Manual Windfc#
V 7.4.13

Contents

Introduction

The program *WINDFC#* replaces completely the old Borland Version *Windf6.* It is converted into C#-Language and compiled with MS Studio 2008. The source code is available at interest (open source) and I used this as a project in my Control Systems and my Visual programming lectures.

Features:

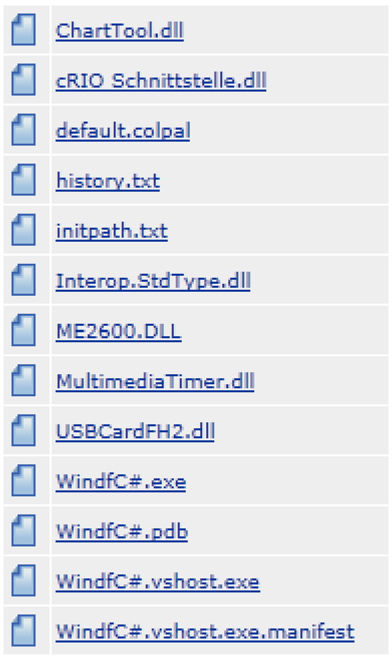
- Graphical display of diagrams (bode plots, measured step responses etc.)
- Display of bode plots of analog filters with given F(p)
- Conversion of F(p) into poles-zero and time constant form of F(p)
- Display of bode plots of digital filters with given F(z)
- Conversion / design of Digital filters (F(p) converted into F(z))
- Execution of digital filters down to 1 ms real time under Win XP/ Win7
- Step and other responses of digital filters
- Read in of scope data from Fluke scope PM 3000 series via rs232
- Read in of GPA-data from Schlumberger 12xx series via rs232
- Identification of many different model types via LS-method with step responses
- Identification of F(z) via measured in- and output values of a system (LS offline)
- Reuter tangent method (2PT1-model identification with inflection tangent method)
- Control loop real time algorithms with simulated and real hardware processes. Interface via different ADC-Cards and USB-interface.

- Interface to Multimeter DT 9062 via rs232
- Different controller design tools: Flash design PIDT1+2PT1, FRA PIDT1+2PT1+delay, digital controller design 4 processes+7 controller types
- Conversion of continuous PIDT1 into digital algorithms
- Graphical display with point finder in the 2PT1 RSR 2 point control identification
- Transfer of data from ufk- window into bode window via zk file.
- In the Advanced Adaptive Controller- Box you can find some sophisticated controller types like the dead beat controllers, Orientation controller ore predictive controller, which could be online adaptive with recursive LS- method.

Installation

The *WINDFC#.exe* file needs the .NET – runtime package 3.5. If not available it can be found on MS- pages in internet. Look for "DotNetRedistributable"- files. They should be free.

Additionally this program needs several other files, which are at this times:



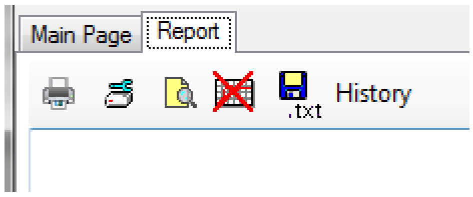
ChartTool.dll
cRIO Schnittstelle.dll
default.colpal
history.txt
initpath.txt
Interop.StdType.dll
ME2600.DLL
MultimediaTimer.dll
USBCardFH2.dll
WindfC#.exe
WindfC#.pdb
WindfC#.vshost.exe
WindfC#.vshost.exe.manifest

They can be found in the Path *WINDFC#/ windfc#7.4.13 nur runtime.rar.*

If you want to compile the *WINDFC#*source you have to download the latest version from same directory *stud/pub/bayerlej/ SolWindfx.x.x.* The compiler MS Visual studio 2008 or newer is needed.

Report function

Many modules have a "Report" button. In that case by clicking on it calculated results a copied into the "Report" tab- page of the main window and can be printed or stored as ASCII text- file. Clear – and Preview- buttons are available.



Click on "History" displays the actual history- file of the actual version.

Some tool- buttons of Chart2D- Curve

If there is a display of any curve, following tool buttons are active.

Load and Save complete Charts

With these buttons you can save and load complete charts into a file.

Print chart

With this button you can print a chart in either horizontal or vertical shape

Titles

With this button you can add a top or a bottom Title.

Toggle Legend

A legend on the right side is displayed or hidden after this click. The curve names and colors can be changed simply by clicking on items of the legend.

Grid lines

With this buttons you can add horizontal or vertical grid lines.

Data Editor

This toggle button switches between the graphical display and a numeric table. Sometimes it is useful to see the exact digits of points. But in this table you can see the x- and y-values of the points. The x-values can not be changed, but the y-values.

Zoom

With this button you can activate the zoom- function. Click after activation on the top left and then on the bottom right corner of the new area. After this this area is displayed in the total display area. To go back to normal display click on this button a second time.

Manual scaling

This is the Manual-Scale-Button. It opens a window to change all axis information including the Left-Right and the Up-Down limits of the graphic display. Also the distance between the horizontal lines can be changed. Menu: File/Load graphic file

Autoscale

This Auto-Scale-Button sets the axis limits to the calculated Min/Max-values of the curves. This is only useful after loading a *.sim-file with wrong or bad axis-information. All other files are automatically scaled during loading. If you delete a curve, the axis limits can't be changed, because the Min/Max-values are not updated. If you want to have new scaling, delete all curves and redraw them in the desired combination or use the next Manual-Scale-Button.

Change Color of Background

This tool button offers a color selection window, where you can easily change the color the background. The curve colors can easily be changed by clicking the square symbol in the legend.

Color of curves or black/white curves

With this button you can change between colored and black/white curves. It is also a toggle key.

