrooutput to the Main Scheme

The block creates additional output from macroblock. That block can't be set on main system scheme and is used only inside macroblocks. While it is set, macroblock of upper level scheme gets one more output. In the properties of block «Macrooutput to main scheme» one can define the output name and prompt, which will be pop up for that output.



Inputs:
Output to the main scheme.

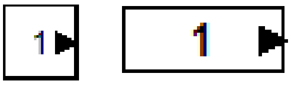
Parameters:

1. Output name.
2. Prompt text.

24.5.7. Shapers

Constant Signal Generator

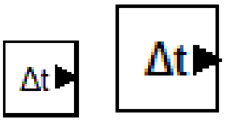
The block outputs the value, given as parameter



Outputs:
Signal value

Parameters:
Signal value

Returns the Value of Calculation Tact in Seconds

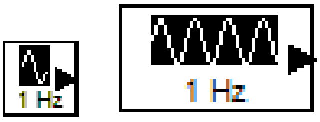


Outputs:
The value of calculation tact in sec.

Parameters:
None

Sine Generator

It outputs sinusoid with unit amplitude and given frequency.

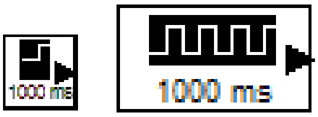


Outputs:
Sinusoidal signal of given frequency.

Parameters:
Signal frequency (Hz)

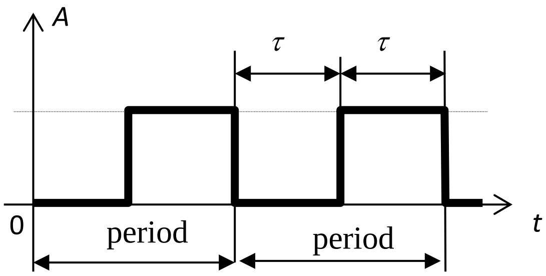
Meander Generator

Meander generator (see the Picture)



Outputs:
Meander

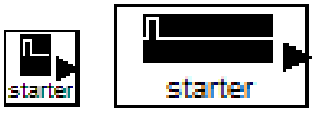
Parameters:
Period (ms)



Meander time diagram

Starter

The block forms single unit impulse at the beginning of modeling.

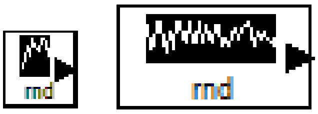


Outputs:
Single unit impulse

Parameters:
None

Random Signal Generator

Outputs random signal with given amplitude. Output signal is quantized in time with a given period.



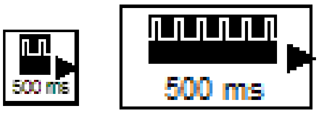
Outputs:
Random signal.

Parameters:

1. Amplitude.
2. Period of changing the input signal.

Short Impulses Generator

The block outputs impulses with one tact duration and given repetition period.



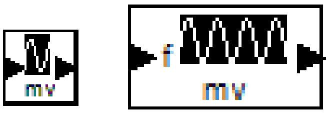
Outputs:
Impulses with one tact duration and given repetition period.

Parameters:
Impulse repetition period (ms)

Controlled Multivibrator

Multivibrator is a device, which generates sinusoid on the output. Controlled multivibrator is the multivibrator, in which oscillation frequency can be regulate by input signal.

It's necessary to form the signal at the block input, which corresponds to the oscillation frequency in hertz.



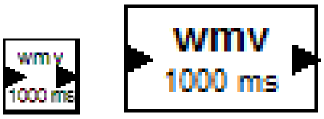
Inputs:
Input to control frequency (Hz)

Outputs:
Sinusoid with given frequency and amplitude.

Parameters:
None

Waited Multivibrator

It generates one impulse with given length when the signal with value «1» is send to the input. If input signal still be «1» after output impulse was finished, then one more impulse will be produce and so on.



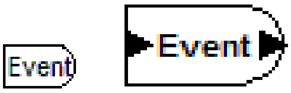
Inputs:
Start with single impulse

Outputs:
Output is «1» during the given time, otherwise «0».

