User Manual

www.simapp.com **Version 2.0**

SimApp

for Windows 95/98/ME/NT4/2000/XP

License and warranty
Upon ordering, downloading, installing or unencrypting any version of the software, you are reaffirming that you agree to be bound by the terms of this agreement.

License terms
One copy of the SimApp software may either be used (1) by a single person who uses the software personally on one or more computers, or (2) by multiple people non-simultaneously on a single workstation, but not both. You may access SimApp through a network, provided that you are the only person who may use the software or you have obtained the appropriate number of licenses for the software covering the number of workstations that will access the software simultaneously through the network. For instance if maximally eight different workstations are to access SimApp simultaneously on the network, you must have a license for eight workstations.
You may transfer all of your rights to another person, provided that you transfer to that person all of the software, diskettes, and documentation belonging to this software. Once you transfer the software you no longer have any right to use it.

Warranty
Buesser Engineering warrants that the physical media and the documentation will be free from defects in materials and workmanship under normal use for 60 days from the date of delivery to you.
If the software of this warranty is defective, the customer is entitled to replacements on return of the incorrect products with proof of purchase within the warranty period.

Limitation on Liability
In no case shall Buesser Engineering be liable for any indirect, incidental, special, punitive, or consequential damages or losses, including, without limitation, lost profits or the inability to use equipment or access data, due to the use of the software.

User Manual

SimApp

Version 2.0

for Windows 95/98/ME/NT4/2000/XP

Edition 07/03

Contents

About this manual

This manual contains a comprehensive introduction to SimApp. If you are new to SimApp, we recommend that you read it carefully. Simulation programs are not so widely standardized as text or drawing programs. Many important details will only be discovered by reading this manual.
The primary aim of this manual is to present you with the concepts, the main tools, and the equipment for successful simulations. Reference information about the use of special tools and commands is only available in the online help. Use the context-sensitive help if you need information about buttons, property sheets, and dialog boxes. If you have trouble using a tool, press the function key F1, the corresponding help key or search in the *contents and index* in the *Help (?)* menu.
We assume that you have a basic of Microsoft Windows. Furthermore, this manual is not a textbook about automatic control systems and system modeling. Please, refer to the bibliography at the end of this manual.

1 Introduction

1.1 What is SimApp?

SimApp is a Windows program for the analysis and optimization of any kind of dynamic system based on the idea of block diagrams. It doesn't assume any predefined structures.
Nonlinear and time variant systems can be simulated and displayed in the time domain. In addition, linear and time invariant systems can be investigated in the frequency domain. It can construct frequency domain plots as Bode plots and Polar plots (Nyquist plots) and compute the eigenvalues of the investigated system or subsystem. The data is also presented in tabular form and can be copied to other applications by means of the Windows clipboard.

The modeling of the systems, i.e. the drawing of the block diagrams, is made in a full graphical way by placing functional elements into a drawing and connecting them by means of signal lines. The most important parameters (gain, time constants, delays, etc.) can be set directly in the drawing without the need to open any dialog boxes.

SimApp contains an object palette with a rich collection of functional elements. Frequently used subsystems can be combined in groups and stored in the palette or in libraries. You can develop custom blocks that give you the opportunity to model and store real world objects for later use in your block diagrams. This lets you adapt SimApp to your special needs.

The system or any subsystem can be analyzed by means of special measuring probes. Time and frequency responses are available at every diagram node and can be compared with those of other nodes. For example, in an automatic control system you can analyze the closed and open loop response in a single run and compare the results in a common diagram without making any changes to the block diagram (e.g. breaking up the control loop).

SimApp has a multiple document interface. This means that you can open more than one document at the same time. Furthermore, the support of the multi-threading capabilities of Windows enables the simultaneous running of lengthy simulations while working on other drawings.

1.2 Who can or should use SimApp?

SimApp is suitable for students, technicians, engineers and scientists who want study the nature of dynamic systems and set great store by easy and intuitive operation.

1.3 Using help

SimApp has several help mechanisms that support you. You can call the SimApp help application that gives you information in structural form or you can use context-specific help on the various interactive components (menus, controls and toolbars). You will also find this manual in the online help.

1.3.1 Launch help application

You can start the help application within SimApp by pressing F1-key or clicking the help command (?) in the main menu. Now you have the possibility to obtain information by referring to the contents or the index list. This strategy is appropriate if you need special help for a general topic. It is not suitable if you need help with special commands, menu, controls and toolbars. Use the context-sensitive help instead.

1.3.2 Context-sensitive help

Contextual information is available anywhere where you discover the ? character in a windows title bar or a button within the open window. Click the ? character or the help button and then the object you need more information about. This opens a help pop-up window for that object. Alternatively you could press the F1 key that displays a context-sensitive help window for the object that has the input focus.

1.4 Technical support

If you have problems or questions first use this manual or the online help.

If you need more information email us to

info@simapp.com

Please do not hesitate to contact us if you have any questions, suggestions or critics. We are glad of any letter or message and will answer quickly and competently.

Do not forget to visit our homepage regularly. There you always find the latest information about SimApp and customer support.

http://www.simapp.com

As registered user you can always download the latest release of your program version and time limited trial editions for newer versions.

Our address for mailed letters:

Buesser Engineering
Wacht 28
CH-8630 Rueti ZH
Switzerland
Tel. **+41 (0)55 260 12 88** (middle European time)
Fax **+41 (0)55 260 12 89**

2 Installation

2.1 License terms

Before you install SimApp, please read the license terms at the beginning of this manual carefully and check if you agree.

Please note: Technical programs have only a limited user circle and they are costly in developments and support. Your support and your fairness motivates us to improve this product constantly and to adapt it to meet your increasing needs. Please help us in that task!

2.2 System requirements

Minimum

- Pentium IBM compatible personal computer
- Microsoft Windows 95/98/ME/NT4-SP6/2000/XP or higher
- Min. 40 Mbytes free hard disk space for installation
- Min. 8 Mbytes free main memory (RAM) for running
- CD-ROM disk drive for installation
- Mouse
- Color monitor, VGA or better
- Installed default printer

Recommended

- TrueColor or HighColor graphics card and monitor (16 or 24 Bit color resolution)
- 1024x768 screen resolution ore more
- 17“ Monitor or bigger
- For large high resolution simulations we recommend at least a 550 MHz Pentium processor and 256 Mbytes free main memory.

2.3 Installation

Before you can run SimApp, you must first install it by launching the setup program. There are two ways you can obtain the installation files:

- By downloading over the Internet
- On a CD-ROM

Note: Running SimApp after installation is only possible if a default printer is installed.

2.3.1 Internet Download

You can freely download the SimApp Trial Edition from www.simapp.com. You can also download full versions of the program that you have purchased.

In either case, the file you download to your computer is an install program. You must run it to install SimApp onto your system.

The simplest way to do this is to download the file to your desktop. This creates an icon on your Windows desktop. You can then run the install program by double clicking on its desktop icon. You can also run the install program using the Run command under the Windows Start menu.

2.3.2 CD

If you purchased SimApp on a CD, all you have to do is put the CD in your CD drive, and the setup program, setup.exe, will start automatically. If this does not happen, you must run setup.exe manually.

The easiest way to do this is to use the Run command, found by pressing the Windows Start button. Run presents a dialog that prompts you for the name of the program you want to run. If you placed the SimApp CD in Drive D, just type:

D:\SETUP

Follow the directions displayed by the Setup program.

2.3.3 Installing SimApp on a network

To install SimApp on a network, you need the SimApp Server edition. SimApp Trial and SimApp Workstation cannot be run on a network.

When you purchase SimApp Server, you get a personal network license file. SimApp measures the number of different users that access the program on the network and limits the number of simultaneous users to the number of licenses you have purchased.

To install SimApp Server on a network, simply install it on the server in the normal way described above. For each client that will have access to the server, create a shortcut (or icon) for SimApp on the client system and run the program once. SimApp will install itself properly on the client system.

The network license comes as a license file *SimAppServer2.lcf*. You must copy this file in the SimApp program folder and overwrite the existing one before you set up SimApp on the clients. The SimApp program folder is the folder where the SimApp.exe file resides.

If you do not copy the license file, SimApp can only be launched on the network server. But SimApp Server is not intended for the direct use on the network +server or the computer where it is installed on. It will be terminated at least in three minutes. The number of purchased licenses applies only to the client stations.

SimApp.nki
Special care in a network environment has to be taken of the network control file "SimApp.nki".
This file is used for logging and license monitoring purposes. It is automatically created in the SimApp program folder at the start of the first instance and disappears as soon as SimApp is closed on the last client station. But if a station crashes while SimApp is running on it, or the network connection is cut, the running instance can not yet cancel itself in the control file. Subsequently you can no longer use the full number of licenses. At 20 licenses, after a crash of one station you will only be able to start 19 instances. Besides you will also find out that SimApp.nki does not disappear even when SimApp is not running on any station.
Rule: *Delete the file "SimApp.nki" in then SimApp program folder if you cannot launch the full number of purchased licenses any longer.* But do this only, if no instance is running. If not, all running instances are immediately closed, without giving the users a chance to save their work!

2.3.4 Repairing and removing

Run the SimApp install program. As SimApp is already installed you can now choose between *Repair* and *Remove*. If you want to fix missing or corrupt files, shortcuts or registry entries select *Repair*. If you have the server edition you must recopy the license file again. Select *Remove* if you want to uninstall SimApp from your system.

3 SimApp desktop

3.1 Overview

After launching SimApp you see the main window. In the program options (*Extras+Options* menu) you can select whether you want SimApp to reopen all drawings that were open at the end of the last program session, otherwise it opens a new empty drawing.

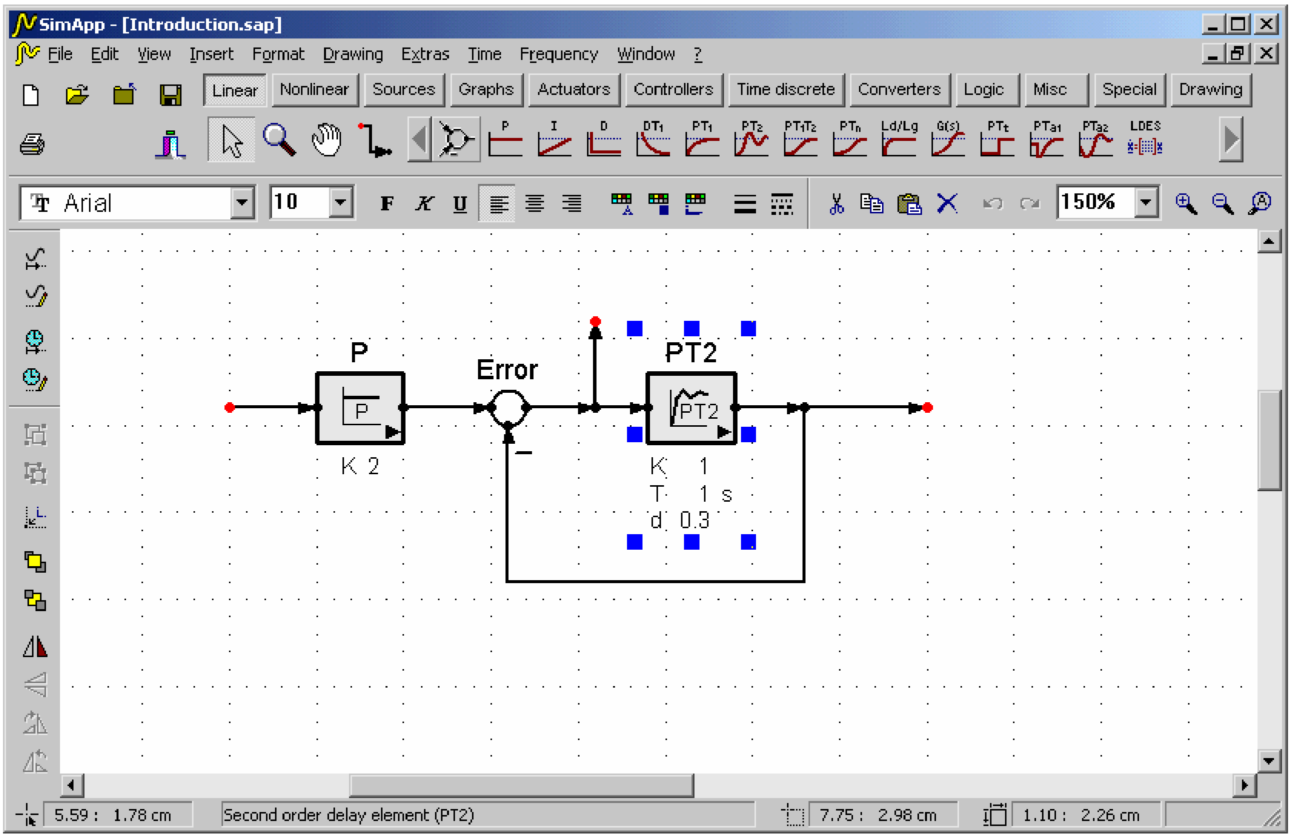


Figure 1: SimApp desktop

3.2 Menus

3.2.1 Main menu

In the menu bar across the top of the SimApp application window you will find most commands. But some contextual or object specific commands are only in pop-up menus.

3.2.2 Pop-up menus

Right-clicking an object opens its pop-up menu. Objects can be anything you see in the application window: palette, buttons, toolbars, panels and even objects in your drawings. By means of pop-up menus you can access the operations of these objects. Pop-up menus are displayed at the pointer's current location so they eliminate the need to move the pointer to the menu bar or a toolbar.

3.3 Toolbars and controls

3.3.1 File toolbar

The file toolbar is in the top left corner of the SimApp main window. It has some buttons for the most important file operations, such as save, open, print, and a button to exit SimApp. This toolbar is not moveable, but you can hide it along with the palette.

3.3.2 Palette

The palette is a multiple page toolbar containing all standard simulation objects and some drawing tools. Select a page by clicking its tab.

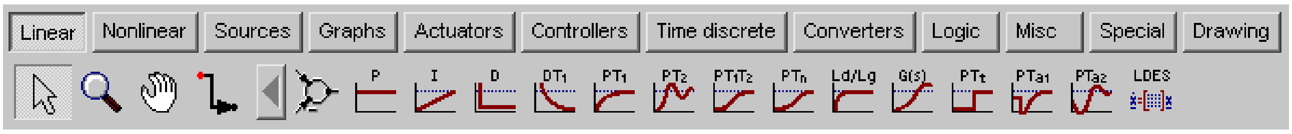


Figure 2: The palette

There are on principle two types of objects:

1. Shapes, lines and text (Page *Drawing*)
   In SimApp you can draw simple shapes and lines. These objects are not primary objects because they are not needed for simulations. You can use them if you want to illustrate your drawings and draw the symbol of custom simulation blocks.
2. Simulation objects
   These objects are the building blocks of your block diagrams, i.e. the models of the systems you want to analyze by simulation.

On the left side of the palette are the most important tools for drawing manipulations (s*election, zoom and dragging*).

Extend the palette by adding your own objects. (See chapter *Working with the palette*).

3.3.3 Moveable toolbars

The various toolbars contain buttons for the most important commands and selections. Each toolbar offers a distinct category of commands. Toolbars are moveable or can be docked along the inner edge of the SimApp main window. They can be individually shown and hidden (*View* + *Toolbars* menu). Double-click a toolbar to quickly change between docking and moving mode.

3.3.4 Library toolbars

Library toolbars are user configurable and can be saved on disk. They are primarily dedicated as containers for custom blocks, but you can also store shapes, text and any kind of object groups. Library toolbars are the visual representation of object libraries. They behave like common toolbars with buttons to select the objects for insertion in your drawings. Open libraries with the menu command *Extras* + *Library* + *Open*.

3.4 Status bar

The status bar is at the bottom of the SimApp main window. It displays information about the current drawing state.

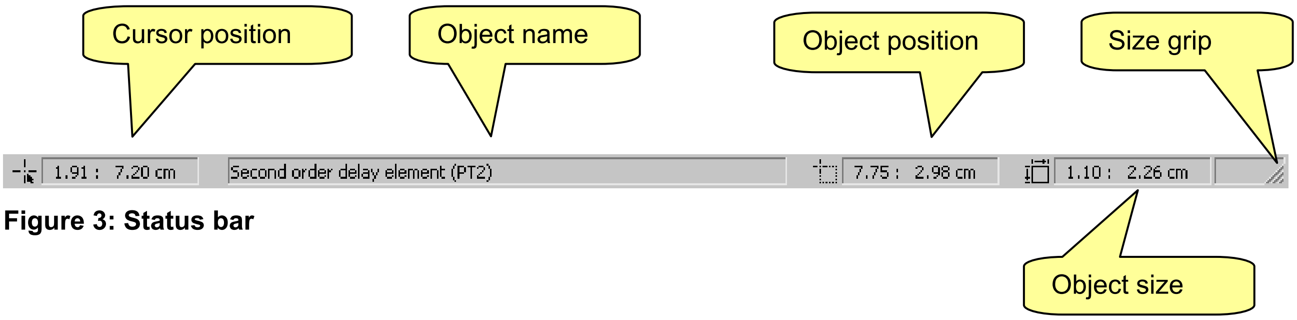


Figure 3: Status bar

4 Introductory example

This chapter is a small project to demonstrate step-by-step creation of a block diagram and simulation of the time and frequency response. The use of time and frequency probes, xy graphs and parameter variation is presented in the following chapters.

4.1 Launch SimApp

Locate the SimApp icon in the Windows program folder of the Start menu and click it. The SimApp main window is opened and contains an empty, maximized drawing window. The SimApp main window is an MDI (Multiple Document Interface) parent window that can contain several child windows.
The drawing windows are fully independent of each other. In every window you can model a system and make simulations. But a system can not be distributed over several windows. However a drawing can cover several print pages in portrait or landscape orientation. New drawings are always in landscape. The page margins are displayed as blue dashed lines.

4.2 Views and page arrangements

There are various commands to manage the view (size and position) of your drawings. You will find all these commands in the standard toolbar.

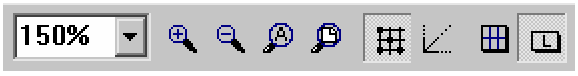


Figure 4: Changing view

4.3 System modeling

The real system is represented by block diagrams (or block schemes). A block diagram consists mainly of blocks and signal lines. The blocks represent the transfer elements that change system data or create new signals. The signal lines are to connect the blocks and so representing and enabling the system's data flow. Each line stands for a system data item whose direction is represented by the arrow head.

4.3.1 From real system to block diagram

You are lucky if your system is already represented by a mathematical description (differential equations) or even by a block diagram. You can just start drawing. Otherwise you still have a lot to do: Analyze the system, find interfaces to other systems, dismantle it in subsystems and find suitable mathematical equations. If you have found a mathematical description you can present it in a graphical form by using objects from the palette and simulate it. However, the convolution from the real system to a block diagram representation is not part of this manual. Refer to the literature about automatic control systems, control engineering, nonlinear control, etc. See the bibliography at the end of this manual.

In palette, all basic elements are divided in categories. As a simple project we draw the following block diagram:

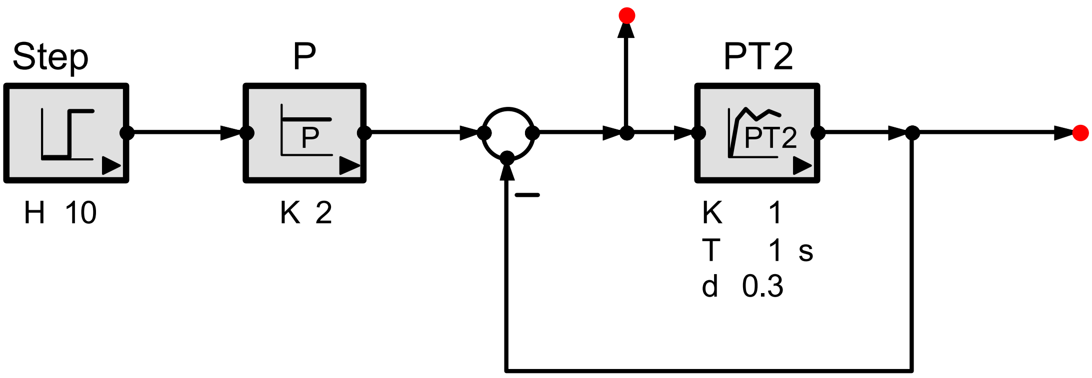


Figure 5: Control loop with step stimulation

Be sure that automatic snapping is on (button ⌗). Get the objects out of the palette by first clicking the tab of the page that contains the category the desired object belongs to and then click the button showing the symbol of the object (release the mouse button again). Position the mouse at the location where you want to paste the object and press the mouse button. If you don't release the button immediately, you see the shape of the object at the pointer's location. You can now exactly position the object by dragging and then releasing the mouse button.

4.3.2 Connecting objects

The signal line tool can be activated by the corresponding button on the left and fixed part of the palette. Click the tool button and position the pointer at the desired starting point in the drawing, usually an output or input node of a diagram block, and press the mouse button. Draw the first segment with pressed mouse button. If you wish to draw a second segment, release the button at the end of the first segment and press it again immediately. Draw the second segment. If you have finished the signal line, double-click at the end of the line. You will notice that the end nodes of the new signal line automatically connect to the block nodes that are at the same locations.
When moving a block, the end nodes of the signal lines are dragged along. Nodes, corners and segments can be moved individually. You can also insert new corners by splitting segments or removing corners by aligning two adjacent segments so that they form a straight line.

4.3.2.1 Some remarks on working with signal lines:

Signal lines are started by pressing the mouse button. Corners are created by releasing the mouse button and immediately pressing it again. A signal line is completed by double-click. Tiny segments that mostly are created by mistake and corners between adjacent segments that virtually form a straight line are removed automatically.

The sign that appears when a signal line runs into a sum block is changed by selecting the signal line and pressing the + or - on the numeric key pad. Alternatively you can right-click the signal line to open the pop-up menu that contains the menu item Change Sign.

SimApp checks the correctness of the data flow. If there are any false connections (e.g. short circuits), the affected signal lines are colored yellow and the arrows are missing.

You can change between two working modes: Point operation is switched on by default. This mode enables you to operate single nodes, corners and segments. Otherwise you can move and stretch a signal line as a whole only. Change the mode in the signal line's pop-up menu.

The following remarks refer to the mode Point operations:

You can move a selected signal line as a whole if you don't touch nodes or corners.

If you move a node, all connected nodes of other signal lines are moved too and stay connected. Break up a node connection by pressing the shift key before moving. Blocks are never moved by node connections.

Move single nodes or corners by pressing the mouse button.

Move single segments by simultaneously pressing the Ctrl-key. If you press the Ctrl-key after pressing the mouse button the signal line is doubled.

Insert new corners or split segments by first selecting the signal line. Then press the Shift-key, click the location where you want to split the segment and drag the new point to a new location. Note, that the new corner is immediately removed if you don't move it significantly.

Remove any corners by pressing the Shift-key and clicking the corners.

Slanting segments can be replaced by a vertical/horizontal segment pairs by pressing the Shift-key and clicking the segment.

4.3.3 Changing block parameters

There are two ways to change the damping d of the PT2 block to 0.3:

Fast editing d in the drawing
Click the numeric value of d. (Not the math symbol). A input box appears with the old value selected. Enter a new value and press *Enter*.

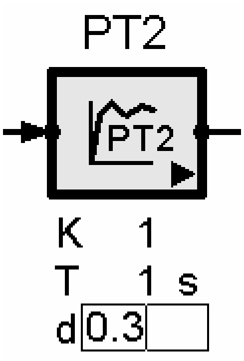


Figure 6: Changing block parameters

Changing d in the block properties dialog box
Double click the PT2 element or select *Simulation properties* in the element's pop-up menu. Locate the input control for the damping, enter a new value and press *Enter*. Note that there are more parameters in the dialog box than visible in the drawing.

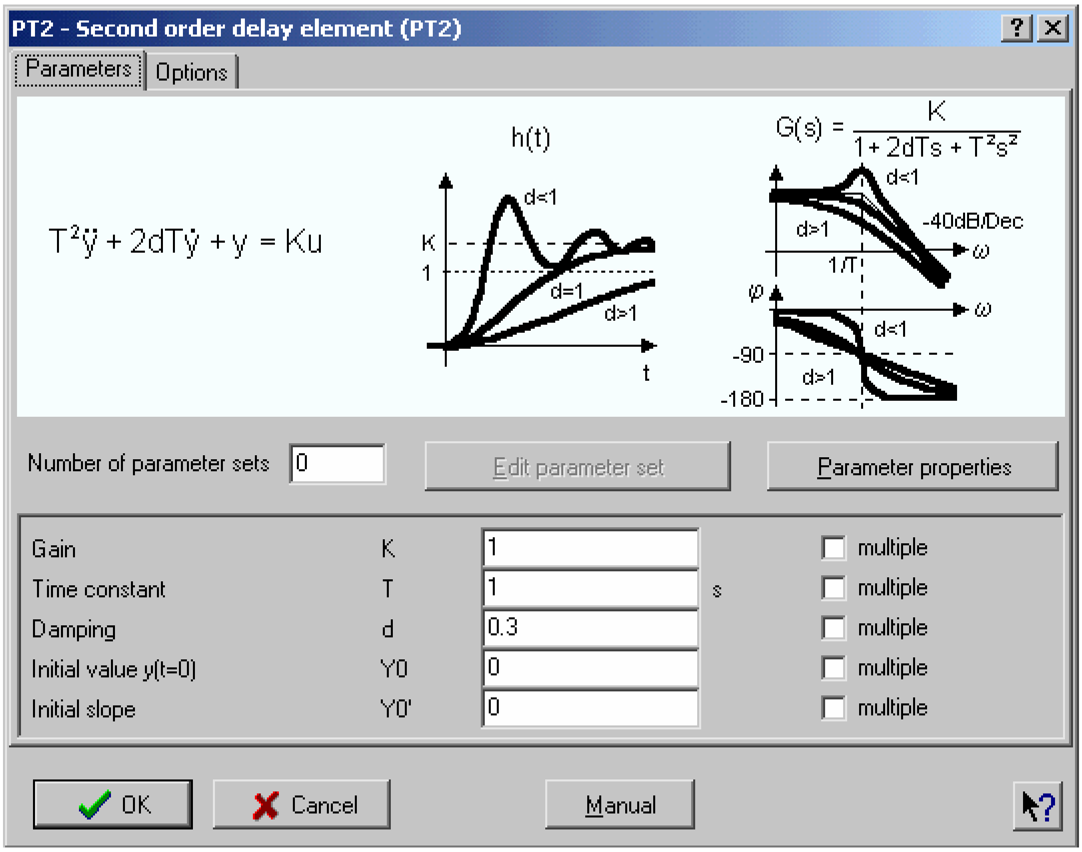


Figure 7: Properties dialog box of the PT2 element

4.4 Simulations

After drawing an error free block diagram you can perform time and frequency simulations.

4.4.1 Time simulation

First, we are interested in the system's step response. Press the start button for time simulation. A new window consisting of several pages appears. On the top page you see the graphical representation of the step response at the system's output nodes.

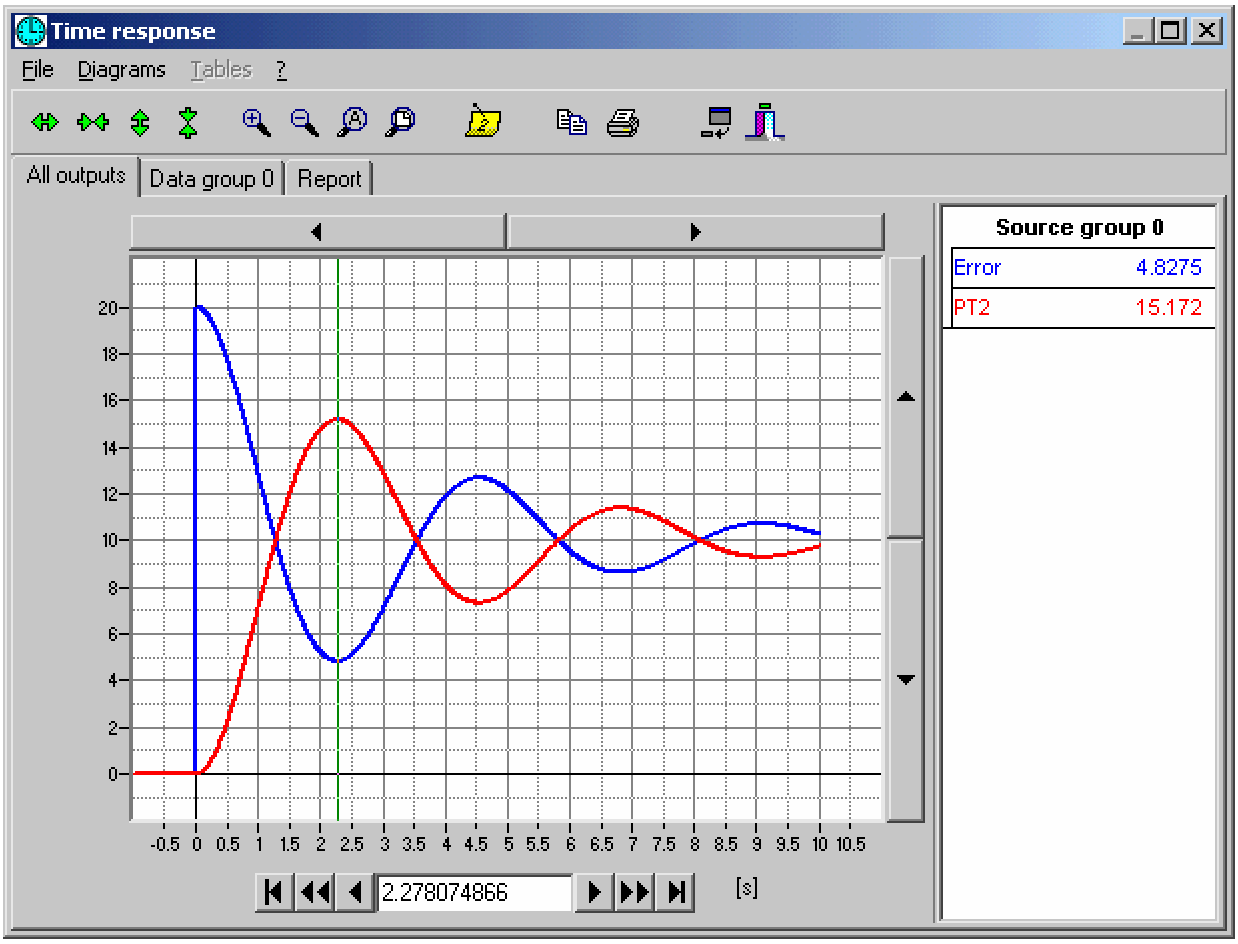


Figure 8: Time diagram

On the second page you find the simulated values in tabular form. The third page shows the parameters and some statistical values of the current simulation. Tables and plots can be printed out or transferred to other applications.

The diagram contains a vertical measurement line that is moveable by mouse or navigator buttons at the bottom of the page. The current time is always displayed in the navigator and the corresponding signal values are displayed in the legend. Zoom any diagram area by dragging a zoom frame with the mouse. The curves can be edited by right-clicking the curve or the corresponding legend item.

The data in the tables can be selected and copied to other applications by means of the Windows clipboard. Columns are adjustable with the mouse.

4.4.2 Frequency simulation

The sample block diagram contains only linear elements so that you can simulate it in the frequency domain without any significant changes. Just delete the step source because you need a definite input node. Don't close the window of the preceding time simulation.

Press the start button for the frequency simulation. A new window appears that is very similar the preceding one.

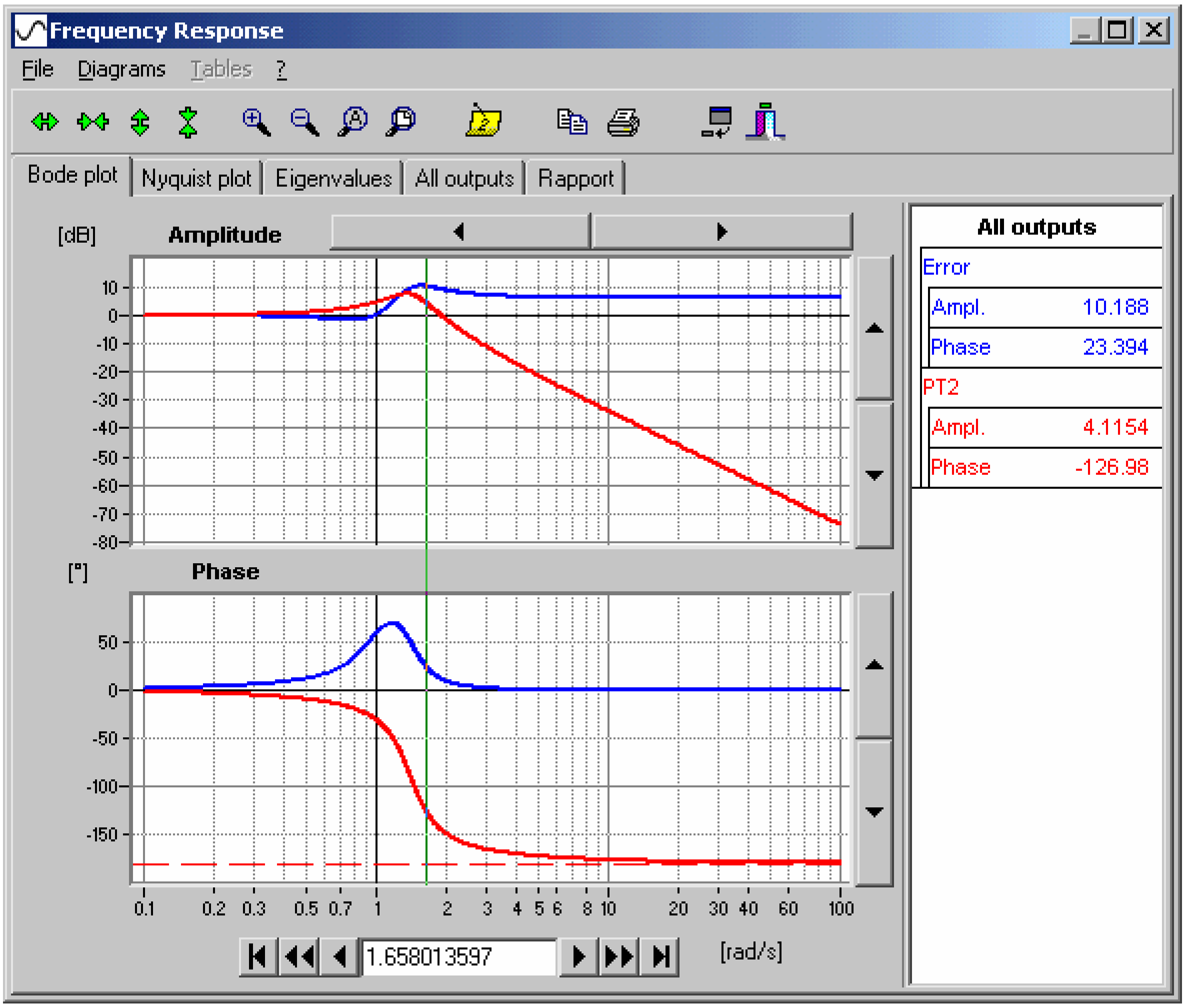


Figure 9: Frequency response

The window contains several pages including a Bode plot, a Nyquist plot, a data table, a table containing the systems eigenvalues and a short report of the simulation.

4.4.3 Simulation list box

Each individual simulation run produces a new data output window. You can leave it open for purposes of comparison or close it instantly. If you close it, all simulation data will be lost. You can hide it temporarily change the text in the title bar.
A small list box always on the top shows a list of all open simulation windows. The titles of the windows appear as list items. Click a corresponding list item when you want to show and move a window into the foreground.

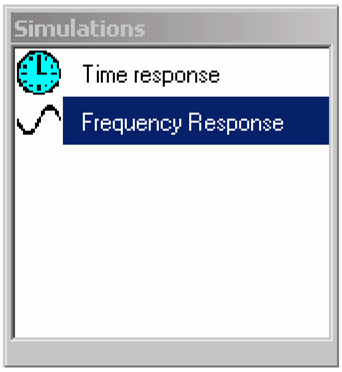


Figure 10: Simulations list box

4.5 Conclusions

By looking at this sample project you have gained some experience with SimApp. Now, you know the basic steps for simple time and frequency simulations and the resulting data representations. Try your own samples to get more accustomed to SimApp.
In the next chapters you will learn how to use time and frequency probes that let you define inputs and outputs in a more flexible way. You will also learn how to change parameters that control the simulation process. In SimApp you can even create your own blocks, store them in libraries, and customize the palette.

5 Drawing functions

5.1 Introduction

SimApp is also a simple drawing application. You can draw shapes, lines and text, change colors, sizes, widths, etc. All these features are for the following purposes:

- To model your systems by block diagrams before running any simulation .Building a block diagram is a drawing process that especially needs the following operations: Pasting, drawing signal lines, selecting and deleting objects.
- To illustrate your drawings and compile a complete documentation that you can store, print and copy to other applications.
- To draw symbols for your own custom blocks.

5.2 Drawing objects

Draw simple shapes, lines and text by selecting the appropriate tool in the *Drawing* page of the palette.

Figure 11: Drawing tools

5.2.1 Lines

Lines are simply drawn by pressing and dragging.

5.2.2 Rectangles and squares

Rectangles and squares are also simply drawn by pressing and dragging. To draw precise squares, press the Shift-key while dragging.

5.2.3 Rounded rectangles and squares

Rectangles and squares are drawn like normal rectangles and squares. But you can also convert a normal rectangle or square to a rounded rectangle or square and vice versa by clicking the *Rounded corners* menu item in the object's pop-up menu.

5.2.4 Ellipses and circles

Draw ellipses and circles by simply pressing and dragging. To draw a true circle, press the Shift-key while dragging.

5.2.5 Polylines

Polylines are lines with multiple straight segments. You can convert a polyline into a polygon and vice versa at any time.
Start drawing a polyline by pressing the left mouse button and dragging the first segment. At the end of the first segment where you want to insert a corner, release the button, press it again and drag the second segment. Double-click at the end of the last segment to complete the line. Select the pop-up's menu item Close Shape to close the shape to a polygon and vice versa.

When you have finished the polyline you can choose between two working modes: Point operation is switched on by default. This mode enables you to work with single nodes, corners and segments. Otherwise you can move and stretch a polyline as a whole only. Change the mode in the line's pop-up menu.

The following remarks refer to point operation mode:
Move a line end or a corner by clicking the mouse and moving it to a new location.
Move a single segment by first pressing the Ctrl-key and then clicking it with the mouse. Drag it to a new location and release mouse button and key. Note: The line is doubled if you press the Ctrl-key after pressing the mouse button.

Insert new corners or split segments by first selecting the line. Then press the shift-key, press on the location where you want to split a segment and drag the new corner to a new location. Note that the new corner is immediately removed if you don't move it significantly.
Remove any corners by pressing the Shift-key and clicking the corners.
Slanting segments can be replaced by vertical/horizontal segment pairs by pressing Shift and clicking the segment.

5.2.6 Polygons

Follow the same rules as for polylines. You can convert polygons in polylines and vice versa by the pop-up's menu item *Close shape.*

5.2.7 Multiline text

Click the drawing at the starting location of the text object and enter any text. Press *Enter* to terminate the input. Line breaks are achieved by pressing *Ctrl+Enter*. Press *Escape* to abort the text entry. To edit existing text, select it and click the text button, or double-click the text.
Note: For element titles will do a single click.

5.2.8 Pictures

By means of picture objects you can paste bitmaps (*.bmp), metafiles (*.wmf; .emf) or icons (.ico) in your drawings. These objects could come from files or from the Windows clipboard. With pictures you can show graphics that you cannot create in SimApp (pixel graphics and sophisticated vector drawings). Create the pictures in a powerful graphics editor and paste it in your SimApp drawing. In SimApp you cannot change them any more, but you can change their size (except icons). Metafiles may be converted into native SimApp objects.

5.2.8.1 Paste pictures from Windows clipboard

Create the picture in a graphics editor and copy it into the Windows clipboard. Back in SimApp, paste the picture by the menu command (Edit + Paste) or by clicking the paste button.
Applications store data to the Windows clipboard in several formats. The paste command always prefers the native format of the pasting application. If it cannot find the own format, it selects the most similar format. SimApp prefers the .sap format. When it cannot find its native format, it selects the metafile format and the bitmap format.
If you don't want the preferred format or if you don't know which format will be pasted, select the Paste special... command (has no toolbar button). You see a list of all known formats in the clipboard. After pasting, the objects are generally shown in original size (if scale is 100%).

5.2.8.2 Insert pictures from a file

Click the picture button on the palette and place a empty picture into the drawing.

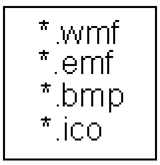


Figure 12: Empty picture

In the picture's pop-up menu you find the following commands:

1. *Paste picture*: Pastes the contents of the clipboard into the empty picture. This corresponds to the paste command as explained above.
2. *Load picture...*: A file dialog box is opened and lets you load a wmf-, emf-, bmp- or ico-file.

5.2.8.3 Edit picture

You cannot edit the contents of the picture. But you can change its size. For restoring the original size select *Restore original size* in its pop-up menu.
You can replace the picture by another one at any time.

5.2.8.4 Convert a picture

You can convert metafiles (wmf, emf) into native SimApp objects by selecting *Convert picture*. Objects unknown to SimApp are omitted.

5.2.9 Arrow

The arrow is a special object and is used for the symbol of a custom block. It shows the direction of the data flow.
Other arrows of any shape and size can be created with the polygon tool.

5.3 Formatting objects

Each drawing object has formatting attributes (e.g. line style, font, filling, etc.). New objects have default formatting attributes that you can change. Changes to formatting attributes apply to selected objects. If no object is selected, the default attributes are changed.

5.3.1 Format toolbar

There is a special toolbar for formatting.

Figure 13: Format toolbar

This toolbar contains controls for the most important attributes.

5.3.2 Format properties

All format properties are accessible on the format property sheet. You can open it in the main menu *Format* or in the object's pop-up menu. (*Format properties*)

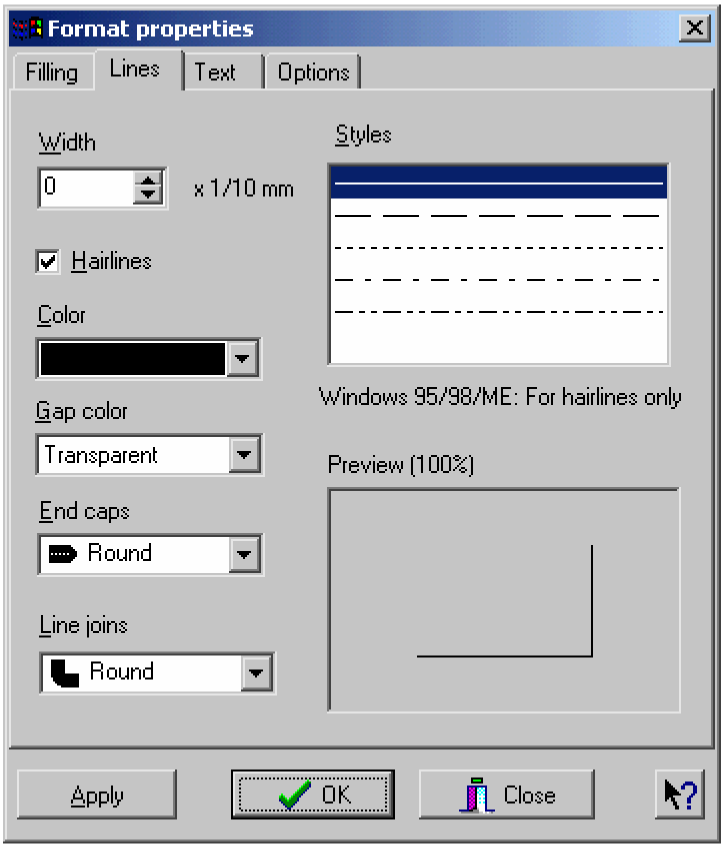


Figure 14: Format properties for lines

The format property sheet always shows the current attributes of selected objects. It is not a modal dialog box. You can keep it open as long as you like. After changing any attributes, the new values are not immediately applied. You must first click the *Apply* button. If you click the *Ok* button, the changes are applied and the property sheet is closed. Press the *Close* button to discard any changes and close the property sheet.

5.3.2.1 Options

On the options page contains the operating options for selected objects. Options can be enabled or disabled. These options are especially important for drawing objects forming the symbol of custom blocks that are explained later in this manual (see chapter: Custom blocks).