Line Thickness

This button changes the thickness of the curves. It toggles from 1-pixel to 2-pixel thickness. If the 2-pixel thickness is chosen, the computer needs more time to draw especially if you have many points.

Short Description of the Menu Items and tool buttons

Menu: File/Deutscher Text

With a click on this button you can switch most of all menu items into German language. Schaltet auf deutschen Menütext um.

-

Menu: Actions/Delete last curve

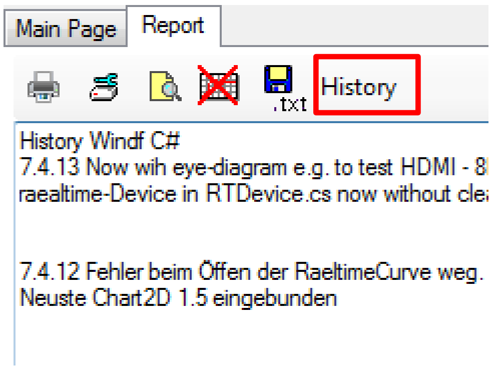
This item deletes the last curve. If this is the first curve the workspace will be cleared.

-

Menu: Actions /Delete all curves

This button deletes all curves.

Menu: File/Info



History Windf C#
7.4.13 Now wih eye-diagram e.g. to test HDMI - 8
raealtime-Device in RTDevice.cs now without cle

7.4.12 Fehler beim Öffen der RaeltimeCurve weg.
Neuste Chart2D 1.5 eingebunden

The the history file is displayed you can read the Version number and some more information for example you can see some latest Version information.

Menu: File/Quit

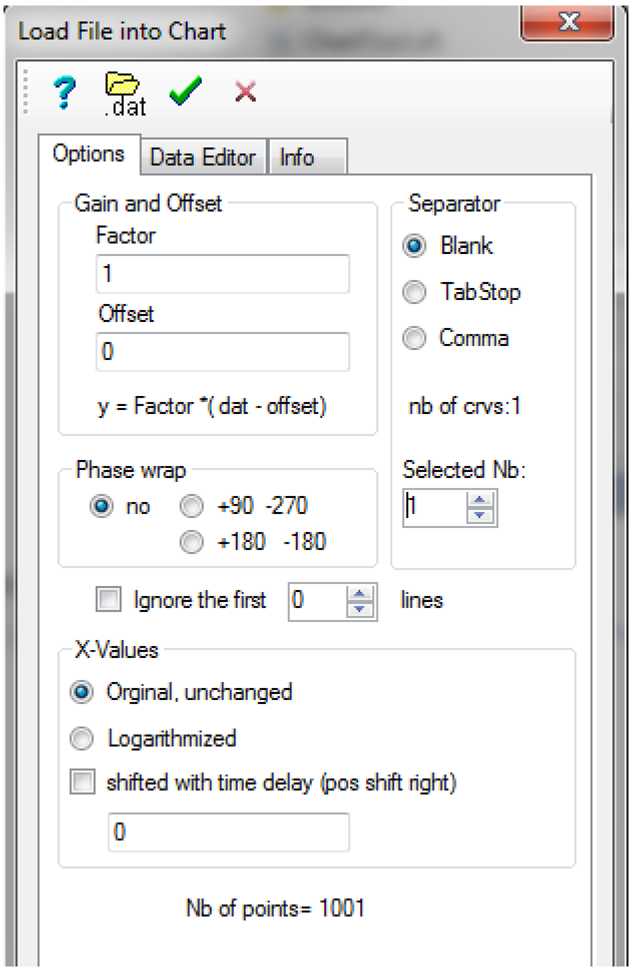
With this key you can quit and escape the program. All data, which is not specially stored, is lost. Of course you can exit the program also through the normal Windows button by clicking on the close-box-button.

File/Print curves

Here you can print the report page. The print out of the curves can be done with the tool Buttons of the Chart. There a change of printer settings is possible, Horizontal paper format (landscape) is recommended for curves, vertical for reports.

Load text files with data points and display

With this item you can load data files with x/y-data points. A x-y diagram will be displayed. The curve memory is not cleared before loading. Maximum number of curves : 10, max number of points : unlimited.
First it is opened an option window, then a file selection window. Select the name of the file (e.g. a .sim – file) and say OK. With this option window you can adapt to the content of the file:


Load File into Chart

Options | Data Editor | Info

Gain and Offset
Factor: 1
Offset: 0
y = Factor *(dat - offset)

Separator: [✓ Blank] TabStop Comma
nb of crvs:1
Selected Nb: 1

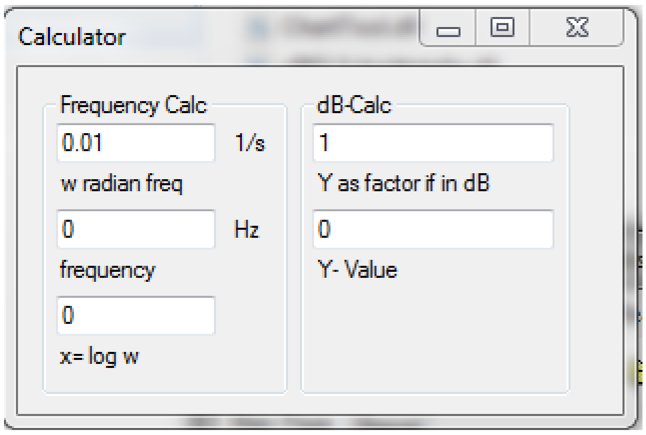
Phase wrap: [✓ no] +90 -270 +180 -180

Ignore the first 0 lines

X-Values
[✓ Orginal, unchanged]
Logarithmized
shifted with time delay (pos shift right): 0

Nb of points= 1001

With the data edition you can see the content of the file. It should have data lines which contains an x-Value followed by a separator an then several y- Values are possible.
In Options you can select three different separators, choose the number of the curve, add a scaling factor and an offset, select a phase wrapper, ignore some header lines. Furthermore for bode plot data you can select a logarithmic horizontal scale. If you have time diagrams you can add a time shift.