Parameters:
Produced impulse length

Event Generator

For any changes of the input analog signal the output produces value «1», otherwise output is 0.



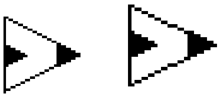
Inputs:
The signal, that if being changed on value more than 1E-3, resulting event production.

Outputs:
«1», if the event occurred, «0» otherwise.

Parameters:
None

Event Generator, Based on Front Edge of the Impulse

When input analog signal increases it produces «1» on output, otherwise output is «0». The input signal is assumed to be «1», if its value exceeds 0.5.



Inputs:
The signal, that if being changed on value more than 0.5, resulting event production.

Outputs:

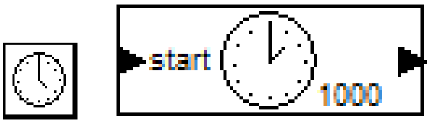
«1», if the event occurred, «0» otherwise.

Parameters:

None

Decrement Timer

Timer counts the time from given value (in ms) to 0. Output is the remaining time. Timer starts on the front edge of the input unit impulse.



Inputs:

Start the timer on the front edge of the impulse.

Outputs:

Time, remained to the end of interval.

Parameters:

Interval (ms)

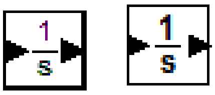
24.5.8. Theory of Automatic Control

This class of blocks includes those blocks which the *theory of automatic control* (TAC) is studied. With these blocks one can describe complex dynamic processes, which are in mechanic, electric and thermal systems. In robots with contour or position control by means of such blocks regulators for executive mechanisms are produced.

One of the important blocks in this section is nonlinear element. Nonlinear element can impose restriction on signal level, transfer the signal from continuous to discrete, quantize by level and so on.

Integrator

The block outputs the time integral of its inputs. The integrator is usually used to form position signal from the signal of velocity, initial conditions. Initial conditions define the output integrator value at the initial time.



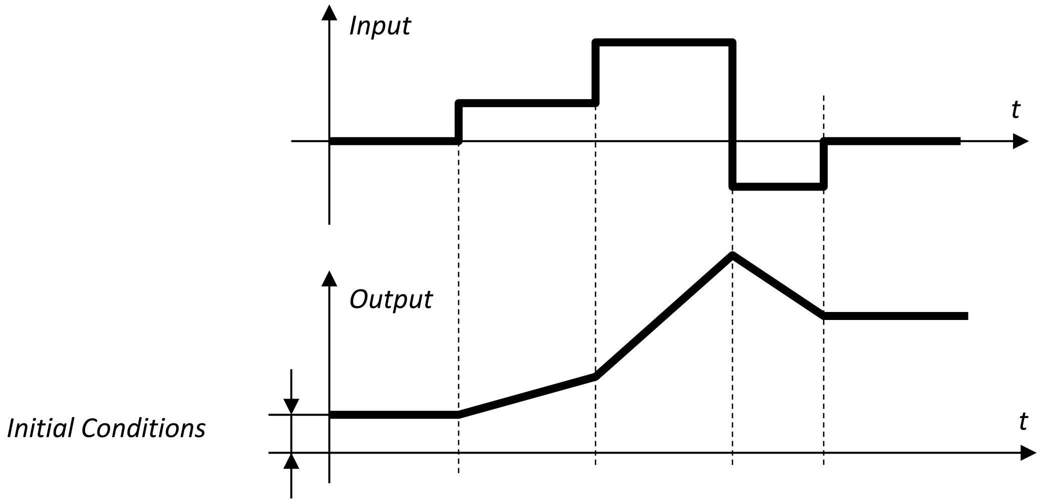
Inputs:

Input signal X(t)

Outputs:

Time Integral of X(t)

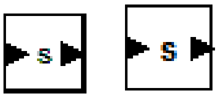
Parameters:



Time diagram for processes in the Integrator

Derivative

Computes time derivative of its input and send it to the output. It usually used to find velocity signal from position signal.