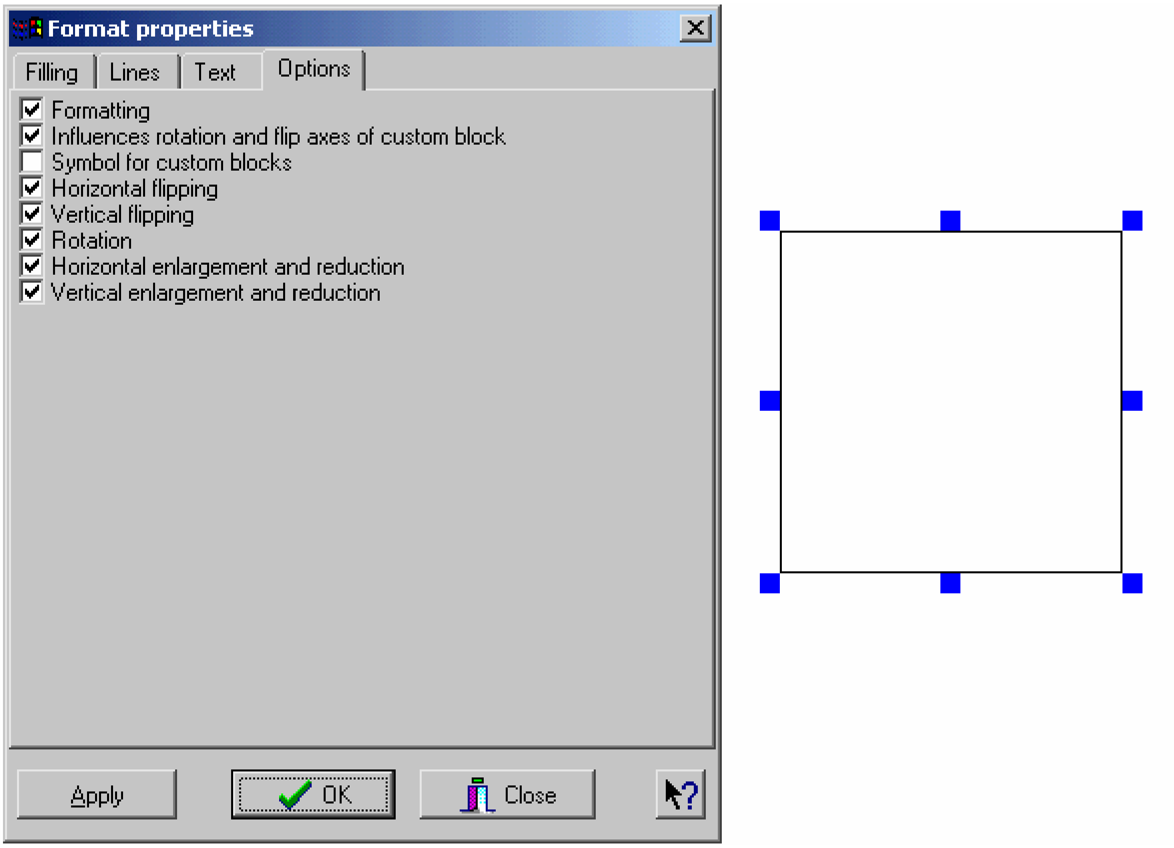


Figure 15: Options for a rectangle

The check boxes can have three states:
The mixed-value state (grayed check mark) indicates that the values for an option of a multiple selection differ (e.g. for a text selection that is partly editable). If you press Ok, that option is not changed for all selected objects. If you click a check box that is in the mixed-value state, the associated value is set and a check mark is placed in it. This implies that the property of all objects in the multiple selection will be set to this value when it is applied.

5.3.3 Default formatting attributes

New objects receive default formatting attributes. Change default formatting attributes in the attribute property sheet while no object is selected.

5.4 Rearranging and changing objects

There are various commands to change and rearrange objects in the main menu, the object's pop-up menu or the drawing toolbar.

The drawing toolbar contains the most important commands:

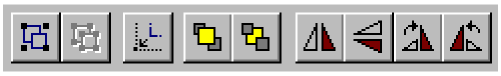


Figure 16: Drawing toolbar

5.4.1 Flip and rotate objects

Rotate selected objects by steps of 90 degrees to the left or to the right or flip them vertically or horizontally .

5.4.2 Object's Z-order

A drawing also has a third dimension - the Z-dimension - because objects can overlap. Z-order refers to the order in which you encounter objects as you start closest to you and move back "into" the screen. The last object inserted is the front object. Change the Z-order for individual objects: brings a selected object to the front and sends it to the back.

5.4.3 Snap objects to grid

If you draw signal lines and insert simulation blocks, it is rather difficult to keep the object on a line. To simplify the precise placing of objects and drawing of exact vertical and horizontal lines use the grid. Choose whether all objects drawn or pasted into the drawing should snap automatically to the grid. This feature is very helpful for drawing block diagrams since you don't need the high resolution SimApp offers. The connecting of nodes is much simplified and you don't need to align line segments all the time.

You can temporarily switch off the snapping (with button), if the grid is too wide for special drawing operations. But if you forget to switch it on again, you will place all subsequent objects beside the grid lines. If you switch it on after some time and try to match the objects to each other, you will realize that it doesn't work. The objects lie on different grids.
SimApp differentiates between a global and an object-oriented grid. The global grid corresponds to the dots in the drawing. The object-oriented grid refers its origin to the location of the object. But the grid width is the same. Thus, if you move an object without snapping, its grid will not match the global grid any longer.

To move selected objects to the grid, press the Lock into grid button .
Tip: Never switch off the snapping function. Press the Alt-key temporarily if you want to place an object more precisely. As long as the Alt-key is pressed the snapping is off.

5.4.4 Group objects

Grouping objects can simplify the task of working with several objects simultaneously. Select all objects you want to put in a group and press the Group button . Ungroup a group by pressing the Ungroup button .

5.5 Important auxiliary keys

When you draw block diagrams you usually use the mouse. For some actions, however, you also need one or two keys. SimApp uses the following keys:

Alt-key
The effects of the Alt-key depend on the current operation:
- Pressing the Alt-key while drawing, inserting or moving objects, the snap function is temporarily off. This is useful if you want to place some objects more precisely.
- Press the Alt-key before pressing the mouse button if you want to move palette pages or palette buttons.

Ctrl-key
The Ctrl-key causes an object to be copied. To copy or double an object, press the Ctrl-key before releasing the mouse button. This is also valid for copying objects to the palette or to a library.

Shift-key
The Shift-key causes objects to be aligned. Use this key, if you want draw an exact square or circle. It also simplifies the drawing of a vertical, horizontal or 45 degree lines. If this mode is switched on permanently pressing the Shift-key switches it off temporary.

Esc-key
The Esc-key is an abort key when drawing polyline, signal lines and text.

Space-bar
Press the space bar to suppress auto scrolling. This is useful if you want to move an object to the palette.

6 Simulation objects

6.1 Description

Simulation objects are the basic elements for block diagrams. They generate process and transmit data. The signal lines represent the time variant data of a system and the blocks the functional relationship of data items. The arrow inside the block symbol shows the data flow direction through this object. The outputs are generally on the right and the inputs on the left edge. For feedback loops, blocks can be flipped by the flip command.
You can also create custom blocks that consist of a system of basic simulation objects or any other user defined blocks.
The basic blocks usually have two different symbols. The default symbol is a schematic representation of the time or frequency response. The second symbol shows the mathematical transfer function in the time or frequency domain.

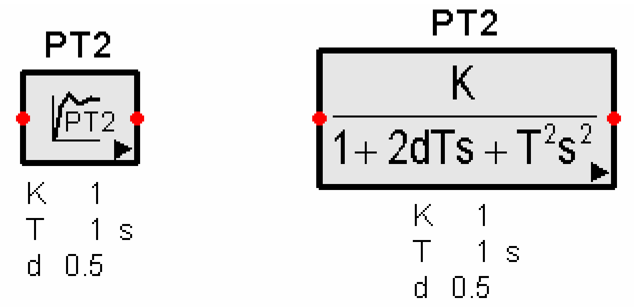


Figure 17: Symbols of the PT_2-block

Choose the default symbol in the options dialog box (menu *Extras + Options)*. Switch between symbols individually in the object's pop-up menu.

The transfer function of a block is controlled by its parameters. Each block has its own set of parameters. Some of them are displayed just below the block in the drawing and can be edited by clicking others can only be changed in the block's property dialog box. You can determine which parameters appear in the drawing. Every parameter has a default value. Those parameters usually not displayed in the drawing are nevertheless displayed if their values have been changed.

6.2 Connecting objects

Blocks are connected by signal lines. You have learned in the introductory example how to work with signal lines. They behave as polylines, except that polylines don't have nodes with connecting capabilities. Another difference is the formatting process. You can not determine the formatting for each signal line individually. In the program options (menu Extras + Options...), you determine style, width, color, etc. for all signal lines.
Signal lines can transport data only in the direction indicated by the arrowhead. You can connect the outputs of two blocks with each other. But SimApp recognizes the collision and changes the color of the signal line to yellow and does not draw the arrowhead. The arrow and thus the data flow can be turned and moved to the other node by the Turn arrow command in the signal line's pop-up menu or by the End- and Home-key.

6.2.1 Addition, subtraction and inversion

Add, subtract or invert signals by means of sum blocks:

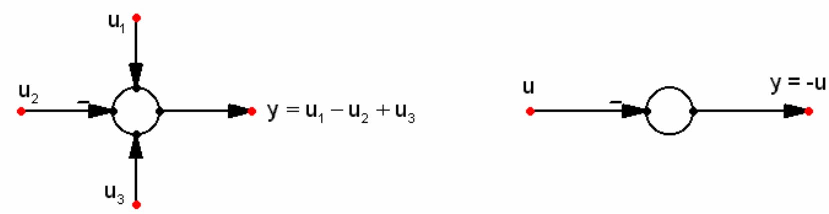


Figure 18: Addition, subtraction and inversion

The polarity of a signal is determined by the sign. You can change the sign by the signal line's pop-up menu or by pressing the - and + key on the numeric key pad. Press the Alt-key if you have difficulties in connecting a signal line to a sum block; this disables snapping temporarily which is sufficient.

6.2.2 Branchings

You need branchings if you want to distribute a signal to more than one input.

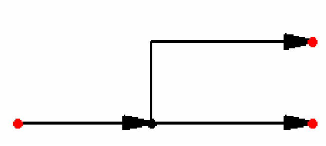


Figure 19: Branching

Create branchings by placing a node of a signal line on a corner or a vertical or horizontal segment of another signal line. Remove a branching by dragging a node away from the branching location while pressing the Shift-key. Also press the Shift-key if you want to move a block without pulling the signal lines.

6.3 Fast editing

You can edit block parameters and titles just in the drawing without opening a dialog box.

6.3.1 Edit block titles

By clicking the title the whole text is selected. Position the cursor by a second click on the desired location. Press Enter if you have finished. Insert line breaks with Ctrl + Enter.
To move the title click the text and hold down the Alt-key.
Note: After changing text the title is repositioned automatically. (Presupposed this feature is switched on in the elements properties.)

6.3.2 Changing parameters

Simply click the parameter value. Move the whole parameter table by clicking and pressing the Alt-key.
Note: After changing values in the parameter table the table is repositioned automatically. (Presupposed this feature is switched on in the properties of the element.)

6.4 Block Properties

This dialog box contains all properties and options that concern object specific simulation settings. You can open it by the *Block Properties* command in the object's pop-up menu.

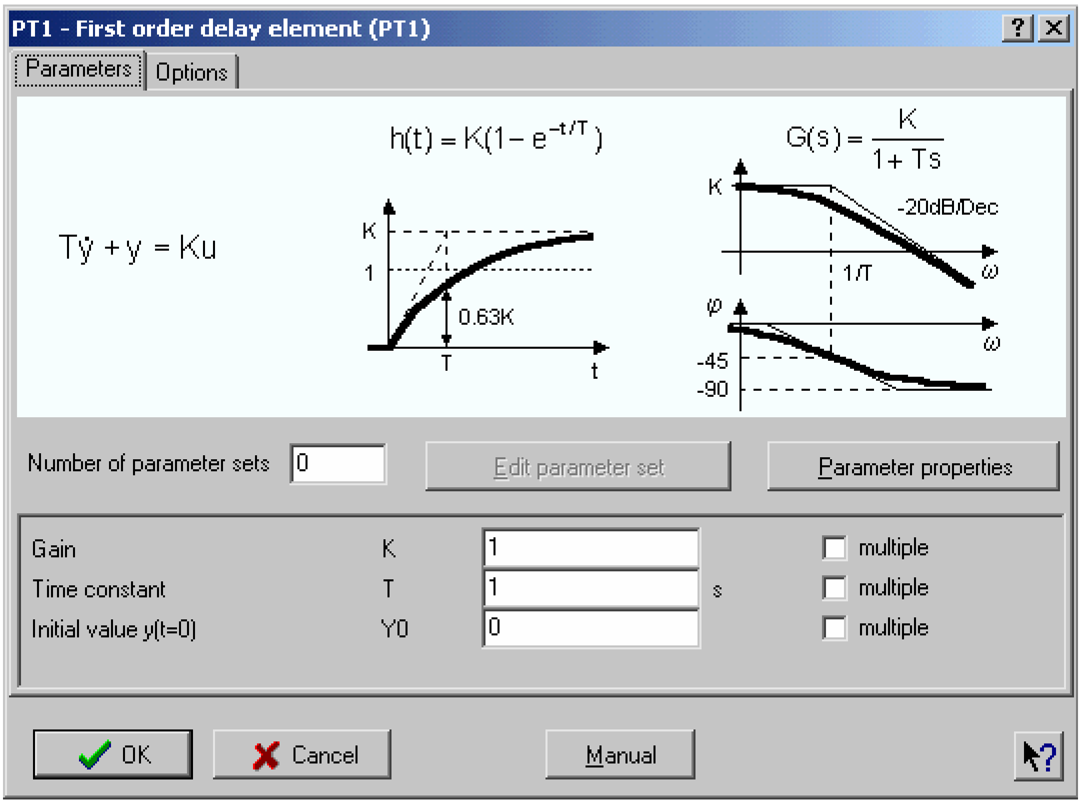


Figure 20: Block properties of the PT_1 element

The dialog box contains two pages. All or most parameters appear on the *Parameters* page. On the *Options* page you can change display options. Some elements still have more pages for additional settings.
Number of parameter sets and *Edit parameter sets* as well as the option boxes *multiple* refer to parameter variation. (See chapter *Parameters variation*.)

6.4.1 Parameters properties

On the *Parameters* page you can only change the numerical values of the parameters. But it's nevertheless possible to customize the parameters by changing name, unit, symbol, etc. Press the *Parameters properties* button.

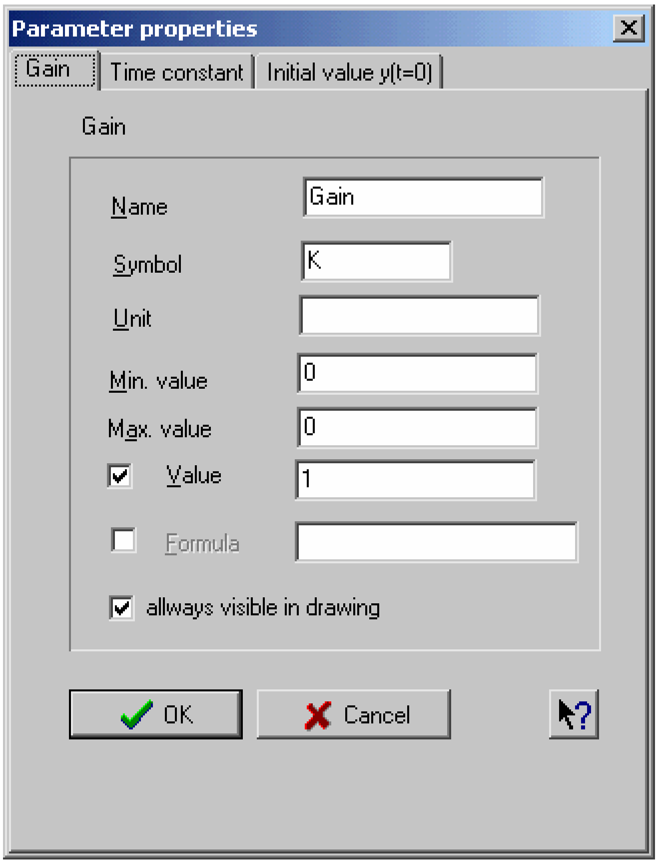


Figure 21: Parameters properties

There is no limitation if you enter zero for minimum and maximum input. The Option *Visible in drawing* determines whether a parameter is displayed in the drawing.
The item *Formula* is used in blocks being part of a custom block. It defines the relations of the block parameters to the virtual parameters of the custom block (more information see later in this manual).

6.4.2 Units

SimApp knows the following default units:

Time	ps	ns	us	ms	**s**	min	h	d	a			
Slope	ps-1	ns-1	us-1	ms-1	**s-1**	min-1	h-1	d-1	a-1			
Frequency	urad/s	uHz	mrad/s	mHz	**rad/s**	Hz	krad/s	kHz	Mrad/s	MHz	Grad/s	GHz

The basic units are bold.

SimApp recognizes these units and uses internally the appropriate conversion factors.

A up/down button pair appears if the unit edit box contains a default unit. Simply press the keys to change between default units of the same type.

If SimApp encounters a unknown unit, it uses the value as it is. For example, if you type "y" instead of "a" for year, SimApp uses the basic time unit "s" (seconds).

Pay attention to capitalization! For example if you enter "h" instead of "H" (Henry) for magneto-electric induction, the value is evaluated as hours and SimApp multiplies it by the factor 3600 (one hour). As "H" is a unknown unit SimApp uses the value as it is.

6.4.3 Options

Set the object title and determine which parameter items are to displayed in the parameter list in the drawing. As default, mathematical symbol, value and unit are displayed.

6.4.4 Labeling objects and signal lines

It is important that you label your block diagrams carefully. The names of objects and signal lines are also used in Bode and Nyquist diagrams and in data tables.

Each element has a name that is displayed as title above the block symbol. The default name corresponds to the basic function of the block. Change the name so that it describes the real operation it performs in system. A water tank can be modeled by an integration block. Therefore, change the name from Integrator to water tank.
Signals take over the name of the block from which they come from. If a block has several outputs, all signals will have the same name. To change the name of individual signals, use the pop-up menu of the associated output nodes by right-clicking the nodes. (The element must not be selected.) After entering a name, you have to position it correctly. Press the Alt-key while doing this so that snapping is temporarily switched off.

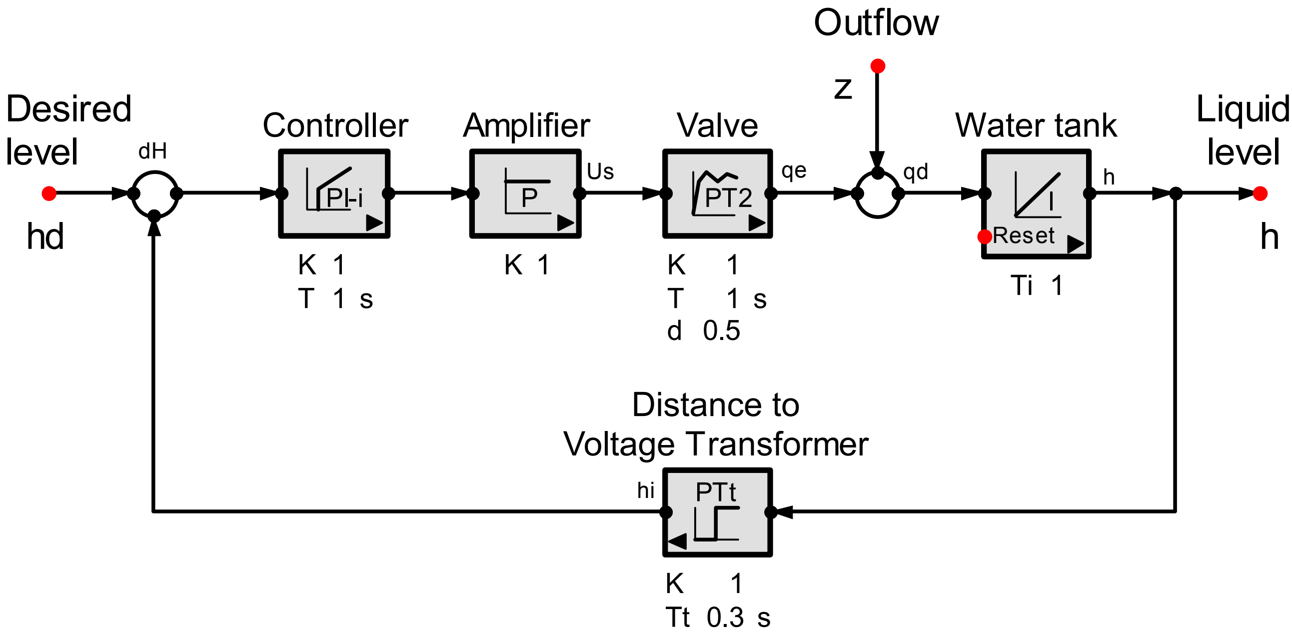


Figure 22: Labeling objects and signals

Remarks:

- Nodes of signal lines can not be labeled.
- The output node of a sum block can not be labeled. Use the block title instead.
- Bold text in the above figure is normal text and not a label.
- Rotate the PT_t-block by pressing or by the block's pop-up menu.

7 Frequency simulation

In the introductory example you learned the basic concepts of frequency simulations. In this chapter you will learn the use of frequency probes and the settings of the simulation control parameters.

7.1 System modeling

The first steps in frequency simulation are entering a block diagram and setting the block parameters. Note that you may only use linear elements or time discrete elements which are also linear elements.

Note: There are some elements with selectable transfer function in nonlinear pages of the palette. (e.g. arithmetic element, function element with one or two inputs). If you select a linear function you may use that element too.

7.2 Frequency probes

After modeling the system you can determine which subsystem to analyze.
The simplest case is a system with only one single input for which you are only interested in the input-output response. In this case, you do not need any probes and can just press the start button.
By means of frequency probes you can determine any subsystem within the system, even when the subsystem doesn't have its own inputs and outputs.

Frequency

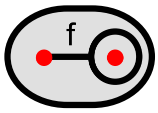


Figure 23: Frequency probe

Each frequency probe has two terminals. The left terminal is an output that feeds the system with a sinusoidal test signal. The terminal on the right is an input that can be connected to several system nodes. The input terminal of the probe is the measurement input. To connect probes with system nodes, you need signal lines. The output signal of the probe can be fed into the system at any node regardless of whether it is already driven by another output. SimApp deactivates all these outputs. You can use as many frequency probes as you like. For every stimulation input, you need one probe. All responses are displayed in the same diagrams and can easily be compared. The names of the probes appear in the legend and as column names in the data tables. SimApp internally runs a simulation for every single probe ignoring all other probes.

A common task is to analyze the open and closed loop of an automatic control system:

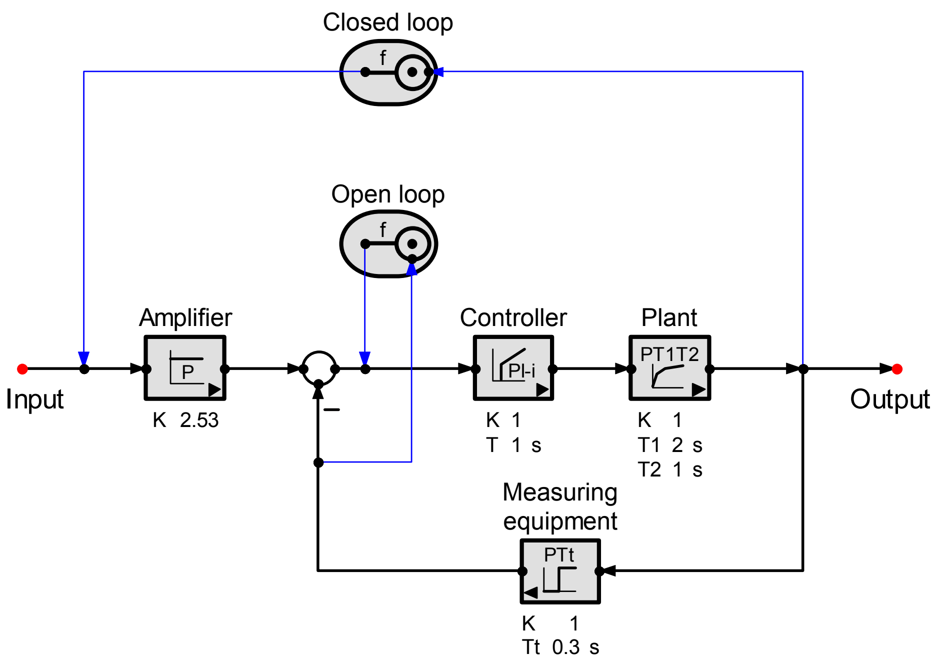


Figure 24: The use of frequency probes

Remarks:

- Inputs and outputs of the system consist of red nodes. If you find other red nodes this means that they are not connected to any other node. This could happen due to improper placing of nodes.
- The system must not have short circuits or node having no signal. If SimApp finds any errors, it paints all incorrect connections and signal lines yellow.
- For frequency simulations you may only use continuous time and linear elements incl. time discrete elements. Nonlinear elements are forbidden.
- Loops without any delay elements are allowed (in contrast to time simulation). If you also want to perform some time simulations with the same system, you should insert fictitious delay elements as PT1-blocks. For more information, refer to chapter *Time simulation*.
- If the system has one input only and one or more outputs, you don't need any frequency probes. For every output, a complete simulation is performed. Inputs with sources are not recognized by SimApp.
- For frequency simulations you don't need sources. If you still have some, they are omitted by SimApp, but they hide the associated input node.
- Label every probe with a unique name. This will help you to have a better overview of the results.

7.3 Simulation properties

You can control the simulation process by control parameters and options. Open the frequency simulation options dialog box (menu *Frequency* + *Simulation Properties...* or button or the drawing's pop-up menu).

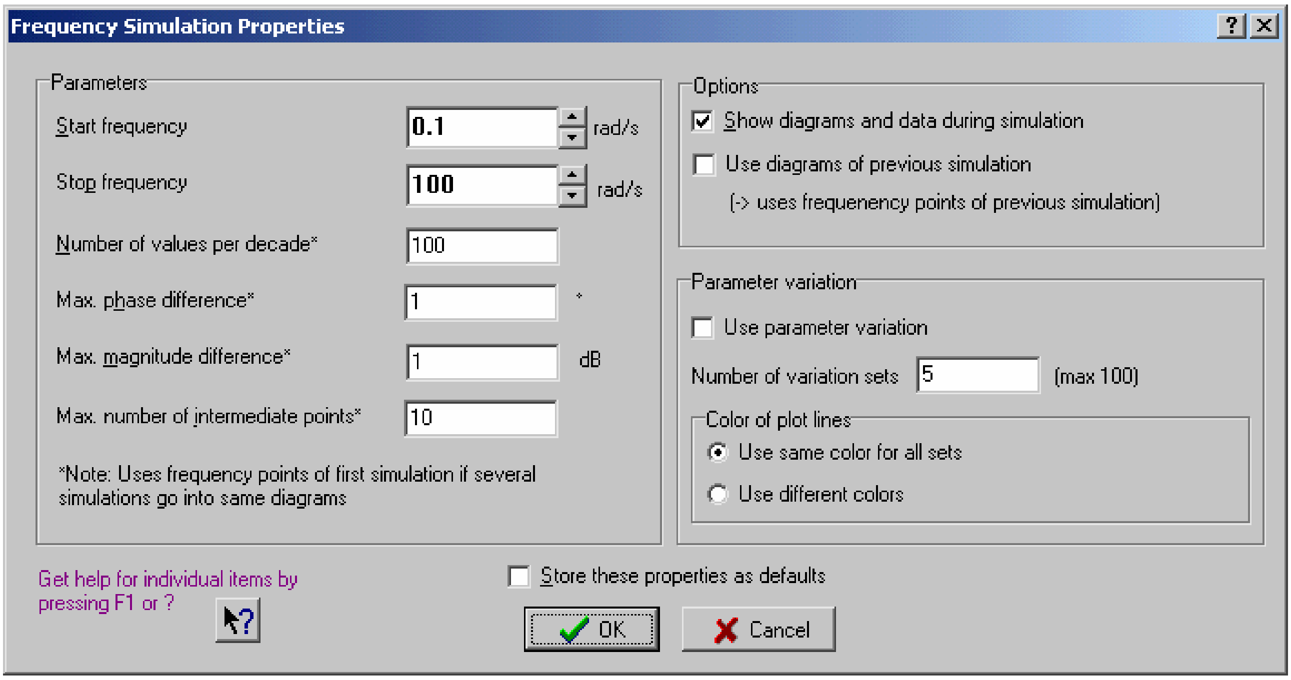


Figure 25: Parameters and options for frequency simulations

Enter a start and a stop frequency. Choose how many frequency instants per decade you want to calculate. Generally, the frequency response differs very much from decade to decade, especially the phase response. To get good resolution in these ranges, you could increase the number of values per decade. But this has the disadvantage of huge memory demands. In ranges where the response is very smooth, you don't need a high frequency resolution. Hence, before you increase the resolution try the following three parameters instead:
Select a smaller maximum phase and magnitude difference between two adjacent frequency instants and set the number of surplus intermediate points if the maximum differences are exceeded.

For more information use the contextual help (?). Refer to chapter *Parameters variation* for more information about parameter variation.

7.4 Start frequency simulation

Start simulation by using the menu command *Frequency + Start* or the start button. The calculation time depends on the size of the system and the simulation parameters. During a simulation you can work on other drawings and even run other simulations.

7.5 Results

7.5.1 Diagrams

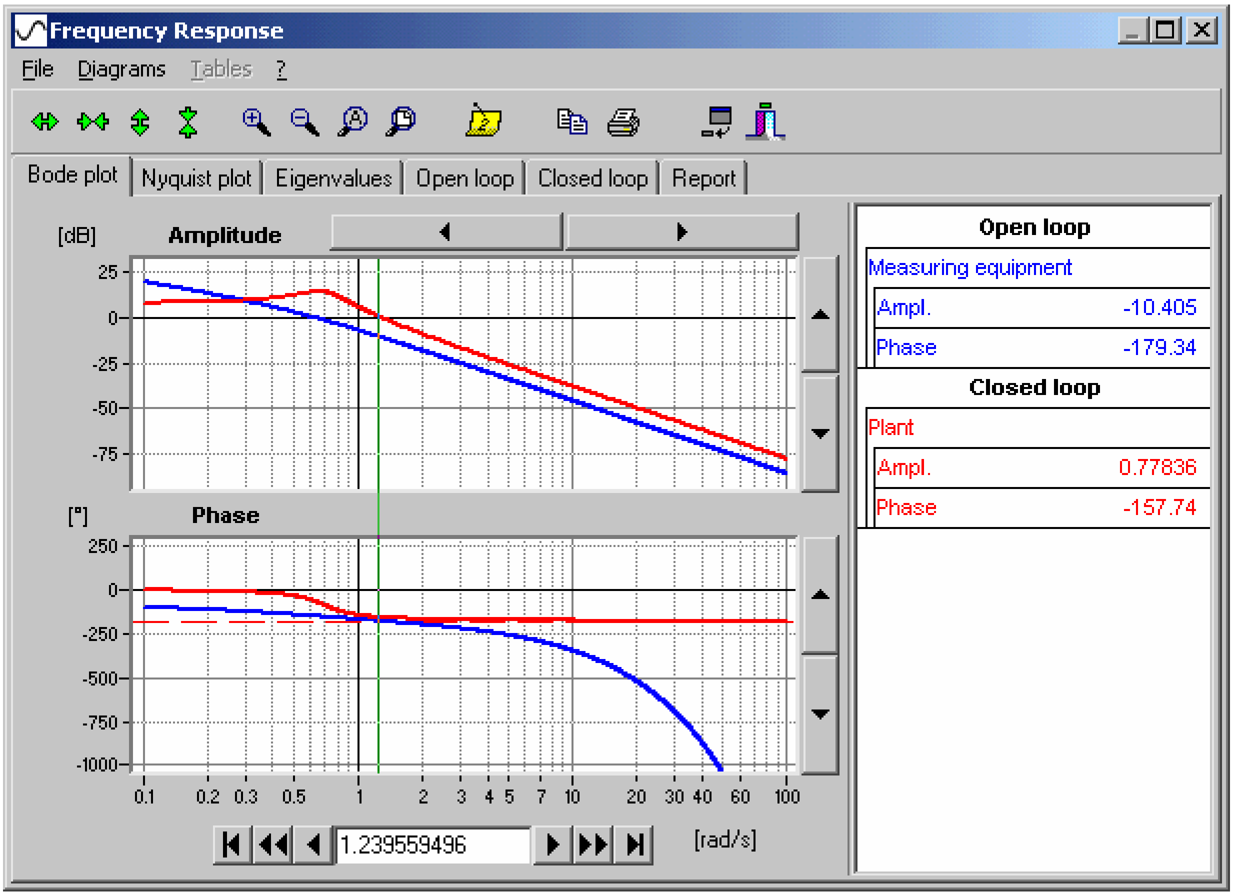


Figure 26: Bode diagram

A diagram can display several curves each having its own color and pop-up menu (right-click the curve) for display settings. Several curves can overlap. Bring a curve to the front by left-clicking the associated legend item.
The toolbar contains several tools for display manipulation. You can zoom, unzoom and shift a diagram. The tools always operate on the active diagram, which are selected by clicking. You can recognize the active diagram by the fine frame around its title.
The mouse pointer also serves as a zoom tool. Draw a frame around that region you want to zoom in. Unzoom by pressing the unzoom button in the toolbar.
The Bode diagram contains a measurement line and the Nyquist a measurement cursor for each curve. You can control the line and the cursors by the navigator at the bottom of the window. The line can also be moved by the mouse pointer. The legend items show the current values which correspond to the navigator position. Measurement line and cursors are always in step. The legend is hierarchically structured. There is a section in the legend for every probe.

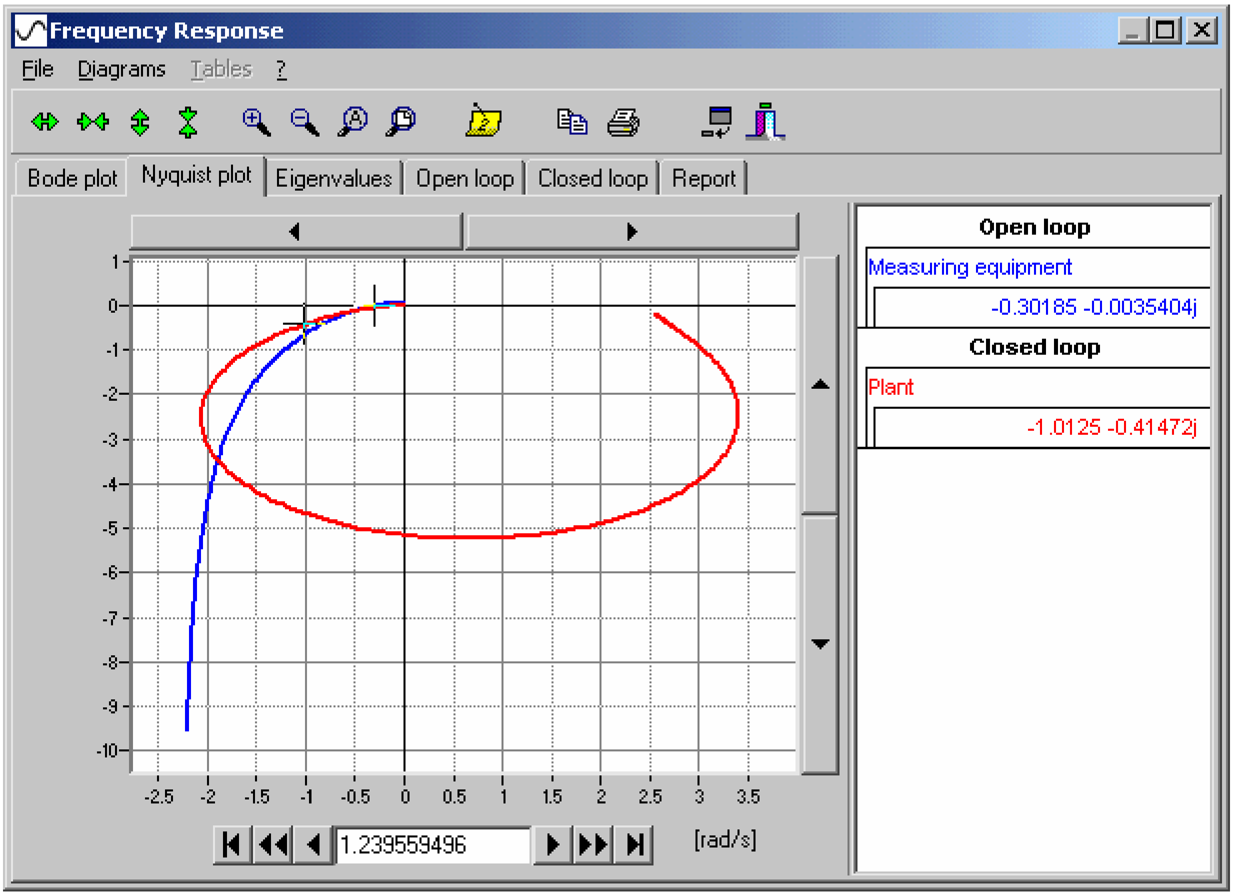


Figure 27: Nyquist diagram

7.5.2 Special effects

The frequency response is calculated numerically by the state-space description. This method has some special effects:

7.5.2.1 Constant phase errors

Phase calculation from the real and imaginary parts of the frequency response matrix uses the atan function [1]. The atan function has a periodicity and therefore an uncertainty of 360°. So the initial phase is always in the range of ±180°. If the real phase is not in this range, SimApp cannot recognize it and produces a constant phase error over the whole frequency range of a multiple of 360°. You can study this by simulating a chain of three integrators. SimApp shows a phase of 90° instead of –270°.

7.5.2.2 Phase garbage

If the phase drop is very fast, the phase difference between two frequency instants can exceed ±180°. As the atan-function has a periodicity of 360° (see above), SimApp produces erroneous phase values at every frequency instant and can even force the phase response in the wrong direction. But this effect can easily be recognized in the diagrams. It usually occurs if the systems contain time delay elements.

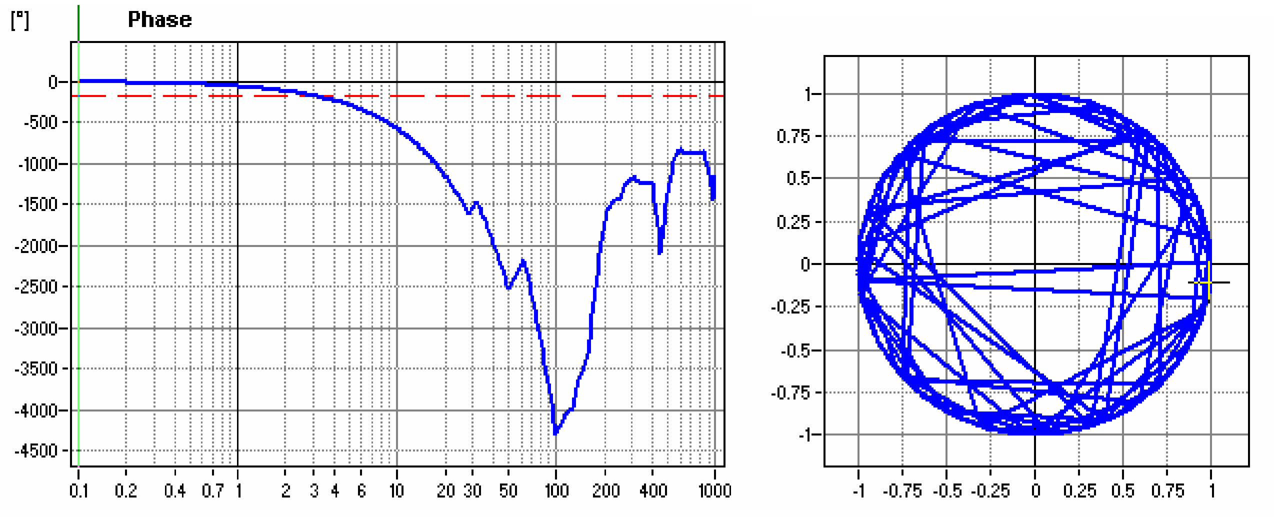
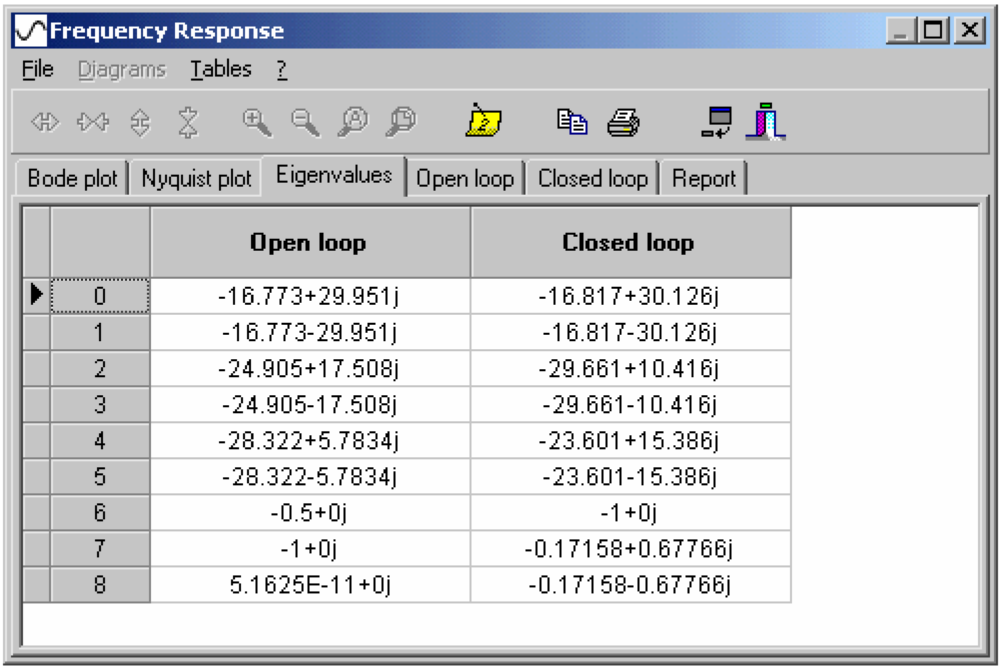


Figure 28: Erroneous phase response of a dead time element

In figure 31 you see the frequency response of a dead time block. In the frequency range above 70 rad/sec, the frequency response shows only garbage. The phase drop in the Bode diagram changes to a phase rise and the Nyquist diagram should show a perfect circle. You can reduce this effect by increasing the number of points per decade or the number of intermediate points if the phase difference is too high. Note however that this effect usually occurs at high frequencies and phase drops of more than 1000 to 2000° only, thus not in the interested frequency range. So you could eliminate this effect simply by reducing the stop frequency.

7.5.3 Eigenvalues

The eigenvalues are calculated for every subsystem (determined by a frequency probe) and displayed in a table.



	Open loop	Closed loop
0	-16.773+29.951j	-16.817+30.126j
1	-16.773-29.951j	-16.817-30.126j
2	-24.905+17.508j	-29.661+10.416j
3	-24.905-17.508j	-29.661-10.416j
4	-28.322+5.7834j	-23.601+15.386j
5	-28.322-5.7834j	-23.601-15.386j
6	-0.5+0j	-1+0j
7	-1+0j	-0.17158+0.67766j
8	5.1625E-11+0j	-0.17158-0.67766j

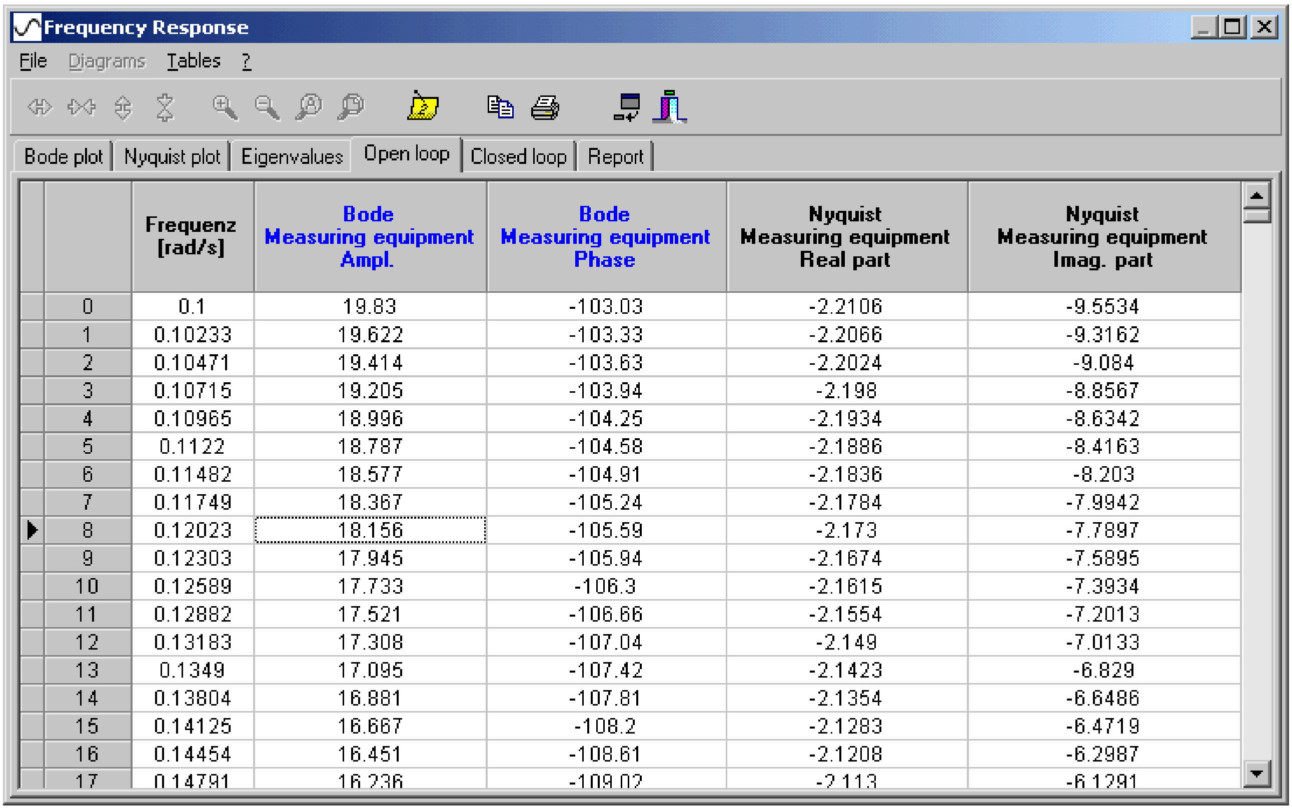
Figure 29: Eigenvalues

There is a column for every single probe.
Notes:

- The eigenvalues are only calculated for linear continuous time and time invariant systems that have no dead time elements. As a compromise you can approximate dead time elements by means of an appropriate Padé-approximation.
- Those system parts that have no effect on the output are omitted and not used for calculation. If you have several outputs, the eigenvalues are valid for those subsystems that contain all other subsystems. If you need the eigenvalues for a single subsystem, you may connect a frequency probe only with the input and output of that subsystem.

7.5.4 Data table

There is a data table for every frequency probe. This table contains all data for the Bode and Nyquist diagrams. Note: The frequency column is usually different for every probe because the various frequency responses do not need the same amount of intermediate frequency instants.