Calculator

Frequency Calc
0.01 1/s
w radian freq
0 Hz
frequency
0
x= log w

dB-Calc
1
Y as factor if in dB
0
Y- Value

If there is a logarithmic frequency scale then the direct logarithmic values x are displayed. This is quite unusual, but the used graphic can't display the original frequency. You can easily calculate the radian frequency ω with the equation ω=10^x if x is the horizontal parameter value and then the frequency is f=ω/2/pi . If you click with the right mouse key you can activate in a popup window a x-y display of values (“Show coordinates”). If you move the mouse on any point of the display you get an information about the x/y-value of that point. With the tool in “Some Tools”→”Frequ.-calculator” you can copy values by a left mouse click and see than some calculated information.

Export last curve

This is the EXPORT-button. With this button you can store the last curve as ASCII-files in *.SIM-file format.

Menu: Actions /Modify analog filter *.UFK

This item offers the input of the direct coefficients of a transfer function in polynomial form, which is described in the left box. The general form is $F(p) = \frac{d_m p^m + ... + d_1 p + d_0}{c_n p^n + ... + c_1 p + c_0} e^{-pT0}$. After the input of the coefficients by hand or loading an *.UFK-file there is the possibility of calculating the poles and zeros of the transfer function with the 'Factorize'-Button Factorize. You can store the result as a *.*UFK* or as a *.*ZK*-file. After leaving this window with the OK-Button (green arrow) the *.*ZK*-results are automatically stored into the analog filter memory and you can go on with bode plot or digital filter design. The first lines could have a comment, the comments starts with a !-sign. If you have used the identification IDA of the Winfact- package, you can import that exported ufk-files here.

**.UFK-File:*
```
!comment
m
n
d0
.
dm
c0
.
cn
T0
```

Menu: Actions/modify analog filter *.zk

This opens a new window, in which you can define the coefficients of a F(p) of an analog filter. Either you can load a data file *.*zk* or you can type in the coefficients by hand. The transfer function is in the time constant format, see below. There nr is the number of real zeroes, nc the number of complex zero pairs, pr the number of real poles and nc the number of complex pole pairs. With the save button you can save the coefficients as a *.zk- file.

Curious is the representation of I- or D-blocks.

Representation of D-Block in ZK-Form:

$$F_D = K_D p \approx K_D \frac{1+10^{10} p}{10^{10}} = K_D'(1+10^{10} p) \quad \text{mit} \quad K_D' = K_D 10^{-10} \quad \text{und} \quad T_0 = 10^{10}.$$

Representation of I-Block in ZK-Form:

$$F_I = \frac{K_I}{p} \approx \frac{K_I 10^{10}}{1+10^{10} p} = \frac{K_I'}{1+10^{10} p} \text{ with } K_I' = K_I 10^{10} \text{ and } T_\infty = 10^{10}.$$

If the relevant filter only is known as standard transfer function (*.*UFK*- data file), you previously have to change it into the time constant form with the *UFK*-button.

In the relevant window you can arbitrarily change all coefficients and restore them into a *.*ZK*-file.

$$F(p) = K \frac{(1+pT_{01})..(1+pT_{0nr})\left(1+2d_{01}p/\omega_{01}+p^2/\omega_{01}^2\right)..\left(1+2d_{0nc}p/\omega_{0nc}+p^2/\omega_{0nc}^2\right)e^{-pT0}}{(1+pT_{\infty 1})..(1+pT_{\infty pr})\left(1+2d_{\infty 1}p/\omega_{\infty 1}+p^2/\omega_{\infty 1}^2\right)..\left(1+2d_{\infty pc}p/\omega_{\infty pc}+p^2/\omega_{\infty pc}^2\right)}$$

```
*.ZK
!comment
nr nc pr pc K
T01
..
T0nr
d01 w01
..
d0nc w0nc
T∞1
..
T∞pr
d∞1 w∞1
..
d∞pc w∞pc
T0
```

After the choice of the coefficients the bode plot parameter should be defined. You can choose the standard logarithmic frequency scale or a linear scale.
You can draw the magnitude bode plot of this analog filter function by leaving this window with the Pen |F|-Button now. The magnitude curve is displayed on the graphic screen of the main window. The keys FA (for magnitude curve) and PA (for phase curve) are active now, so you can easily replot the curves without calling the digital filter window.
If you leave the window with the red-X key no bode plot is drawn.
A nice feature is the 3D-magnitude - mesh-plot. It is the only function, which **is not available** yet in C#.
With the save button you can save the bode plot data as ASCII- files. There are three file formats: *.bdf contains the frequency, separated with blank the magnitude and separated with blank the phase in degrees one line each point. The *.gpf file contains only frequency and magnitude, the *.gpp file the frequency together with the phase in the same way. Check it with notepad. Later you can reload these data with the load button in the main menu (see above). In the tab sheets of the page control you can find this ASCII data in Memo-fields and can check or edit.

Menu: Actions/Draw mag / phase analog filter

Only active, if the right key has been pressed previously and a bode plot is already calculated. With this item you can redraw the magnitude curve |F| or the phase curve φ of the transfer function of the analog filter, if the drawing points are available.

Menu: Actions/Calc digital filter

Only active, if the right button has been pressed previously or equivalent menu item has been chosen. With this item it is possible to calculate a corresponding digital filter from an analog one. You can either use the rectangular approximation with the replacement $p \approx (1-z^{-1})/T_0$. In this case the integration is replaced by a sum of rectangular slices and the tangent is replaced by the secant during differentiation.
You can use the trapezoidal approximation, too. This method replaces the integration by a sum of trapezoidal slices. This sometimes is also called the bilinear or Boxer-Thaler-transformation. The conversion equation is then $p = \frac{1}{T_0}\ln(z) \approx \frac{2(z-1)}{T_0(z+1)} = \frac{2(1-z^{-1})}{T_0(1+z^{-1})}$.
After doing this the you can edit the coefficients in the next digital filter window.