Inputs:

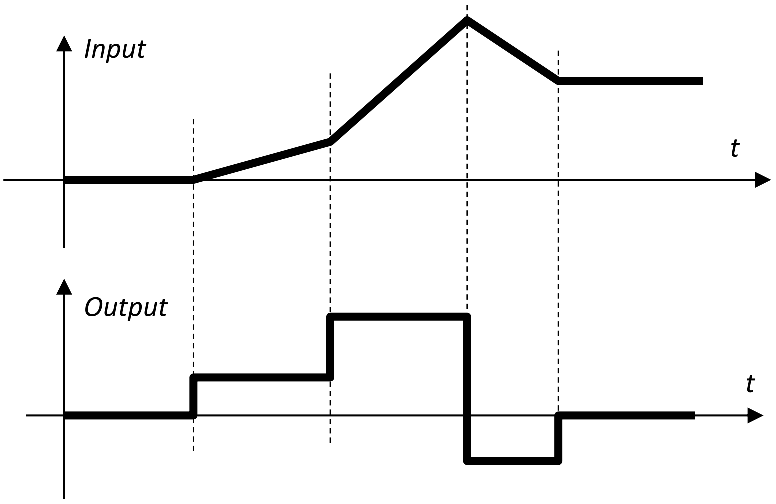
Input signal X(t)

Outputs:

Time derivative X(t) of input.

Parameters:

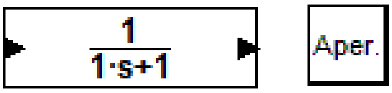
None



Time diagram for processes in derivative block

Aperiodic Element

It’s also inertial. It smoothes Input signal over time. The more is time constant the more are smoothing properties

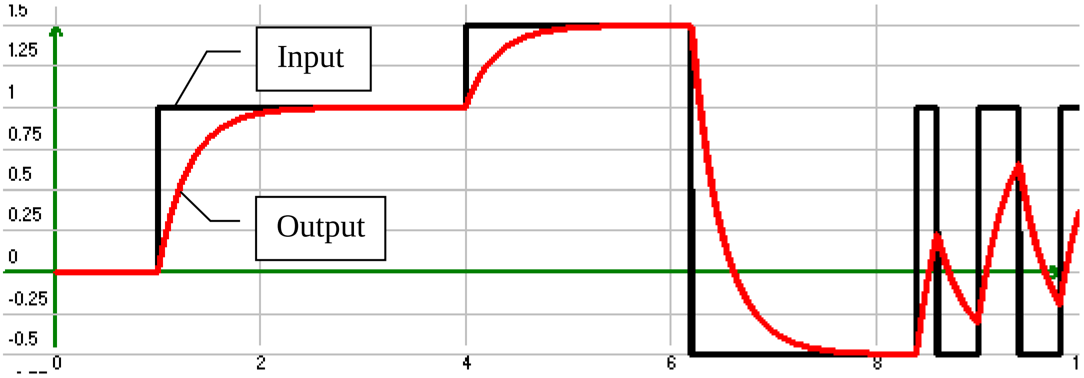


Inputs:
Input signal

Outputs:
Output signal

Parameters:

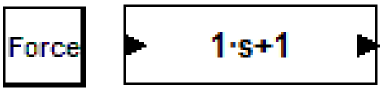
1. Time constant
2. Initial conditions



Time diagram for processes in aperiodic element

Forced Element

Forced Element is artificial element without any analog in the nature. Nevertheless the forced elements are wide used to make corrective filters in automatic control systems. It isn't recommended to use forced elements in simple control systems.



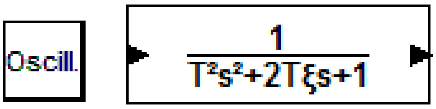
Inputs:
Input signal

Outputs:
Output signal

Parameters:
Time constant

Oscillatory Element

It forms the oscillatory process. It isn't usually used in control systems, but it's wide used to model processes in control object, for example, the model of hydraulic actuator is described by oscillatory element.