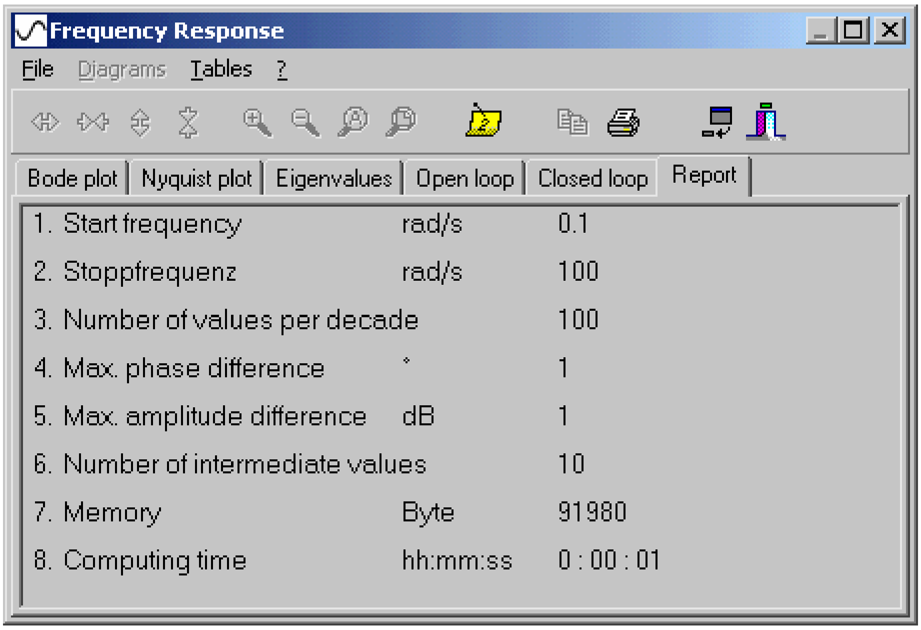
	Frequenz [rad/s]	Bode Measuring equipment Ampl.	Bode Measuring equipment Phase	Nyquist Measuring equipment Real part	Nyquist Measuring equipment Imag. part
0	0.1	19.83	-103.03	-2.2106	-9.5534
1	0.10233	19.622	-103.33	-2.2066	-9.3162
2	0.10471	19.414	-103.63	-2.2024	-9.084
3	0.10715	19.205	-103.94	-2.198	-8.8567
4	0.10965	18.996	-104.25	-2.1934	-8.6342
5	0.1122	18.787	-104.58	-2.1886	-8.4163
6	0.11482	18.577	-104.91	-2.1836	-8.203
7	0.11749	18.367	-105.24	-2.1784	-7.9942
8	0.12023	18.156	-105.59	-2.173	-7.7897
9	0.12303	17.945	-105.94	-2.1674	-7.5895
10	0.12589	17.733	-106.3	-2.1615	-7.3934
11	0.12882	17.521	-106.66	-2.1554	-7.2013
12	0.13183	17.308	-107.04	-2.149	-7.0133
13	0.1349	17.095	-107.42	-2.1423	-6.829
14	0.13804	16.881	-107.81	-2.1354	-6.6486
15	0.14125	16.667	-108.2	-2.1283	-6.4719
16	0.14454	16.451	-108.61	-2.1208	-6.2987
17	0.14791	16.236	-109.02	-2.113	-6.1291

Figure 30: Data table

Data tables can be copied as a whole or by selecting rows or columns. The data in the frequency column is always taken even if it is not selected.

7.5.5 Report

The report shows the control parameters and some statistical data of the current simulation.



1. Start frequency	rad/s	0.1
2. Stoppfrequenz	rad/s	100
3. Number of values per decade		100
4. Max. phase difference	°	1
5. Max. amplitude difference	dB	1
6. Number of intermediate values		10
7. Memory	Byte	91980
8. Computing time	hh:mm:ss	0 : 00 : 01

Figure 31: Report

8 Time simulation

In the introductory example you learned the basic concepts of time simulations. In this chapter, you will learn the use of time probes and XY graphs and how to set the simulation control parameters.

8.1 System modeling

The first steps in time simulation are entering a block diagram and setting the block parameters. In contrast to the frequency simulation, you are no longer restricted to pure linear elements. You may use now any element from the palette.

8.2 Insertion of signal sources and use of time probes

For time simulation you need sources which stimulate the system and time probes or XY graphs (see next paragraph) which measure the signals at various nodes.
For frequency simulations, probes also served as sinusoidal signal sources. For time simulations however, you need more signal forms and real systems usually have more than one signal source.
Time probes have one input terminal that can be connected in the same time to several nodes in the system.

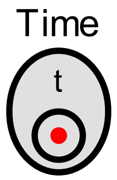


Figure 32: Time probe

Use signal lines to connect system nodes to time probes. Signal lines which connect probes to the system change their graphical representation. Open inputs (red nodes, not in custom blocks) are valid but are fed by zero. Generally, you have one source of main system stimulation as reference input and some optional sources as disturbance inputs.
If no probe is available, all outputs are led to a fictive probe. If you have at least one probe, all outputs which are not connected to probes are omitted.

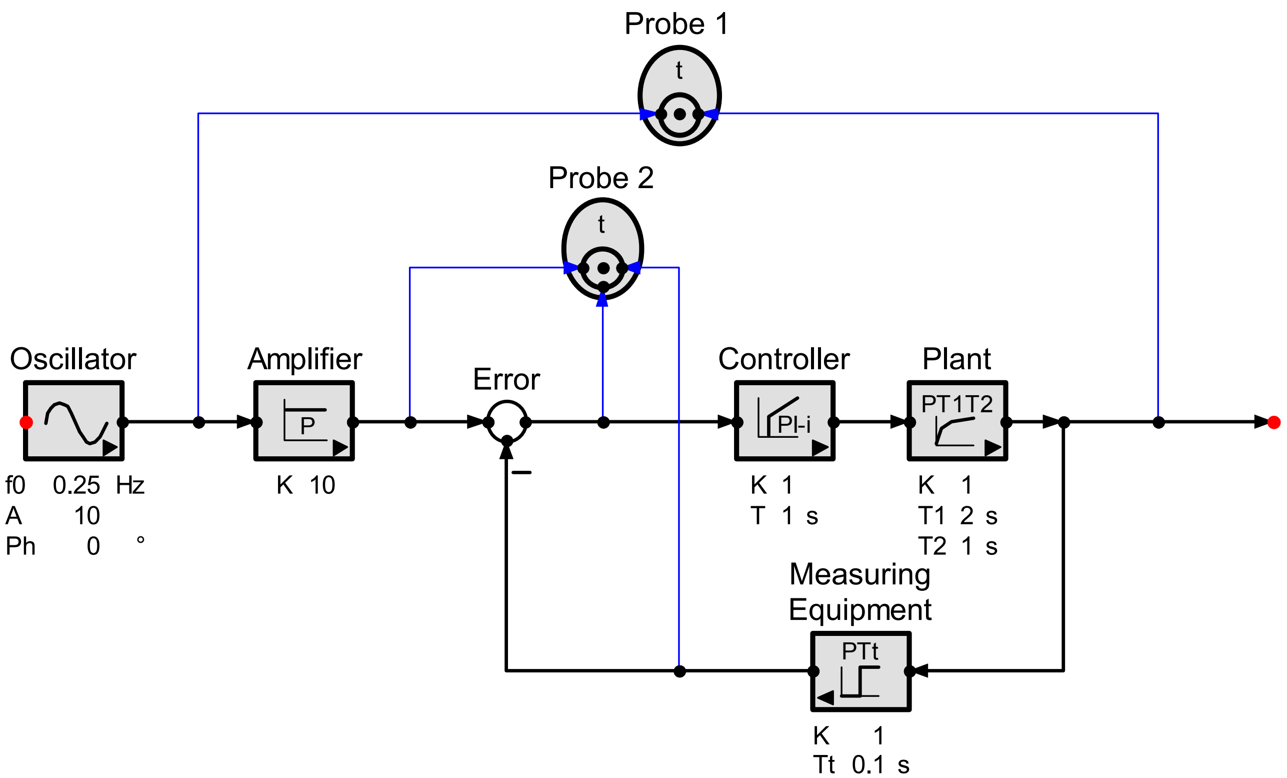


Figure 33: Using time probes

Time probes can be grouped. (Do not confuse with drawing groups). The membership of a source to a distinct group is defined by its group number. The group number is displayed only if it is greater than zero.

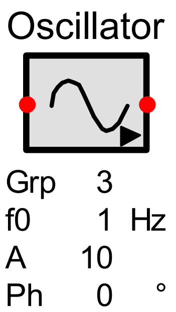


Figure 34: Source with group number (Grp)

SimApp runs one simulation for every single group. Sources not belonging to the current group are switched off. The time responses of all groups are displayed in the same diagram so that you can stimulate your system in different ways and compare the results.

The sources of group 0 are special. These sources are used by all groups. For example, you may use a source from group 0 as reference input and two other sources as disturbance inputs. SimApp will perform two simulations, first with disturbance 1 and then with disturbance 2 and display the results in the same diagrams.

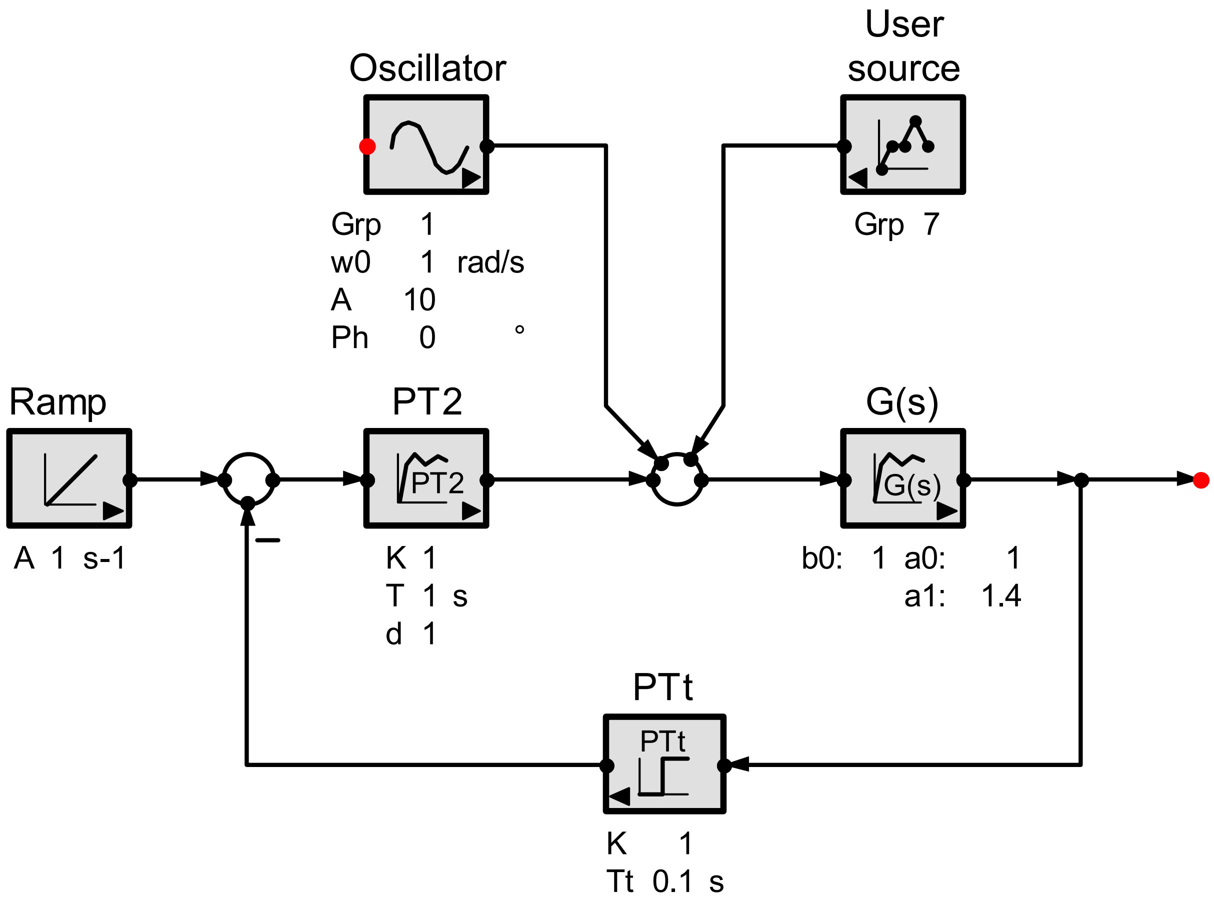


Figure 35: Example for source groups

Remarks:

- Inputs and outputs of the system are red marked. If you find other nodes which are also red marked, this means, that they are not connected to any other node. This could happen due to improper placing of nodes.
- The system must not have short circuits or nodes having no signal. If SimApp finds any errors, incorrect connections and signal lines are yellow marked.
- System parts which have no effect on measured outputs are not included in the simulation process.
- Rigid feedback loops without time delays are, in contrast to frequency simulation, not allowed. In such cases, insert a fictive PT1-element in the loop and set the time constant so small that the system behavior doesn't change. This means that you have to shorten the integration interval as well (approx. 1/10 of the shortest time constant in the system).

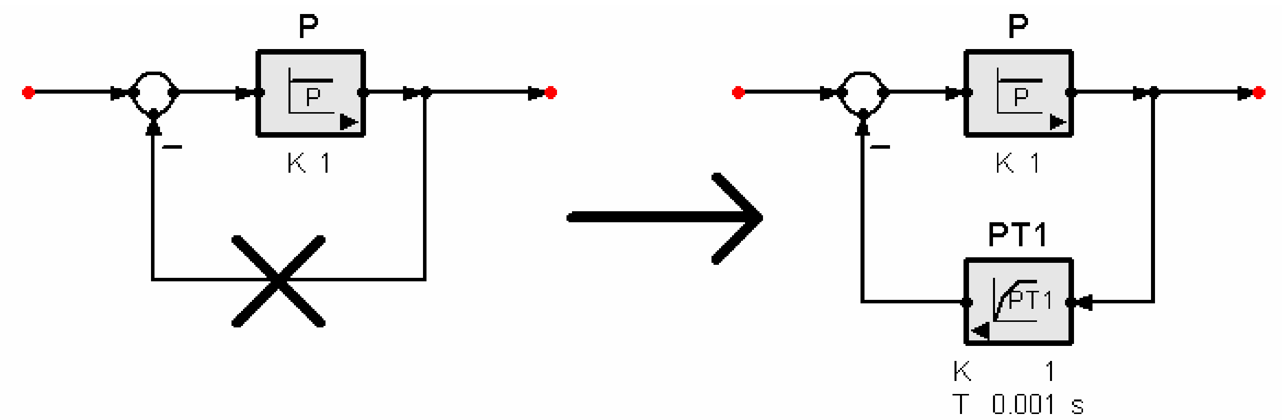


Figure 36: Avoid feedback loops without delays

8.3 XY Graphs

XY graphs are suitable for two-dimensional data representations. You may use time probes and XY graphs simultaneously. The results are displayed in the same data window but in different diagrams.

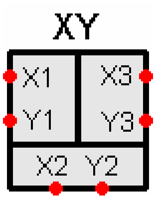


Figure 37: XY Graph

The XY graph plots data in the x input against data in the y input. It has three input channels for displaying up to three X-Y plots simultaneously. Unused channels can simply be left unconnected.
The plots can be printed out. Time information is printed as labeled dots.

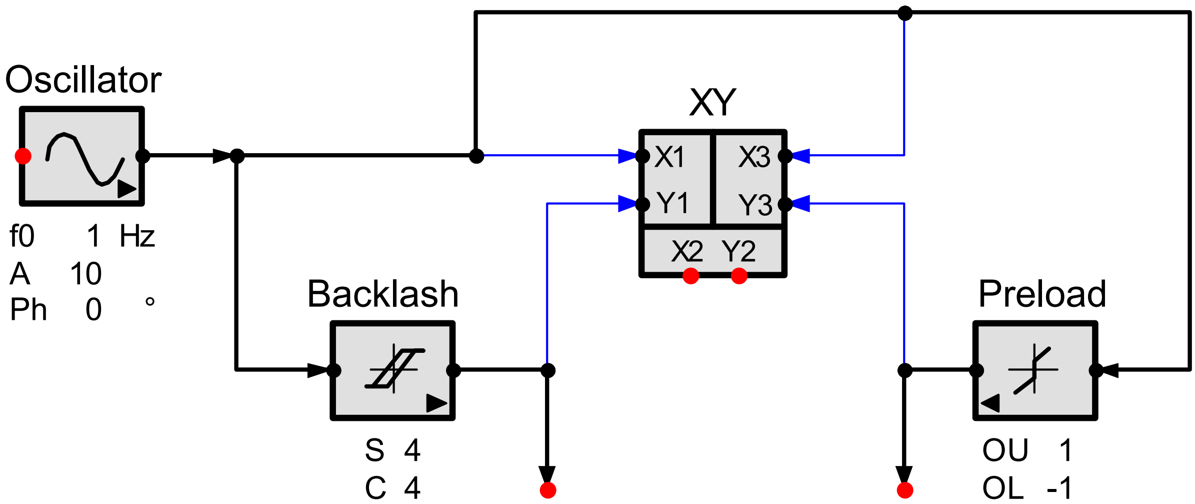


Figure 38: Comparing backlash and preload characteristics by using a XY graph element

8.4 Simulation Properties

The simulation process is controlled by several parameters and options. Open the associated dialog box by selecting then menu item *Time + Simulation properties...*, pressing button or opening the drawing's pop-up menu.

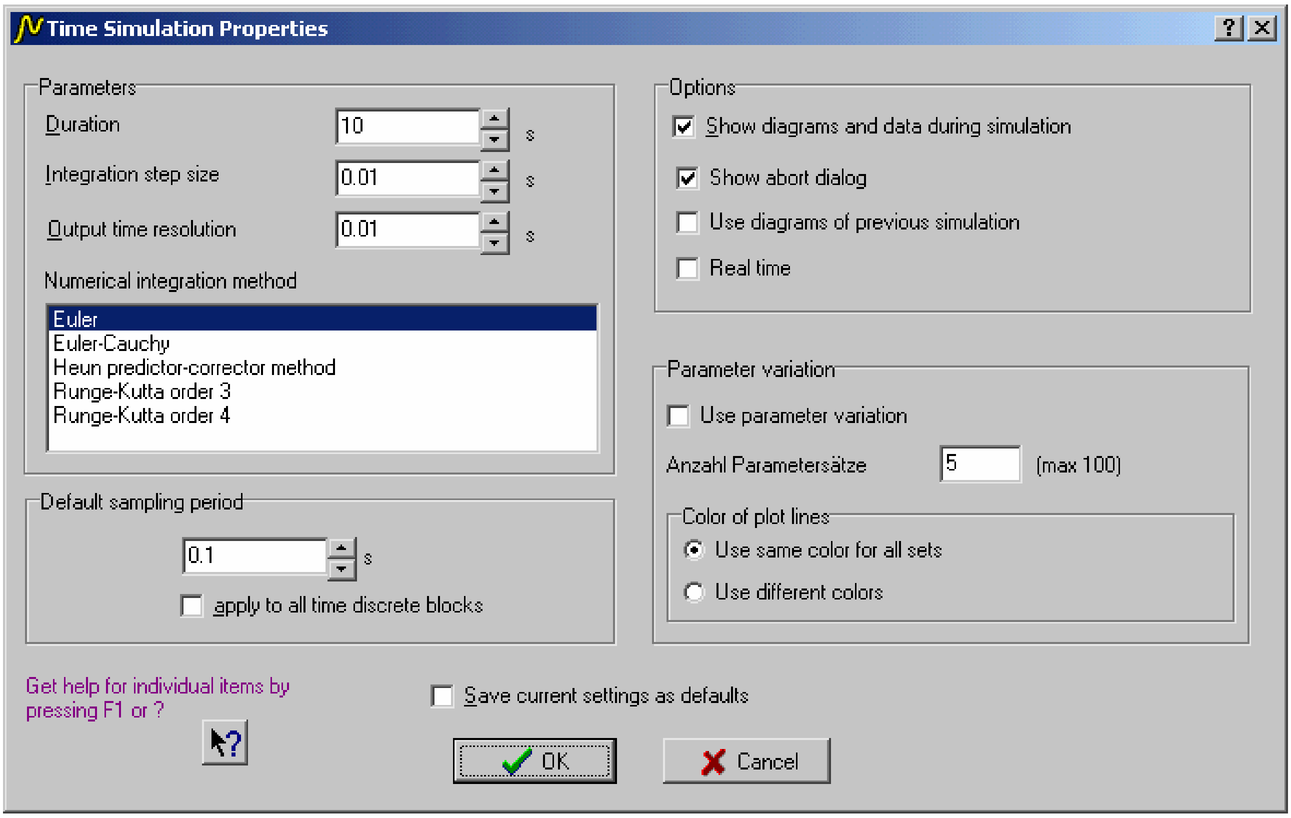


Figure 39: Parameters and options for time simulation

Enter a simulation duration. The simulation always begins at time 0.

Numerical Integration:
The simulation is made by numerical integration of differential equations as defined by the block diagram. The simulation time is divided into short time intervals of equal length. During these intervals the actual solution curve is approximated with straight line segments. The manner of computing the segments varies from method to method. The accuracy depends on the method and the interval. Smaller intervals and the use of complex approximations will lead to more accuracy but results in longer computing times.

The optimal interval depends solely on the simulated system. Difficult systems tending to instabilities may need some attempts to find out the best integration method and interval. As a general rule, follow these guidelines:

- Use an interval that is smaller than the reaction time of the system. For good results use an interval that is 10 times smaller than the shortest time constant in the system. You can quickly estimate a rough value if you know the field the system comes from. In heating plants you have time constants of seconds up to hours. In motors they are in the range of micro- to milliseconds.
- The delay time of dead time elements must be exactly a multiple of that interval. If this is not true, the actual dead time can vary by 1 interval in both directions. If the dead time is less than half the interval, it disappears.
- In the first place, select the method of Euler. It is a simple numerical method and leads to short calculation times. If the system tends towards numerical instability, choose the Runge-Kutta fourth-order method and shorten the interval. You can recognize numerical instability if the variation of the integration method and integration interval leads to very different solutions.
- Step-like signals can cause inaccuracy and distortions with high order integration methods. Use the Euler methods and short intervals instead.

The quality of the display of the solution curve is determined by the time resolution which has no impact on the accuracy. The time resolution defines the interval the computed values are stored in memory and displayed in tables and plots. Generally you need fewer values for displaying than for the simulation process. In general, use a resolution interval 10 times smaller than the integration interval.

For time discrete elements you can set the sampling period individually. But usually, the sampling period is unique in a system. In this case, you are more efficient if you set the sampling period in this dialog box globally.

8.5 Starting time simulation

Start time simulation by menu item *Time + Start* by button or by the drawing's pop-up menu..

8.6 Simulation results

8.6.1 Time diagram

The time diagram displays all measured signals from one probe. Every probe and xy graph has its own diagram.

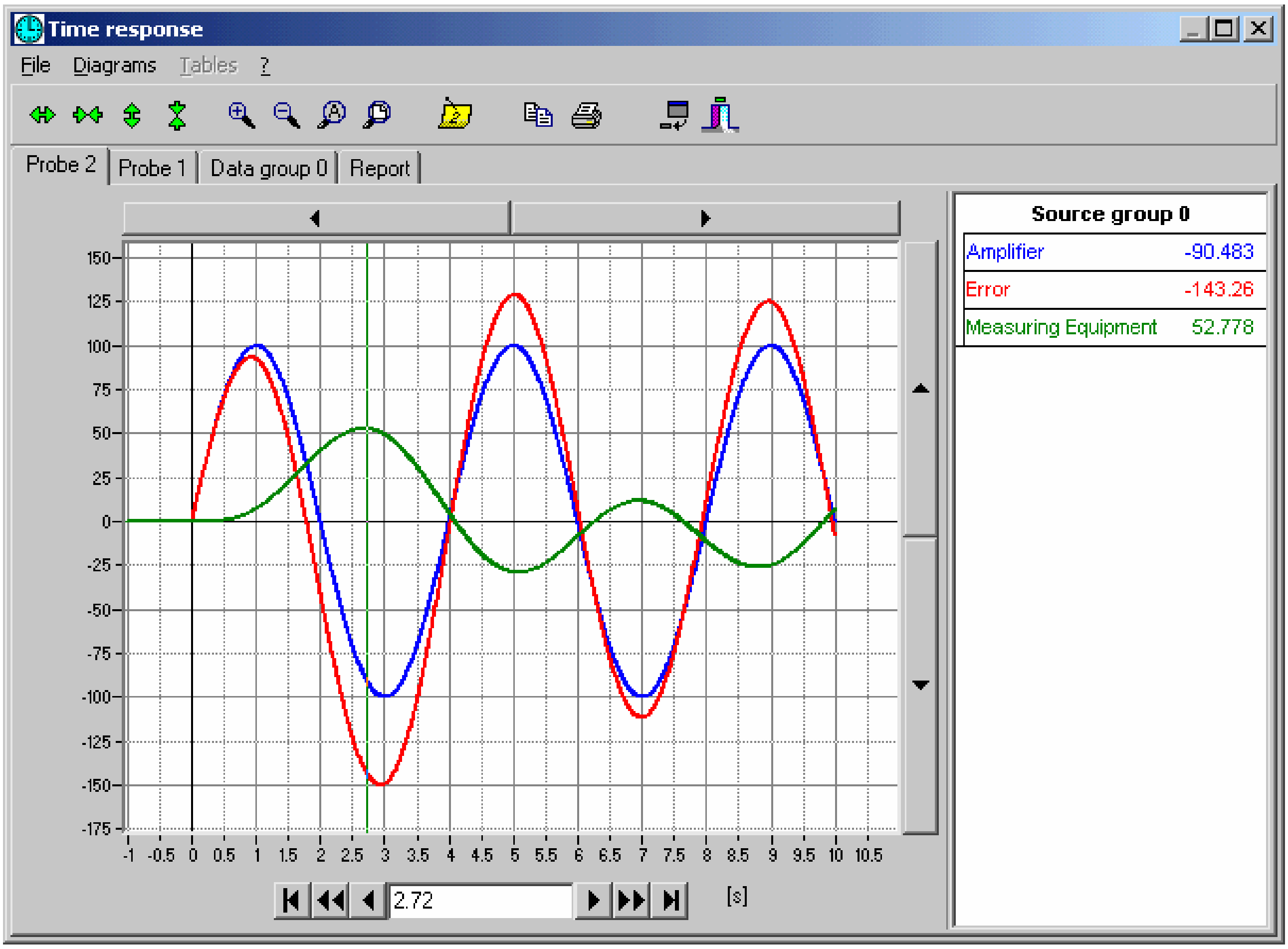
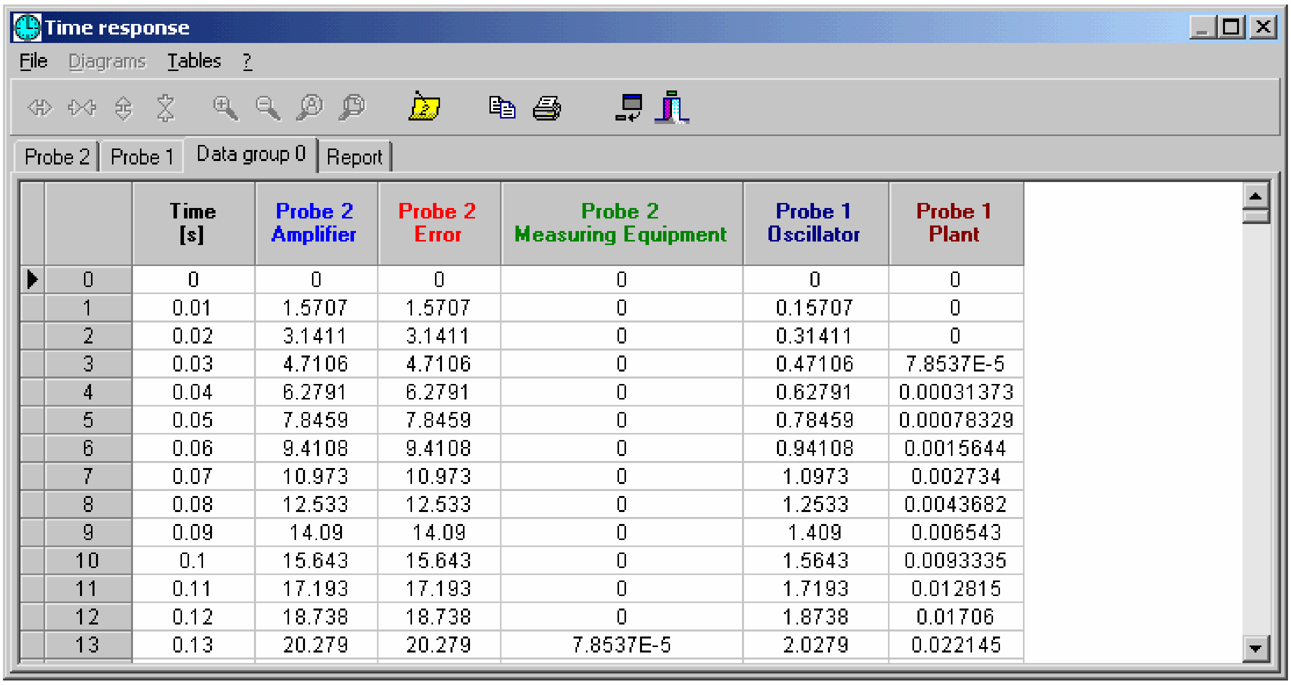


Figure 40: Time diagram

The legend is structured hierarchically. Every source group has its own section.

8.6.2 Data tables

Every source group creates a data table. The table can be copied - in part or in whole - by selecting appropriate sections. The time column is always included even when not selected.

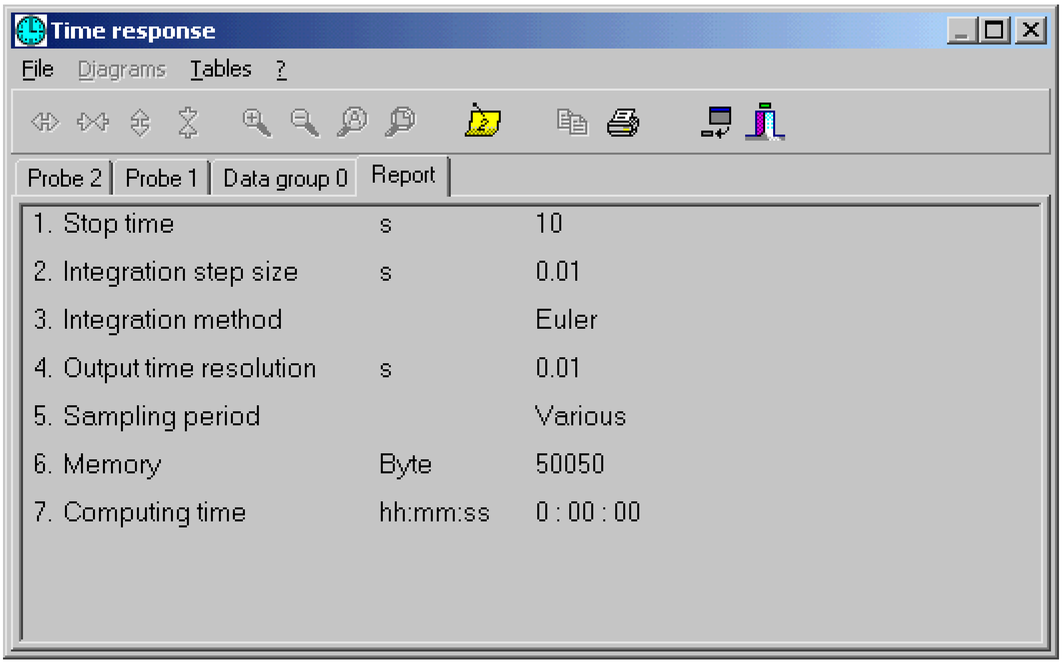


	Time [s]	Probe 2 Amplifier	Probe 2 Error	Probe 2 Measuring Equipment	Probe 1 Oscillator	Probe 1 Plant
0	0	0	0	0	0	0
1	0.01	1.5707	1.5707	0	0.15707	0
2	0.02	3.1411	3.1411	0	0.31411	0
3	0.03	4.7106	4.7106	0	0.47106	7.8537E-5
4	0.04	6.2791	6.2791	0	0.62791	0.00031373
5	0.05	7.8459	7.8459	0	0.78459	0.00078329
6	0.06	9.4108	9.4108	0	0.94108	0.0015644
7	0.07	10.973	10.973	0	1.0973	0.002734
8	0.08	12.533	12.533	0	1.2533	0.0043682
9	0.09	14.09	14.09	0	1.409	0.006543
10	0.1	15.643	15.643	0	1.5643	0.0093335
11	0.11	17.193	17.193	0	1.7193	0.012815
12	0.12	18.738	18.738	0	1.8738	0.01706
13	0.13	20.279	20.279	7.8537E-5	2.0279	0.022145

Figure 41: Data table

8.6.3 Report

The report shows the simulation parameters and some statistical values.



1. Stop time	s	10
2. Integration step size	s	0.01
3. Integration method		Euler
4. Output time resolution	s	0.01
5. Sampling period		Various
6. Memory	Byte	50050
7. Computing time	hh:mm:ss	0 : 00 : 00

Figure 42: Report

8.7 Numerical solutions of differential equations

The following example shows how to solve differential equations in SimApp.

A differential equation and the initial value of the solution curve are given as

$$y' = -2 \, x \, y^2, \qquad y(0) = 1$$

Compute the approximation of the solution curve with different integration methods and compare the results.

First draw the block diagram that represents the differential equation:

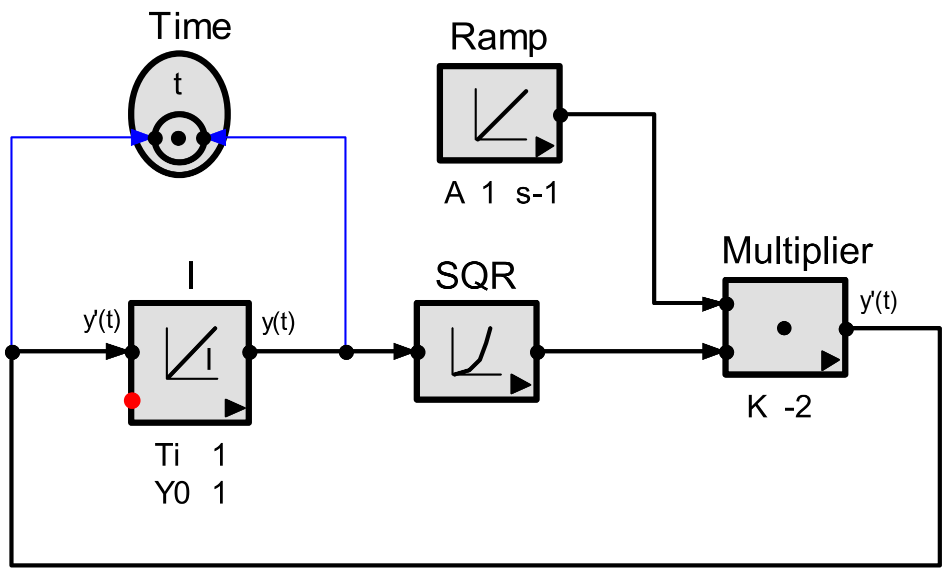


Figure 43: Block diagram for the differential equation y' = -2xy

Calculate for each integration method the first 10 values of the solution curve with an integration interval of h = 0.1. Copy the values into an Excel or a Word table and compare the results with the exact solution $y(x) = 1 / (x^2 + 1)$.

x	exact	Euler	Euler modified	Heun	Runge-Kutta 3. order	Runge-Kutta 4. order
0	1	1	1	1	1	1
0.1	0.9901	1	0.99	0.99	0.99013	0.9901
0.2	0.96154	0.98	0.96118	0.96137	0.9616	0.96154
0.3	0.91743	0.94158	0.91674	0.91725	0.91751	0.91743
0.4	0.86207	0.88839	0.8611	0.86195	0.86216	0.86207
0.5	0.8	0.82525	0.79889	0.80003	0.80009	0.8
0.6	0.73529	0.75715	0.73418	0.73553	0.73537	0.73529
0.7	0.67114	0.68835	0.67014	0.67159	0.6712	0.67114
0.8	0.609756	0.62202	0.60895	0.6104	0.6098	0.60976
0.9	0.552486	0.56011	0.55191	0.55329	0.55252	0.55249
1	0.5	0.50364	0.49964	0.50092	0.50002	0.5

Since the function has no steps, the Runge-Kutta 4. order method has the best performance at all times. For a comprehensive discussion about numerical methods refer to [6].

9 Parameters variation

By means of the parameter variation you can examine the system behavior with varying parameter values fast and easily. Each parameter can take up to 100 different values. SimApp automatically runs a simulation for each value and displays the results in one single output window.

If one or more parameters of an element contain varying values, one can speak of parameter sets. For example all i-th values of the parameters of an element form the i-th parameter set of that element. All i-th parameter sets of all elements of a system form the i-th parameter set of that system.

The number of parameter sets for elements can be individually set. The number of parameter sets used for the simulation is set in the simulation properties of the time and frequency simulation. But the number of parameter sets in the elements may be different.

The default parameter value, displayed below the block symbol in the drawing or in the block properties dialog, generally doesn't contribute to the parameter variation. SimApp uses additional values for that feature.

An alternative to parameter variation would be to run several simulations with different parameter values each and display all results in the same diagram (see option in the simulation properties). The disadvantage of this procedure however is the badly arranged curves in the diagrams and the impossibility to store all the values in the file.
By using parameter variation the default parameter values are not changed and therefore don't have to be noted for re-use.

9.1 Properties of parameter variation

The properties of the parameter variation cannot be set in one single property sheet. It depends whether the setting concerns a single element or the system as a whole.

9.1.1 Block property dialog

In the element's property sheet you can set the number of parameter sets (of that element only), the individual values and designate those parameters which participate in the parameter variation.

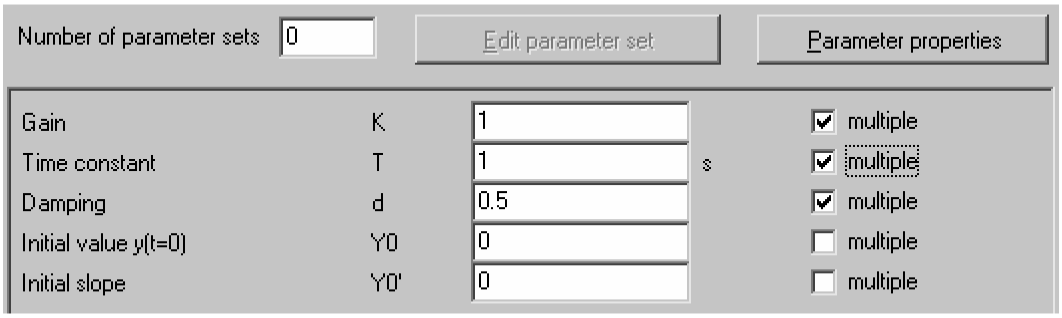


Figure 44: Parameters variation settings

The following settings are important for the parameter variation:

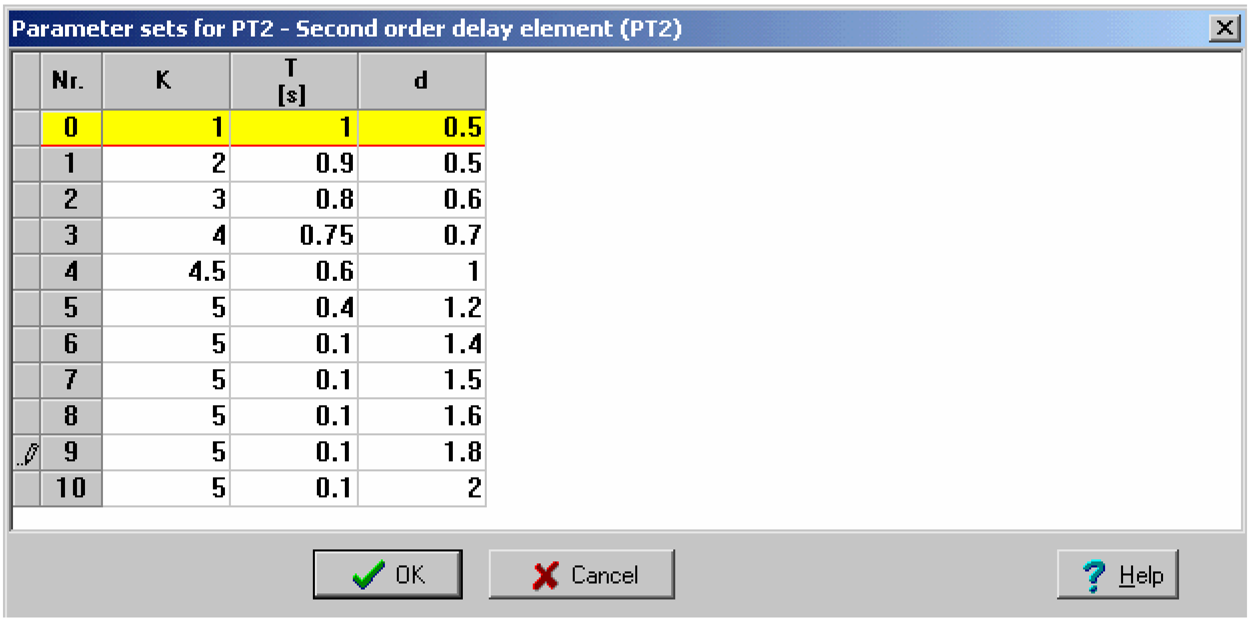
multiple
Must be checked for all parameters contributing to parameter variation.

Number of parameter sets
Denotes the number of additional values a parameter can store for parameter variation. It affects a parameter only if the check box *multiple* is checked at the same time.

Edit parameter set
This button opens a input table for editing the variable values of all parameters of the current block. Each parameter whose multiple check box is checked gets a column.

All parameters contributing to the parameter variation are red marked in the drawing.

9.1.2 Input table for variable parameter values



Nr.	K	T [s]	d
0	1	1	0.5
1	2	0.9	0.5
2	3	0.8	0.6
3	4	0.75	0.7
4	4.5	0.6	1
5	5	0.4	1.2
6	5	0.1	1.4
7	5	0.1	1.5
8	5	0.1	1.6
9	5	0.1	1.8
10	5	0.1	2

Figure 45: Input table for varying parameter values

The parameters default values appear in the first yellow marked line and cannot be modified as they serve for better overview only. In line 1 follows the first parameter set. Parameters not contributing to parameter variation do not appear in that table and provide their default parameter values.

9.1.3 Simulation properties

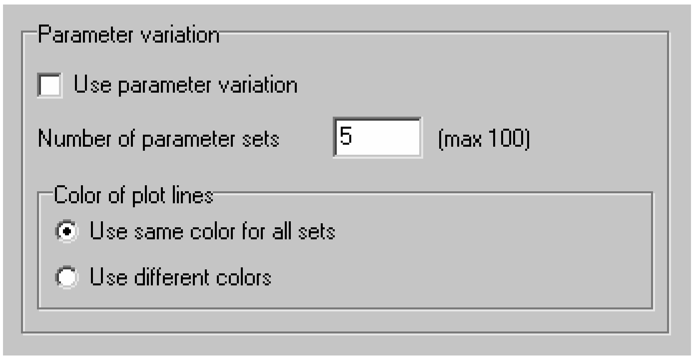


Figure 46: Settings for parameter variation in the frequency and time simulation properties

Use parameter variation must be checked.

Number of parameter sets denotes the number of parameter sets used in the next simulation run. The number of parameter sets in the individual elements may not equal this value. The elements may have more or less sets.
For those elements having more parameter sets, all surplus sets are omitted.
For those elements having less parameter sets, the last existing set is used for the missing sets.
For elements having no parameter sets at all, the default parameter values are used instead.

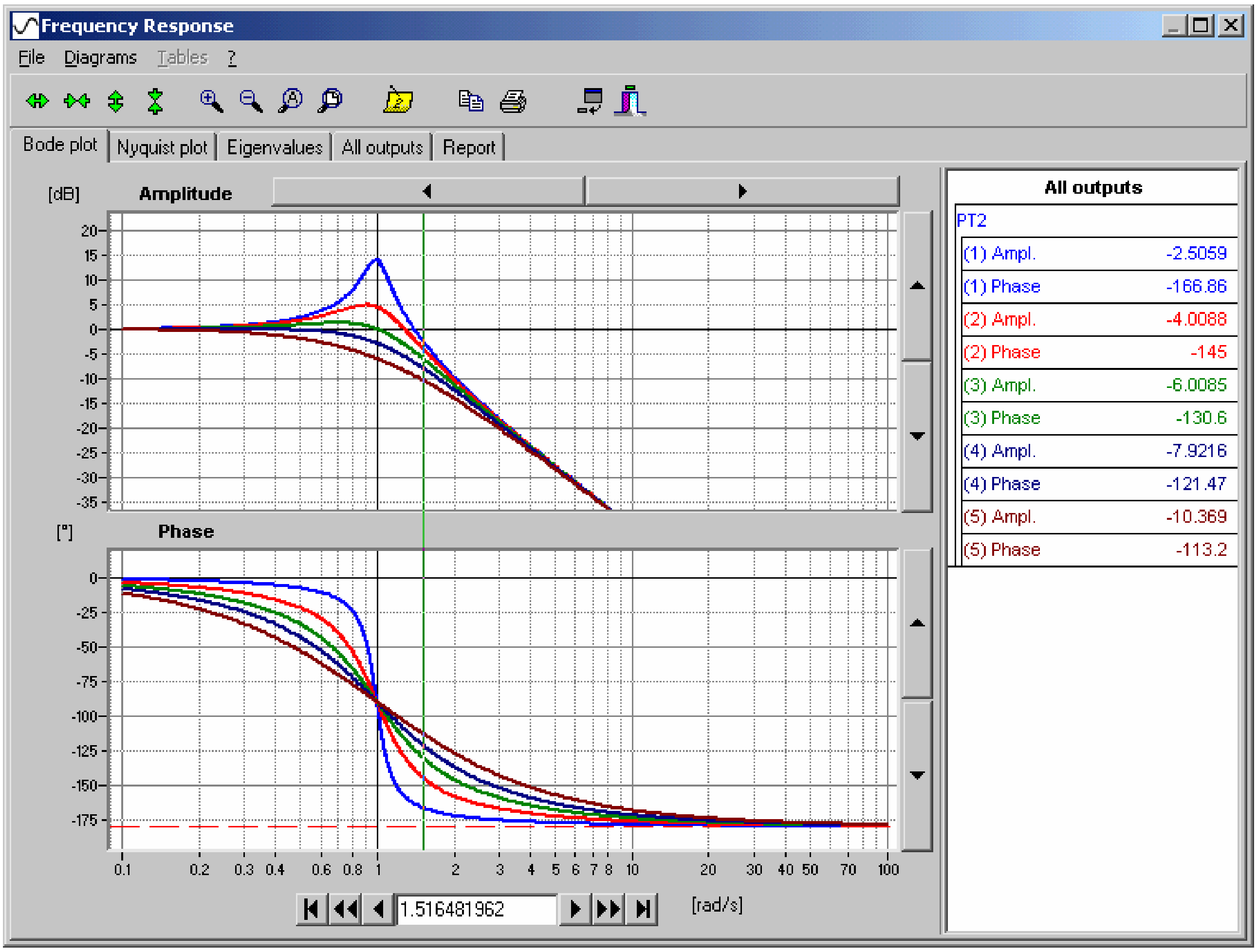
You can give different settings for time and the frequency simulations.