---

*Menu: Actions/Modify digital filter *.fz*

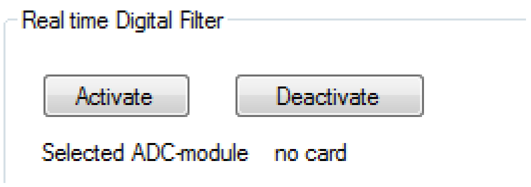
This item defines the coefficients of a digital filter. Either you load a file of the format *.fz or you type in the coefficients by hand or they are already calculated by the program from an analog filter (key 'AtoD'). The form of the transfer functions F(z) is:

$F(z) = \dfrac{b_0 + b_1 z^{-1} + \ldots + b_m z^{-m}}{1 + a_1 z^{-1} + \ldots + a_m z^{-m}} z^{-d}$ of the format *.fz .

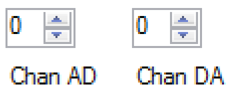
```
*.FZ
!comment
m
1 b0
..
am bm
t0
d
```

Now you can draw a bode plot of this digital filter function by leaving this window with the Button (magnitude) or with the phase button. The same buttons (for magnitude or phase curve) of the main menu are active now, so you can easily replot the curves without activating the digital filter window.

Additionally you can activate a real-time digital filter, if an ADC-Card (at this time no card is selected) is connected to your PC. Choose the relevant card in the main menu “ADC-Cards”). If you have different AD-Cards, contact me for adaption software. Select the channel of adc-connection with the left selectors. With this choice is between AD-channel 0 and 12-Bit DA-channel 0 the digital filter active now. This runs up to 1ms sampling time. Depending on the Windows operating system (either WIN95/ 98/ME or NT/XP Win7) the real time operation is different. We saw good behavior with Windows XP. I have measured interrupt times down to 1 ms with an XP. The number of interrupts in average remains constant. If you have chosen 1ms run time, you get real 1000 interrupts per seconds.



Selected ADC-module no card



The ‘design T0-time’ has no influence on neither the bode plot nor the digital filter, it only displays the sampling time of the design of the filter. Only the ‘runtime T0-time’ changes the effective filter response.

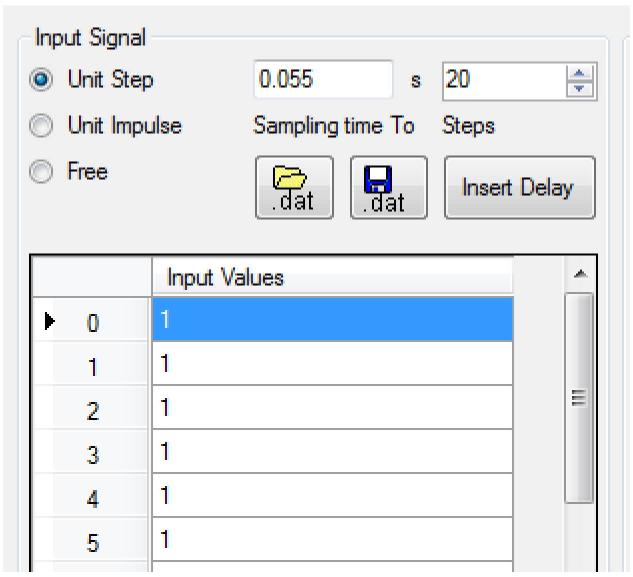
To check real time property I have added a 1V-impulse at DA channel 1. The period is the actual sampling time, the pulse duration indicates the calculation / delay time. If you leave the window with the red-X key no bode plot is drawn.

Menu: Actions/Draw mag /phase digital filter

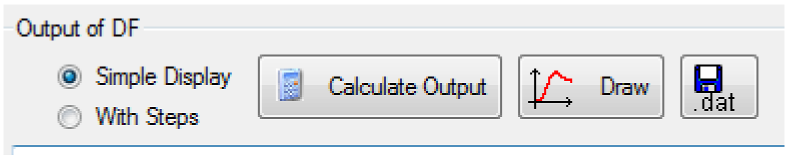
Only active, if the right buttons has been pressed previously or equivalent menu item has been chosen. With this item you can draw the magnitude curve |F| or the phase curve φ of the transfer function of the digital filter.

Menu: Actions/Calc response of dig. filter

Only active, if the right button has previously been pressed or equivalent menu item has been chosen. Here you can calculate the step response, impulse response and response of any input signal the chosen digital filter. In the opened window you can choose the number of points and see the values of the calculated points. If you leave this window with the draw button, the data is copied into the main diagram and added to the other curves.
You can load and store input signals and store the calculated output signal in .sim – ASCii files. If you click the radio button "with steps" before calculation, then you can see in the output signal with steps caused by a Sample & Hold (SH) block.
With the button "Insert Delay" you can shift the input signal one step into the future.


Input Signal
[✓ Unit Step] Unit Impulse Free
Sampling time To: 0.055 s
Steps: 20

	Input Values
0	1
1	1
2	1
3	1
4	1
5	1


Output of DF
[✓ Simple Display] With Steps

Interface to digital scope

This function and the next function gives directly an interface via RS232 / COM port to a measurement device. You can directly download measured data from our digital -scope (Fluke PM33xxA series) via RS232-cable and display or store it or compare it with simulated data. You can also use this data with the identification functions (see below). You can read out each of the four channels and the contents of the scope memory.

With the button, which is enabled after the first data download, you can redisplay the transferred data until new data is transferred.