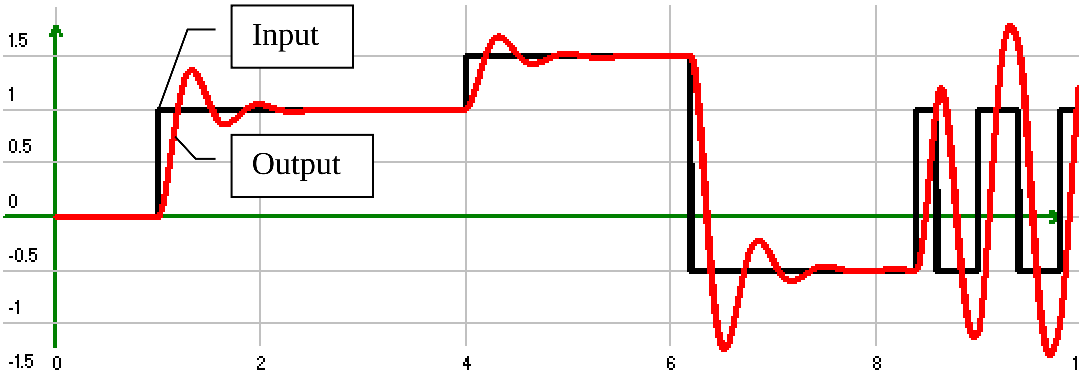


Inputs:
Input signal

Outputs:
Output signal

Parameters:

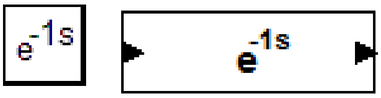
1. Time constant
2. Damping coefficient
3. Initial conditions



Time diagram for processes in oscillatory element

Delay Block

The block delays input signal for a specified number of seconds.



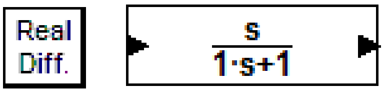
Inputs:
Input signal X(t)

Outputs:
Output signal Y(t) = X(t-T)

Parameters:
Delay time (sec.)

Real Differentiator Element

The block realizes real differentiator element function. It’s used to model processes in analog differentiators.



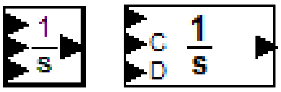
Inputs:
Input signal

Outputs:
Output signal

Parameters:
Time constant

Integrator with Reload

The block realizes ordinal integrator function (see above). But additionally, it can reload its output by the value in input D. The reload is made on the base of input C front edge.



Inputs:

Input signal

C is the signal, on the front edge of which the integrator is reloaded by the value in the input D.

D is the value for reload.

Outputs:

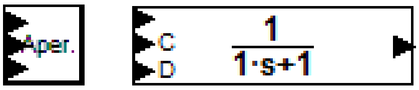
Time Integral of X(t).

Parameters:

Initial output value.

Aperiodic Element with Reload

The block realizes ordinal aperiodic element function (see above). But additionally, it can reload its output by the value in input D. The reload is made on the base of input C front edge.



Inputs:

Input signal

C is the signal, on the front edge of which the block output is reloaded by the value in the input D.

D is the value for reload.

Outputs:

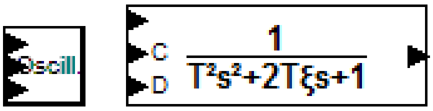
Output value

Parameters:

1. Time constant
2. Initial output value.

Oscillatory Element with Output Reload

The block realizes ordinal oscillatory element function (see above). But additionally, it can reload its output by the value in input D. The reload is made on the base of input C front edge.



Inputs:

Input signal

C is the signal, on the front edge of which the block output is reloaded by the value in the input D.

D is the value for reload.

Outputs:

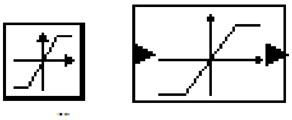
Output value

Parameters:

1. Time constant
2. Damping coefficient
3. Initial output value.

Nonlinear Element

The block computes output value depends on the input value according given graphic. The input-output function graphic is defined as a parameter. Such block gives signal restrictions, characteristics as “sensitive zone”, “relay” and so on.



Inputs:

Input signal X

Outputs:

Output signal Z=f(X), where f(x) – the function, given graphically.

Parameters:

The graphic editor is open to edit the graphic of the nonlinear block characteristic. It is convenient to choose one of the functions from library. The library has patterns of frequently used function graphics.

24.5.9. Other

Text comment

The block adds text comment to any place of the scheme in editor. It does not participate in calculations.

Parameters:

Text comment