9.2 Starting parameter variation

Start parameter variation as you would do for normal simulations. Parameters variation is performed only, if the check box *Use parameter variation* in the simulation properties is checked. SimApp runs a simulation for each parameter set and presents the results in the same output window.

Every legend entry is preceded by the index of the parameter set. The colors can be controlled in the simulation properties.

Parameters variation



10 Custom blocks

Large and complex systems can be difficult to survey. But such systems mostly consist of several subsystems. You could graphically separate subsystems by drawing lines or using colored planes and include descriptive text, but this would enlarge the drawings even more. Generally, it is not necessary to show all the details of a system at the same time. Furthermore, there are some well known structures on all fields that can be used again and again but with changed parameter values.

For this purpose, in SimApp you can create your own blocks. You can draw any system or subsystem and pack it into a new block with its own parameters. This is not even restricted to block diagrams. You can put any kind of graphical representation you draw with SimApp into custom block. User-defined blocks have own block symbols and don't differ from basic blocks. You can store them to the palette or collect them in libraries.

There are two methods of creating higher integrated blocks:
By the first method you can select all objects in a drawing and gather them into a single block. By the second method you use a special workshop in SimApp, the block folder.
The first method is very fast and is suitable for temporarily simplifying a drawing without any need for re-use. The second method is more costly, but has no restrictions and is suitable for long-term solutions.

10.1 Creating custom blocks by selection

You can select any object in a drawing and put it into a new block. After selection, use the menu command Extras + Create custom block. All selected objects are immediately replaced by the new block. It has a default symbol and looks like a basic block.

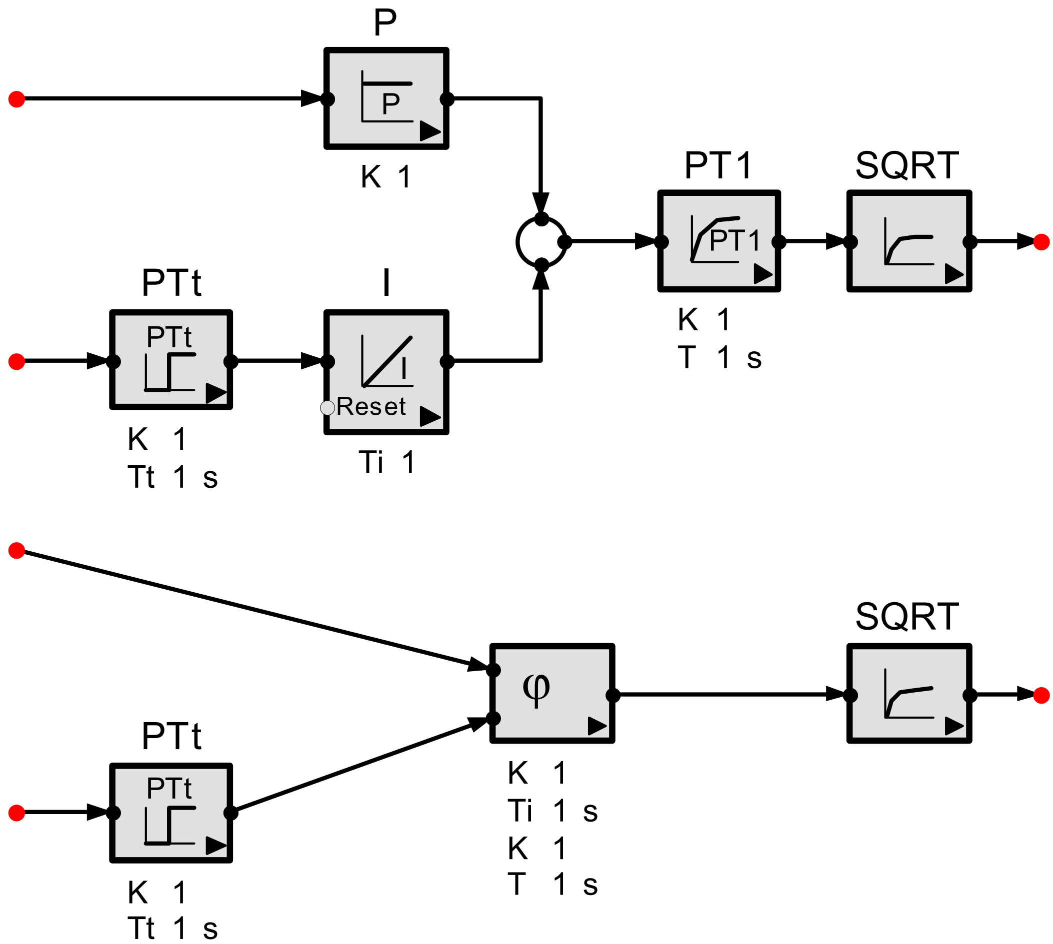


Figure 47: Create custom block by selection

The input and output nodes result from the connections to non-selected elements or are open nodes of the selected objects. The connections to external elements are not broken.

Remarks
- Do not select signal lines to non-selected blocks. This will break up connections.
- If you select adding elements, select input and output signal lines as well.
- Disable open control input nodes of individual elements to prevent them from appearing in the new block.
- Use the block title to give the block a descriptive name.
- All parameters of the contained elements appear below the new block and behave like native parameters. You can change their properties as you do in basic blocks.
- The parameters which you don't want to be changeable must be hidden in advance. These parameters act as constants and do not appear on the drawing or any property dialog of the new block.
- Parameters of grouped elements are hidden.

If you don't like the default symbol, you can finish the block in the block folder. In the block folder you can also edit the inner structure and create new parameters.

Custom blocks can be broken up with the menu command *Extras* + *Break up custom block* at any time. It is immediately replaced by its inner structure. The symbol gets lost, but connections to external blocks are retained.

10.2 Creating custom blocks in the block folder

In a block folder you can create and alter custom blocks. You have access to the inner structure and the block symbol. A special feature is the ability to create new virtual parameters and connect them to the real parameters of the contained blocks.

10.2.1 Introductory example

In this example you will learn the most important steps to create a custom block in the block folder. The lesson is to create a custom block from the block diagram of a DC motor. This block should be supplied with a descriptive symbol.

10.2.1.1 Mathematical modeling

The laplaced transformed equations of a DC motor are [1],[21]:

Armature current: $I_A = \frac{K_A}{1+sT_A} U_A$ $\quad K_A = \frac{1}{R_A}$ and $T_A = \frac{L_A}{R_A}$

Differential voltage: $U_R = U_A - e_m$ $\quad U_A$ = Armature voltage

Back-emf: $e_m = K_F \omega$ $\quad K_F$ = Torque constant

Motor torque: $M_A = K_F I_A$

Differential torque: $M_B = M_A - M_L$ $\quad M_L$ = load torque

Angular velocity: $\omega = \frac{1}{Js} M_B$ $\quad$ J = Inertia of motor and load

Revolutions per minute (r.p.m): $n = 60\omega/2\pi$

The equations describe the function of the new block. It has three nodes:
- Inputs: Armature voltage U_A and load torque M_L as disturbance input.
- Output: r.p.m

To make the block usable for different designs of DC motors, the following coefficients must be changeable:

Name	Symbol	Unit
Armature inductance	La	H
Armature resistance	Ra	Ohm
Back-emf constant	Kf	Nm/A
Inertia of motor and load	J	kgm2

These coefficients forms the new virtual parameters of the block.

10.2.1.2 Open block folder

Open the block folder (menu *File +New block folder)*. A new window appears with the title "Block folder 1". The window consists of two pages and a table. The first page is for the design of the block symbol and the second page for modeling the block system. Click the tabs if you want to change the page. First we will look at the System page.

10.2.1.3 Draw the internal system

The next figure shows the block diagram representation of the dc-motor simply derived from the equations:

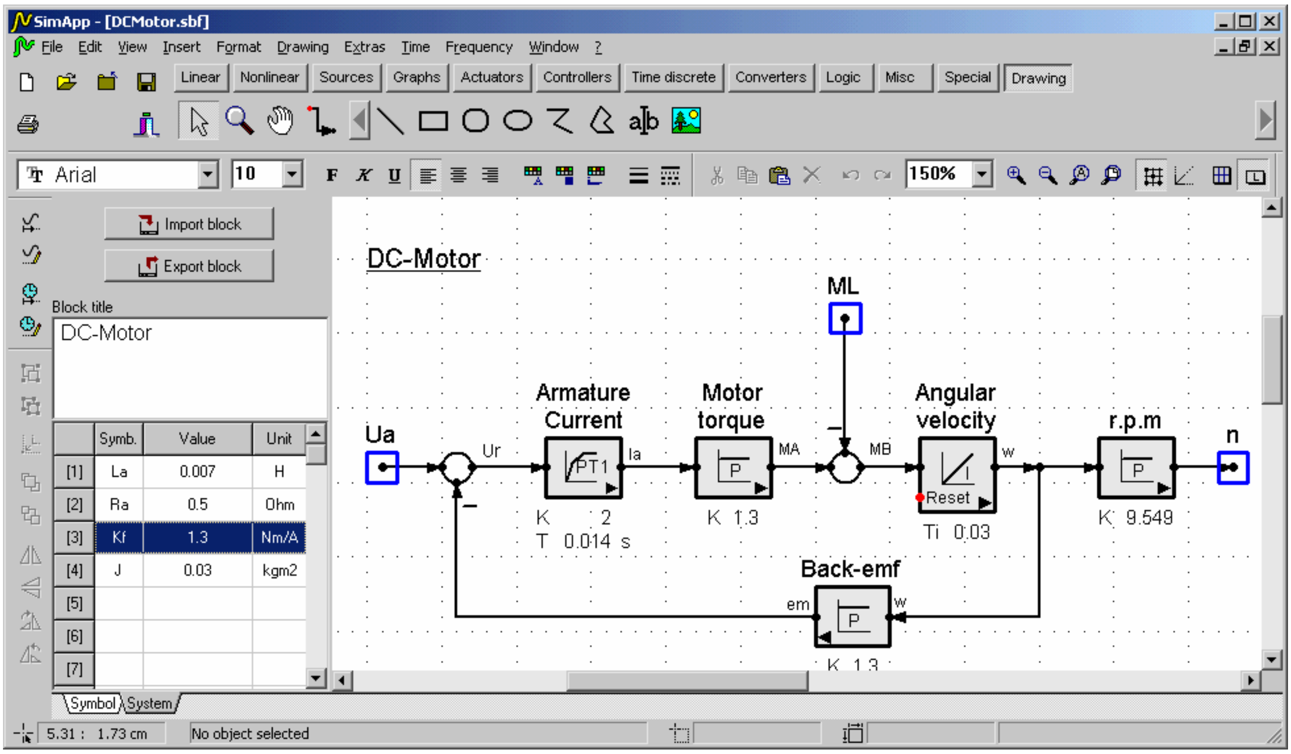


Figure 48: System page of the block folder

The inputs and outputs are specially denoted by node objects. You will find the node objects on page *Special* in the palette. Place the node objects exactly on the input and output nodes of the input and output signal lines. The node objects are automatically numbered, however, we recommend to label them with the associated signal names.

10.2.1.4 Creating parameters

The parameters of the native blocks can not be used as motor parameters. For example, the time constant of the PT_1-element is the quotient of armature inductance and armature resistance. You can create new parameters in the parameter table. In the simulation properties dialog box of the individual elements, you can then define the relations of their own parameters to the new virtual parameters.

Steps:

1. Double-click the first row of the parameter table. In the dialog box, enter the properties of the armature inductance L_A:

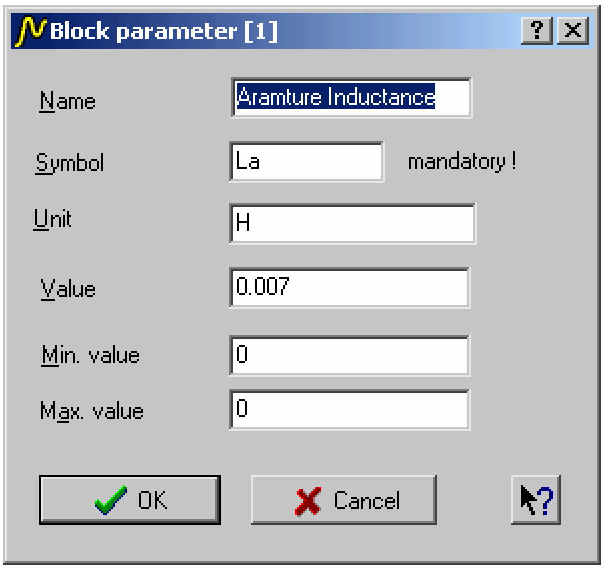


Figure 49: Defining block parameters

2. Enter the mathematical symbol, and a typical value for the armature inductance.
3. Close the dialog box. The symbol and the value appear on the first row of the parameter table. Repeat this procedure for the armature resistance, back-emf constant and inertia.
4. Repeat step 1 and 2 for all other parameters.
5. Until now, the new parameters have not had any effect. Now you have to define their relationship to the parameters of the individual elements. Open the simulation properties dialog box of the PT_1-element „Armature current". Press the *Parameters properties...*button and click the tab for the time constant. In the formula file, enter the expression [1]/[2].

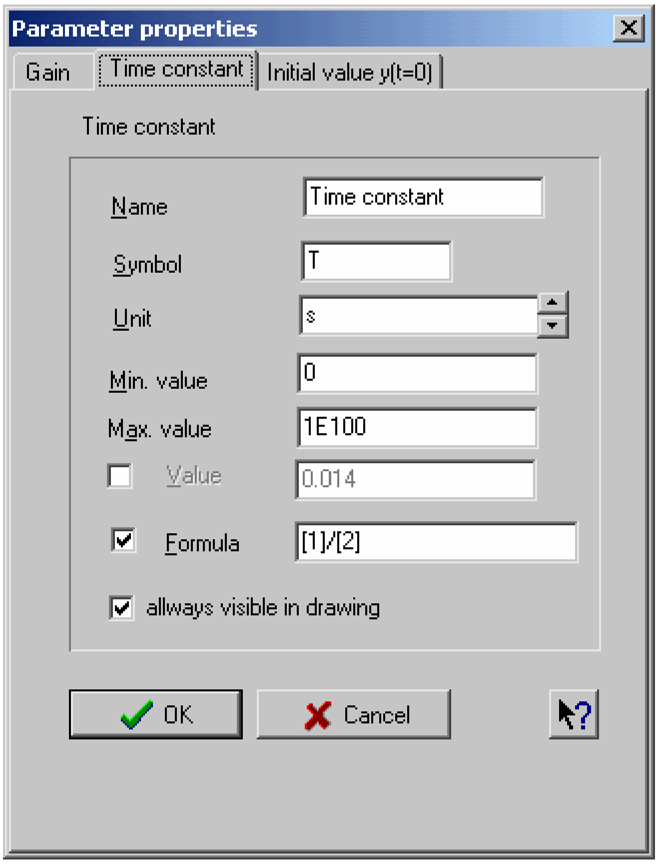


Figure 50: Entering formulae

6. The values in the square brackets are the indices of the new block parameters in the parameter table. [1] refers to the armature inductance L_A. The formula [1]/[2] means the quotient of L_A and R_A. The formula is interpreted by SimApp at every simulation. Besides the basic operators +, -,* and / , there are numerous

additional functions available and you can also use parenthesis. Repeat this procedure for the gain of the PT_1-element.

7. Now you have defined the relations between the parameters of the PT_1-element and the new motor parameters. Repeat step 3 for all elements. For the revolutions at the proportional element, you don't need a formula. Simply enter 9.549 or the formula 60/(2*Pi).

10.2.1.5 Designing the block symbol

After you have defined the functional part you need a symbol for the motor block. Change to the *Symbol* page. You see an empty frame and free nodes that are labeled with the input and output names. If you did not assign names, they will be numbered as (1),(2) and (3). These are the input and output nodes of the new block. Move them onto the frame. Input nodes on the left and output nodes on the right. Do not press the Alt-key for positioning or switch off the snapping permanently. The nodes must lie exactly on the grid. If you are not sure if they do so, move it to grid by the button.
The empty frame has the standard size of 1.6 by 1.6 cm, but you can enlarge or reduce it.
Now paint the block symbol with the SimApp's drawing tools.

10.2.1.6 Block assembly

Now we can join the functional and graphical part to a whole. Select the command *Extras* + *Block Folder* + *Export Block to Windows clipboard*. The ready to use block is now in the Window clipboard.
Change to another drawing or open a new one and paste the block for further use.

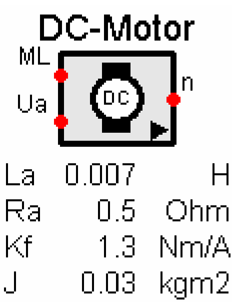


Figure 51: The finished block

If you have found any errors, go back to the block folder and correct them. If your block is ok, you can store it in a library (see chapter *Libraries*), into the palette (see chapter *palette*) or into a common SimApp drawing.

10.2.2 Summary

You have learned the necessary basic steps for creating custom blocks. As an overview we will enumerate all steps:

1. Open a new block folder.
2. Define new parameters in the parameter table
3. Draw the block diagram of the system you want to put in the block.
4. Enter the formulae for the parameters of the system blocks.
5. Test the inner structure
6. Denote the input and outputs by connecting them with node objects.
7. Design the block symbol and move the nodes on the block frame.
8. Join the system and the symbol by exporting the block into the Windows clipboard.
9. Paste the block into a drawing.
10. Store the block into a library or into the palette.

10.2.3 Relations between symbol and system window

The inner system of a custom block is connected to other blocks over the external nodes. The external nodes can not be created in the symbol window. They are created by inserting node objects into the system window. Node objects designate the inputs and outputs of the inner system. The node object and the associated node in the symbol window build an inseparable pair. Deleting a node object in the system window also deletes the associated node in the symbol window and vice versa. Every new pair gets a unique ID so that you can see which node and node object build a pair. However, it is advantageous to replace the number by a descriptive name.

10.2.4 Parameters table

By means of the parameter table you define the external parameters of the custom block which behave like native parameters of basic blocks. The table has 3 columns. Each row may contain a parameter. The first column contains the space saver which you use in formulae. The second column contains the symbol and the third a typical value.

Double-click a row to enter a new parameter into the table. You could also select a row by the mouse and press Enter. Blank rows are allowed. The parameters in the finished block appear in the same order as in the table. If you leave empty rows, you can later insert new parameters. Existing parameters can not be moved to another row. They must be deleted and inserted again. You can remove single parameters by selecting the row and pressing the Ctrl- and Del-keys simultaneously.

If you double-click a row, you can define a new parameter or change an existing one. Note that the input of a symbol is mandatory. If the field of the symbol is blank, the parameter is immediately removed when you close the properties dialog box. The values of the parameters can be arbitrary, but it is advantageous to enter typical values as they are used for testing the formulae.

10.2.5 Enter the system

Use any object of the palette to draw the inner system or even other custom blocks. By using sources be aware that their group number is zero, otherwise they will produce additional simulation runs. You can also connect the group number to an external parameter.

10.2.6 Formula editor

The inner parameters have fixed values that cannot be changed in the finished block or they are related to external parameters and can be changed indirectly. The relation can be very simple. For example, you could determine that an inner parameter always has the same value as the external parameter. But you can establish much more complex relations. The formulae are entered in the parameter properties dialog box of the inner blocks.
If you check the button for the formula, the actual parameter value is calculated by the formula and the external parameter values. In the background works a formula interpreter. The formulae are entered as plain text that must comply with special rules.

All the functions, operators and constants you can use are listed below:

Function	Description
+ - * / ^	Addition/ Subtraction/ Multiplication/ Division/ Power
!	Faculty
sin( )	Sine (Radiant)
cos( )	Cosine (Radiant)
tan( )	Tangens (Radiant)
arcsin( )	Arcsine (Radiant)
arccos( )	Arcosine (Radiant)
arctan( )	Arctangens (Radiant)
ln( )	Natural Logarithm
log( )	Decade Logarithm
exp( )	Exponential
sqr( )	Square
sqrt( )	Square root
int( )	Integer part
frac( )	Fractional part
abs( )	Absolute value
rnd( )	Round to integer
Pi	π = 3.14159...

You can also use parentheses. Upper and lower case is not differentiated. The formula is syntactically tested and numerically evaluated when closing the dialog box. If you have a relation to external parameters, their typical values are used. If the formula contains any syntactical or numerical errors (for example division by zero, overflow, etc.), the point in the formula where the interpreter encountered an error is designated by a question-mark. Therefore, by entering typical and valid values for the external parameters you can avoid simulation aborts produced by formula errors.

Some examples for correct formula:

5.67	Parameters with constant value 5.67
[3]	The value is always identical to the value of the block parameter in row 3 of the parameter table.
sin([3])	Sine of the parameter in the third row

2.5 * exp(-4/[6]) * sin(1.76)
Pi^[4]
[2] * (1 – [4] * (Pi – sin([1])))

Note: SimApp applies zero if you enter a relation to a non existent parameter. There will be no error message if no syntactical or numerical errors occur!

10.2.7 Testing the inner system

Before you have finished your block you should test the inner system. In the system window you can perform time and frequency simulations but you cannot store the results. They are lost if you close the block folder. Alternatively you could test the system in a standard SimApp drawing. In this case, the formulae are omitted and the current parameter values are used instead.

10.2.8 Inserting node objects and block nodes

The input and output nodes of the inner system must be connected by node objects. The insertion of node objects causes the creation of block nodes in the symbol window. Related node objects and block nodes always have the same name. When you change the name in one window, the name of the associated object in the other window is immediately adjusted. The name appears in the complete block as node label.

10.2.9 Designing block symbols

The empty frame and the arrow are created automatically. If you have many nodes, you can enlarge the frame. The filling color of the frame corresponds to the standard color also used by basic blocks. You can designate any shape to use this filling color by setting the option Background color for simulation blocks in the format properties dialog box.

Do not insert simulation objects into the symbol window. Direct inserting such objects is however not prevented by SimApp and are removed by block exportation.
Use the block title edit box for block title editing. Common text objects are not recognized as block title.

Draw a symbol which clarifies the functionality of the block. If necessary, use objects from other drawing programs and insert them as pictures. This particularly applies to curve paths (Bezier curves), which you cannot produce with SimApp. Curve paths can be approximated by polylines however. For that, select a great enlargement (10 - 30%). All curve paths in the standard objects were produced in this way.

Very important: Be sure that all nodes lie exactly on the grid. If you are not sure if they do so, move it to grid by the button. Do not switch off the automatic snapping function.

Set the operation behavior of the objects in the block symbol.

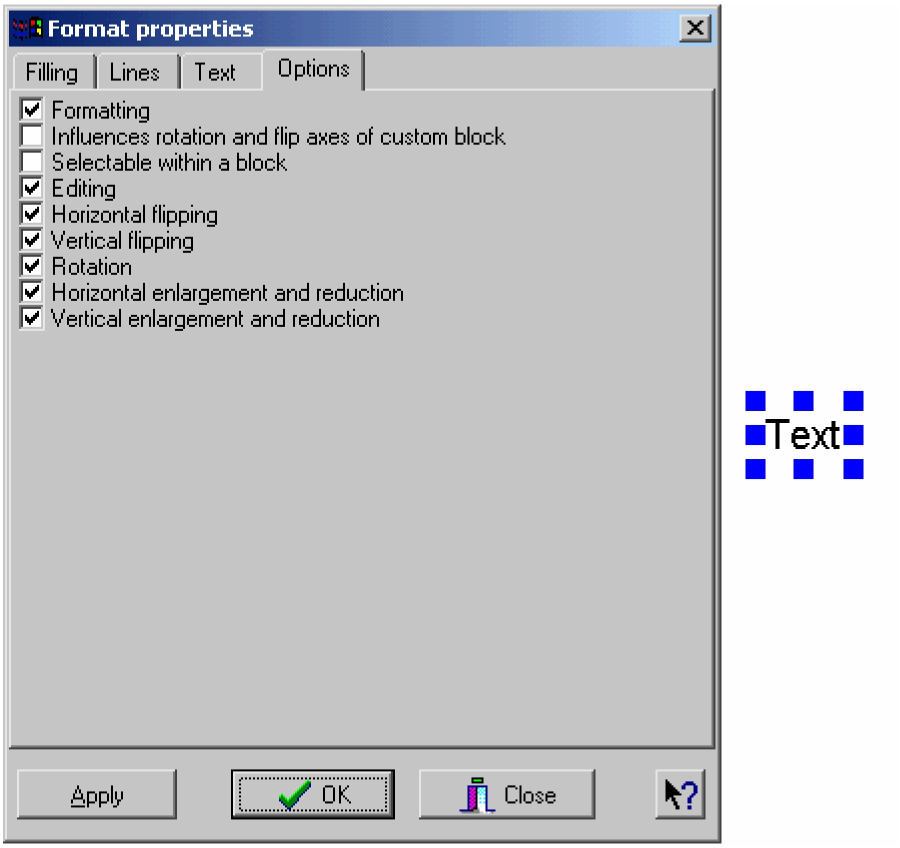


Figure 52: Options for text objects

The check boxes have three states: Checked, unchecked and indeterminate. The indeterminate state is reflected by a grayed check mark that appears if several objects not having the same state are selected. When closing the dialog box with *Ok* grayed check marks let the underlying objects unchanged. By clicking the check box the option can be changed to set or unset and is now available for all selected objects.

Within block symbols only text objects can be modified by position, size and contents.

Settings for text objects:

- Neither selectable nor capable for editing: No changes at all.
- Selectable but not editable: change size and position by clicking and pressing the ALT-key. (But the text is not accessible.)
- Editing but not selectable: Edit text by clicking. Position is fixed, size depends on text.
- Selectable and capable for editing: Select and drag the object by clicking and pressing the ALT-key. Edit by clicking without pressing the ALT-key.

10.2.10 Joining system and symbol (exporting)

Join the system and the symbol by copying the block into the Window's clipboard. For that however, you must not use the standard copying command. Select in the main menu *Extras* + *Block folder*+ *Export block to Windows clipboard* or simply press the *Export block* button. The complete block is now in the clipboard and ready for use.

10.2.11 Using and saving blocks

After copying into the clipboard you can use your new block like a standard element. Do not forget to store it at least at one place. This can be within a normal drawing file, in the palette or in a library. If you have saved the block, you don't need the contents of the block folder any longer. All information is stored in the block. You can see the internal structure of the block and change it at any time by copying it into an empty block folder.

10.2.12 Block revision

Formerly produced blocks may be changed by copying them back into the block folder. This is done via the Window's clipboard. Copy the block by Edit + Copy to clipboard. To import it into the block folder must not

use the standard command Edit + Paste however. This would insert the complete block into the system or symbol window without opening it. Select menu Extras + Block folder + Import block from Windows clipboard instead. The block is opened and inserted into the two windows. It does not even matter which window happens to be visible.
Alternatively you may open the custom block's pop-up menu and click the Open custom block in block folder command.

10.2.13 More remarks on the design of custom blocks

- You can open several block folders and design several blocks at the same time.
- The system and the symbol window can be stored together in a file. Select *File* + *Save* or *File* + *Save as....* The file is given the extension **.sbf**.
- SimApp also accepts time and frequency probes in blocks and affect the simulation in the common way. The application of such probes is questionable since they are not visible in the drawing, but nevertheless influence the simulation. A possible application would be the automatic monitoring of an internal transmission circuit or an internal node. But if you wish to monitor internal nodes, it is better to designate outside test nodes or use transmitter/receiver pairs.
- Custom blocks have one block symbol only in contrast to standard elements.
- If a block contains one or more time discrete elements, you must designate a block parameter for the sampling period. The reference must be direct (e.g. [3]) and may not contain any formula expressions ([1]/[2] or 5*[5]).
- For testing purposes internal signals can be transmitted to outside by using transmitter/receiver pairs.

11 Working with the palette

The palette contains all standard objects that can be inserted into drawing. It consists of several pages, accessible by tabs. Each page contains a certain category of objects.

You can extend the palette as you wish. For example, you may create new pages and store your custom blocks therein.

Use the following functions for palette processing

- Create new pages
- Move pages, change arrangement
- Extend palette by new objects
- Rename pages
- Delete pages
- Move buttons
- Load and save palette

11.1 Creating, deleting and renaming palette pages

Open the palette's pop-up menu by clicking the right mouse button on the palette (but not on a button), then select the appropriate menu option.

New page
Click *New page*. Enter a name which does not yet exist in the palette.

Delete a page
Click *Delete page*. The front page is deleted immediately. If there are still buttons in it, SimApp requires confirmation that it may remove all buttons and the associated objects. Standard pages are excluded from deletion.

Rename a page
Click *Rename page*. Enter a new name. Standard pages can not be renamed.

11.2 Moving pages and buttons

Click with pressed Alt-key the tab or button which you would like to move. Move the mouse pointer to the desired place and release it.

11.3 Storing objects in palette

You can store any objects into the palette. These could be standard elements, custom blocks, text objects or even grouped objects. The storing takes place simply by dragging and dropping.

11.3.1 Storing objects from drawings

Click the tab of the page in which you would like to store the object or a selection of several objects. Then select the desired object in the drawing and shift it toward the palette. If you come to the edge of the drawing window, normally the autoscrolling starts. You can prevent this by keeping the spacebar pressed.
The object is inserted into the page by releasing the mouse button. If you wish to copy the objects, you must press the Ctrl-key when you release the mouse button otherwise the objects will be moved into the palette and are no longer available in the drawing A new button with a standard image appears. You can, as described later, still move the button within the current page. The characteristics of the new button can be adjusted according to your requirements (see *Process buttons*).

11.3.2 Storing objects from libraries

Click the tab of the page in which you would like to store your objects. Open a library. Press the Alt-key and press a library button. Move the button to the palette and release it at the desired insertion position. The button is copied from the library to the palette.

11.4 Working with palette buttons

If you move objects from the drawing to library or palette, SimApp creates standard buttons. The images of these buttons are not very meaningful. You should replace the text by a suitable name for the contained objects. It is even better if you design your own image.

11.4.1 Properties of palette buttons

Right-click the desired button and select *Format properties* in the pop-up menu.

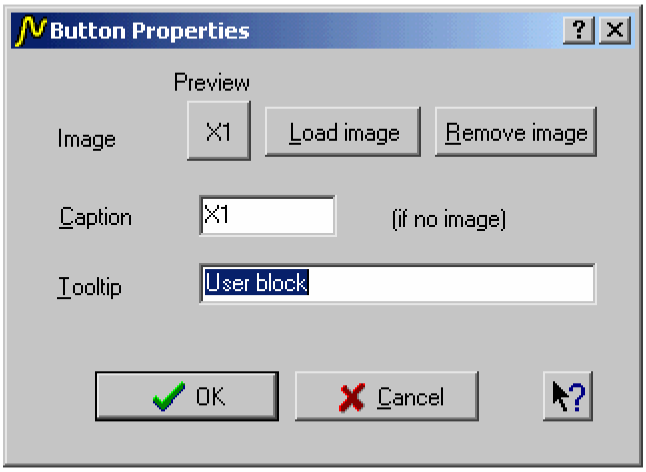


Figure 53: Palette buttons properties

You can load any bitmap of size 24 x 24 pixels. But you cannot show bitmap and button text simultaneously. Create the bitmap with the Paint program of Microsoft Windows or with another paint application. Paint is a standard accessory of Windows. If you have not installed Paint, you will find it on the installation CD of Windows.
As further characteristic you can assign a tool tip to the button, a small pop-up window appearing when moving the mouse pointer over the button. The tool tip should contain a short description of one to three words.

11.4.2 Designing button images with Microsoft Paint

Open Paint and select *Image + Attributes*. Select the following adjustments:

Width:	24
Height:	24
Unit:	Pixel
Color:	Colors

Select the largest zoom factor and paint the image. In order to clarify the different switching status, the color of the background is controlled by SimApp. You must determine however, which pixels belong to the background. SimApp detects the background by means of the pixel in the lower left-hand corner. All other pixels having the same color belong to the background. Which color you select is insignificant for SimApp. It is only important that you do not use it anywhere else.

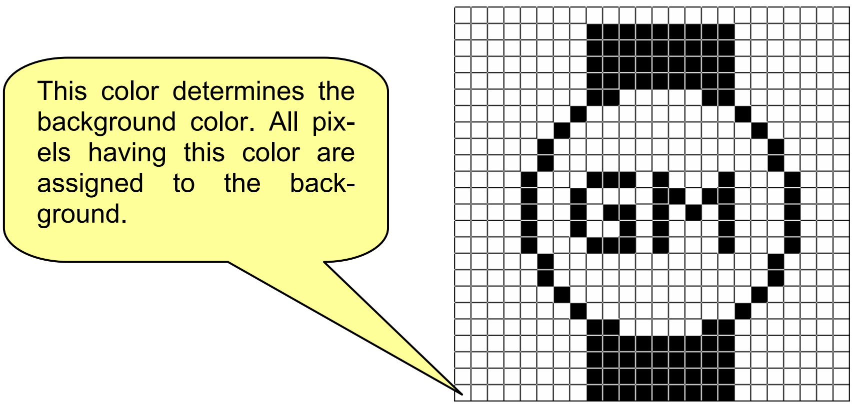


Figure 54: Button image

Store the image under any name. Change to SimApp and load it into the desired button. The image is completely copied. The bitmap is no longer needed after load.

11.5 Loading, saving and restoring the palette

The palette can be saved and reloaded. At program start, SimApp loads the last current palette. Just after installation the original palette *SimApp.spl* is loaded from the SimApp's program folder.
If you change the palette and save it on disk, SimApp reloads this palette as default palette on the next program launch. If you wish another palette you can load it at any time.
If you changed the origin palette by mistake you can restore it by the restore command and overwrite the palette file *SimApp.spl* in the program folder.
Commands for loading, saving and restoring palette are in the palette's pop-up menu (right-click) and in the program's main menu.

12 Libraries

Each field has characteristic system structures which are used again and again. For these structures you can create your own custom blocks. You can divide your blocks according to certain criteria in groups and store into libraries. By using libraries you are however not limited to custom blocks. You can select any number of objects and signal lines or grouped objects for storing in libraries for re-use. Even standard elements of the palette can be copied into libraries. Libraries behave on the display like normal toolbars.

12.1 Example

By means of a simple example you will learn the basics for creating and using libraries. So that you do not have to master a large drawing pool, a new library is created which is only filled with standard elements. Putting standard elements into libraries has, in practice, no advantages. It is much more meaningful for you to store your custom blocks.

Open a new drawing and place therein two standard elements from the palette.
Create a new library with *Extras* + *Library* + *New*. A window having a yellow background appears. Move an element from the drawing into that window. Then move the second element into the library while pressing the Ctrl-key. Let's assume that you have moved the first element and copied the second one.
Select the third element in the palette. Before clicking ist, press the Alt-key. Move the button into the library window and release the mouse button. Your library now looks like this:

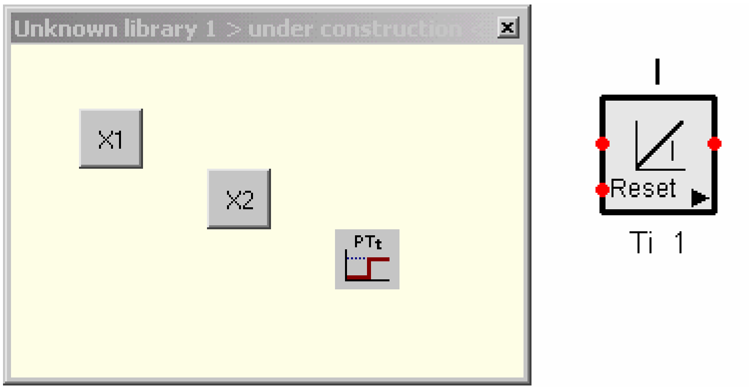


Figure 55: Creating libraries

You can rearrange the buttons as you like. Then right-click the library window and select *Rename*. Enter "Test" as new text in the window's title bar.

The two objects inserted from the drawing now have gray buttons with designation X1 and X2. The object which you moved from the palette remained unchanged. You can change the texts X1 and X2 or even replace them by images. Please read the appropriate paragraph in the chapter *Working with the palette*.

Reopen the library's pop-up window and click *Processing mode.* By switching off the processing mode, the library window will immediately be reduced and the buttons pushed together.

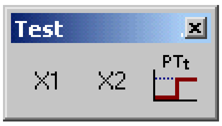


Figure 56: The finished test library

Open the pop-up menu. Click *Save* and save the library under the name *Test*.
Click *Close* in the pop-up menu. The library disappears but still remains in memory since it is only hidden.
Select in the SimApp main menu *View* + *Toolbars...*. You will now find the name of your library in that listbox. Set the check mark of the library and close the dialog box with Ok. The library is displayed again.
Click *Remove* in the pop-up. If you have not saved the library yet, a request to store will appear. Afterwards, the library disappears again and you will not find it in the toolbar list.
Open the library again with *Extra* + *Load Library...*. and load *Test.slb*. Now it is back again.

13 Catalog of standard elements

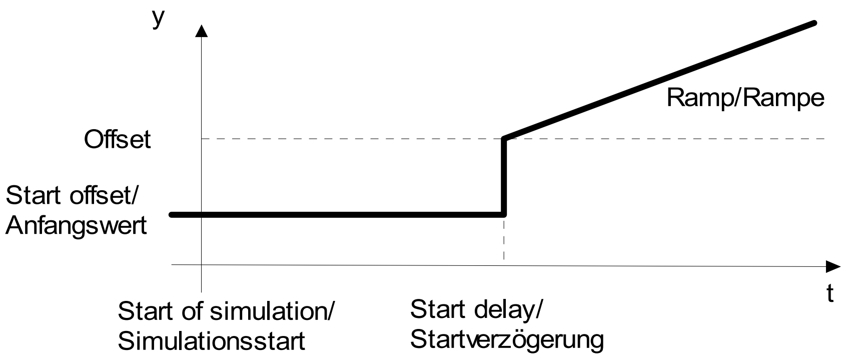
13.1 Sources

Each source has specific parameters to control the output signal. But all sources (except constant source) dispose of the following basic parameters:

Offset
Shifts the characteristic curve up and down.

Start delay
Elapsed time before the output signal is applied. May be used for synchronizing different sources to each other.

Start offset (initial value)
Initial output value before the start of the simulation and output value until the start delay time has elapsed.



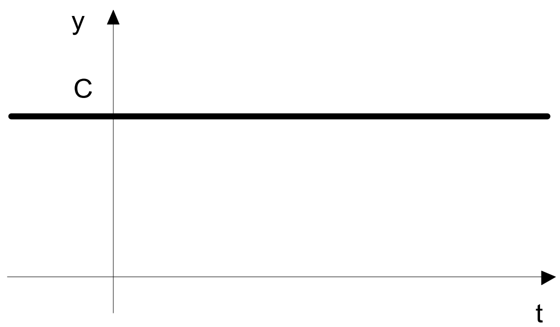
All basic parameters are set to zero as default.

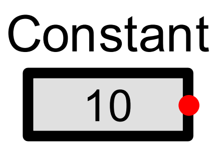
13.1.1 Constant

The constant source applies an arbitrary positive or negative value, which is never changed during simulation. Offset, start delay and start offset do not exist.

Parameters: C, Grp >= 0

Function: y(t) = C



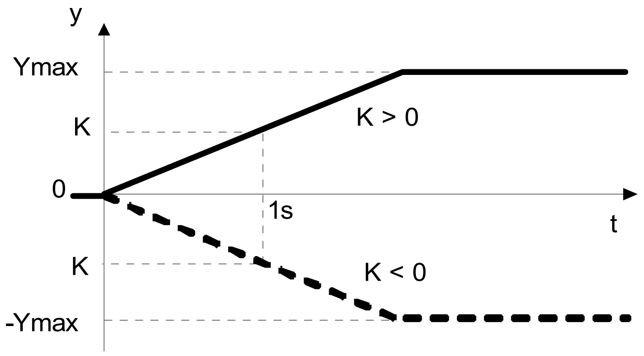


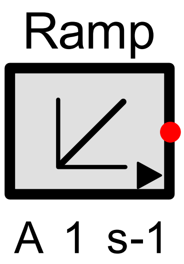
13.1.2 Ramp

The ramp applies a constant slope up to a specified maximum. The saturation Ymax is always edited as positive number and works as maximum at positive slopes and as minimum with negative sign at negative slopes.

Parameters: Grp >= 0, A, Ymax >= 0, Y0, TD>=0, OS

Function: y(t) = A t



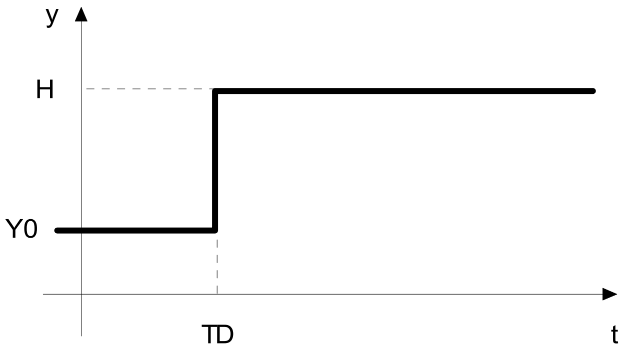


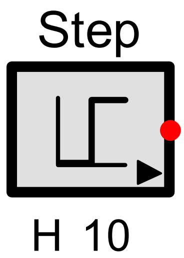
13.1.3 Step

The edge time is determined by the start delay parameter.

Parameters: Grp >= 0, H, Y0, TD>=0, OS

Function: y(t) = H σ(t-TD)

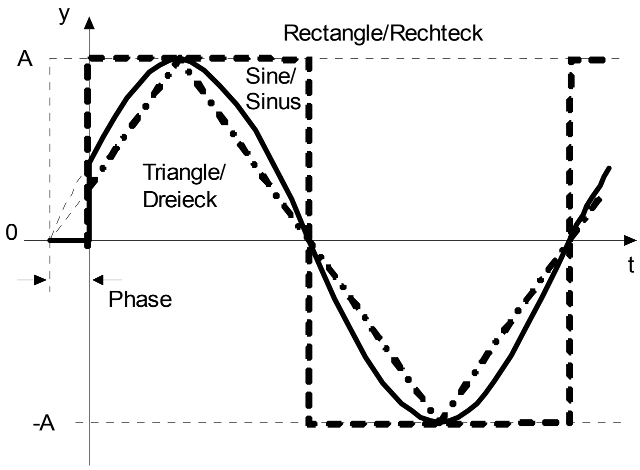


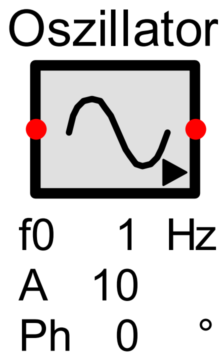


13.1.4 Oscillator

The oscillator generates sinusoidal, rectangular or triangular output signals. Frequency, amplitude and initial phase are adjustable.

Parameters: Grp >= 0, A
f0 >= 0, [Hz] or [rad/s] (Angular frequency)
Ph, [°]
Y0, TD>=0, OS



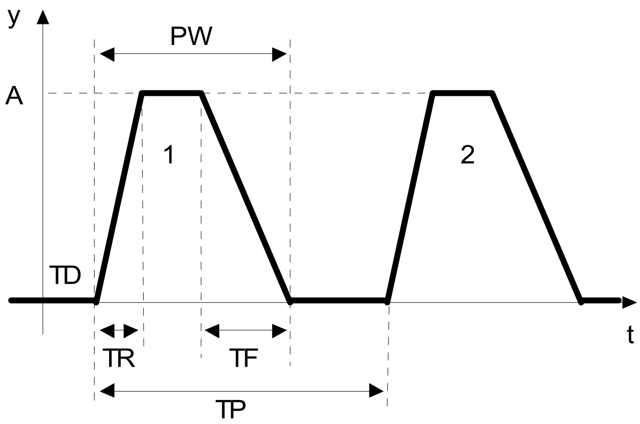


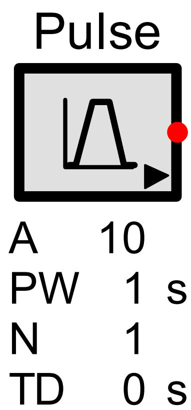
By means of the modulation input port the oscillator may be operated as voltage controlled oscillator (VCO). Either the output frequency is proportional to the input signal (f = Mod x f0, linear modulation) or exponential to the input signal (10^Mod x f0, exponential modulation).
When the modulation input port is not used, the output frequency is constant f = f0.

13.1.5 Pulse

The pulse source applies a series of identical pulses. Either the sequence consists of one single pulse only (N=1) or of several pulses (e.g. N=6) or of an infinite number of pulses (N<0).