Interface to Gain Phase Analyzer

This is similar to the above function and can read out the data of our Schlumberger 1260 Gain Phase Analyzer via RS232. So it is very easy to compare measured and theoretical bode plots in one display. This is a very powerful feature of this program and we use it in our Lab in several experiments.

With the left buttons, which are enabled after the first data download, you can redisplay the transferred data until new data is transferred.

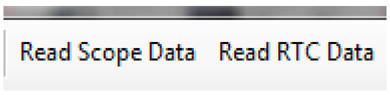
Menu items without tool buttons

Menu Actions → Comma Info

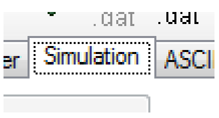
Here you get information about the actual setting of the decimal separator. German PCs usually uses a comma, US- PCs a point. Problem appears, if a text file contains floating point data. Then a conversion has to be done. If the textfile is created by a German PC and should be converted by a US- PC or vice versa, this must fail. The problem is increased by the “thousand-separator” to group large numbers. In German PCs this is the point and in US PCs this is the comma. So for example is the number 1,000 the German One or the US Thousand? The computer doesn’t know it.
I have decided to write my own conversion function. It can be found in FormMain.cs and is called *public void ConvertToDouble(string text,ref double nb).* This conversion is independent of German/US version and takes any separator (point or comma) as decimal separator and replaces it with the actual used setting on the running PC. Only if two or more separators appear in the number then this function takes the right one as decimal separator. So the number 1,000 or 1.000 are always read as One.

Menu: Identification →By time function LS

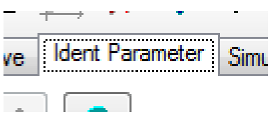
(see also the identification tutorial at the end of this paper)
This is the way to identify dynamic systems by measurement of step responses. A least square method is used to calculate system function parameters by minimizing the least square sum between a measured and a calculated curve. The minimum-search is done by a modified Gauss Newton method.
To start this function you have to load the measured data into the memory of the identification module. This can be done by loading a file with measured data (any ASCII file is possible).



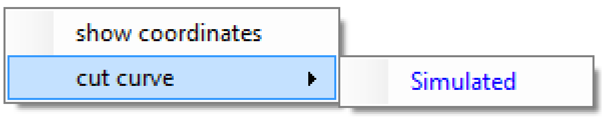
A second loading method is by copying direct generated data from the real-time-controller window or via scope interface. There you can measure step responses of real processes via ADC or with simulated models (see there) or transmitted scope data with step responses. After generation of data the relevant Buttons are enabled. (Either RTC-Data from realtime- controller window or “Read Scope-Data” from Scope interface window). Then you can transfer the data with one mouse click to the identification memory.



If you only want to play with this identification, you can use the ‘Simulation’-tab sheet. Then you can calculate simulated data, on desire with random noise, and try to identify the original parameters. If no noise is used, the original values should be the result.

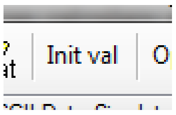


The identification is prepared for at this time 11 different models: The step responses of the standard process models PT1, IT1, 2PT1, PT2 (d<1), PIDT1 and DT2 (d<1) all with possible offset voltages and additional delay. I have added four more functions: First the polynomial function with up to the degree of 6 (7 coefficients) then an exponential function and 2 hyperbolic functions. But if you want to have the identification with other type of functions, please contact me and we talk about a solution. I need only the equation for the curve and all derivatives to the identified parameters. In the parameter window you can see an information window by clicking on the ‘?’-button with all equations. The default start model is the PT1-model.
If the curve is now loaded or generated, the identification can be started.

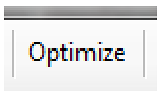


If there is some saturation or any other nonlinearity at the left part of the curve, you can cut off the saturated part by opening the Popup menu with right mouse click. After choice of the item and the curve a box appears with a confirmation question, and with a 'Yes' the curve is reduced to the valid points.

But first you have to know, that for nonlinear equations (and an exponential function is rather nonlinear!) the solution can only be calculated with an iterative procedure.

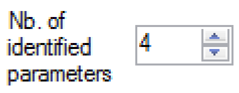


This needs good starting values. The problem can be compared with a search of a minimum or valley in a mountain. If you start in the wrong region and look only for negative gradients maybe you can't find the absolute (or global minimum) and you get only a local minimum. In other words you need good initial values before optimization procedure should be started (Except of the polynomial function, this identification gets from any starting set of values the result in one step!). This initial values can be automatically generated with the 'Init val'-Button. Therefore I have prepared some methods (taught in my CS-lecture) to get good starting values for most of the step responses. I try to find out some characteristic values (turn or inflection tangents, maximum points and so on). After clicking you can directly see the 'initial val'-result, because the program displays a second curve drawn with these values. Then you can decide to optimize the result 'by hand' or try to start an automatic optimization. Always if you click the 'curve'-button the program calculates and draws the curve with the parameter in the parameter tab sheet.



Now the moment to try an optimization has come. After clicking the 'Optimize'-button you can directly see the result by looking on the second curve. If this curves fits with your measured data, you can finish identification.

A second point of remark is the number of parameters the program will optimize. You can change this number with the *numericalUpDown*-field, which indicates the number of parameter starting from top of the list of parameters. Is for example in the 2PT1-model this number =3, the optimization changes only K, T1 and T2. If you increase to 4, the Uoffs-value also will be optimized and with 5 additionally the delay time is changed to optimum. The program starts to optimize step responses with fixed delay time, because this is a sensible parameter often causing non-convergence, if the starting value of delay is rather wrong. All parameter fields changed by optimization are colored.



In the parameter box the number behind S= .. gives in percentage the average least square error between measured and identified curve. In low noise systems this should be lower than 1%.