Parameters: Grp, A, PW >=0, TP > 0, TR>=0, TF>=0,
N, Y0, TD>=0, OS



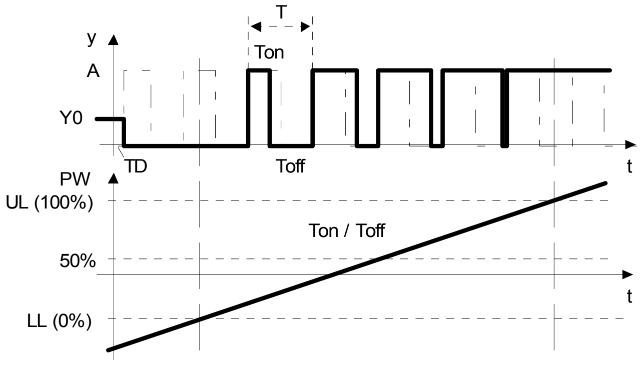


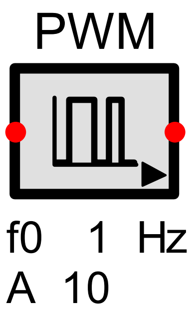
The rising slope always precedes the falling slope. Before the beginning of the rising slope, the falling slope must have been finished. If the rising slope is longer than the period, it finishes at the end of the period and the falling slope is omitted.

13.1.6 Pulse width modulator (PWM)

The pulse width modulator generates a rectangular output signal of desired frequency, phase and amplitude. The ratio between Ton / Toff is controlled by the input signal between 0 and 100%. The rising edge matches always the beginning of a new period of the underlying fundamental wave. The falling edge is controlled by the input signal. The control values for 0% (LL) and 100% (UL) can be set on any value (LL may be greater than UL or even negative).

Parameters: Grp >= 0, A
f0 >= 0, [Hz] or [rad/s] (angular frequency)
UL, LL,
Y0, TD>=0, OS



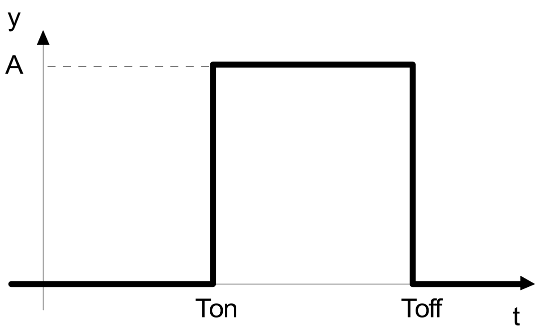


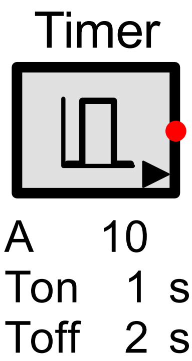
13.1.7 Timer

The timer generates a single rectangular pulse. On/off time and pulse height can be specified.

Parameters: Grp >= 0, A, Ton >=0, Toff >= 0, Y0, TD>=0, OS

Function: y(t) = H [σ(t-Ton) - σ(t-Toff)]



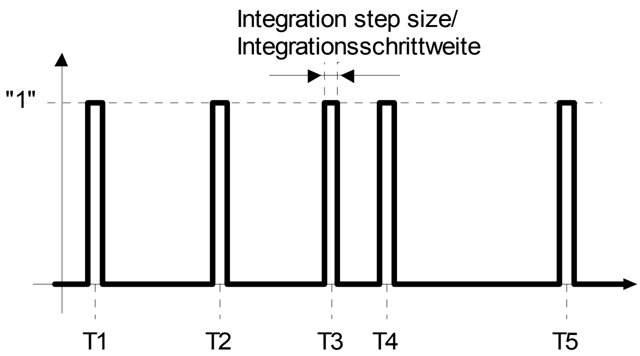


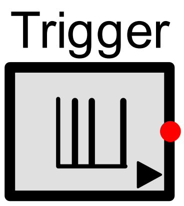
13.1.8 Trigger source

The trigger source supplies a predefined sequence of pulses. The height of the pulses corresponds to logic 1 level and their width to the current integration step size. This source is useful for triggering time dependant events in a controlled process.

The pulse sequence is defined in the simulation properties of the source. By setting the option *Iterative* the sequence is continuously repeated. If the first pulse is at time 0, the sequence starts its repetitions at pulse no. 2, because the last pulse of the sequence has priority.

Parameters: Grp >= 0, V0, TD>=0, OS





The time sequence can be stored into a text file and loaded at any time. The maximum number of pulses is 10'000.

The data format in the text file is the same as for the file source and user source, except that it needs only one data value per line.

Format description for the text file

- One single data value per line. Valid decimal separators are comma and point. A data line may be finished by a comment preceded by an asterisk (*)
- Pure comment lines which start with a numeral must place an asterisk (*) in front. Preceding spaces are allowed.
- The file must be an ANSI- or ASCII text file. Default file name extension is **.txt**, but any other valid Windows file name extension is allowed.

By saving the data in a file the used decimal separator depends of the associated setting in the program options.

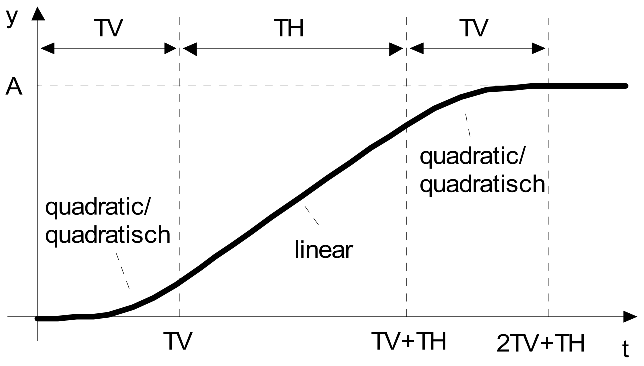
```
Trigger source sample file
Created by Bruno Buesser, Feb. 29, 2003
0           *First pulse at simulation start. Excluded from repetitions.
3.67        *Second pulse. Time values are in ascending order.
                                    (>Blank lines are skipped
5
6,2         *Decimal separator may also be a comma
8           *Last pulse
* 5 values  >> Asterisk * is used as comment starts with a numeral
By setting the option Iterative, the sequence from 3.67 to 8 is
successively repeated.
End of sample file
```

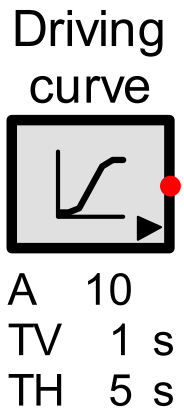
13.1.9 Driving curve

Start and speeding up characteristic.

Parameters: Grp >= 0, A, TH>=0, TV>=0,Y0, TD>=0, OS

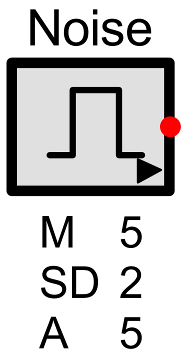
Function: y(t) = f(A,TV,TH)





13.1.10 Noise, random number generator

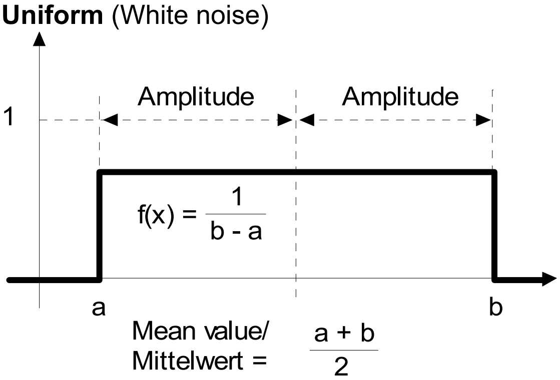
By means of this source that can also be used as random number generator you can evaluate the behavior of automatic control systems in presence of random noise.



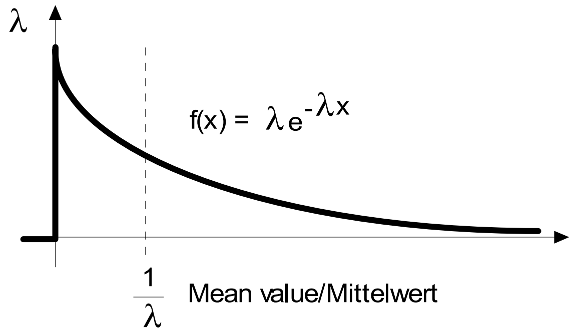
Parameters: Grp >= 0, A, SD>0, M, Y0, TD>=0, OS

The noise source supports the following four noise deviates.

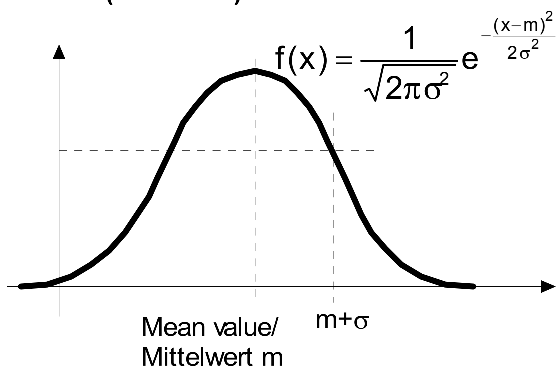
Uniform (White noise)



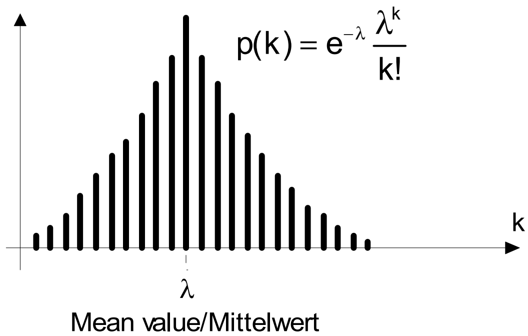
Exponential



Normal (Gaussian)



Poisson



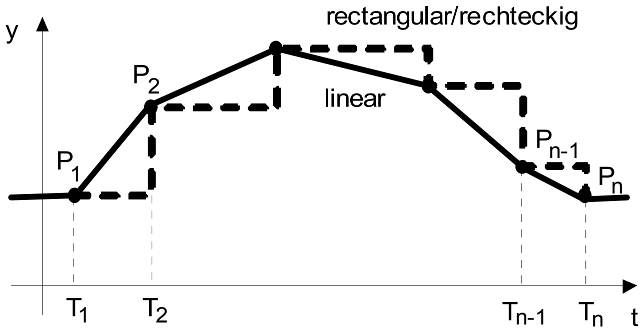
The *Reproducible* option controls the starting point of the internal random generator.
If *Reproducible* is set, it applies identical noise sequences at every simulation run, even when the drawing is saved and reloaded.
If *Reproducible* is disabled, it applies different noise sequences in subsequent runs.
If the system has more than one noise source, all noise sequences differ to each other. (Independent of *Reproducible*)

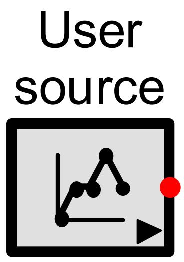
13.1.11 User source

This characteristic consists of a series of points that form the output signal. You can select between a linear and a rectangular interpolation.

Parameters: Grp >= 0, TD>=0, OS, P1..Pn

Function: $y(t) = f[P_1...P_n, t]$





The characteristic starts always with the value of the first point. Steps can be realized by setting the same time for two adjacent points. The law of causality must be observed therefore the points must be timely arranged in ascending order. If three or more points have the same time, only the first and the last one are taken.

By setting the option *Iterative* the characteristic is continuously repeated. If the first point is at time 0, the characteristic starts its repetitions at point no. 2, because the last pulse has priority.

The values are stored in the source, but can also be saved and reloaded into/from a text file.

Description of the data format in the text file:

- One Y-t pair per line.
- Y- and t-values must be separated by space, tab or semicolon. Valid decimal separators are point and comma. A data line may be completed by a comment, preceded by asterisk (*).
- All lines not starting with numeral, comma or point are interpreted as comment.
- The file must be an ANSI or ASCII text file. Default name extension is **.txt**, but all other valid Windows extension are also accepted. The number of data points is limited to 10'000. By loading a bigger filer the surplus points are ignored.

If the data are stored in a text file, the value pairs are separated by TAB. The decimal separator depends of the setting in the program options.

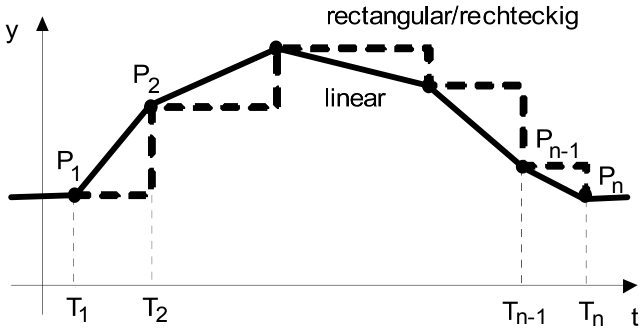
```
User source sample file
Create by Bruno Buesser, Feb. 27, 2003
This file contains 5 pair of values
2.45; 1      *That's the first point. Time first then output value
3.67; 2.5    *Second point. The time values must be in ascending order.
5 3.14       *The values are now separated by space
                                     (>blank lines are omitted)
6.23 2,5     *Comma is also a valid decimal separator
Lines which do not start by numeral, point or comma are skipped
8;0          *That is the last pair of values
* 5 pairs are red  >> asterisk * is mandatory as the comment starts with a numeral
End of sample file
```

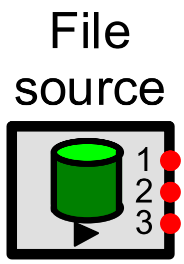
13.1.12 File source

The file source may have up to 50 simultaneous outputs. The data cannot be stored in the source and is loaded from a text file during simulation. Only the file path is stored in the source. If you copy the drawing to another computer, you must also copy the data file. Be aware that the stored file path in the source may not be valid in another environment. The file path may be absolute (i.e. with drive letter) or relative. A relative path refers to the directory of the drawing.

Parameters: Grp >= 0, TD>=0, OS, P1..Pn (per curve)

Function: $y(t) = f[P_1...P_n, t]$ (per curve)





Every output signal requires an output node. The number of output nodes can be set in the block properties dialog box from 1 up to 50. The number of data series in the text file must at least match the number of output nodes. Surplus series are allowed but omitted. The output signals start all with the first point.

By setting the option *Iterative* the series are repeated each time the stored time interval has elapsed. If the first point occurs at time 0, it is skipped in repetitions as the last point has priority.

If the simulation lasts longer than the stored sequences and repetitions are not required the last point is taken as constant value for the rest of the simulation.

Description of the data format
The data streams and the associated time form a table. Columns are separated by tab, spaces or semicolons (or mixed). Valid decimal separators are point and comma. The first column contains the time values, all other columns the associated output values. All output values of a line are referred to the time value in the first column. The times are arbitrary (no equal spaced time interval needed), but the value must be in ascending order. Steps can be realized by setting the same time for two adjacent points.
A data line can be followed by a comment that is started by asterisk (*). Pure comment lines must only be started by asterisk if the comment starts with a numeral. The file must be an ANSI or ASCII text file. Default name extension is **.txt**, but all other valid Windows extension are also accepted.

```
File source sample file
Created by Bruno Buesser, Feb. 27, 2003
This file contains 3 output signals with 5 data points each
2.45; 1; 5; 8       *This is the first set of points. Time first then data.
3.67; 2.5; 4.3; 6   *Second point. Time values are in ascending order.
5 3.14 4.5 3.3      *The values are now separated by spaces
                                  (>blank lines are skipped)
6.23 2,5 4.1; 2,5   *Comma is also accepted as decimal separator
Lines which start not with numeral, point or comma are skipped
8;0 3,8; 1.4        *That is the last set of points
 * 5 sets of points are loaded >> * is mandatory, as the comment starts with a numeral
End of sample file
```

The series are best edited in a spread sheet (like Microsoft Excel) and then exported to an AINSI text file.

Note: The time space of two successive data points should be a multitude of the integration step size. If not, the integration step size should be less than half the space of two successive data points, otherwise signal filtering occurs (undersampling) that may distort the output signal. Make a visual check in cases of doubt.

13.2 Linear elements

Linear elements obey the principles of superposition and gain.

$f(u_1 + u_2) = f(u_1) + f(u_2)$ principle of superposition

$f(Ku_1) = Kf(u_1)$ principle of gain

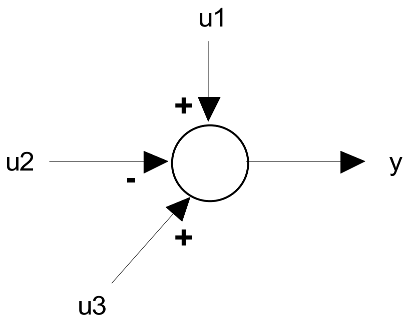
With linear elements you can perform time simulations as well as frequency simulations.

13.2.1 Adder

The adder (adding element, summing element) has one or more signal lines as inputs and one single signal line as output. For each input signal the sign can be set individually. With only one input line the adder can form an inverter.
You can change the sign of the inputs by selecting the signal line and pressing the + or - key on the numeric keypad or by using the signal lines pop-up menu.

Function: $y(t) = \pm u_1(t) \pm u_2(t) \pm \ldots \pm u_n(t)$

Example: y = u1 - u2 + u3



13.2.2 Proportional element (P element)

The input signal is amplified by gain K.

Parameters: K

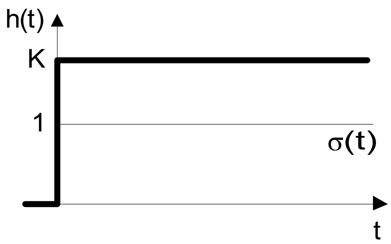
Functions

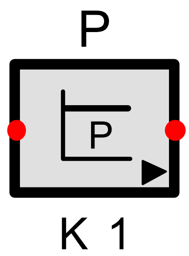
$y(t)=Ku$

$G(s)=K$

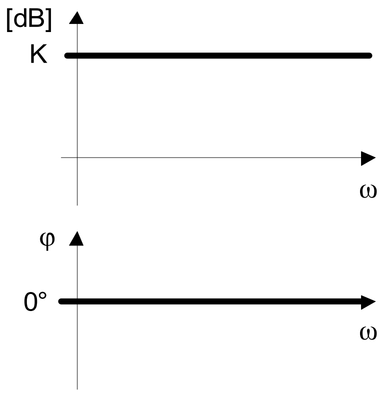
Step response

$h(t)=K\sigma(t)$

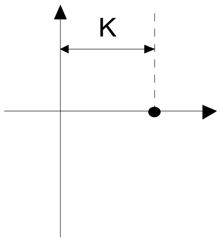


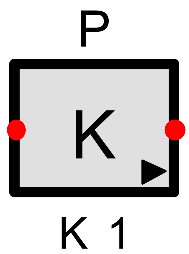


Magnitude and phase response



Polar plot





13.2.3 Integrator

The output of an integral element is the integral of the input signal over the time. It responds on an input step by an unlimited time output ramp. Thus, it represents an unlimited storage device.
For t <= 0 the output is equal to the initial value Y0.
The reset time is the time at which the magnitude of the step response is equal to the constant magnitude of the input step (if Y0 = 0).

The output is limited by Ymin and Ymax (anti-windup). When the maximum value is reached, the integration is stopped and continues in opposite direction only if the input value has changed its sign. The limits are active only, if Ymax > Ymin

Logic high value at Reset input terminal always resets the integrator to initial state Y0. The logical threshold Vth is at 2.5 by default however may be adjusted individually.

In frequency simulations, the Reset input is omitted. In time simulations, if you do not need the reset feature, you can let it open (unconnected).

Parameters: Y0, Ti > 0, Ymax, Ymin, Vth

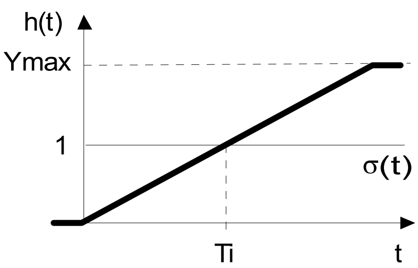
Functions

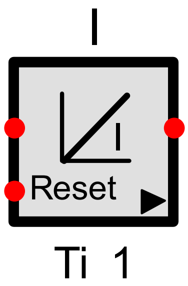
$$y(t) = \frac{1}{T_i}\int_0^t u(\tau)d\tau$$

$$G(s) = \frac{1}{sT_i}$$

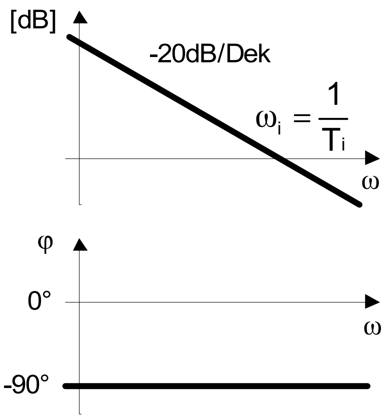
Step response

$$h(t) = \frac{1}{T_i}t$$

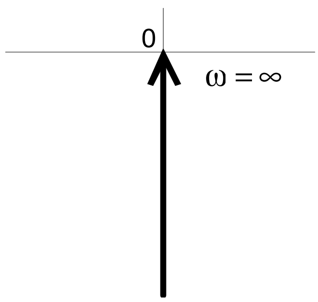


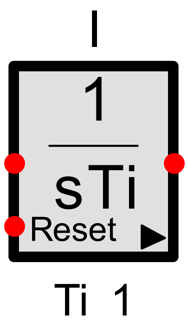


Magnitude and phase response



Polar plot





13.2.4 Differentiator, derivative element

A pure differential behavior can not exist in a real system and can not even be realized by computers. This derivative element is a best approach to the ideal differentiator. For time simulations the approach depends on the integration step size and for frequency simulations it depends on stop frequency.

An ideal differentiator responds by a infinitely high pulse that lasts an infinitely short time interval only. In simulations however, the pulse height is limited to TD/h and the pulse length to h where h is the Integration step size. The shorter h the better the approximation to the ideal behavior. As an ideal behavior can not occur in real systems the approximation in SimApp is not a disadvantage.

Parameters: Y0, TD >= 0

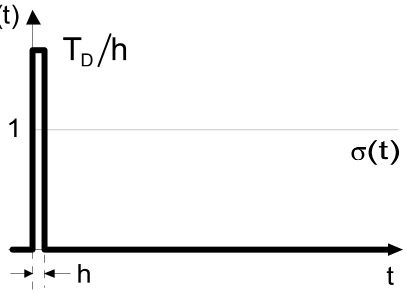
Functions

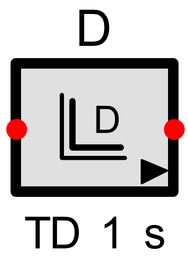
$$y(t) = T_D \dot{u}(t)$$

$$G(s) = T_D s$$

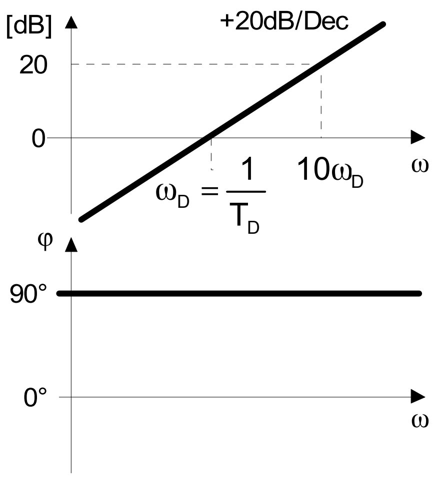
Step response

$$h(t) = T_D \delta(t)$$

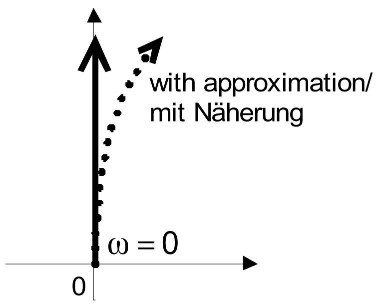


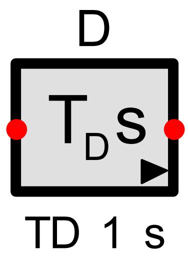


Magnitude and phase response



Polar plot





Implementation notes:

As a pure differentiator can not be realized by programming, SimApp uses a real differentiator whose coefficients are set in such a way that they approaches the ideal behavior as far as possible.

Transfer function of a real differentiator: $G(s) = \frac{T_D s}{1 + Ts}$. Ideally T is 0.

For **time simulations**: T = h (integration step size)

For **frequency simulations**: $T = \frac{1}{100\omega_S}$ where ω_S = simulation stop frequency (set by user) or $T = T_D / 100$, depending on which value is smaller.

13.2.5 Rate element (DT1)

The DT_1 Element is the combination of an ideal differentiator and a first-order lag element. Its step response is strongly damped and so limited in size and the settling time is significantly increased.

Parameters: TD >= 0, T1 > 0, Y0

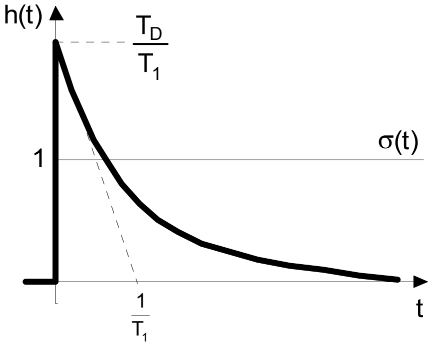
Functions

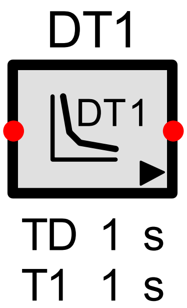
$$y + T_1\dot{y} = T_D\dot{u}$$

$$G(s) = \frac{T_D s}{1 + T_1 s}$$

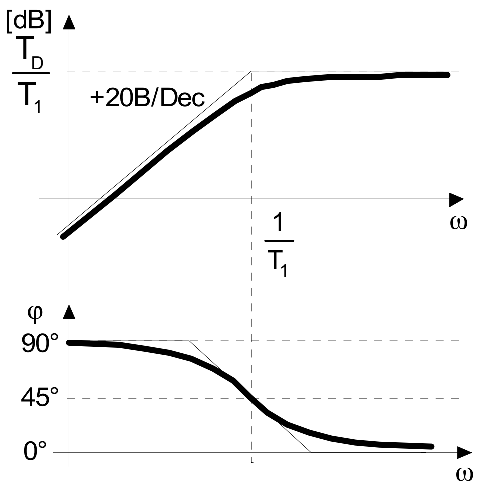
Step response

$$h(t) = \frac{T_D}{T_1} e^{-\frac{t}{T_1}}$$

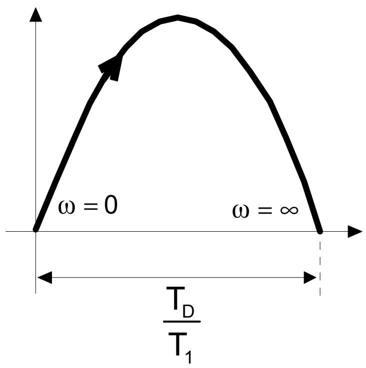


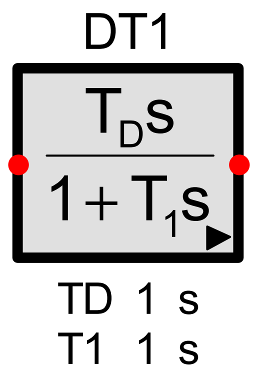


Magnitude and phase response



Polar plot





13.2.6 First order delay element (PT1)

The PT1 element has one internal energy store. Therefore, it cannot follow a step instantaneously. The output signal is strongly smoothed. The damping is so strong that an overshoot can not occur. After a long time, the step response follows proportionally the input signal.

Parameters: K , T >= 0, Y0 (if T = 0, Y0 is omitted)

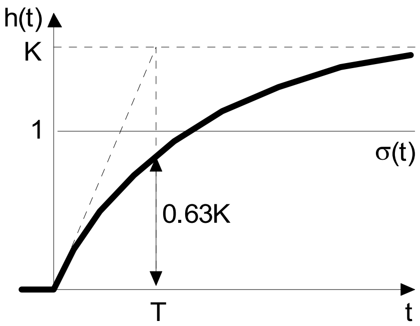
Functions

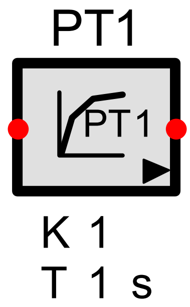
$$T\dot{y} + y = Ku$$

$$G(s) = \frac{K}{1 + Ts}$$

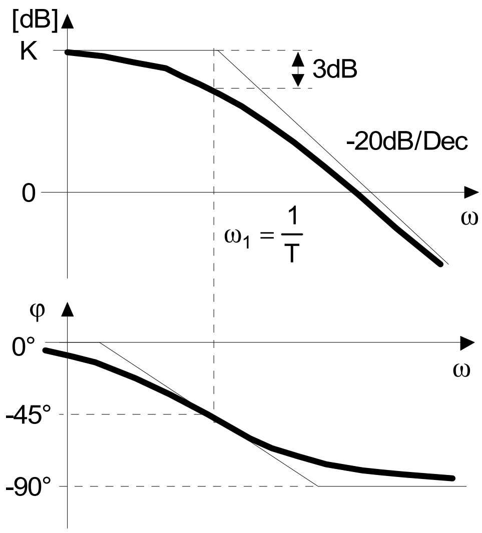
Step response

$$h(t) = K(1 - e^{-t/T})$$

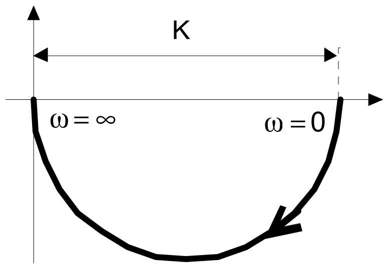


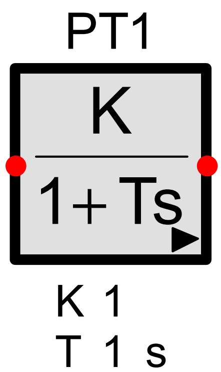


Magnitude and phase response



Polar plot





13.2.7 Second order delay element (PT2)

The PT_2 element contains two independent energy stores. The step response depends on the damping d and the resonance frequency. There are four cases:

1. Undamped: d= 0. The step response is an undamped sinusoidal signal
2. Underdamped: 0 < d < 1
3. Critically damped: d = 1
4. Overdamped: d > 1

Parameters: K, d >= 0, T >= 0, Y0, Y0' '
When T = 0: d, Y0 and Y0' are omitted

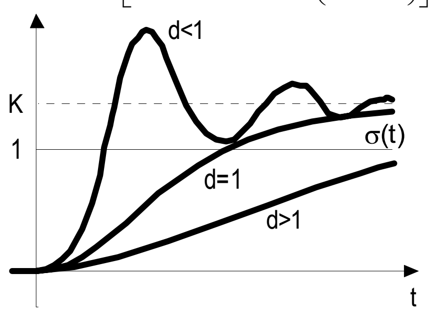
Functions

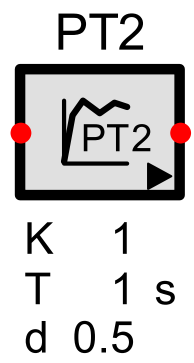
$$T^2\ddot{y} + 2dT\dot{y} + y = Ku$$

$$G(s) = \frac{K}{1 + 2dTs + T^2s^2}$$

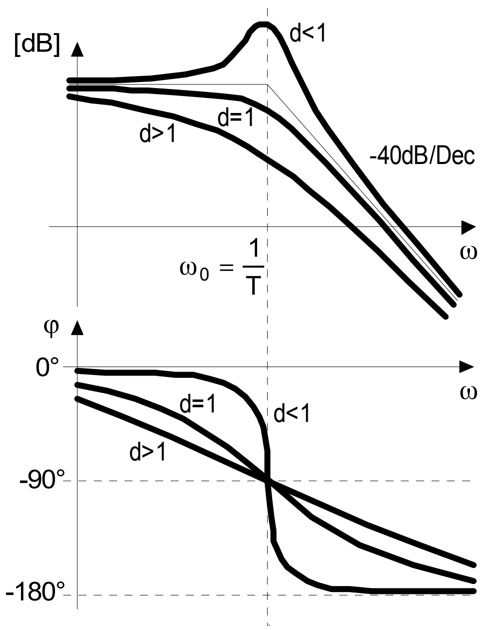
Step response

$$h(t) = K - \frac{K}{\sqrt{1-d^2}} e^{-(d/T)t} \sin\left[\frac{\sqrt{1-d^2}}{T} t + \arctan\left(\frac{\sqrt{1-d^2}}{d}\right)\right]$$

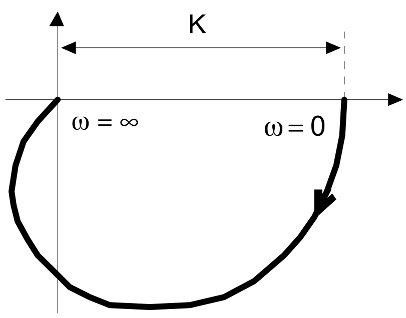


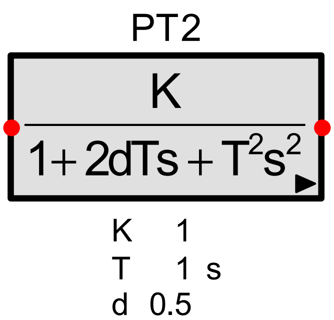


Magnitude and phase response



Polar plot





For d ≤ 1 we get:

$$G(s) = \frac{K}{(1+T_1 s)(1+T_2 s)}$$

$$T_1 = T\left(d + \sqrt{d^2-1}\right)$$
$$T_2 = T\left(d - \sqrt{d^2-1}\right)$$

13.2.8 Second order delay element, non oscillating (PT1T2)

This element corresponds mathematically to the PT2 element, but the damping is always greater or equal to 1. This is the overdamped case, where the denominator of the transfer function can be decomposed in two linear factors with the time constants T1 and T1.

Parameters: K, T1 >= 0, T2 >= 0, Y0,Y0'
If T1 or T2 = 0, Y0' does not have any effect
If both T1 and T2 = 0, Y0 and Y0' do not have any effect

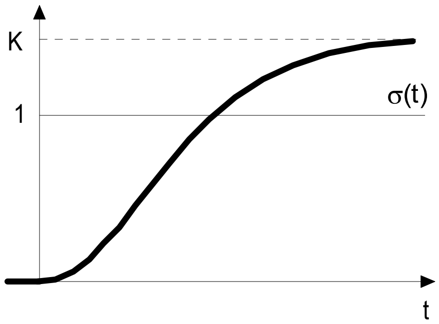
Functions

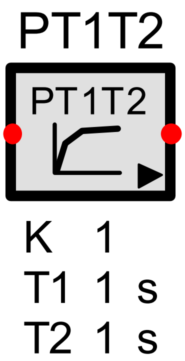
$$y + \dot{y}(T_1 + T_2) + \ddot{y}T_1T_2 = Ku$$

$$G(s) = \frac{K}{(1+sT_1)(1+sT_2)}$$

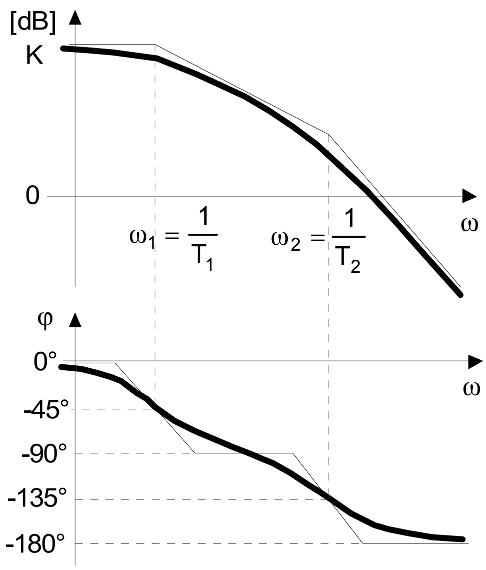
Step response

$$h(t) = K\left(1 - \frac{T_1e^{-t/T_1} - T_2e^{-t/T_2}}{T_1 - T_2}\right)$$

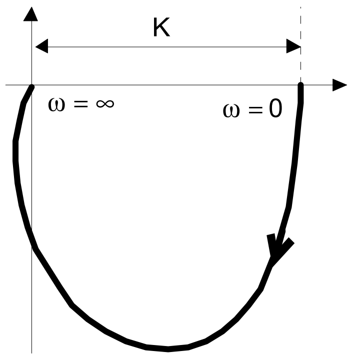


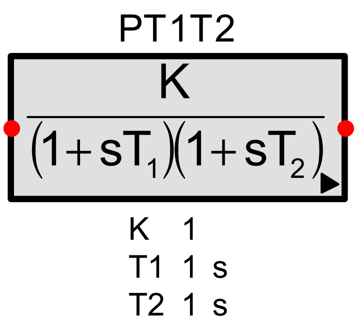


Magnitude and phase response



Polar plot





13.2.9 Delay element n-th order (PTn)

This element has n identical time constants. The greater n the flatter the step response.

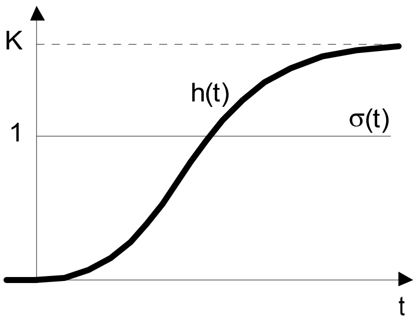
Parameters: K, T > 0, n >= 0, n-1 initial values

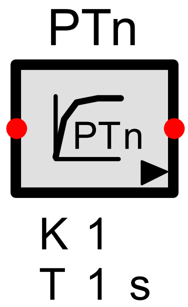
Note: The Ptn element in SimApp version 1.x corresponds to the current G(s) element.

Functions

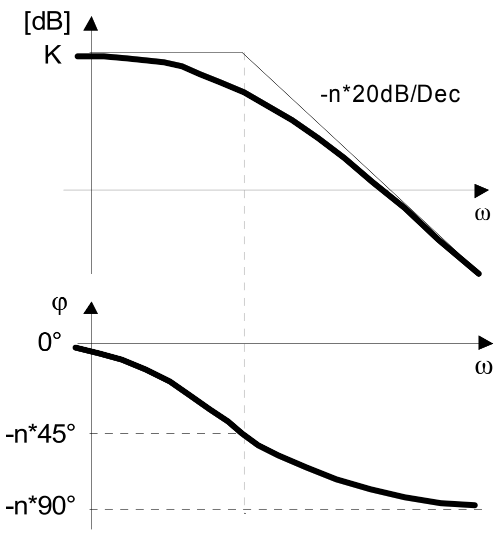
$$G(s) = \frac{K}{(1+Ts)^n}; \quad n \geq 0$$

Step response

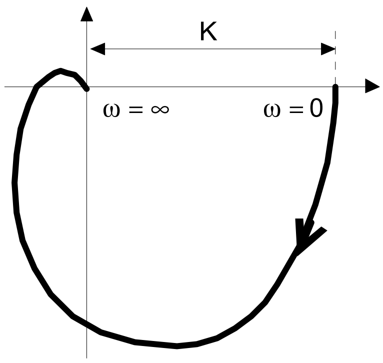


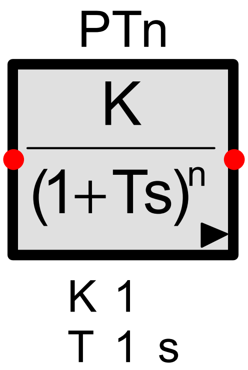


Magnitude and phase response



Polar plot





13.2.10 Lead/Lag element

This element is a first order rational element. Dependent on the ratio of T1 to T2 it can be configured as a leading or lagging phase element.

Parameters: K, T1 >= 0, T2 > 0, Y0

Do not confuse the lead/lag element with the lead/lag controller, which has a second order transfer function. (In SimApp version 1.x the controller was called lead/lag element.)

Functions

$$y + T_2\dot{y} = K(u + T_1\dot{u})$$

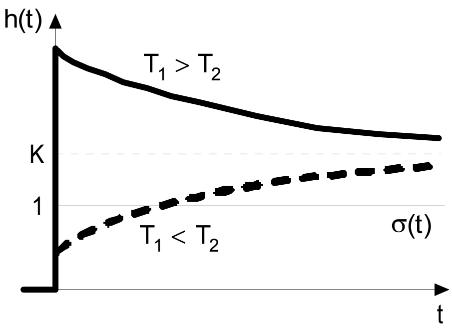
$$G(s) = K\frac{1 + T_1 s}{1 + T_2 s}$$

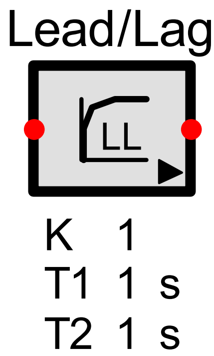
Leading: $T_1 > T_2$

Lagging: $T_1 < T_2$

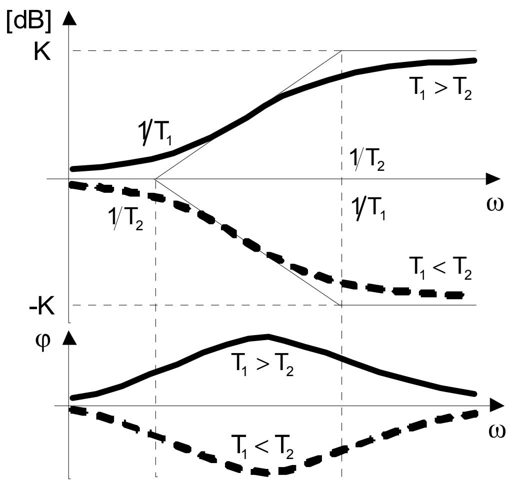
Step response

$$h(t) = K\left[1 + (T_1 - T_2)e^{-\frac{t}{T_2}}\right]$$

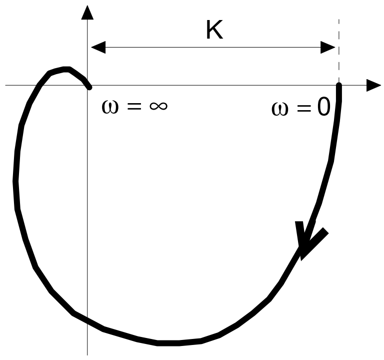


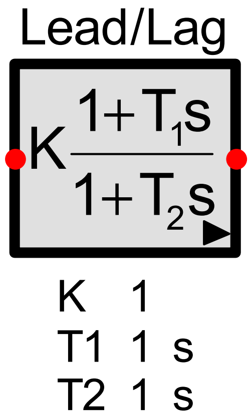


Magnitude and phase response



Polar plot





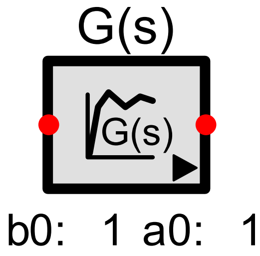
13.2.11 Rational transfer element (G(s))

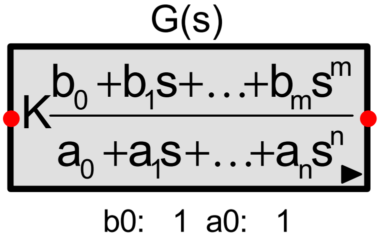
The G(s) element is the general form of a linear transfer element Y(s) = G(s) U(s) where G(s) is a rational function

Parameters: K, m>=0, n >=0, coefficients ai, bi and initial values

Transfer function:

$$G(s) = K\frac{b_0 + b_1 s + \ldots + b_m s^m}{a_0 + a_1 s + \ldots + a_n s^n}\ ; \quad a_n <> 0 \ \text{ and } \ m <= n$$





P-, I-, PT1-, PT2- and PT1T1 elements are special cases of the G(s) element. Since the nominator order m must not be greater than the denominator order n, the element can not have a differential behavior. Magnitude and phase response, polar plot and step response depend on the order of the polynomials and the choice of the coefficients.

Note: In SimApp version 1.x this element was called Ptn element.

13.2.12 Dead time (PTt)

The dead time element delays the input signal by the dead time Tt, but does not change the signal shape.

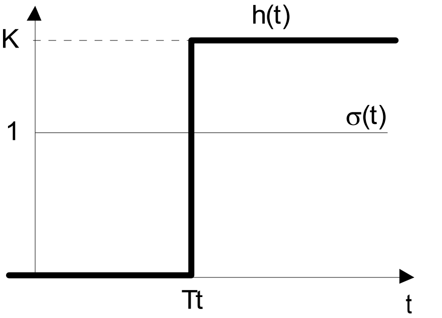
Functions

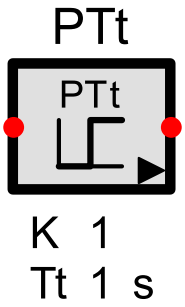
$$y(t) = Ku(t - T_t)$$

$$G(s) = Ke^{-sT_t}$$

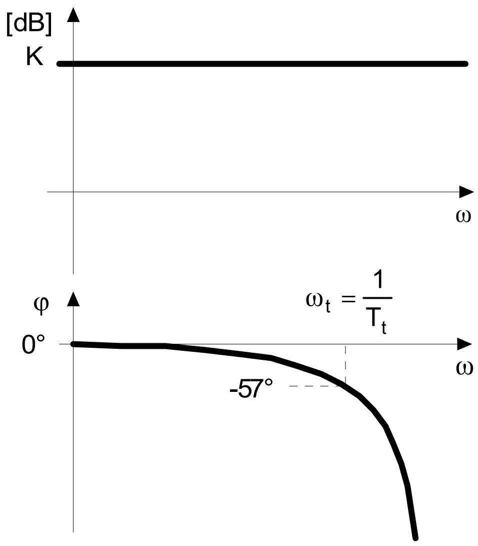
Step response

$$h(t) = K\sigma(t - T_t)$$

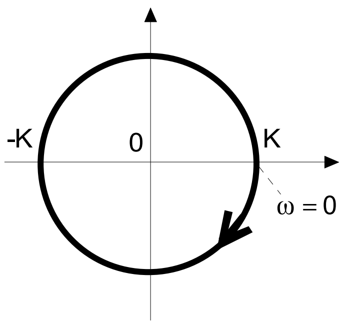


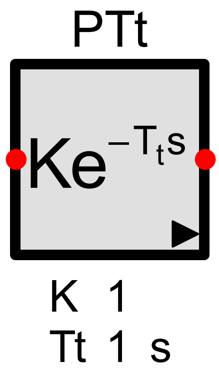


Magnitude and phase response



Polar plot





Padé-Allpass

The transfer function of the dead time element has neither a nominator polynomial nor a denominator polynomial. Since poles and zeros are not defined, eigenvalues do not exist and SimApp will not try to calculate them.