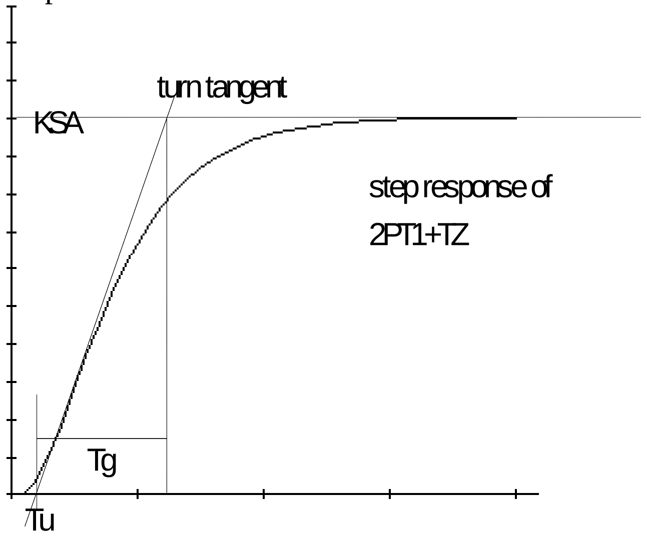
You can store the last curve values in a *.SIM-type file after identification, but the parameters are not stored. You can printout the curves and in a bottom line the parameters are documented. By leaving this window with the green arrow-button the parameters are stored for further purpose (controller design) in the PC-memory. Leaving with the red-X button clears all results.

Menu: Identification →Reuter 2PT1- Identification

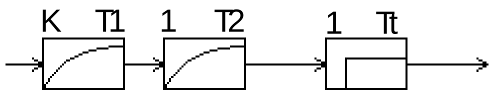
(No tool button)

This method uses the turn or inflection tangent (tangent at the point of inflection, inflection tangent, that with the steepest gradient) of the step response of the 2PT1-process or similar ones. The final value must be known to determine the crossing point at T_g . Measure and plot the step response, draw the turn tangent (see above)) and determine the times T_u and T_g (see picture). For test purposes you can type in a process with any time constants T_1 and T_2 and recalculate the times T_u and T_g . Is a pure 2PT1-model not possible automatically a delay time will be added with T_z.

Step response:



Signal block diagram of the identified process:



Menu: Identification →By Off-line LS-method

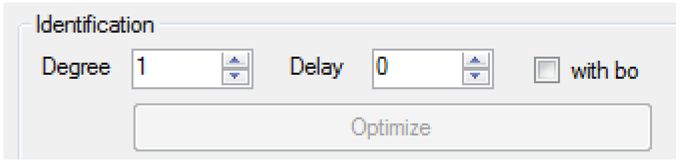
(No tool button)

This is a second method to identify dynamic systems. Input data is not necessarily a step function, but can be any function, which is useful. Even the controller output of a running system is useful, so this method can be used, if the system which should be investigated, can not be exited with steps or other generated signals, but can be measured directly in an industrial process, which can't be disturbed.

The only condition is steady state before measurement starts. Only then you can be sure, that any change in the output signal is caused by the input signal.

The output of system is the transfer function F(z) of a digital filter with unlimited order (see starting curve tab sheet with the max number of parameter the program is compiled). The order must be guessed by yourself, but this is quickly done by testing with low order and increasing step by step until no improvement can be observed.

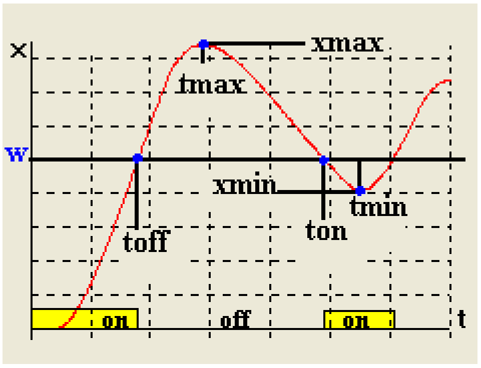
Details: Load the input (left button) and the output (right button) of a system as two different data files with the 'load'-buttons or with the 'Get RTC Data' -button, if this is enabled by an internal measurement, for example in the real-time-controller box. So any signals generated by an open or closed loop process can be used.



If there is some saturation at the left part of the curve, you can cut off the saturated part by popup menu.
After loading choose the degree and a possible delay as integer multiple of the sampling time T0 and if you want to have a filter with the b_0 -constant unequal to zero. After this you can click the ‘Optimize’ button to get an identification. The result appears in a result-box. If the model degree is one or two and bo=0, the program tries a recomputation of a F(p)-filter function. This is successful, if the model is equal to a PT1, IT1 or a 2PT1-model.
At the same time the program draws a third curve (in green), which is the output of the identified model with the given input data. If the identification is OK, you can’t see the red curve, because the green identified curve is drawn over the red measured curve. In the group box “Identified model” you can see the coefficients of F(z) and the DC-gain of the model. Additionally you can see the sum of the least squares as the average distance between measured and model curve. In a good result this should be in the region of some percent. It should be mentioned, that not all measurements get a converging result, sometimes no combination of degree, delay or bo-factor gives a well-fitting model curve.
The only restriction is, that the process should be linear (no limitation of signal x) and the start is in steady state.
Again the printout is possible including the resulting coefficients. If you close this box with the green arrow-button, the results are copied to the memories of the other boxes to design some controllers.
You can save the resulting model as transfer function .fz with the .fz- button.

.fz Calc new output data with F(z) and input data This module allows to calculate the output of any digital filter to any input. Load or edit input data in the Tab “ASCII in data” of data input, load or edit your filter coefficients with the load-FZ-button and then calculate the output data by clicking on button “calc new output data…”.

Menu : Identification →2- point controller Identification



After input of all 6 values toff, tmax, ton, tmin, xmax, xmin (see left picture) of the measured reference step response of a closed loop with a two point controller (often used in temperature control), you can calculate a new model function with the button ‘identify’. At the identify--> beginning you should try a simple 2PT1-model with fixed or zero delay time T_z. If this is not possible (error message) you can start an automatic T_z -search. A good indicator is the recalculated set value w. If that value is identical to the really used set value, the identification is OK. Check here also the sensitivity.
With the recomputed- button you can recalculate the equivalent 4 points from a given process. Additionally you can see a display of the measured curve, if available and loaded and the curve calculated from the identification. You can also simulate a curve with the ‘simulate- button’ . then the program calculates the step response of a 2PT1- system with given parameters.

<--recompute

If you have loaded the 2-point controller response curve either with the ‘load .dat’-button, then you can use the point finder, started with the relevant button. Then by clicking with right mouse button on any point of the curves this copies the coordinates into two edit-fields. If you have found the correct points you can accept them with OK-button and look for the next point. Same can be done with values from the data editor in the data folder, with the zoom button you can enlarge the curve.

Menu: Identification → PT1 with 2 points

This methods need only two points of the step response of a PT1. With only two points (and the zero-starting point) of the PT1-step response you can calculate the parameters K and T of the PT1. Condition is that the second time t2 is two times the time t1 (t2=2*t1). The final value is not necessary! But the accuracy increases with the times t1 and t2 relating to the time constant T. Check the sensitivity and look the resulting change in the parameters by changing the input values by 1% with the sensitivity-button! Following equations are used, if U_0 is the input step amplitude:

$$U_0 K = \frac{U_1}{2 - U_2/U_1} \; and$$

$$T = \frac{-t_2}{\ln(1 - U_2/U_0 K)}$$

Menu: Identification →Schwarze 2PT1-Identification

This is a similar method from the book ‘Grundriss der praktischen Regelungstechnik’ from Erwin Samal, Verlag R. Oldenburg 1970. You have only to measure the point-pair of 30% and 70% of the final value, which of course has to be known. Or you can measure the pair with 10% and 90% of the final value. This method avoids the turn tangent, but is more sensitive (check this with the sensitivity-key).

Menu: Realtime functions→Student Control Box

This module is used in the CSII Lab and described there in more detail. It is a reduced module depending on the more comfortable one called “Adaptive Advanced Controller”. See there.

Menu: Realtime functions→Data logger DT 9062

This function can read the data of multimeter type DT90xx or DT4000 from Digitek, available at ELV-Elektronik in Leer. This Multimeter is a low cost (about 40 €) measuring device (volt ampere AC / DC and temperature etc) with RS232 interface. Connected to a COM port gives possibility to read in all displayed values, to present it as 7 segment display, to store in a memo, to save as sim-file or to display it as a diagram in main menu.

Menu: Realtime functions→ Inverted xy pendulum

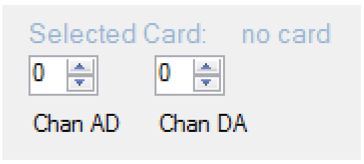
This contains the software to drive a xy- Plotter, which is used to balance a two dimensional inverted pendulum connected to a joy stick. Not implemented yet.

Menu: Realtime functions→ Crane model MUK

This module drives the crane model which is used as demonstration model at MUK in Lübeck some years ago.

Menu: Realtime functions→ Active digital RT- Filter

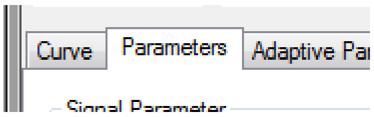
This module converts the PC into an active real-time filter, if an AD-card is connected. First select the card in Menu ADC-Cards, then select a filter either by choosing a filter type and calculate the F(z) or directly load a F(z)- function from a .fz- file. Connect the signal generator or any other signal source with the AD- Input selected in the channel selector and measure the filtered output at the selected DA-output. Sampling times down to 1ms depending on speed of AD-Cards are possible.



With the group box "timing" you can supervise the timing of your hardware.

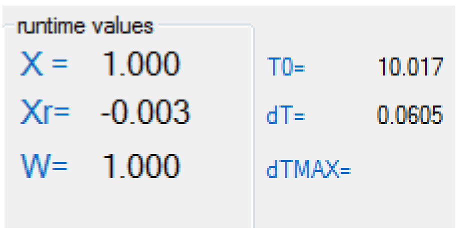
Menu: Realtime functions→ Adaptive Advanced Controller

This is a powerful module of this program. It is designed to test and simulate some digital control loops up to the possibility to control real time processes via connected AD-cards. The lowest sampling time is 1 ms, which needs of course a fast computer, which is today absolutely no problem. The diagrams are totally independent of the main graphic data base, all generated bode plots and other type of curves in the main window remain undisturbed.
Again this database can contain maximum 10 curves and unlimited number of points.
Before you start a control loop action, you should note that a process should be connected via ADC-Card or simulated with menu "ADC-Cards→Hardware Simulation".
Secondly a set of controllers should be designed with the menu Controller Design→Design of digital PI/PIDT1 + div. If you have left that module with the green arrow button, all controllers are available in this section.

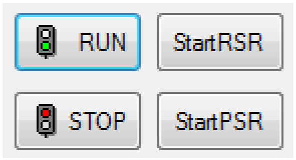


Then define some parameters in the tab page "Parameters":

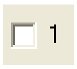
If you click the 'Run'-Button after this definitions, the real-time process is enabled and runs with the chosen T0. Now the settings of the groupBox "Initial conditions" are active. There are several possibilities to define this starting situation before a reference or disturbance step response: Do nothing or drive process with constant value at process input or Control output with some controllers to the given Uoff.



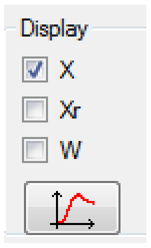
Furthermore you can find some more controls at the top, which relates to the graphic display. At the same time you can observe the input signal Xr and the output signal X of the process and the desired value w in V. These both values are continuously actualized and you can now see reactions of the loop. The PC runs the loop with the chosen initial conditions. The actual sampling time and calculation time is measured.