To eliminate this disadvantage the dead time element must be described by state variables. But since the dead time element can not be described by a finite number of state variables, we must use an approximation.
SimApp offers two approximations due to H.E. Padé [1].

Padé-Allpass

The rational transfer function of the Padé allpass is

$$G(T_t s) = \frac{1 + \sum_{i=1}^{n} (-1)^i a_i T_t^i s^i}{1 + \sum_{i=1}^{n} a_i T_t^i s^i} \quad \text{where} \quad a_i = \binom{n}{i} \frac{1}{2n(2n-1)\ldots(2n-i+1)} \quad \text{i= 1,...,n}$$

The allpass shows good results in the frequency domain. The magnitude characteristic is exact and the phase characteristic matches that of the pure time delay up to a frequency of $10\omega_t$. But in the time domain the step response is rather unpleasant, the initial value depends on the order n being equal to 1 or –1 (although it should be 0) followed by an intensive oscillation.

By means of the second approximation you can obtain better results in the time domain.

Padé-Approximation

The rational transfer function of the Padé approximation is

$$G(T_t s) = \frac{1 + \sum_{i=1}^{n-1} b_i T_t^i s^i}{1 + \sum_{i=1}^{n} a_i T_t^i s^i} \quad \text{where} \quad a_i = \binom{n}{i} \frac{1}{(2n-1)(2n-2)...(2n-i)} \quad \text{i = 1,...,n}$$

$$b_i = (-1)^i \binom{n-1}{i} \frac{1}{(2n-1)(2n-2)...(2n-i)} \quad \text{i = 1,...,n-1}$$

The Padé approximation does not have the exact allpass characteristic of the Padé allpass. But the step response is much better. The initial value now is zero.

Which method to use?

If you perform time simulations, you should use the ideal dead time element without any approximation. The response is absolutely exact without any distortions or over- and undershoots. But the need of buffer memory can be rather high for long dead times. If your computer lacks main memory, an approximation might be indispensable.
If the dead time element is close to the input of the system where effective steps and spikes can occur, you should use the Padé approximation because of its better time response.

At frequency simulations the ideal dead time element yields exact values for the magnitude and the phase curve. For the calculation of the eigenvalues it is replaced by a unit-gain element without any phase lag. Therefore, the eigenvalues are not computed. For valid eigenvalues you should use an approximation of Padé. The Padé allpass shows a good frequency response. Note that the poles and zeros of the Padé allpass affect the whole system and influence the number and magnitudes of eigenvalues.

13.2.13 All-pass element, type I (PTa1)

The PTa element has a constant magnitude characteristic and is independent of the frequency. The step response shows a negative undershoot near t = 0. Systems with allpass characteristics are difficult to control.

Parameters: K, Ta >= 0, Y0

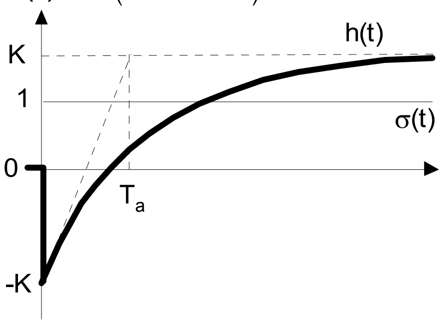
Functions

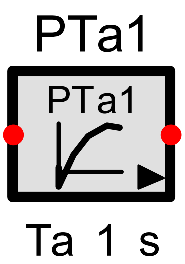
$$y + T_a\dot{y} = K(u - T_a\dot{u})$$

$$G(s) = K\frac{1 - sT_a}{1 + sT_a}$$

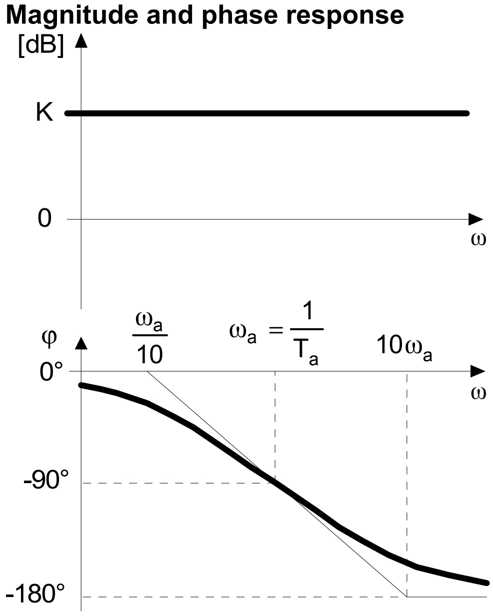
Step response

$$h(t) = K\left(1 - 2e^{-t/T_a}\right)$$

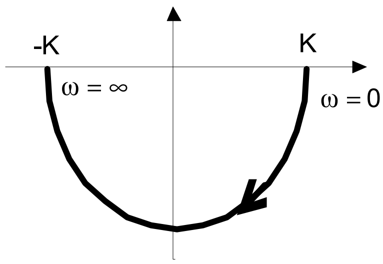


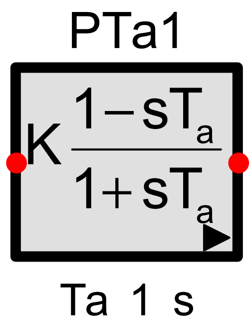


Magnitude and phase response



Polar plot





13.2.14 All-pass element type II (PTa2)

This element has a similar undershoot as the first order all-pass. But in the second part of the step response it shows a dying oscillation, dependent on the damping in the denominator.

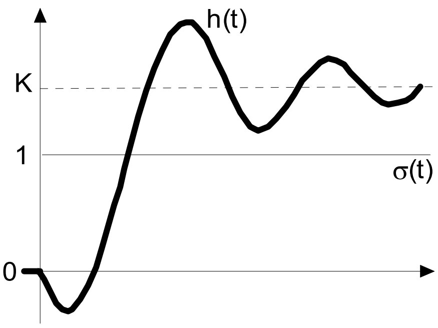
Parameters: K, d >= 0, T1 >= 0, T2 >= 0, Y0
If T2 = 0 then must be T1 = 0.

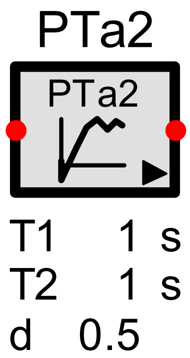
Functions

$$y + 2dT_2\dot{y} + T_2\ddot{y} = K(u - T_1\dot{u})$$

$$G(s) = K\frac{1 - T_1 s}{1 + 2dT_2 s + T_2^2 s^2}$$

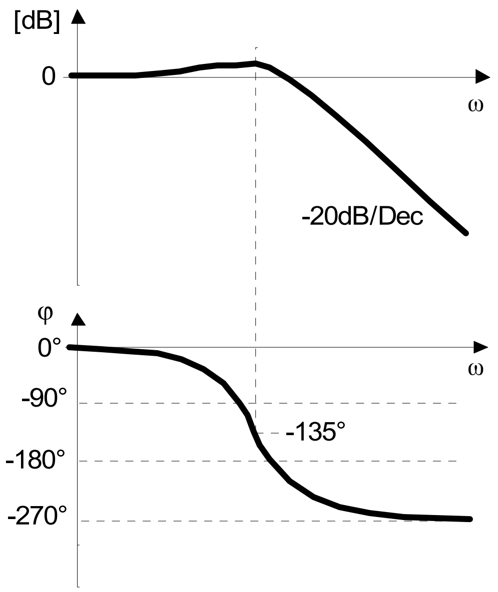
Step response



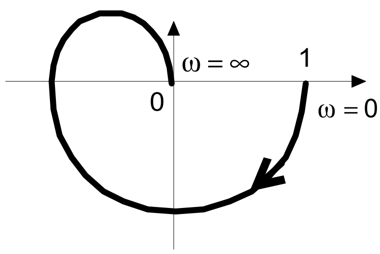


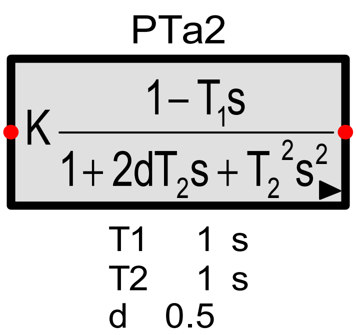
Magnitude and phase response

Bsp. für T1=1, T2=1; d=0.5



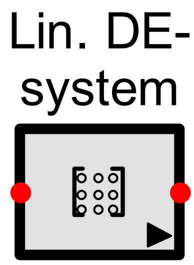
Polar plot





13.2.15 Linear differential equation system

Free configurable linear and time invariant differential equation system.



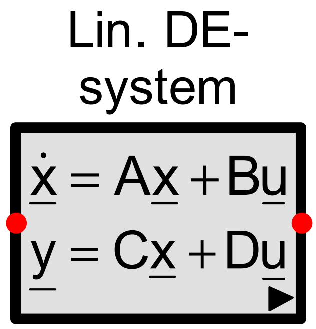
Vector form

System matrix equation: $\dot{\underline{x}}(t) = A\underline{x}(t) + B\underline{u}(t)$

Output matrix equation: $\underline{y}(t) = C\underline{x}(t) + D\underline{u}(t)$

Transfer function

$$G(s) = C(s)[sI - A(s)]^{-1}B(s) + D(s) \quad s = j\omega$$



The system has n states, p input quantities and q output quantities.

$\underline{x}$	State vector (n x 1)
$\underline{u}$	Input vector (p x 1)
$\underline{y}$	Output vector (q x 1)
A	System matrix (n x n)
B	Input matrix (n x p)
C	Output matrix (q x n)
D	Direct transmission matrix (q x p)

Parameters: n<=50, p<=50, q<= 50, coefficients Aik, Bik, Cik, Dik,
Initial values xn(i)

Matrix representation

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ . \\ \dot{x}_n \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} & . & A_{1n} \\ A_{21} & A_{22} & . & A_{2n} \\ . & . & . & . \\ A_{n1} & A_{n2} & . & A_{nn} \end{bmatrix} * \begin{bmatrix} x_1 \\ x_2 \\ . \\ x_n \end{bmatrix} + \begin{bmatrix} B_{11} & B_{12} & . & B_{1p} \\ B_{21} & B_{22} & . & B_{2p} \\ . & . & . & . \\ B_{n1} & B_{n2} & . & B_{np} \end{bmatrix} * \begin{bmatrix} u_1 \\ u_2 \\ . \\ u_p \end{bmatrix}$$

$$\begin{bmatrix} y_1 \\ y_2 \\ . \\ y_q \end{bmatrix} = \begin{bmatrix} C_{11} & C_{12} & . & C_{1n} \\ C_{21} & C_{22} & . & C_{2n} \\ . & . & . & . \\ C_{q1} & C_{q2} & . & C_{qn} \end{bmatrix} * \begin{bmatrix} x_1 \\ x_2 \\ . \\ x_n \end{bmatrix} + \begin{bmatrix} D_{11} & D_{12} & . & D_{1p} \\ D_{21} & D_{22} & . & D_{2p} \\ . & . & . & . \\ D_{q1} & D_{q2} & . & D_{qp} \end{bmatrix} * \begin{bmatrix} u_1 \\ u_2 \\ . \\ u_p \end{bmatrix}$$

In real systems the direct transmission matrix is mostly D = 0.

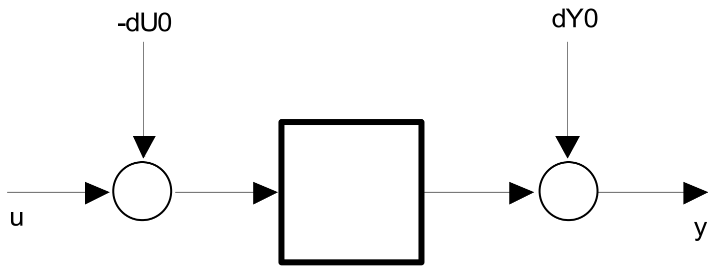
Due to the linear form, no formula evaluation must be done during simulation, so there is no speed loss compared with other elements.

13.3 Nonlinear elements

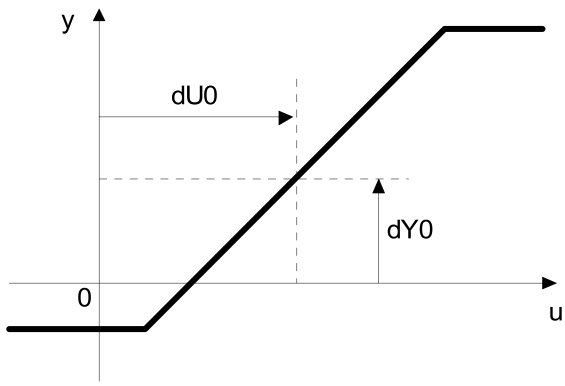
For time simulations it makes no difference if you take nonlinear elements or pure linear elements. This is in contrast to the frequency simulation where you may only take linear elements (incl. time discrete elements). The most important nonlinear functions are available as own elements. By means of the multi-function elements you can apply numerous additional, frequently used transfer functions. If you cannot find the proper function, try to combine standard elements or use the user characteristic element by which you can realize any characteristic.

Input and output shift

The characteristic of many nonlinear elements can be shifted in the coordinate system. By means of a constant value subtracted from the input signal, the characteristic is shifted to the right and by means of a constant value added to the output signal, the characteristic is shifted upwards. The parameters for the input and output shift are not shown below the block symbol in the drawing if they are equal to zero. You can edit them in the simulation properties dialog box of the corresponding blocks.



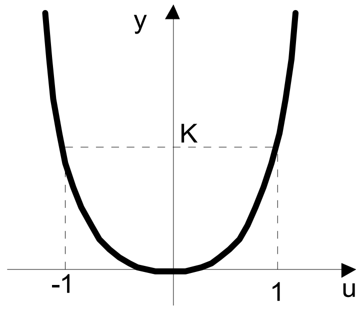
Example: The saturation characteristic is shifted by dU0 and dY0

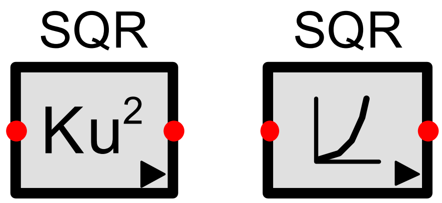


13.3.1 Square

Parameters: K, dU0, dY0

Functional relation: $y = Ku^2$

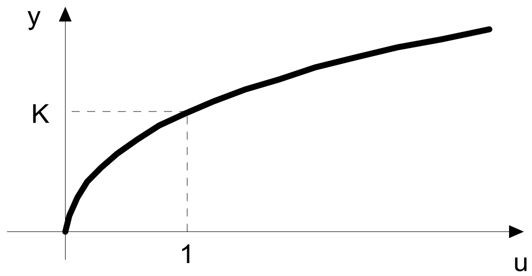


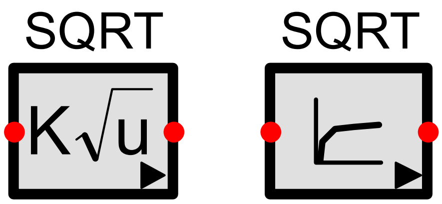


13.3.2 Square-root

Parameters: K, dU0, dY0

Functional relation: $y = K\sqrt{u}$



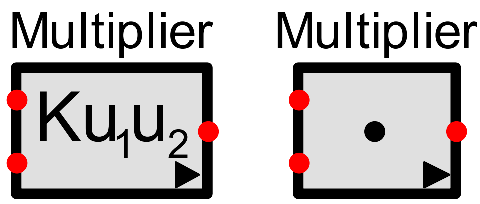


13.3.3 Multiplier (Product)

The multiplying element computes the product of two input quantities and gain K.

Parameters: K

Functional relation: $y = K\, u_1\, u_2$

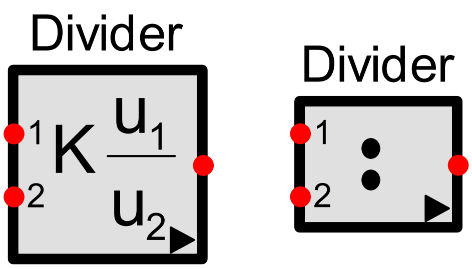


13.3.4 Divider

The dividing element divides the input value u_1 by the input value u_2 and multiplies the result by gain K

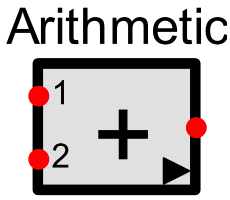
Parameters: K

Functional relation: $y = K\frac{u_1}{u_2}$



13.3.5 Arithmetic element with multiple inputs

This element supports basic arithmetic functions of multiple input values as addition, subtraction, multiplication and division.



Parameters: Number of inputs, function

The number of input nodes can freely be chosen between 2 and 50. Unused inputs (having no connection) are ignored.

The sign of the input values can be changed in the pop-up menu of the associated signal lines or by pressing the + or - key in the numeric keypad when a signal line is selected.

Addition and Subtraction

Functional relation: $y = \pm u_1 \pm u_2 \pm \ldots \pm u_n$

The subtraction of inputs is achieved by inverting the signal in the signal line.

Multiplication

Functional relation: $y = \pm u_1 \times \pm u_2 \ldots \times \pm u_n$

The negative weighting (minus sign) of factorials is achieved by inverting the signal in the signal line.

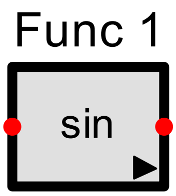
Division

Functional relation: $y = \pm u_1 \div \pm u_2 \ldots \div \pm u_n$

Dividend is the signal of the first, the top most input node. All other inputs are divisors. The sign of the dividend and the divisors can be set by changing the sign of the input signal line. (see subtraction.)

13.3.6 Function element with single input

Only the most important functions have their own element. This element provides additional 29 single input functions. There are linear and nonlinear functions. Linear functions may also be used in frequency simulations.



Parameters: A (used for 3 functions only), U0, Y0

Trigonometric and inverse functions

sin	cos	tan	Cotan	arcsin	arccos	arctan

Hyperbolic and inverse functions

cosh	sinh	tanh	arcosh	arsinh	artanh

Exponential and logarithmic functions

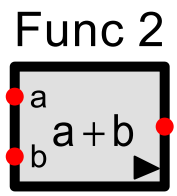
exp	2^x	10^x	x^2	x^3	ln	Lb	lg	x^A	A^x	$\log_A$	$\sqrt{x}$

Miscellaneous functions

$\lvert x \rvert$	sign	Deg > Rad	Rad > Deg	Cycle > Rad	Rad > Cycle

13.3.7 Function element with double input

This element provides 10 dual input functions



a + b	a - b	a x b	a / b
a^b	$a^{1/b}$	$\sqrt{a^2+b^2}$	
arctan(a/b)	min(a,b)	max(a,b)	

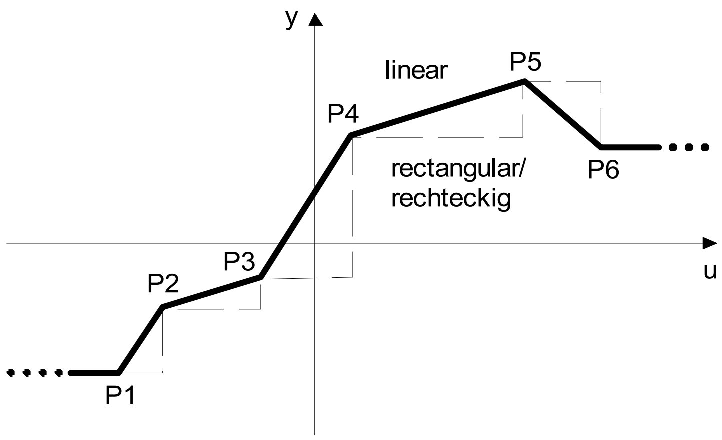
a is the lower and b the upper input node.

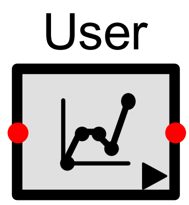
13.3.8 User characteristic

This element has a static input-output characteristic with up to 10'000 points. You can choose between a linear and a rectangular interpolation.

Parameters: dU0,dY0

Functional relation: y = f(P1,P2, ...,Pn,t); n = 1..10'000





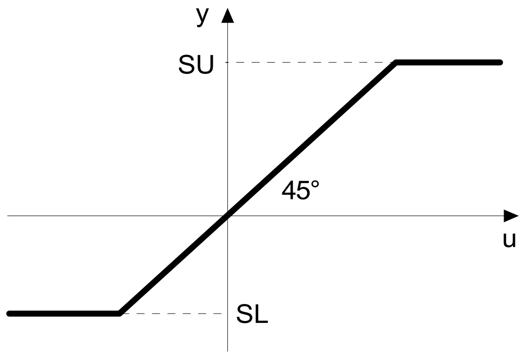
Steps can be achieved by setting the same input value u for two adjacent points. The output values of the first and the last point are also valid beyond the given characteristic.

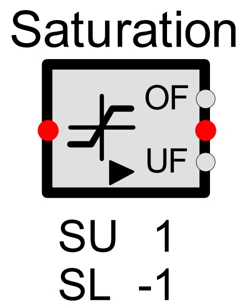
The values are stored in the element, but can also be saved in text files. This enables the data exchange with other applications. For file and data format description please see *User Source*.

13.3.9 Saturation

The saturation element is the most commonly encountered non-linearity in control engineering. It limits the input value to the upper and lower saturation values SU and SL.

Parameters: SU, SL, V0, dU0, dY0



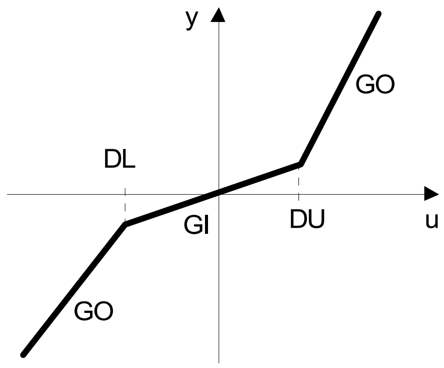


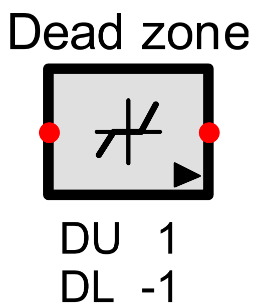
If the input signal is greater than SU, the overflow output (OF) is set to logic 1. If the input signal is lower than SL, the underflow output (UF) is set to logic 1. But the analog output signal stays always between the limits.

13.3.10 Dead zone

The dead zone element suppresses small signals within the dead zone DL..DU. Between DL and DU the gain is GI, GO otherwise. Default value for GI is 0. So the output remains 0 until the amount of the input signal has crossed the thresholds DL and DU.

Parameters: DU, DL, GI, GO, dU0, dY0



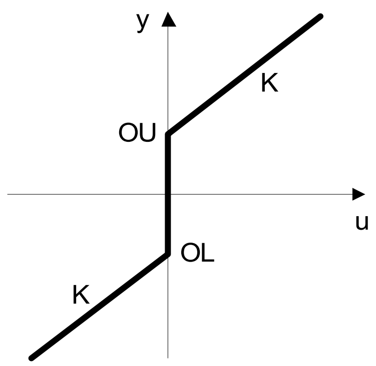


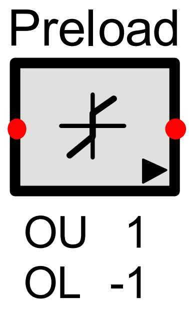
13.3.11 Preload (Offset)

This element adds a positive or negative offset to the input value.

Parameters: OU, OL, K, dU0, dY0

Functional relation: u >=0: y = OU + Ku
u < 0: y = OL + Ku

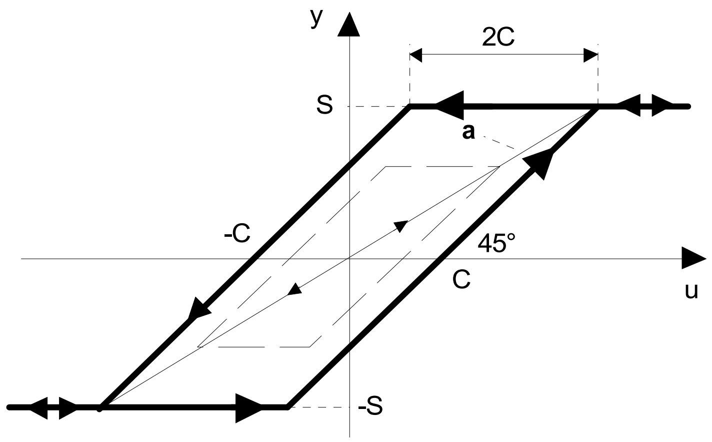


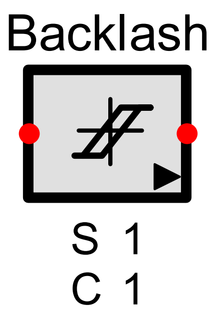


13.3.12 Backlash

S is the maximum saturation value. If the input signal can not reach S, the characteristic forms a smaller backlash. If the input starts at zero, the characteristic follows the new curve a.

Parameters: S, C, dU0, dY0

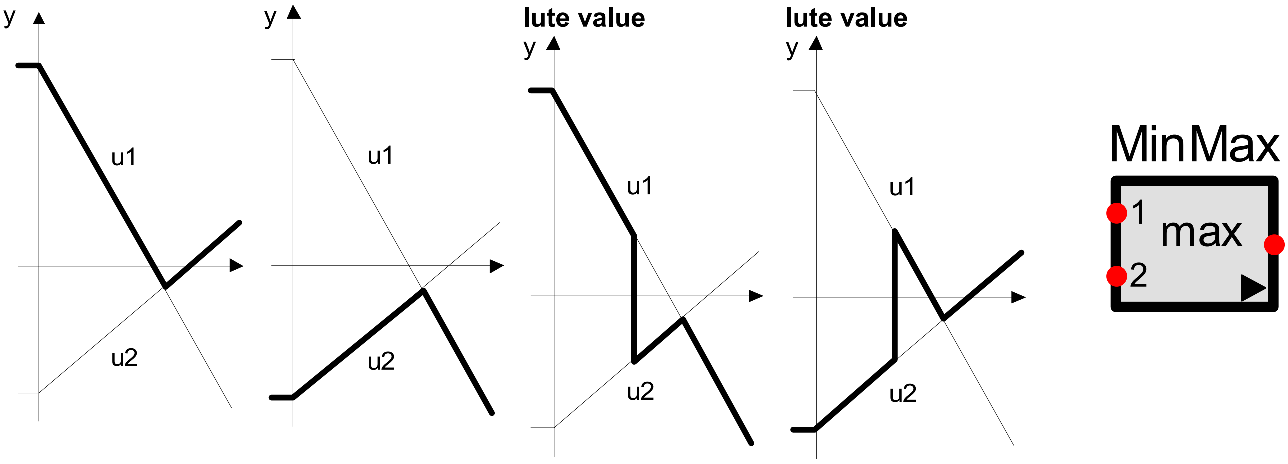




13.3.13 Minimum/Maximum (MinMax)

This element applies those input signal at its output which is currently the smallest or the largest. You can also use absolute values for minimum-maximum determination.

Maximum **Minimum** **Maximum of absolute value** **Minimum of absolute value**

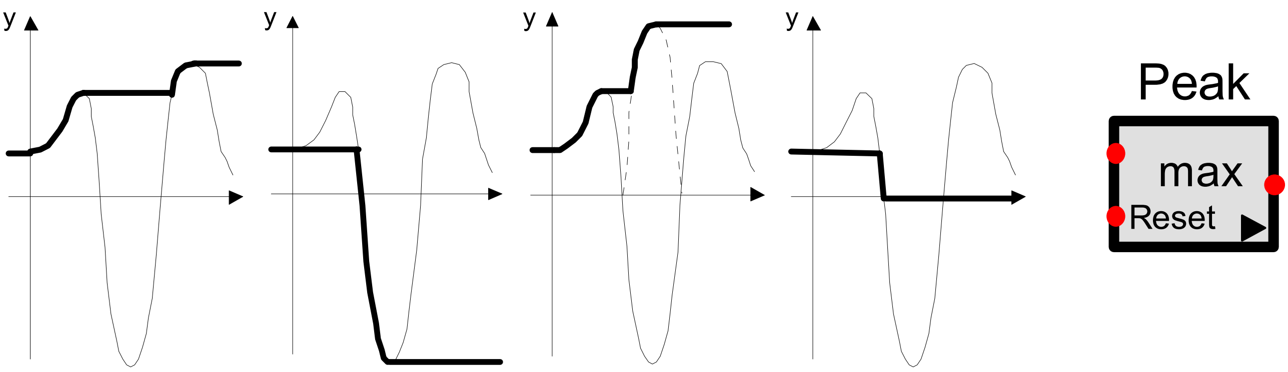


13.3.14 Peak detector

This element tracks the input signal and holds the maximum value which has occurred since simulation start or since the last time the reset signal has been applied. On the rising edge of the reset signal, the peak detector is reset to the current input signal. As long as reset is set the output follows the input.

Parameters: Vth>= 0

Maximum peak **Minimum peak** **Maximum peak of absolute value** **Minimum peak of absolute value**



13.4 Actuators

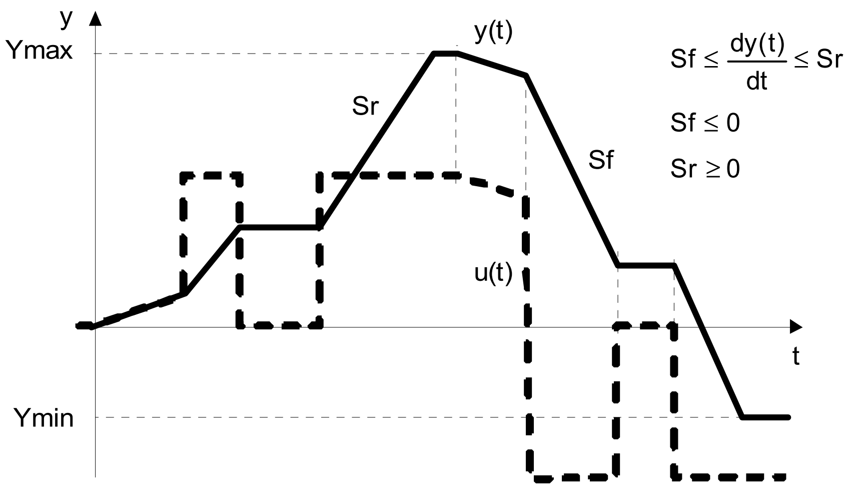
13.4.1 Rate limiter

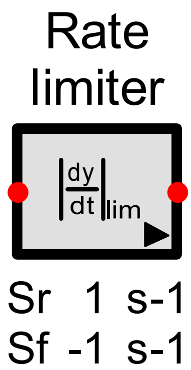
The rate limiter shows the characteristic of an actuator which is too slow to follow the input control signal.

It limits the rising and falling rates of the input signal, i.e. the first derivative of the signal passing through it, and the amount of the upper and lower saturation limits.

The initial value can be specified by user or tied to the input signal at time 0. In the second case, Y0 is ignored.

Parameters: Sr>=0, Sf<=0, Ymin, Ymax, Y0



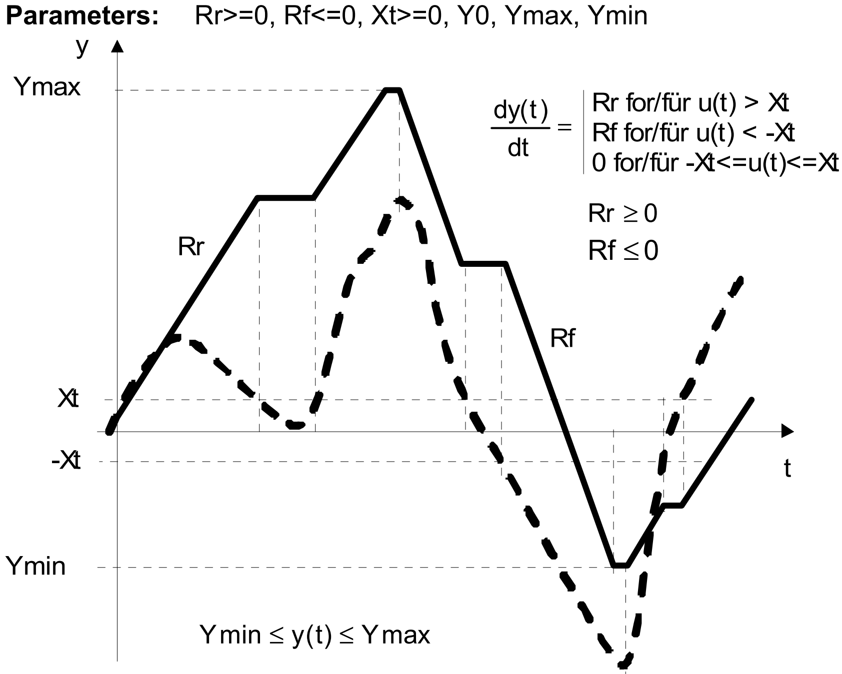


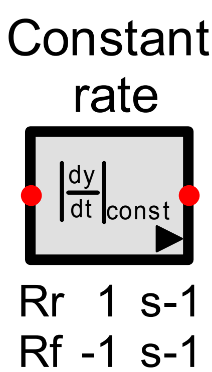
13.4.2 Constant rate

This element knows 3 output signal courses only. If the input signal is too small, the output does not change (dead zone), otherwise it rises or falls by a constant rate, dependent on the input signal being positive or negative. Finally, there is also a saturation that limits the output signal in both directions.

The initial value can be specified by user or tied to the input signal at time 0. In the second case, Y0 is ignored.

Parameters: Rr>=0, Rf<=0, Xt>=0, Y0, Ymax, Ymin



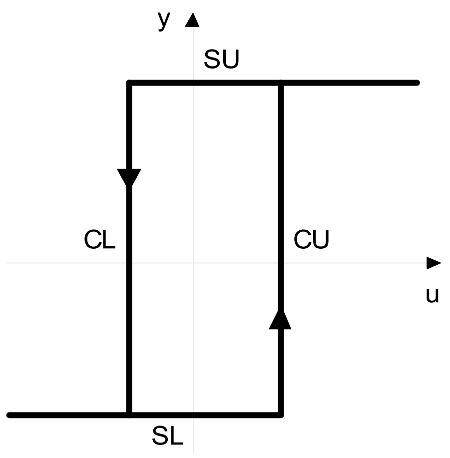


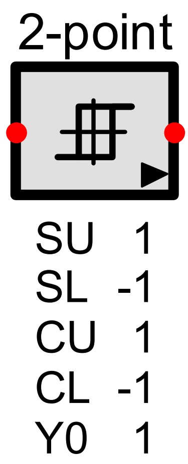
13.5 Controllers

13.5.1 2-point step controller

The output signal changes between two constant values.

Parameters: SU, SL, CU, CL, Y0, dU0, dY0



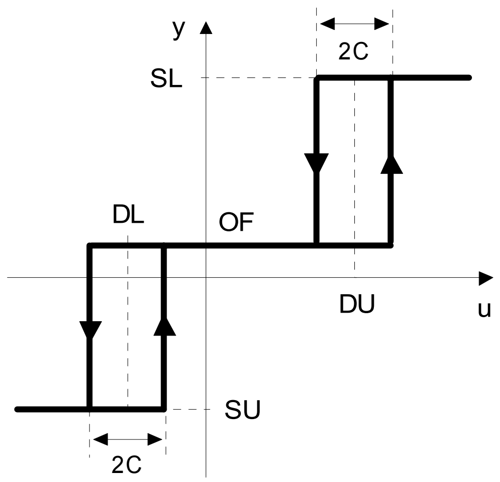


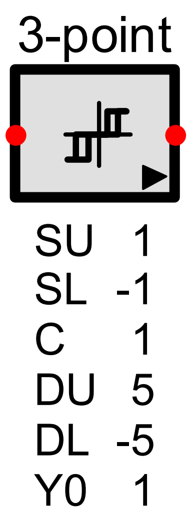
Y0 is the initial value at time t<= 0. May be different to SU and SL.

13.5.2 3-point step controller

The output signal changes between three constant values.

Parameters: SU, SL, C>=0, DU, DL, Y0, dU0, dY0





Y0 is the initial value at time t<= 0. May be different to SU,SL and OF.

13.5.3 Ideal PI controller (PI-i)

The ideal PI controller is designed for phase-lag compensations. It is used if the controlled system has an insufficient steady-state performance. It is not used if the system itself has a integral characteristic. The ideal PI controller eliminates the steady-state error in the step response.

Parameters: K, T > 0, Y0

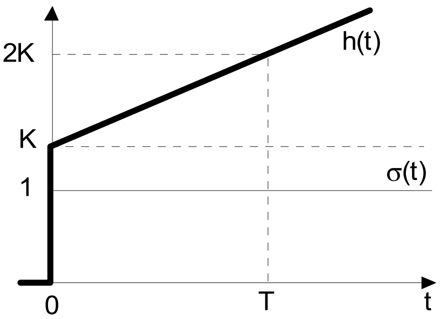
Functions

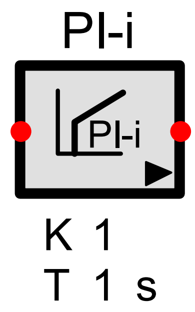
$$y = K\left(u + \frac{1}{T}\int_0^t u(\tau)d\tau\right)$$

$$G(s) = K\frac{1+sT}{sT}$$

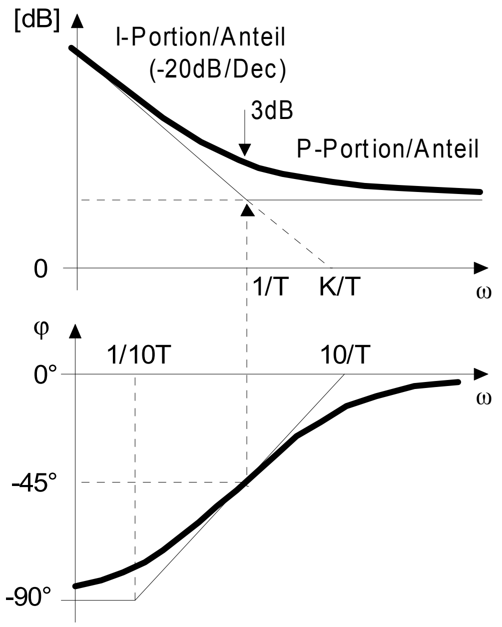
Step response

$$h(t) = K\left(1+\frac{t}{T}\right)$$

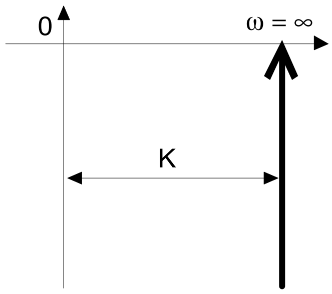


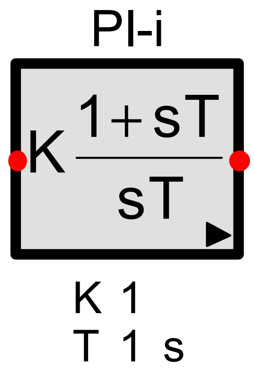


Magnitude and phase response



Polar plot





13.5.4 Modified PI controller (PI-m)

The modified PI controller is designed for phase-lag compensations. It is used if the controlled system has an insufficient steady-state performance. It is not used if the system itself has a integral characteristic.
The modified form of the PI controller is used for the same purpose as the ideal PI controller but can not achieve the steady-state accuracy of the ideal PI controller.

Parameters: K, TR >= 0, TN > 0, Y0

Functions

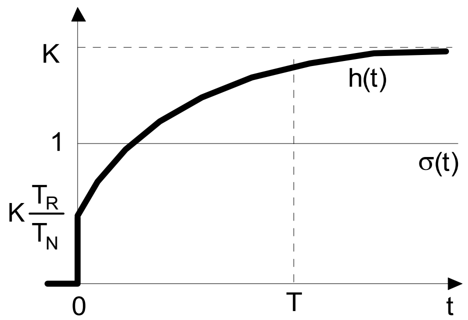
$$y + \dot{y}T_N = K(u + \dot{u}T_R)$$

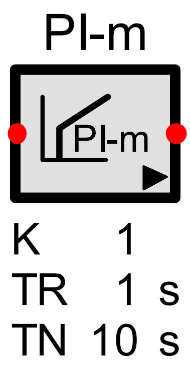
$$G(s) = K\frac{1+sT_R}{1+sT_N}$$

$$T_N \gg T_R$$

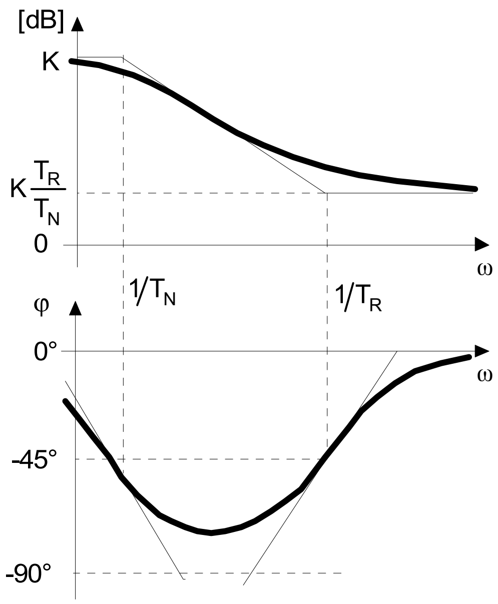
Step response

$$h(t) = K\left[1-(T_N - T_R)e^{-\frac{t}{T_N}}\right]$$

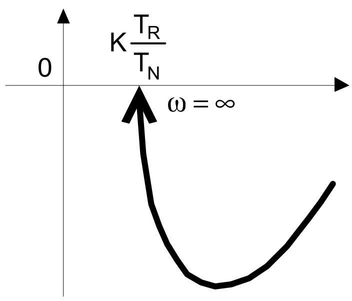


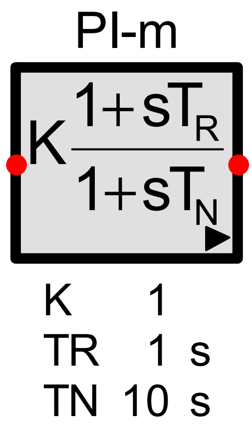


Magnitude and phase response



Polar plot





13.5.5 Ideal PD controller (PD-i)

The ideal PD controller is designed for phase-lead compensations. It is used if the transient response is insufficient or if the system is unstable (phase rise of 0° to 90°). It is not used if the controlled system itself does not have a integral characteristic.
The ideal PD controller is mostly used for educational purposes only. In practice it is difficult to realize and the D-term amplifies the noise in the control system.

Parameters: K, T >= 0, Y0

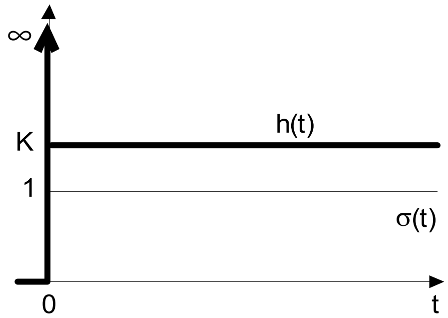
Functions

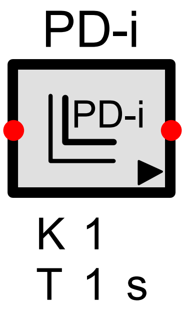
$$y = K(u + T\dot{u})$$

$$G(s) = K(1 + sT)$$

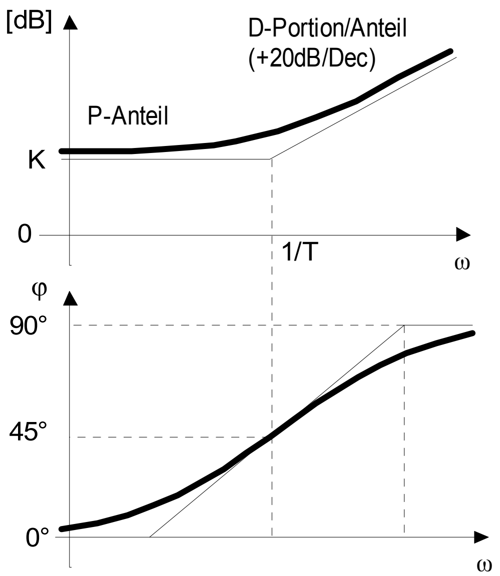
Step response

$$h(t) = K[\sigma(t) + T\delta(t)]$$

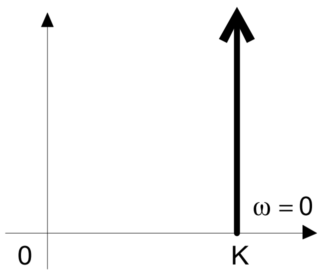


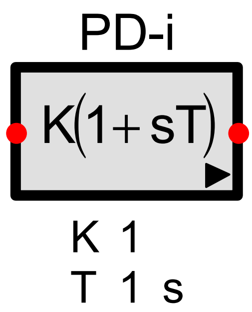


Magnitude and phase response



Polar plot





13.5.6 Real PD controller (PD-r)

The real PD controller is designed for phase-lead compensations. It is used if the transient response is insufficient or if the system is unstable (phase rise of 0° to 90°). It is not used if the system itself does not have a integral characteristic.
The real implementation of the PD controller has similar effects as the ideal PD controller, but it is easy to realize and keeps the noise within tolerable limits.

Parameters: K, TR >= 0, TN > 0, Y0

Functions

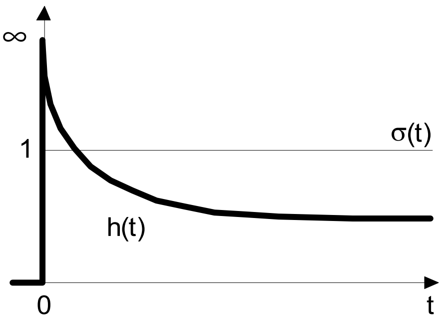
$$y + \dot{y}T_N = K(u + \dot{u}T_R)$$

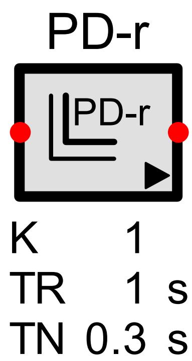
$$G(s) = K\frac{1+sT_R}{1+sT_N}$$

$$T_N < T_R$$

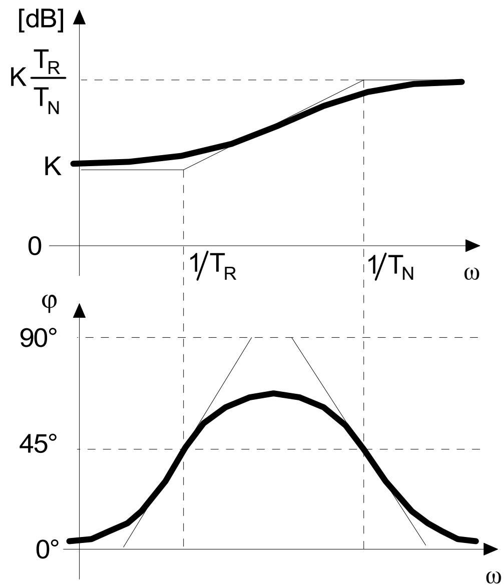
Step response

$$h(t) = K\left[1 + (T_R - T_N)e^{-\frac{t}{T_N}}\right]$$

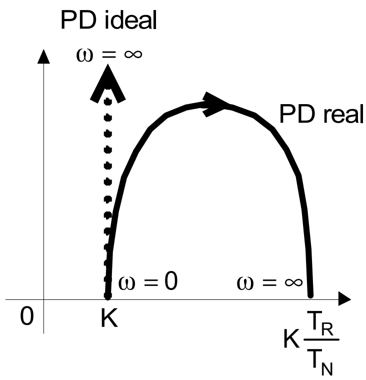


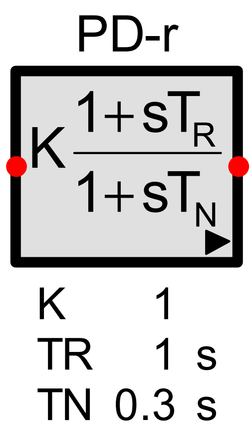


Magnitude and phase response



Polar plot





13.5.7 Ideal PID controller type I (PID-I)

PID control structures are used if the steady-state and the transient performance of the control system are not to standard.
The first form of the ideal PID controller corresponds to the parallel connection of a unit-gain, an integrator and an ideal differentiator followed by a saturation element. As parameters, the gain, the rate time and the reset time are used. In addition the falling edge of the differentiator can be delayed by TD.

The initial value of the output signal is determined by the input value, the gain and the integrator's initial value Y0I: Y0:= U0 * K(1 + Y0I);

The output signal can be limited to the range Ymin ... Ymax by two selectable anti-windup measures.

Windup appears due to the fact that the integral term increases too greatly during saturation.
Thus, during saturation the increase should be slowed down.

Anti-Windup-Hold
The integration is stopped when the saturation's input signal leaves the valid range (the upper or lower limit of the saturation). As soon as the signal is reduced the integrator is released.

Anti-Windup-Reset
If the saturation's input signal exceeds the limit, the integrator's output signal is reduced so that the sum of integrator, differentiator and unit-gain equals exactly the limit (Ymax or Ymin).

Parameters: K, TV >= 0, TN > 0, TD >= 0, Y0I, Ymax, Ymin

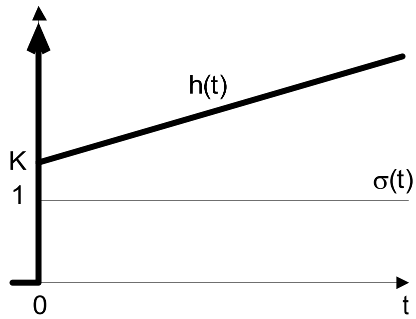
Functions

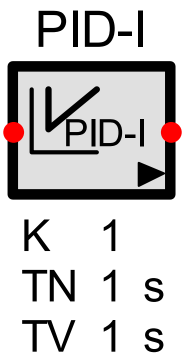
$$y = K\left(u + \frac{1}{T_N}\int_0^t u(\tau)d(\tau) + T_V\dot{u}\right)$$

$$G(s) = K\left(1 + \frac{1}{sT_N} + sT_V\right)$$

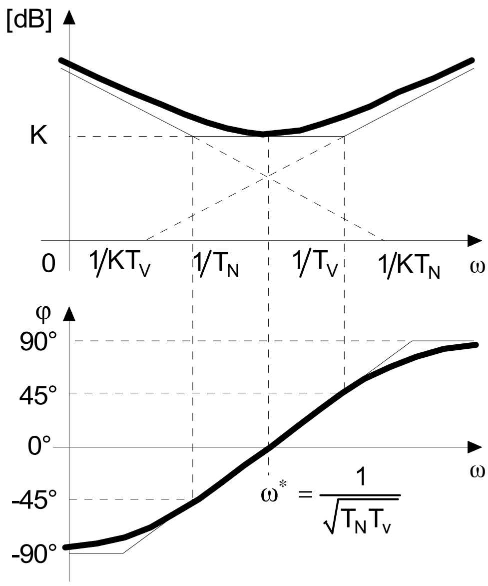
$$T_N > T_V$$

Step response

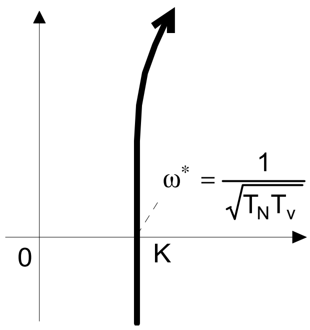


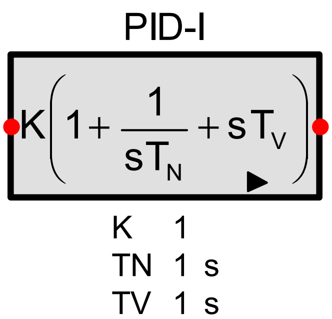


Magnitude and phase response

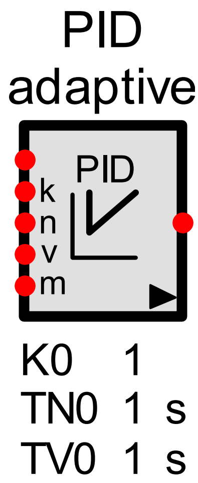


Polar plot





13.5.8 Adaptive PID controller



This controller corresponds to the ideal PID controller, type I, whereby the specified parameters K, TV and TN can be modified during simulation by control input signals. The parameter control can be multiplicative or additive. Thus, the actual parameter values are:

for multiplicative parameter control:
K = K0 * dK
TN = TNO * dTN
TV = TV0 * dTV

for additive parameter control:
K = K0 + dK
TN = TNO + dTN
TV = TV0 + dTV

With open control inputs (unconnected) the associated parameters keep their default values

The limits can also be multiplicative changed by means of the input signal at node m:
Ymin = Ymin0 * dYL
Ymax = Ymax0 * dYL
If the input node is open the limits are unchanged.

Parameters: K, TV >= 0, TN > 0, TD >= 0, Y0I, Ymax, Ymin

13.5.9 Ideal PID controller type II (PID-II)

PID control structures are used if the steady-state and the transient performance of the system are not to standard.
The second form of the ideal PID controller comes from the importance of the poles and zeros. The assignment of the parameters to ideal basic elements is not so obvious as for the type I controller.

Parameters: K, TR1 >= 0, TR2 >= 0, Y0

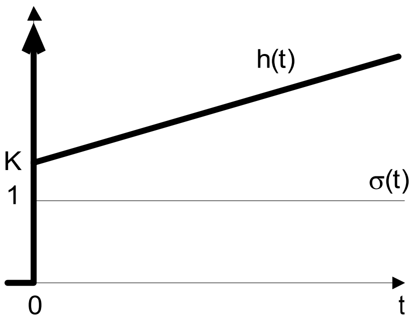
Functions

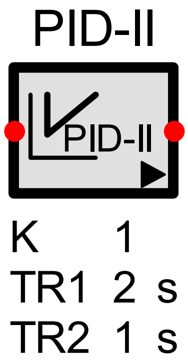
$$y = K\left[(T_{R1}+T_{R2})u + \int_0^t u(\tau)d\tau + \dot{u}T_{R1}T_{R2}\right]$$

$$G(s) = K\frac{(1+sT_{R1})(1+sT_{R2})}{s}$$

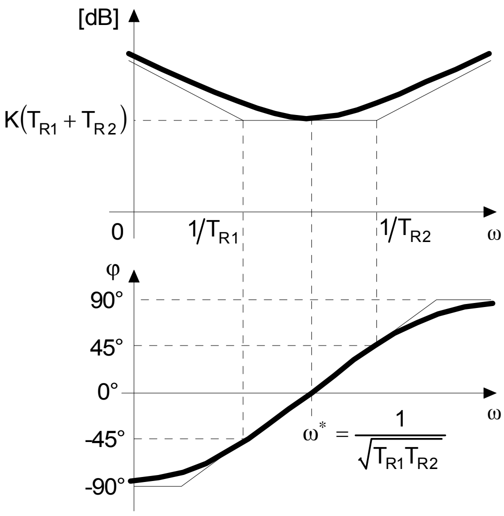
$$T_{R1} \geq T_{R2}$$

Step response

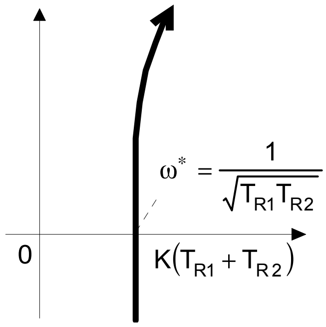


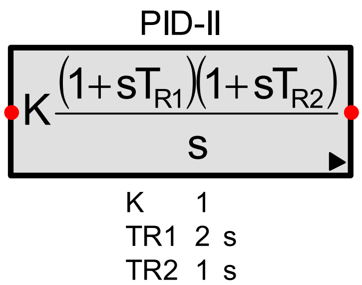


Magnitude and phase response



Polar plot





13.5.10 Real PID controller (PID-r)

The real PID controller lessens the influence of the differentiator in the high frequency range (noise) and can easily be realized.

Parameters: K, TR1 >= 0, TR2 >= 0, TN > 0

Functions

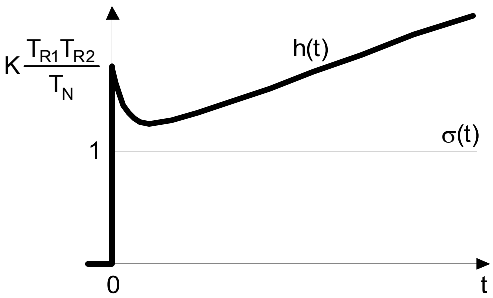
$$y + T_N \dot{y} = K\left[(T_{R1} + T_{R2})u + \int_0^t u(\tau)d\tau + T_{R1}T_{R2}\dot{u} \right]$$

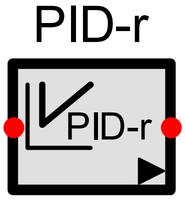
$$G(s) = K\frac{(1 + sT_{R1})(1 + sT_{R2})}{s(1 + sT_N)}$$

$$T_{R1} \geq T_{R2} > T_N$$

$$\omega^* = \frac{1}{\sqrt{T_{R1}T_{R2} - (T_{R1} + T_{R2})T_N}}$$

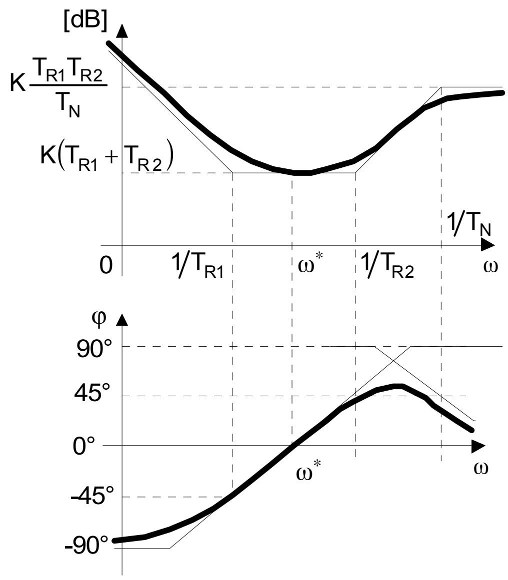
Step response



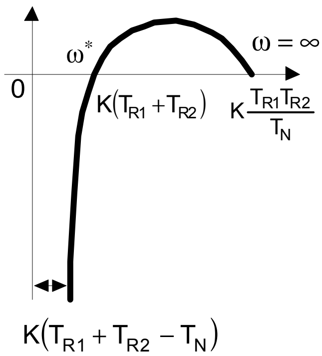


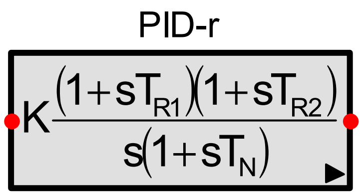
K 1
TR1 3 s
TR2 1 s
TN 0.5 s

Magnitude and phase response



Polar plot





K 1
TR1 3 s
TR2 1 s
TN 0.5 s

13.5.11 Modified PID controller (PIDm)

The modified PID controller is suitable for specific and flexible pole and zero cancellations in the controlled system.

Parameters: K, TR1 >= 0, TR2 >= 0, TN1 > 0, TN2 > 0, Y0

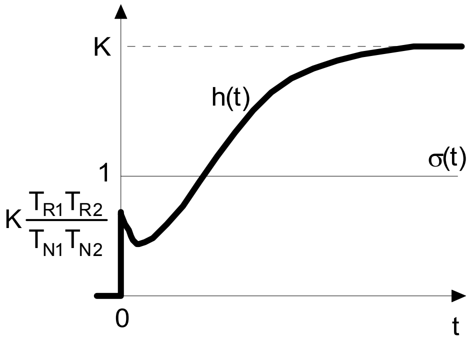
Functions

$$y + \dot{y}(T_{N1} + T_{N2}) + \ddot{y}T_{N1}T_{N2} = K[u + \dot{u}(T_{R1} + T_{R2}) + \ddot{y}T_{R1}T_{R2}]$$

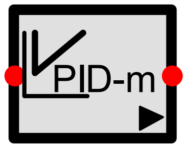
$$G(s) = K\frac{(1+sT_{R1})(1+sT_{R2})}{(1+sT_{N1})(1+sT_{N2})}$$

$$T_{N1} >> T_{R1} \geq T_{R2} > T_{N2}$$

Step response

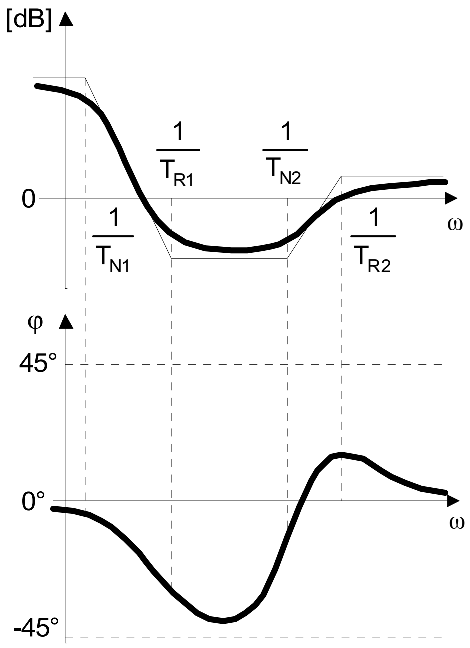


PID-m

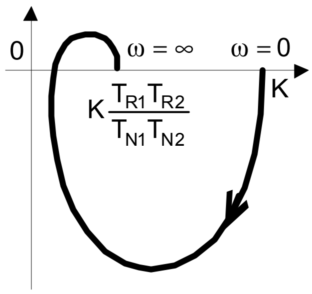


K	1	
TR1	2	s
TR2	1	s
TN1	10	s
TN2	0.5	s

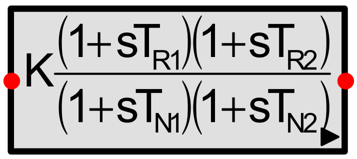
Magnitude and phase response



Polar plot



PID-m



K	1	
TR1	2	s
TR2	1	s
TN1	10	s
TN2	0.5	s

13.5.12 Generalized PID controller (PID-a)

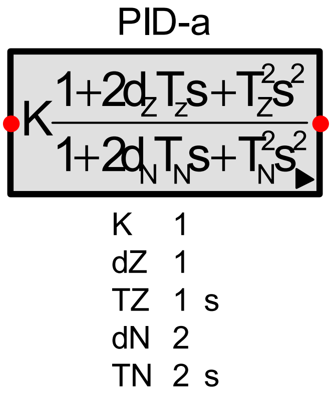
By means of the generalized PID controller complex roots in the polynomials of the nominator and denominator are realizable. However, all other structures are also realizable by using suitable parameters.
A typical step response, magnitude or phase curve or Nyquist plot can
not be shown due to the great variety.

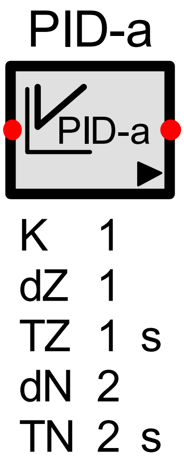
Parameters: K, dZ >= 0, TZ >= 0, dN >= 0, TN >= 0, Y0

Functions

$$y + \dot{y}2d_N T_N + \ddot{y}T_N^2 = K\left[u + \dot{u}2d_Z T_Z + \ddot{u}T_Z^2\right]$$

$$G(s) = K\frac{1 + 2d_Z T_z s + T_Z^2 s^2}{1 + 2d_N T_N s + T_N^2 s^2}$$





13.5.13 Lead/Lag controller

PID control structures are used if the steady-state and the transient performance of the controlled system are not to standard.
This controller corresponds to the modified PID controller, where as the separation into a leading and a lagging part is particularly clear.

Parameters: K, a >= 1, 0 < b < 1, T1 > 0, T2 >= 0, Y0

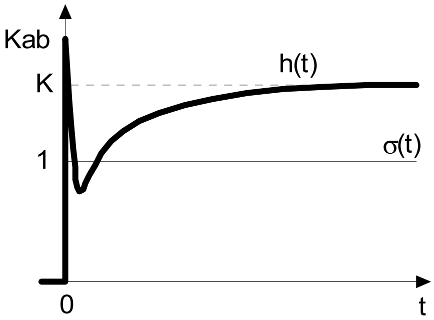
Functions

$$y + \dot{y}(T_1 + T_2) + \ddot{y}T_1T_2 = K[u + \dot{u}(aT_1 + bT_2) + \ddot{u}abT_1T_2]$$

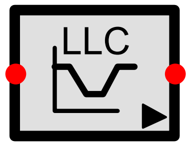
$$G(s) = K\left(\frac{1 + aT_1s}{1 + T_1s}\right)\left(\frac{1 + bT_2s}{1 + T_2s}\right)$$

$T_1 < T_2$
$a > 1$
$b < 1$

Step response

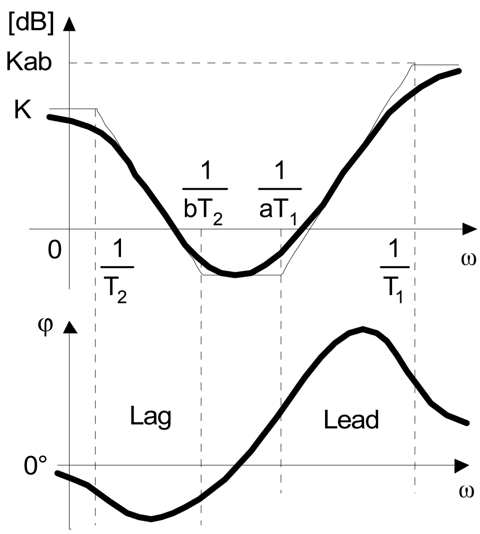


Lead/Lag controller

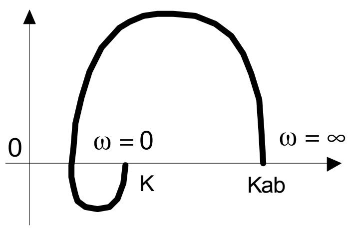


K 1
a 4
T1 0.1 s
b 0.5
T2 1 s

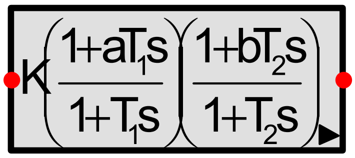
Magnitude and phase response



Polar plot



Lead/Lag controller



K 1
a 4
T1 0.1 s
b 0.5
T2 1 s

13.6 Time discrete transfer elements

13.6.1 Introduction

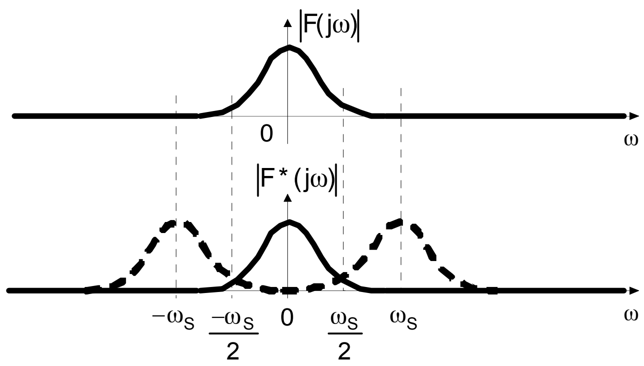
In a sampling control loop the controlled variable consists of a sequence of time discrete values at time t = kT (k = 0...n) which are supplied to a time discrete controller. The signal between the sampling points is ignored.
The time discrete controller transforms the input sequence into a appropriate output sequence which is also defined at the same sampling points t = kT only. As the following actuator or controlled system needs a continuous signal the output signal of the controller is hold between the sampling points and results in a step function which is finally smoothed by a low pass filter.

SimApp can simulate time discrete elements in the time domain as well as in the frequency domain, whereby however the following differences have to be considered:

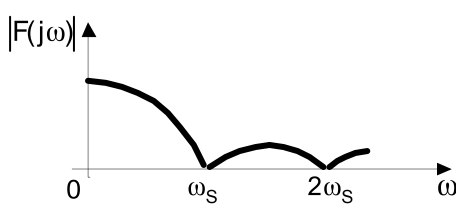
At time simulation all time discrete elements have a virtual sampler at its inputs and a virtual hold element at its outputs. Time discrete elements may be placed anywhere in the system. Sampling and hold elements are allowed but not required. In both cases the signal course is unchanged.

At frequency simulations however the transition from the time discrete to the time continuous processing and vice versa is very important. The sampling process produces constant weighted sideband signals by the periodic repetition of the spectrum of the time continuous signal. The hold element exhibits a low pass characteristic with frequency dependant weighting.

Sampling



Holding

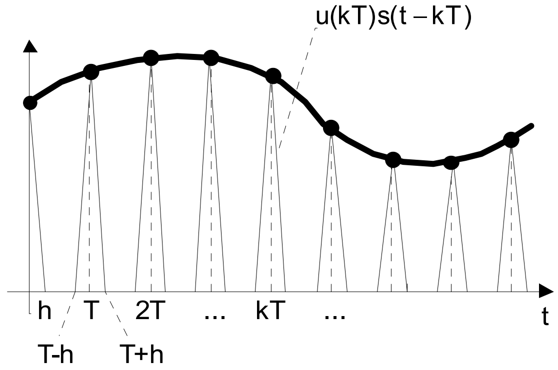


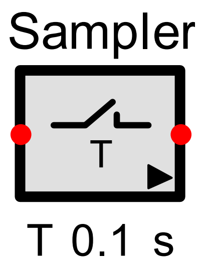
To get a valid frequency response, the sampling and the hold elements must be correctly placed. They may be omitted however if the simulated frequency range is significantly smaller than the sampling period.
As the system for frequency simulation is purely linear, the sampling and the hold element may be replaced by a combined sample and hold element.

13.6.2 Sampler

The sampling element samples the input signal at sampling points kT (T = sampling period) and supplies a pulse train at its output.

Parameters: Sampling period T > 0

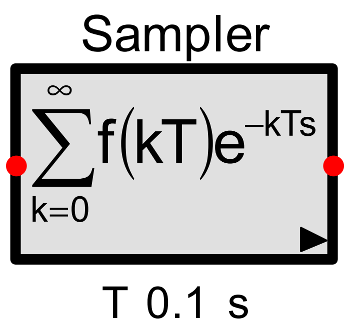




Time domain:

$$y(t) = \sum_{k=0}^{\infty} u(kT)s(t-kT)$$

$$s(t) = \begin{vmatrix} 1 & -h/2 <= t < h/2 \\ 0 & \text{sonst} \end{vmatrix}$$



Frequency domain:

$$Y^*(\omega) = \frac{1}{T}\sum_{n=-\infty}^{\infty} U\left(\omega - n\frac{2\pi}{T}\right)$$

where $U(\omega)$ is the spectrum of the continuous input signal.

The spectrum of an ideally sampled signal u(t) is equal to original time continuous signal weighted by the sampling period T and periodically repeated by the sampling frequency $2\pi/T$

The description of a time discrete signal as a triangular pulse train does not fully correspond to reality. After sampling, the signal is only defined at sampling points kT. All following time discrete elements uses only these values. So the shape of the pulses is not important as it is not considered in the following processing. Therefore it is not allowed to supply the output signal of a sampler to a time continuous element. The resulting signal would not correspond to reality.
Behind a sampler only time discrete elements are allowed which consider the signal at sampling points only and do not care about the values between.

To supply a sampled signal to a time continuous system, it must be hold between the sampling points by a hold element. But most real and time discrete elements already fulfill this condition by itself as they hold the value of the last sampling point until the new value is applied. That's also true for all time discrete elements in SimApp during a time simulation. But at frequency simulations the sample and the hold elements must exist by pairs. (See introduction to time discrete elements.)

13.6.3 Zero order hold (ZOH)

The hold element creates from a series of time discrete values u(kT) a stair like time continuous output signal by holding the current value until a new value is applied. This procedure corresponds to a hold element of 0-th order (zero order hold).
By means of the hold element one can also create time continuous step functions from time continuous input signals as the hold element also performs a inherent sampling at its input.
In the frequency domain however, a frequency dependant weighting of the signal occurs and the dynamic systems behavior is only valid in combination with a sampling element.

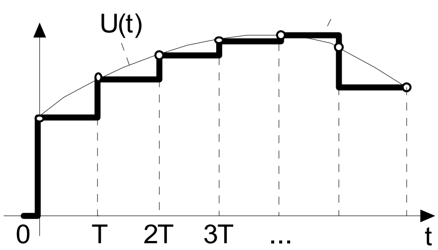
Parameters: $T > 0$

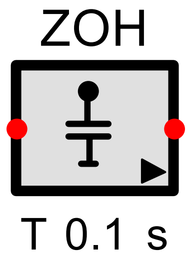
Functions

$$y(t) = \sum_{k=0}^{\infty} u(kT)\left[\sigma(t-kT) - \sigma(t-kT-T)\right]$$

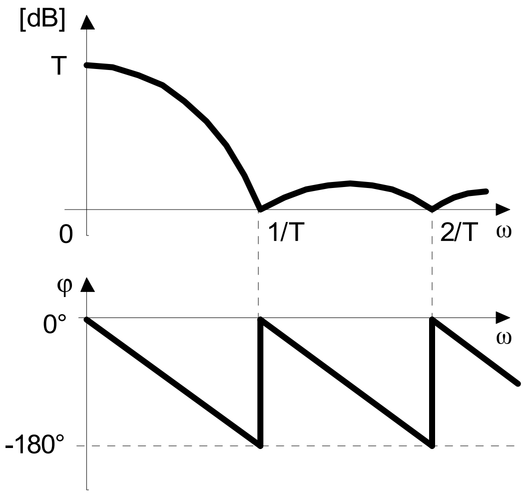
$$G(s) = \frac{1-e^{-Ts}}{s} = T\frac{\sin\omega T/2}{\omega T/2}e^{-j\omega T/2}$$

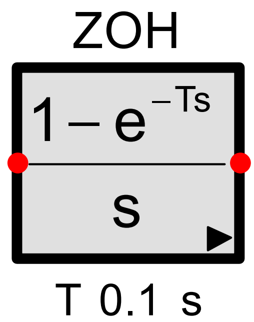
Time response





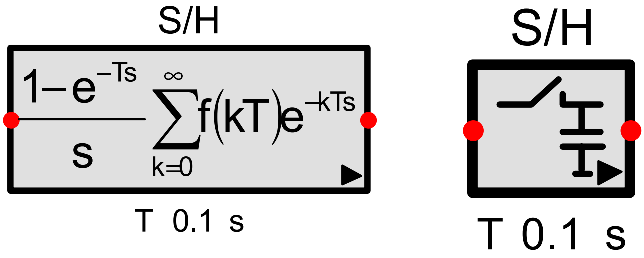
Magnitude and phase response





13.6.4 Sample and hold (S/H)

The sample and hold samples the input signal at times kT (T = sampling period) and holds the sampled value at its output until the next value is sampled at k(T+1).



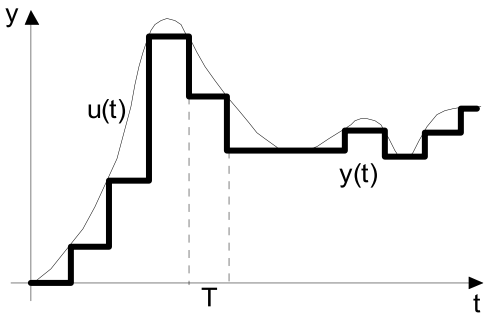
Parameters: T > 0

Functions

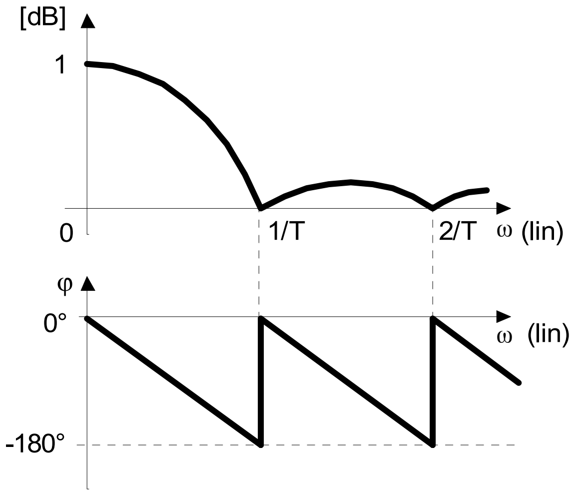
$$y(t) = \sum_{k=0}^{\infty} u(kT)[\sigma(t - kT) - \sigma(t - kT - T)]$$

$$Y(s) = \frac{1 - e^{-Ts}}{s} \sum_{k=0}^{\infty} u(kT) e^{-kTs}$$

Time response



Magnitude and phase response



13.6.5 Time discrete integrator (Iz)

There are several methods to realize a time discrete integrator. SimApp uses a polygonal approximation to transform an ideal time continuous integrator to its time discrete counterpart.

In the range of half the sampling frequency and above occur strong distortions (poles and alias frequencies). Therefore, in sampling control systems, the sampling period must be sufficient short and the harmonic distortion produced by aliasing must be eliminated by a low pass filter before the sampling process.

Parameters: T > 0, Ti>0

Functions

Polygonal approximation:

$$y_k = y_{k-1} + \frac{1}{T_i}\frac{u_k + u_{k-1}}{2}T$$

State equation:

$$x_{k+1} = x_k + u_k$$

$$y_k = \frac{T}{2T_i}(2x_k + u_k)$$

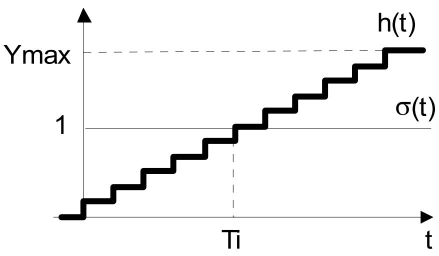
$$\underline{x}_k = \underline{x}(kT)$$

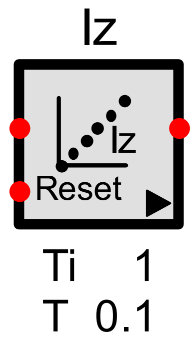
Transfer function:

$$G(z) = \frac{T}{2T_i}\frac{z+1}{z-1}$$

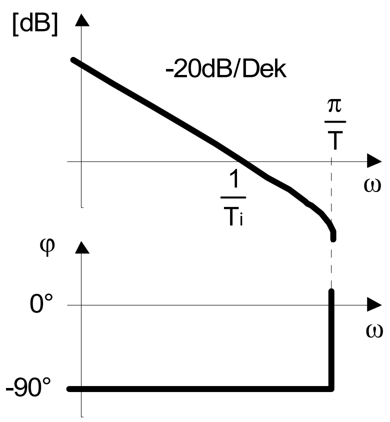
$$z = e^{sT} = e^{j\omega T}$$

Step response

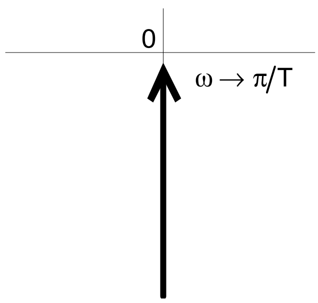


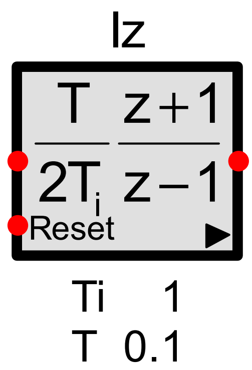


Magnitude and phase response



Polar plot





The time discrete integrator can also be derived from the analog implementation by the bilinear transformation:

Transfer function of the analog integrator is $G(s) = \frac{1}{sT_i}$

Bilinear Transformation: $s = \frac{2}{T}\frac{z-1}{z+1}$

Replacing s in G(s) by the bilinear transformation yields G(z)

13.6.6 Time discrete differentiator (Dz)

The time discrete differentiator is derived from the analog implementation by difference expression.

In the range of half the sampling frequency and above occur strong distortions (poles and alias frequencies). Therefore, in sampling control systems the sampling period must be sufficient short and the harmonic distortion produced by aliasing must be eliminated by a low pass filter before the sampling process.

Parameters: T > 0, TD>=0

Functions

Difference expression:

$$y_k = T_D \frac{u_k - u_{k-1}}{T}$$

State equation:

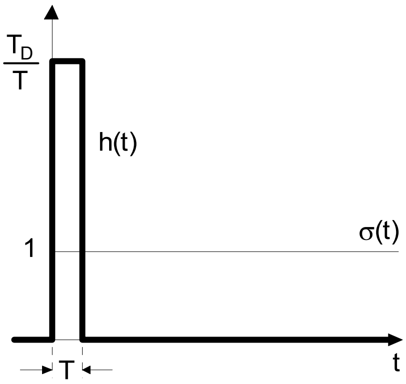
$$x_{k+1} = u_k$$

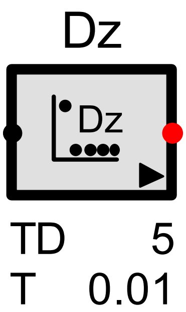
$$y_k = \frac{T_D}{T}\left(-x_k + u_k\right)$$

Transfer function:

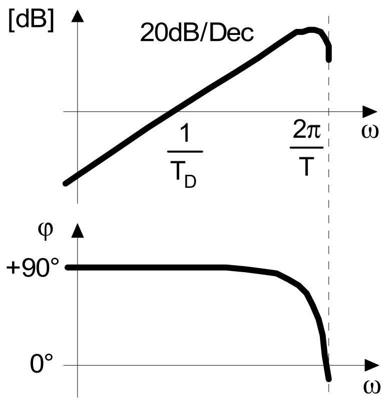
$$G(z) = \frac{T_D}{T}\frac{z-1}{z}$$

Step response

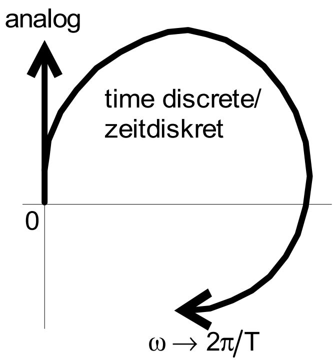


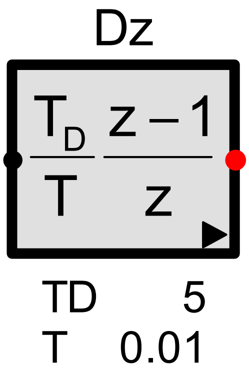


Magnitude and phase response



Polar plot





13.6.7 Unit delay (z element)

The unit delay delays a time discrete or analog signal by the sampling period T.
The block can be rotated in steps of 90°.

Parameters: T > 0, Y0

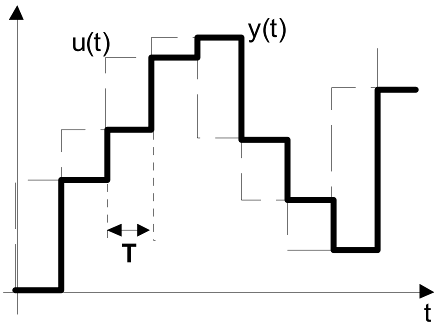
Functions

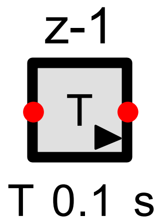
$y(t) = u(t - T)$

Step response:
$h(t) = \sigma(t - T)$

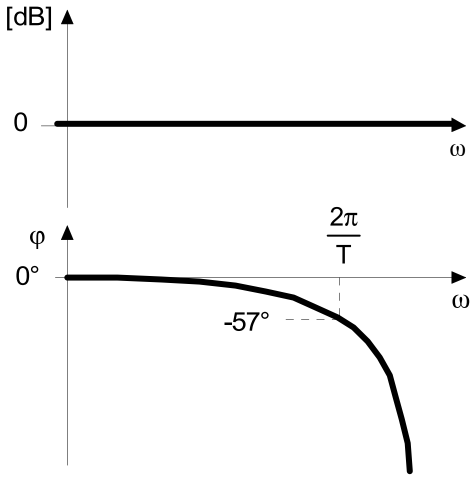
Transfer function:
$G(z) = z^{-1}; \quad z = e^{sT}$

Time response

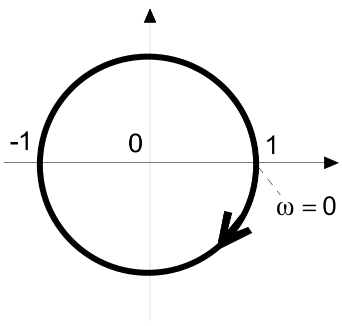


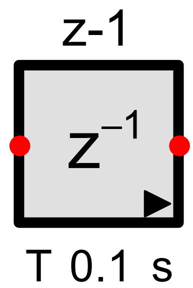


Magnitude and phase response



Polar plot





13.6.8 Time discrete PID controller (PIDz)

This controller is the counterpart to the ideal analog PID controller, type I. It consists of the parallel connection of a P-, a Iz and a Dz element, which can individually be switched off.
As the same parameters exist (gain K, rate time TV and reset time TN) it allows a direct comparison between the analog and the dime discrete implementation.

The output signal can be limited by two selectable anti-windup measures in the range Ymin ... Ymax.

Anti-Windup-Hold
The integration is stopped when the saturation's input signal leaves the valid range (the upper or lower limit of the saturation). As soon as the signal is reduced the integrator is released.

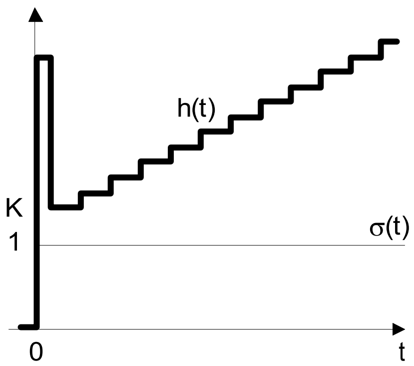
Anti-Windup-Reset
If the saturation's input signal exceeds the limit, the integrator's output signal is reduced so that the sum of integrator, differentiator and unit-gain equals exactly the limit (Ymax or Ymin).

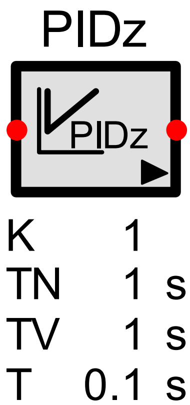
Parameters: K, TV >= 0, TN > 0, TD >= 0, Y0I, Ymax, Ymin, T>0

Functions

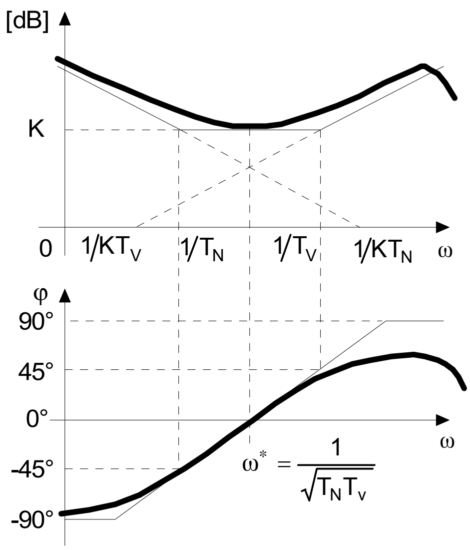
$$G(z) = K\left(1 + \frac{T/T_N\,(z+1)}{2(z-1)} + \frac{(z-1)}{T/T_V\,z}\right)$$

Step response

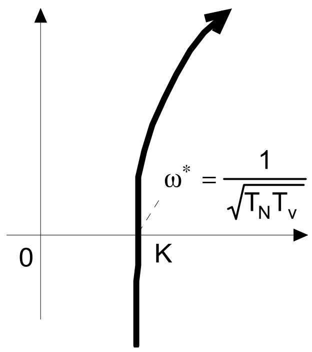


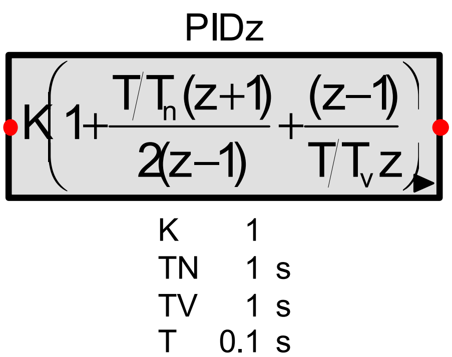


Magnitude and phase response



Polar plot





13.6.9 Rational time discrete transfer element (G(z))

The G(z) element is the general implementation of a linear time discrete transfer element. It is the counterpart to the analog G(s) element and is also known as digital controller.
A real time computer program must be causal, i.e. for computing the new output value y(kT), one can only use the input values u(kT), u(kT-T), ... From that is concluded that the order of the denominator must be greater or equal to the order of the nominator.

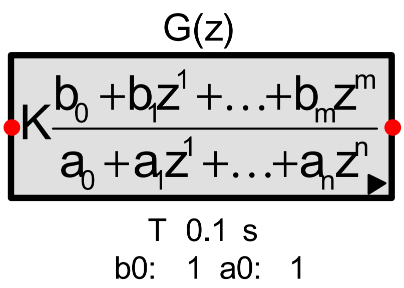
Parameters: K, T>0, a_i , b_i

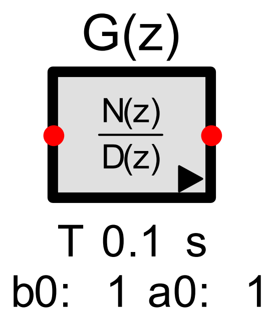
Functions

$$G(z) = K\frac{b_0 + b_1 z^1 + \ldots + b_m z^m}{a_0 + a_1 z^1 + \ldots + a_n z^n}$$

$m \le n\,,\ a_n \ne 0$

$m \le 31 \quad n \le 31$





The meaning of the initial values is depicted by the matrix representation of the state equations:

$$\underline{x}_{k+1} = \begin{bmatrix} 0 & 1 & 0 & \ldots & 0 \\ 0 & 0 & 1 & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ & & & & 1 \\ -a_0^* & -a_1^* & -a_2^* & \ldots & -a_n^* \end{bmatrix} \underline{x}_k + \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1/a_n \end{bmatrix} u_k$$

$$\frac{y_k}{K} = \begin{bmatrix} b_0 - a_0^* b_n & b_1 - a_1^* b_n & \ldots & b_{n-1} - a_{n-1}^* b_n \end{bmatrix} \underline{x}_k + b_n^* u_k$$

$$a_i^* = \frac{a_i}{a_n} \qquad b_i^* = \frac{b_i}{a_n}$$

13.6.10 Time discrete filter (z-Filter)

As the powers in the transfer function are negative, this elements has not the restrictions of the G(z) element. The orders of nominator and denominator can be freely selected. The law of causality does not apply here.

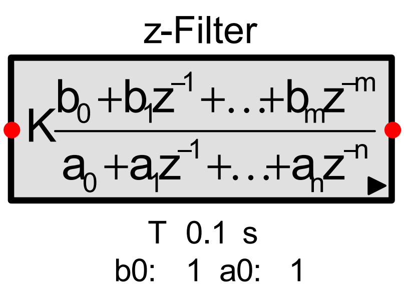
Parameters: K, T>0, a_i , b_i

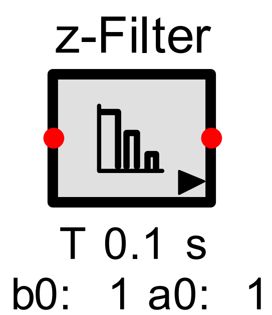
Functions

$$G(z) = K\frac{b_0 + b_1 z^{-1} + \ldots + b_m z^{-m}}{a_0 + a_1 z^{-1} + \ldots + a_n z^{-n}}$$

$a_0 \ne 0$

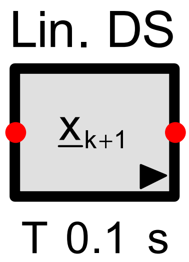
$m \le 31 \quad n \le 31$





13.6.11 Linear difference equation system

Free parameterizable, linear time invariant difference equation system.
This element is the counterpart to the analog implementation.



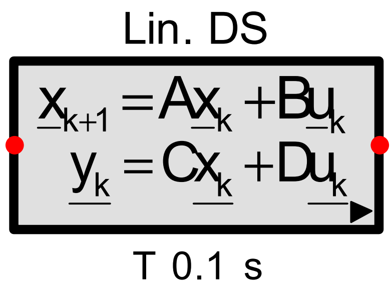
Vector form

System matrix equation $\underline{x}_{k+1} = A\underline{x}_k + B\underline{u}_k$

Output matrix equation $\underline{y}_k = C\underline{x}_k + D\underline{u}_k$

where $\underline{x}_k = \underline{x}(kT)$

The system has n states, p input quantities and q output quantities.



$\underline{x}_k$	State vector (n x 1)
$\underline{u}_k$	Input vector (p x 1)
$\underline{y}_k$	Output vector (q x 1)
A	System matrix (n x n)
B	Input matrix (n x p)
C	Output matrix (q x n)
D	Direct transmission matrix (q x p)

Parameters: n<=50, p<=50, q<= 50, coefficients Aik, Bik, Cik, Dik, Initial values xnk(i)

Matrix representation:

$$\begin{bmatrix} \dot{x}_{k+1,1} \\ \dot{x}_{k+1,2} \\ . \\ \dot{x}_{k+1,n} \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} & . & A_{1n} \\ A_{21} & A_{22} & . & A_{2n} \\ . & . & . & . \\ A_{n1} & A_{n2} & . & A_{nn} \end{bmatrix} * \begin{bmatrix} x_{k,1} \\ x_{k,2} \\ . \\ x_{k,n} \end{bmatrix} + \begin{bmatrix} B_{11} & B_{12} & . & B_{1p} \\ B_{21} & B_{22} & . & B_{2p} \\ . & . & . & . \\ B_{n1} & B_{n2} & . & B_{np} \end{bmatrix} * \begin{bmatrix} u_{k,1} \\ u_{k,2} \\ . \\ u_{k,p} \end{bmatrix}$$

$$\begin{bmatrix} y_{k,1} \\ y_{k,2} \\ . \\ y_{k,q} \end{bmatrix} = \begin{bmatrix} C_{11} & C_{12} & . & C_{1n} \\ C_{21} & C_{22} & . & C_{2n} \\ . & . & . & . \\ C_{q1} & C_{q2} & . & C_{qn} \end{bmatrix} * \begin{bmatrix} x_{k,1} \\ x_{k,2} \\ . \\ x_{k,n} \end{bmatrix} + \begin{bmatrix} D_{11} & D_{12} & . & D_{1p} \\ D_{21} & D_{22} & . & D_{2p} \\ . & . & . & . \\ D_{q1} & D_{q2} & . & D_{qp} \end{bmatrix} * \begin{bmatrix} u_{k,1} \\ u_{k,2} \\ . \\ u_{k,p} \end{bmatrix}$$

In real systems, the direct transmission matrix is mostly D = 0.

Due to the linear form, no formula evaluation must be done during simulation, so there is no speed loss compared with other elements.

13.7 Converters

SimApp makes a difference between binary and digital converters. This is not common in practice but it is important to describe two types of digital transmission used in SimApp.

A digital input or output in SimApp consists of a single node which can transmit all bits at one time. The digital numerical value is transmitted as positive analog integer signal, e.g. binary 0 corresponds to 0, 1101 (13 decimal) corresponds to 13. It does not make a sense to mix digital and analog signals, therefore the digital signal lines differ in color from the standard signal lines. The advantage of this analog coding of digital values lies in the small space requirement of elements and transmission (one line used only).

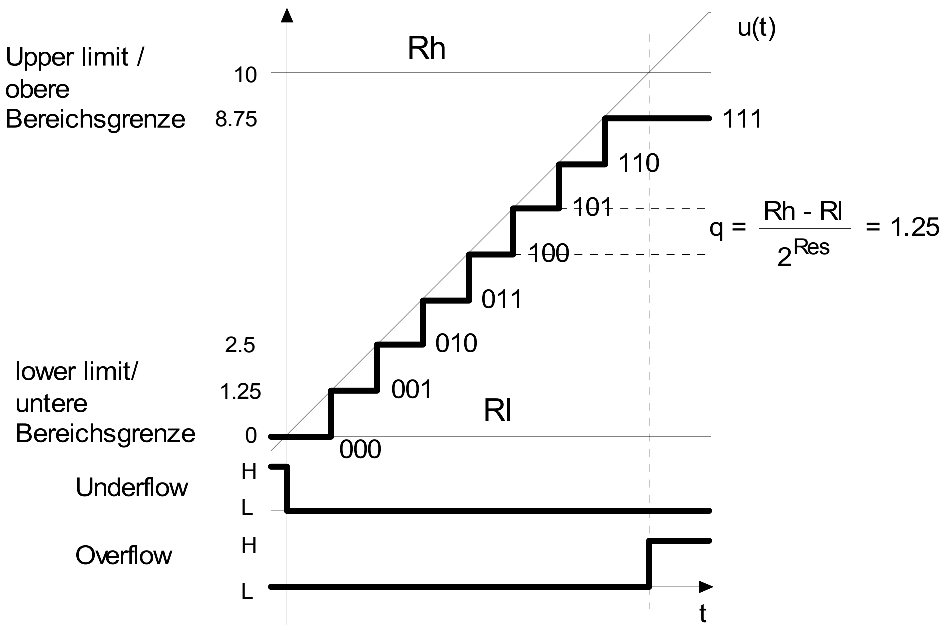
The binary input and output in SimApp corresponds more to the usual conceptions. For each individual bit a individual signal line is used. The disadvantage of this solution is the huge space requirement for large numbers (big elements and multiple lines).

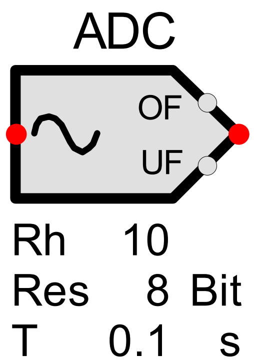
13.7.1 Analog to digital converter (ADC)

The ADC samples the input signal at constant time intervals T (sampling period) and converts it into a digital signal. The digital output signal is supplied on one single digital signal line. The ADC incorporates a sampler a converter and a quantizer which rounds the signal to positive integer values.
The sampling period, the resolution of the quantizer and the upper and lower limit can be specified. The default value for the lower limit is zero.
When the input signal leaves the valid range, the Overflow or the Underflow output is set to logic high.

Parameters: Rh, Rl, 0<= Res <= 32, T > 0

Example: Characteristic of an ADC with 3 bit resolution in the range of 0 to 10.





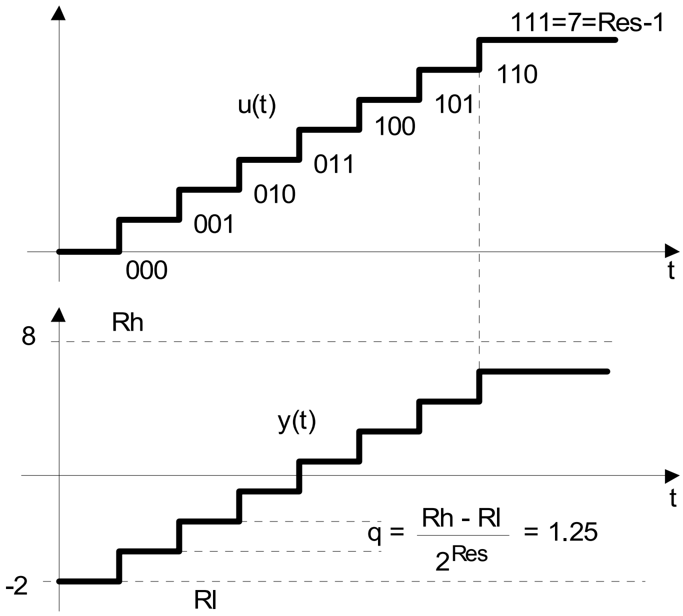
Note: The upper limit Rh corresponds to the binary value 1000 = 111 + 1.

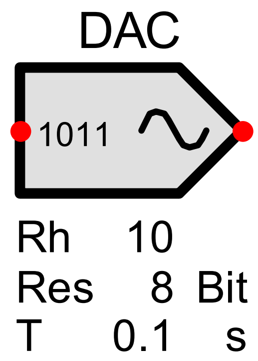
13.7.2 Digital to analog converter (DAC)

The DAC converts the digital input signal into a analog output signal. The sampling period, the resolution and the upper and lower limit as well must be specified. Default value for lower limit is zero. The sampling period must be equal to those the ADC uses to produce the input signal.

Parameters: Rh, Rl, Res > 0, T > 0

Example: DAC with resolution of 3 bit and range from -2 to 8.



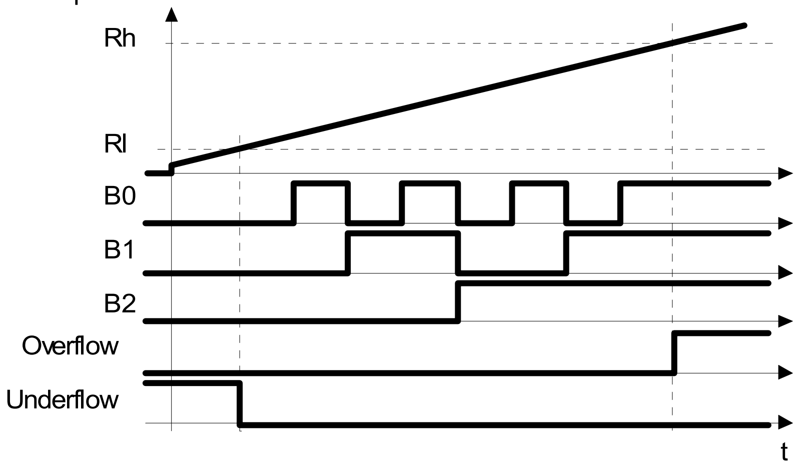


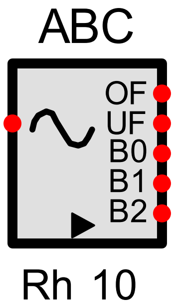
13.7.3 Analog to binary converter (ABC)

The ABC converts a analog input signal into a binary weighted output signal. Each digit has its own output node. The valid input range and the resolution as number of bits (= number of output nodes) is selectable. Valid resolution range is from 1 to 31. When the input signal leaves the valid range, the Underflow (UF) or Overflow (OF) output is set. In the underflow case the binary outputs are cleared, in the overflow case all binary outputs are set.

Parameters: Rh, Rl, VO

Example: 3-Bit ABC



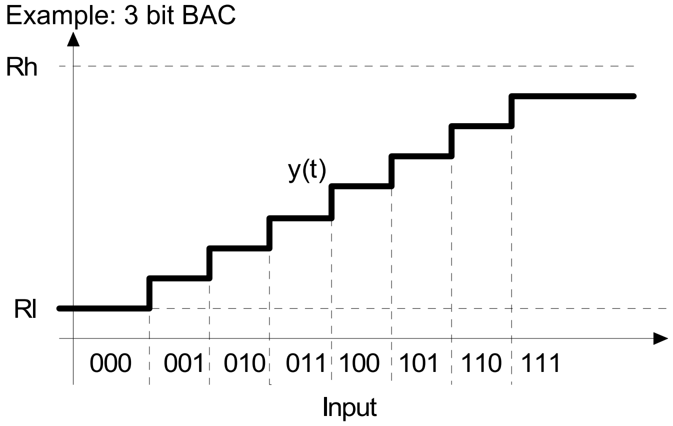


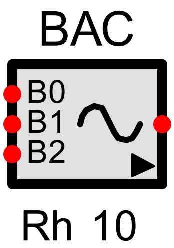
13.7.4 Binary to analog converter (BAC)

This converter is the counterpart to the ABC. It converts the binary input signal that comes on several signal lines into an analog output signal. The resolution (number of bits = number binary input nodes) can freely be selected between 1 and 31. The two selectable limits correspond to the binary values 0 and 11111..1 +1. The upper limit can never be reached as it represents the value 2^n. The maximum input value is $2^n - 1$. (n = number of bits, resolution.).

Parameters: Rh, Rl, Vth

Example: 3 bit BAC





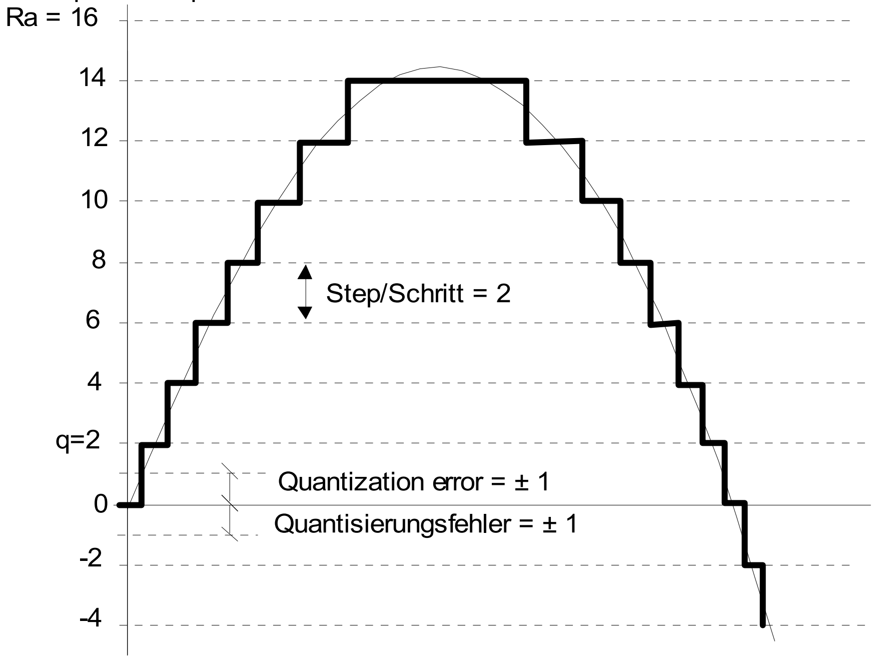
13.7.5 Quantizer

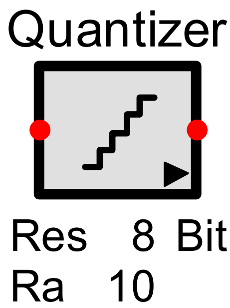
The output signal of a quantizer can assume integer values only, which are a multiple of a basic integer value. It follows a ramp by creating a step function. The step size depends on the selected range and the resolution. Example: For a range of 16 and a resolution of 3 bits, the step size amounts to q = 16 / 2^3 = 2. The output signal however is not limited by the selected range, as it is only used for step size computing.

The quantizer can be used for simulating digital systems where only discrete values can occur. The time discrete elements itself do not quantize the signal, they only sample and compute the signal at discrete times.

Parameters: Res >0, Ra

Example: 3 bit quantizer





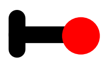
13.8 Logic

Logic elements perform Boolean operations with binary and logical signals. In SimApp there is no difference between logic and analog signals. Logic elements can be supplied by output signals of analog elements and vice versa.
Input signals being smaller than the logic threshold are interpreted as low (logic 0 or false) whereas signals being greater than the logic threshold are interpreted as high (logic 1 or true). Default threshold value is 2.5 and default logic high output value is 5. Logic 0 is always at 0.

Feedback with logic elements
By default, logic elements in SimApp have no propagation delay. Thus, there are the same restrictions as for other elements: Feedback loops without any delay elements are not allowed.
But in logical circuits it is common to build feedback loops. For example you can form a set/reset flip flop by two feedbacked NAND gates.
Furthermore, logic circuits often need propagation delays for proper operation, even when they don't have feedback loops. For example, in a synchronous counter with a chain of JK flip-flops all having the same input clock signal, it is essential, that the states at the J and K inputs and so the outputs of the preceding flip-flops, change after the positive going edge of the clock signal. For details, see the counter example in the programs example directory.
To enable such cases, all logic elements have the option *Propagation delay*. The output signal of the element changes only at the beginning of the next integration interval if this option is set. So it performs a short delay of one integration step size. If the integration step size is set as recommended (see chapter *Time simulation*) this delay can be neglected. But if you build large logical circuits or long chains of logic elements the sum of delays may become unacceptable long.
If you encounter such problems, you can take the following two steps.
1. Enable the *Propagation Delay* option only for one logic element in the feedback loop.
2. Reduce the integration step size by factor 10.

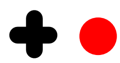
13.8.1 GND (Ground, logic 0, false)

Connects an input port to ground and acts as a source which supplies always value 0.
Normally, open logic input ports are interpreted as ground connection.
For correct positioning in drawing you can rotate this element by 90 degree steps.



13.8.2 V+ (logic 1, true)

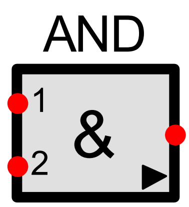
Connects an input port to logic 1 and acts as a source which supplies always the associated value for logic 1.
For correct positioning in drawing you can rotate this element by 90 degree steps.



13.8.3 AND gate

This multiple input logic element performs the Boolean operation

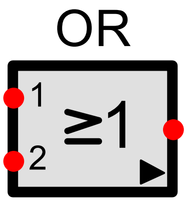
$$Y = I_1 \bullet I_2 \bullet \ldots \bullet I_n$$



The number of inputs is freely selectable between 2 and 50.
By means of the propagation delay the output is delayed by one integration step. The NAND function achieved by a following NOT gate.

13.8.4 OR gate

This multiple input logic element performs the Boolean operation:

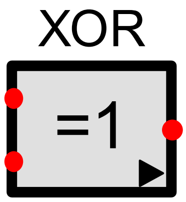


$$Y = I_1 + I_2 + \ldots + I_n$$

The number of inputs is freely selectable between 2 and 50.
By means of the propagation delay the output is delayed by one integration step. The NOR function is achieved by a following NOT gate.

13.8.5 Exclusive-OR gate (XOR, non-equivalence)

This logic element performs the Boolean operation:



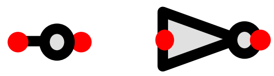
$$Y = I_1 \oplus I_2 \text{ resp. } Y = I_1 \neq I_2 \text{ resp. } Y = \overline{I_1} \bullet I_2 + I_1 \bullet \overline{I_2}$$

The number of inputs is two and is unchangeable.
By means of the propagation delay the output is delayed by one integration step.
The EXNOR function $Y = I_1 = I_2$ (equivalence) is achieved by a following NOT gate.

13.8.6 Inverter (NOT gate)

This element performs the Boolean operation:

$$Y = \overline{I} \text{ or } Y = \text{not } I$$



By means of the propagation delay the output is delayed by one integration step.
The NOT gate has two symbols:
The first symbol is a small circle so that the inversion of an input or output signal of another gate needs as little place as possible. The second symbol is a triangle for independent use.

13.8.7 SR flip-flop

A high level at Set input sets the output (Q to high, Q- to low) and a high level at Reset input resets the output. If both inputs are reset the output is not changed.

If both Set and Reset are set the output state depends on the option *Set dominant*.

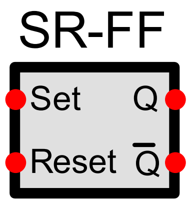
If *Set Dominant* is true, the output is set, when both Set and Reset are high at the same time.
If *Set Dominant* is false, the output is reset, when both Set and Reset are high at the same time.

If *Positive edge triggered* is active, state changes only occur at positive input edges otherwise the static values are decisive. At positive edge triggering there is no output change if the inputs fall from 1 to 0.

The logic initial value is selectable.
By means of the propagation delay the output can be delayed by one integration step.

Parameters: VO, Vth

Inputs		Outputs		
Set	Reset	Q	$\overline{Q}$	
L	H	L	H	
H	L	H	L	
L	L	Q_0	$\overline{Q}_0$	
H	H	H	L	Set dominant
H	H	L	H	Reset dominant



Q0 = level of Q before the last change of Set or Reset.

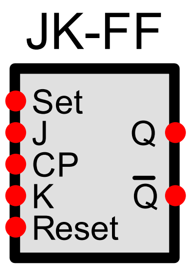
13.8.8 JK flip-flop

A high level at Set or Reset input sets or resets the outputs regardless of the levels of the other inputs. When Set and Reset are inactive (low), data at the J and K inputs are transferred to the outputs on the positive-going edge of the clock pulse. Data at J and K input may be changed without affecting the levels at the outputs. This flip-flop can be used as toggle flip flop by tying J and K high.

The logic initial value is selectable.
By means of the propagation delay the output can be delayed by one integration step.

Parameters: VO, Vth

Inputs					Outputs	
Set	Reset	CP	J	K	Q	$\overline{Q}$
H	L	X	X	X	H	L
L	H	X	X	X	L	H
H	H	X	X	X	H	H
L	L	_|‾	L	L	Q_0	$\overline{Q}_0$
L	L	_|‾	H	L	H	L
L	L	_|‾	L	H	L	H
L	L	_|‾	H	H	Toggle	
L	L	L	X	X	Q_0	$\overline{Q}_0$



X = don't care
Q0 = level of Q before the last input change or clock pulse
_|‾ = 0-1 CP edge

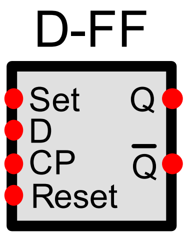
13.8.9 D flip-flop

The D-type flip-flop has asynchronous Set and Reset inputs and complementary (Q,Q-) outputs. Information at the input is transferred to the outputs on the positive going edge of the Clock Pulse. After the Clock Pulse the Data input is locked out and information present will not be transferred to the outputs until the next rising edge at the Clock Pulse input.

The logic initial value is selectable.
By means of the propagation delay the output can be delayed by one integration step.

Parameters: VO, Vth

Inputs				Outputs	
Set	Reset	CP	D	Q	$\overline{Q}$
H	L	X	X	H	L
L	H	X	X	L	H
H	H	X	X	H	H
L	L	↑	H	H	L
L	L	↑	L	L	H
L	L	L	X	Q_0	$\overline{Q}_0$



13.8.10 Monoflop

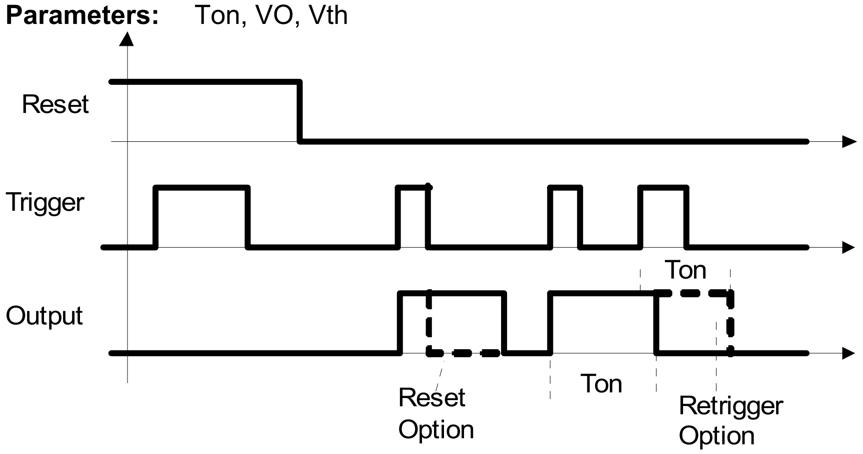
The monoflop has a positive-transition-triggered input (T) and a asynchronous reset input (R). Triggering sets the output. Once triggered, all input transitions at trigger port are ignored, as long as the output is high. (See options.)
The output is reset after the on-time Ton has elapsed or when Reset is asserted. The output cannot be set when Reset is high.

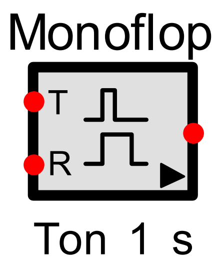
Options

Reset option: The output is cleared as soon as the trigger signal goes back to low, otherwise it waits Ton.

Retrigger option: The monoflop can be retriggered, i.e. when the trigger input fires again, the high state is extended by another Ton interval.

Parameters: Ton, VO, Vth





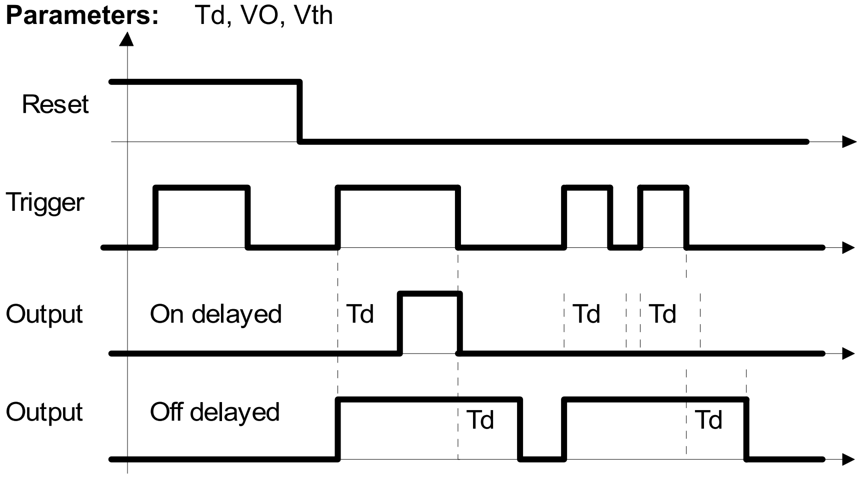
13.8.11 On/off delay

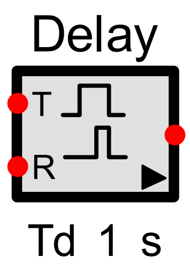
This logic delay element has a positive-transition-triggered (T) and a asynchronous Reset (R) input. In contrast to the non delayed monoflop, the on-time of the output is not specified. The on-time is determined by the duration of the Trigger pulse and an additional delay time. The rise and fall times depend on the *Off delayed* option.

Options

When Off delayed is disabled, the output is set the delay time *Td* after the positive transition of the trigger signal and forced low again on the negative transition of the trigger. If the trigger pulse is shorter than the specified delay time, the output signal remains low.
When Off delayed is enabled, the output is set on the positive transition of the trigger signal and is forced low delay time *Td* after the Trigger signal went low. If the trigger input is set again before the *Td* has elapsed, the output high state is extended until Td after the negative transition of the second trigger pulse (= retriggering).

Parameters: Td, VO, Vth





13.9 Miscellaneous

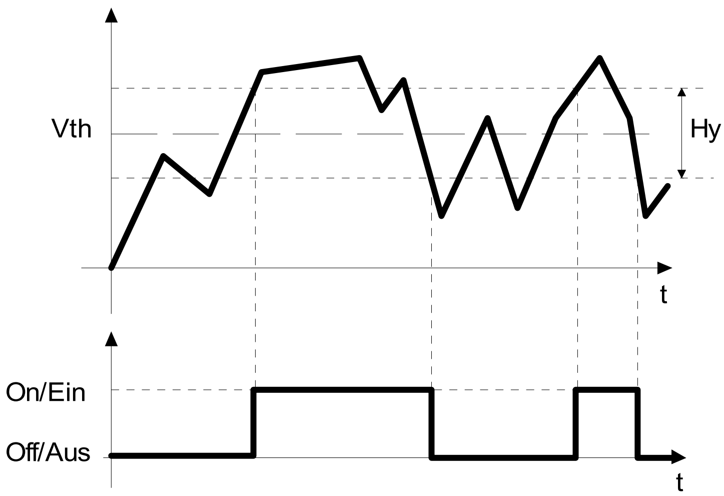
13.9.1 1:2 Switch

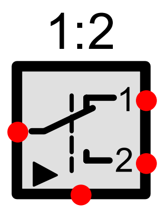
This element is a 1-line to 2-line demultiplexer. The Select input determines to which output the input signal is routed.

When the select signal is low (off state) the input is routed to the upper output, the lower one supplies zero. When the select signal is high (on state) the input is routed to the lower output, the upper one supplies zero.

The Schmitt Trigger select input prevents output jittering, when the select signal is slowly changing and superimposed with noise. Switching threshold and hysteresis are adjustable. By default, threshold is set to logical threshold.
The switch can be used for logical and digital signals as well as for analog signals.

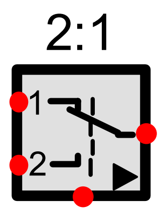
Parameters: Vth, Hy





13.9.2 2:1 Switch

This element is a 2-line to 1-line data selector/multiplexer. The select input determines which input is routed to the output.
When the select signal is low (off state), the upper input is routed to the output.
When the select signal is high (on state) the lower input is routed to the input.



For information about the Schmitt Trigger select input see *1:2 Switch.*

The switch can be used for logical and digital signals as well as for analog signals.

Parameters: Vth, Hy

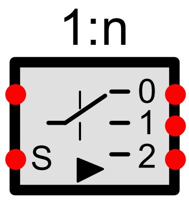
For more information about switching characteristic see *1:2 switch*.

13.9.3 1:n output switch (demultiplexer)

The signal at select input (S) selects 1 of several outputs to be connected to the single input. All other outputs supply 0. The routed signal may be of any type (analog, logical or digital). This multiswitch can have 2 to 50 output pins.

Parameters: n (number of outputs)

S	O0	O1	...	On-1
<0	0	0	...	0
0	1	0	...	0
1	0	1	...	0
⋮	⋮	⋮	⋱	⋮
n-1	0	0	...	1
≥n	0	0	...	0



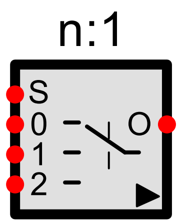
All outputs supply 0, if the value at S is < 0 or greater than the number of outputs - 1

13.9.4 n:1 Input switch (multiplexer)

The signal at select input (S) selects 1 of several inputs to be connected to the single output. The routed signal may be of any type (analog, logical or digital). This multiswitch can have 2 to 50 input pins.

Parameters: n (number of inputs)

S	Out
<0	0
0	I0
1	I1
⋮	⋮
n-1	In-1
≥n	0



The output supplies 0, if the value at S is < 0 or greater than the number of inputs - 1.

13.9.5 Triggered sample and hold

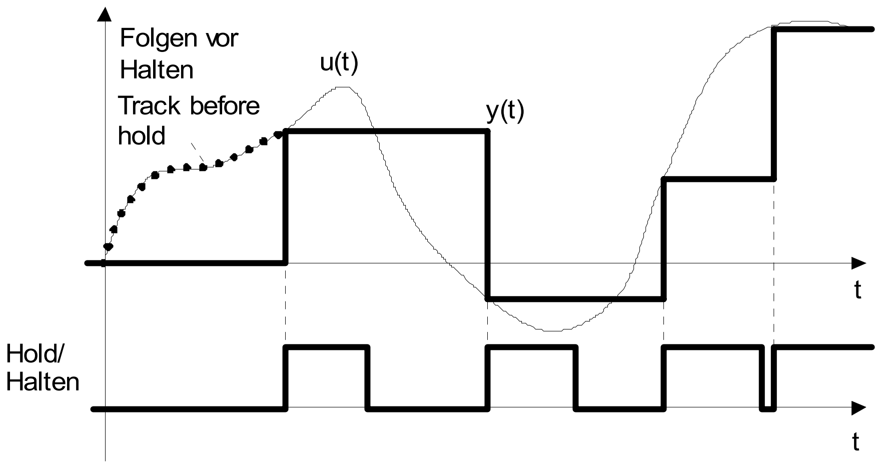
This element has an analog and a logic input terminal. On the rising edge at the trigger input the analog input signal is sampled and the value hold at the output until the end of the simulation or until the next trigger event is applied.

Before the first trigger event, the shape of the output signal is determined by the option *Track before hold*.

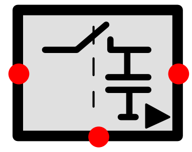
If *Track before hold* is enabled, the output tracks the input signal until the first trigger event is applied, otherwise the output signal remains zero. After the occurrence of the first trigger event this option does not affect the output any longer.

Note the difference to SimApp's ordinary sample and hold element which samples and holds the input signal at every sample period.

Parameters: Vth



Trig. S/H

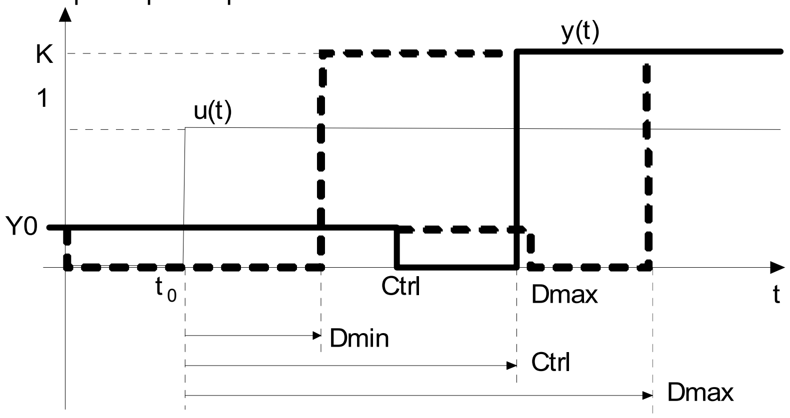


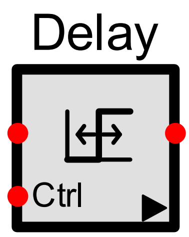
13.9.6 Controllable delay element

The current delay time in seconds [s] is controlled by a analog input signal at the control input (*Ctrl*). The delay range can be limited by Dmin and Dmax. The lower limit is 0 if Dmin < 0 and the upper limit is ∞ if Dmax is < 0.

Parameters: K, Dmin, Dmax , Y0

Example: Input step at time t = t0:





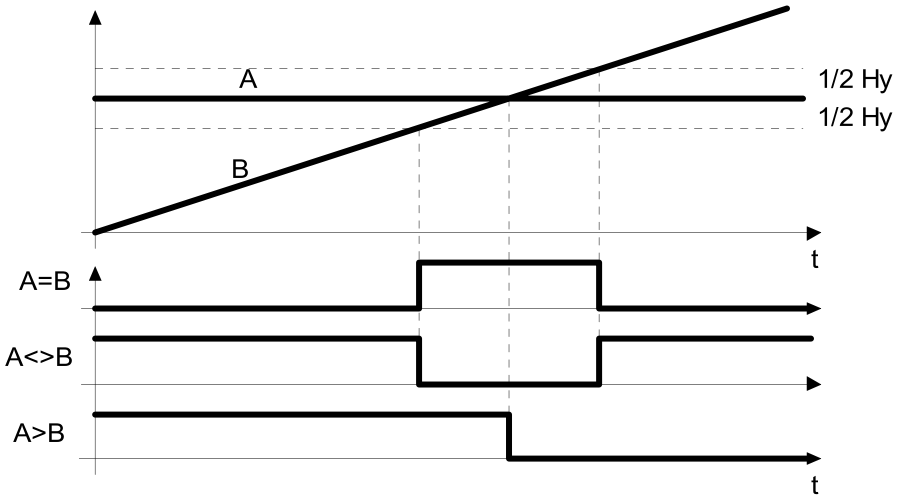
13.9.7 Relation

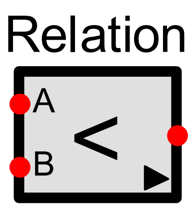
The Relation element offers 6 relational operators:

A < B A <= B A = B A >= B A > B A <> B

The hysteresis is used for the equal (A=B) and unequal (A<>B) operations only, since comparing two floating point numbers is very critical. If both input signals are within this band, they are interpreted as equal or unequal respectively.

Parameters: Hy, VO





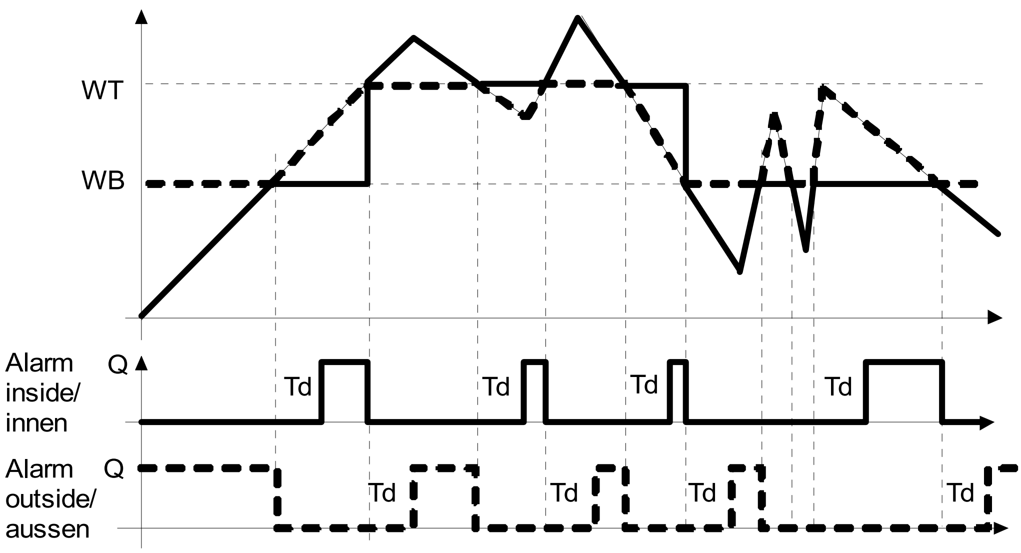
13.9.8 Window comparator

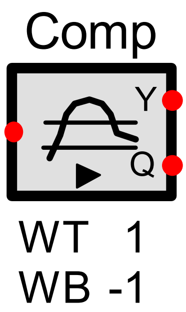
This element detects if the input signal is inside or outside the range (window) WB...WT. The state (input inside or input outside) is indicated by the logic Q output port. The output port Y supplies the input signal if it is in the valid range otherwise it supplies WB or WT.

Designation of the valid range is determined by the *Outside* option.
If *Outside* is enabled, the valid range is between WB and WT. If the input is not in this range, the Q output is set.
If *Outside* is disabled, the valid range is outside of the range WB...WT. If the input is between WB and WT the Q output is set.

The start of the rising edge at the Q output can be delayed by the *Alarm delay* parameter so that short time range violations are ignored.

Parameters: WT, WB, Td, VO





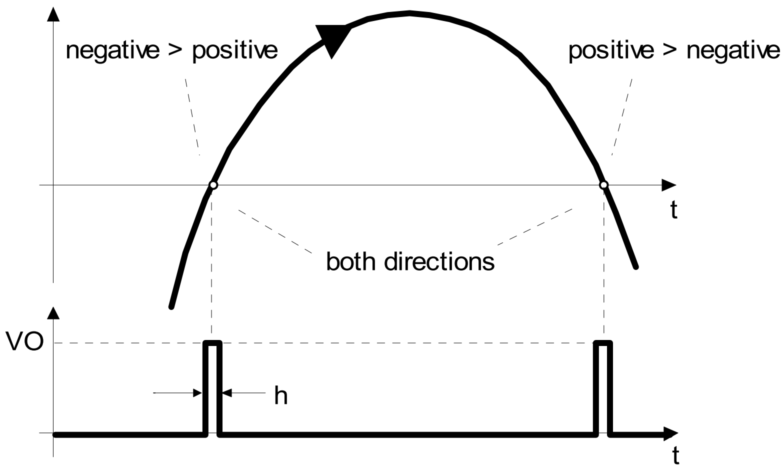
13.9.9 Zero crossing detector

This element observes the input signal and supplies a short logic pulse if it crosses the zero line. The pulse width is one integration step size, so that it is not visible if the output time resolution is longer than the integration step size.

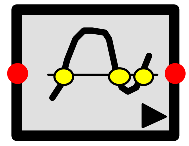
There are three detection modes:

Both Directions: Fires an output pulse if the input goes form negative to positive or from positive to negative.
Negative > Positive: Detects crossings only if the input goes from negative to positive.
Positive > Negative: Detects crossings only if the input goes from positive to negative.

Parameters: VO



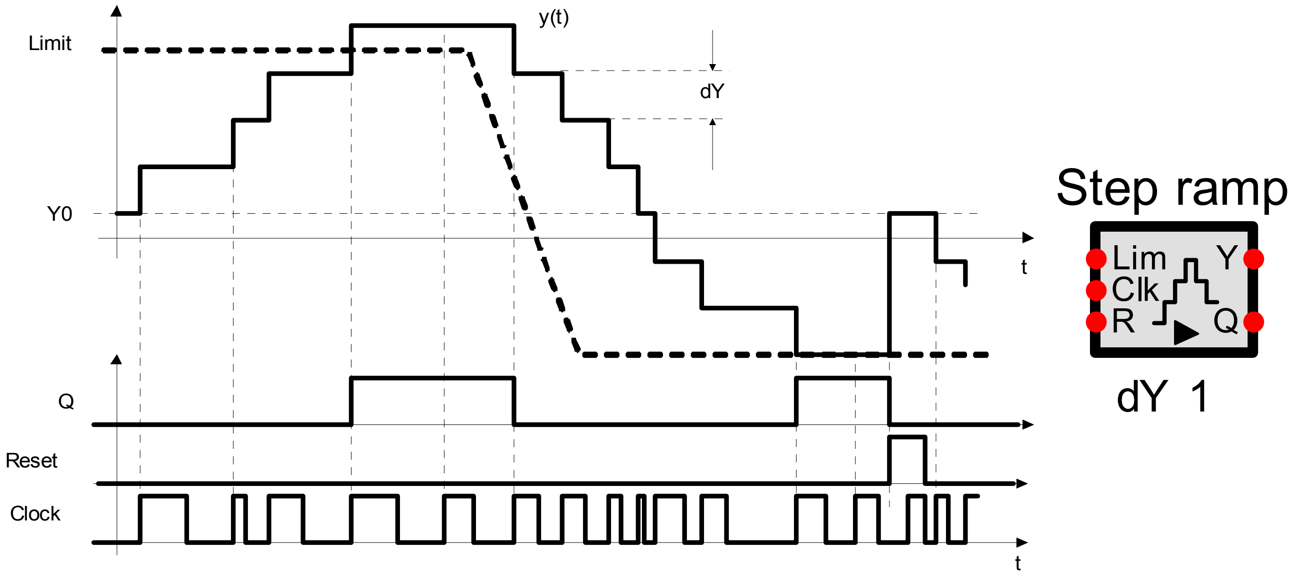
Zero-crossing



13.9.10 Step ramp

This elements supplies a stair shaped ramp at its output Y. The step size can be specified and remains constant during simulation. On each positive going edge at the clock (Clk) input, the ramp is increased or decreased by one step. The ramp starts at a given size and grows towards the variable limit signal. By applying Reset the ramp is reset to initial size. The ramp stops if it has reached the limit and the output Q is set. If the limit signal is changed, the output Q is cleared and the ramp approaches again the new value of the limit signal on every clock edge. By changing the limit signal the slope of the ramp can be changed at full speed.

Parameters: dY, Y0, VO, Vth



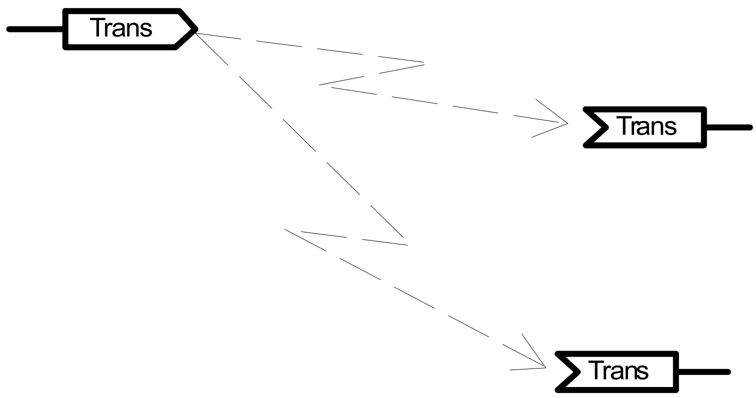
13.10 Special

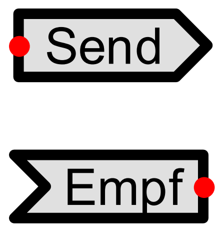
13.10.1 Transmitter and Receiver

By means of transmitter-receiver pairs you can build block diagrams which are more clearly. Instead of a direct visual connection of two blocks, you can use a wireless transmission. The connection is established by a common name which must be unique in the drawing. Two or more transmitters having the same name forms a short circuit. A transmitter however can transmit a signal to more than on receiver if all have the same name.

It is also possible to transmit signals out of custom blocks without using nodes. But this is recommended for testing purposes only as the transmitter is not visible from outside and can get forgotten.

Example: One transmitter and two receiver blocks with the name *Trans*:





14 Bibliography

Deutsch

[1] Föllinger Otto: Regelungstechnik, Hüthig-Verlag Heidelberg, 1994
[2] Orlowski Peter: Praktische Regeltechnik, Springer-Verlag Berlin, 1994
[3] Mann Heinz, Schiffelgen Horst, Froriep Rainer: Einführung in die Regelungstechnik, Carl Hanser-Verlag München, 1997
[4] Wegener Adolf: Analoge Regelungstechnik, Carl Hanser-Verlag München, 1995
[5] Bossel Hartmut: Modellbildung und Simulation, Vieweg-Verlag Wiesbaden, 1994
[6] Schwarz Hans Rudolf: Numerische Mathematik, B. G. Teubner Stuttgart, 1993
[7] Föllinger Otto: Lineare Abtastsysteme, Oldenburg Verlag München, 1993
[8] Föllinger Otto: Nichtlineare Regelungen I, Oldenburg Verlag München, 1993
[9] Föllinger Otto: Nichtlineare Regelungen II, Oldenburg Verlag München, 1993
[10] Föllinger Otto: Laplace- und Fourier-Transformation, AEG-Telefunken, 1980

English

[21] Kuo Benjamin C.: Automatic Control Systems, Prentice Hall, 1995
[22] Kuo Benjamin C.: Digital Control Systems, Holt, Rinehart and Winston, Inc, 1980
[23] Kailath Thomas: Linear Systems, Prentice Hall, 1980
[24] Gibson John E.: Nonlinear Automatic Control, McGraw-Hill, 1963
[25] Atherton D. P.: Nonlinear Control Engineering, Van Nostrand Reinhold Company London, 1975
[26] Slotine, Jean-Jacques E. and Weiping Li: Applied Nonlinear Control, Prentice-Hall,1991

Français

[41] BHALY: Boucles de Régulation, études et mise au point, Kirk Editions, 1990
[42] C.Sueur, P.VanHeeghe, P.Borne: Automatique des Systèmes Continus, Edition Technip, 1997