This initialization-state of the real time process ends either with clicking a start of a reference or a process step response (*Start RSR* or *Start PSR*) or with the *Stop*- button.
If you click on the *Start PSR*-button (PSR=Process step response), the controller is not active. The generator signal is directly applied to the process input and the process output X is displayed. The input signal is the step function, you will see the step response. If the field with the '*1*' is clicked, the program normalizes this response to unit input and you see the unit step response. Notice, that in this case the voltages on the screen are different to the voltages of the process.

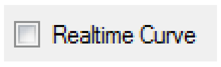


If you click on the *Start RSR*-button (RSR=reference step response), the controller remains activated (since *RUN*-button) and now the signal generator output is connected to the reference input w of the closed loop. The chosen controller now tries to control the process output x to the desired value w= Amp + Uoff. After the chosen number of points you can see the response on the display and the control stops.



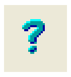
Additionally to the process output x you can decide to add a further curve at each response: either the desired value w (to see control errors) or the controller output (to check amplitudes of the actuator or to identify with Off-line LS) or both. After a measurement you can any time repeat the drawing with the draw-button, because all values are stored internally.

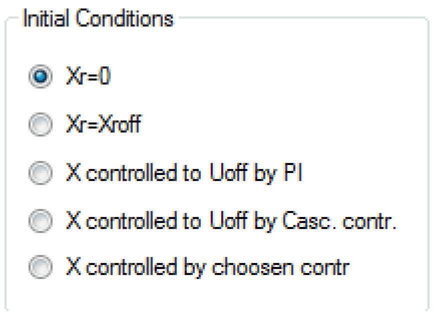
Now some more details to the parameter setting tab sheet. In the group box "*Signal Parameter* you can choose the offset Uoff and amplitude voltage Amp and the kind of the signal. At this time there are four offers: simple step function ending automatically after chosen number of points, continuous running binary random impulse functions with pulse width Pw. Third is a continuous running rectangular function with pulse width Pw and period defined by the number of points. This is sensefull only with the realtime curve, which must be activated.



This continuous signals stops only by clicking on the *stop* button. The forth method to define the input of the loop is directly reading the signal from an input of the AD-card. (not implemented yet)

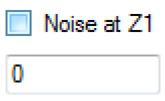
Next choice in this box is, where the input "Amp" of the signal is applied. Either a reference (w) or a disturbance step response is possible. Z1 is the disturbance input in front of the process and Z2 at the end of the process. See the block diagram information button '?'.



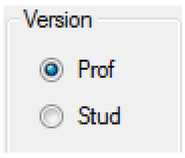

Initial Conditions
Xr=0
Xr=Xroff
X controlled to Uoff by PI
X controlled to Uoff by Casc. contr.
X controlled by choosen contr

Last but not least you should define the initial conditions before the signal is applied, and the output signal after ending of control process. Either you can start this initialization with constant values Xr at the process input (0 or any other value) or you can activate a PI- controller with arbitrary values or the cascaded Controller for the breathing system (works only there!) or the chosen controller to force the process output to the offset voltage Uoff of the signal. Independent of your choice you should not start a response until all changes of X and Xr are nearly zero (wait for steady state). But this controlled initial conditions should be used carefully, because the control loop can be unstable with wrong control parameters and then no initial conditions are defined.

The next parameter setting possibility is the '*Control Parameter*'- group Box. Here you can see or change the digital control parameter coefficient p and q. A change in the grid array only becomes active after clicking the -->*p/q*-button. Other changes will be ignored. Furthermore you can decide to run an ideal algorithm (delay time only caused by the computation time) or the real algorithm (output of the control output at the start of the next sample to get constant, known delays of T0). In some controllers 'ideal algorithm' is not implemented. Last a compensation of a nonlinear effect of the process called dead zone or starting friction is possible. Additionally you can add a random noise to the controller output. This is suitable for the identification method Off-line LS, see above.



The number of displayed parameters depends on the type of controller. In the case of the '*free*' controller you can use those PID-controllers, which are designed in the menu *control-lers/K,T→p,q*. (see below the relevant chapter).



If the stud-Version Check-box is clicked, internal student- controller versions are active. This is only senseful during lab experiment, because the build in stud-versions are all empty.
If you click on the green arrow -button, the window get closed and data is accessible in some simulation programs.

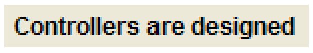
If you leave the RTC-box, the last values of x and (if clicked) the Xr are copied to internal memory to directly transfer the data to the identification procedures accessible with the '*RTC Data*'-Buttons.

Menu: Realtime functions→ CS II Pendulum

This is the module for Experiment 3 of German DRT Lab and UC CS II Lab experiment. Students have to fill the algorithms to balance a standing pendulum.

Menu: Controller Design→Design of digital PI/PIDT1 + div

In this part of the program many digital controllers can be designed for different processes. At this time four different processes and for each process seven different digital controllers are automatically designed by your computer. The processes are the three simple PT1, IT1 and 2PT1 processes and a free choice of a digital filter function (maybe defined in the digital filter window). You can choose for the simple ones the K/T-values, for the digital filter the coefficients of the filter function F(z).
In the case of the three simple processes the four classical controllers P, PI, PD and PID are designed with the standard methods of my lecture with frequency response approach (phase margin). For the digital filter function these classical controllers are not prepared. Use for design of arbitrary filters and processes the program Regdelph, also available at the FHL.
For all four processes the two classical Dead-Beat-Controllers are available as well as the Orienteering Controller coming up from the university of Chemnitz. Try them.
If you later will run an ideal or an real algorithm defines the delay time of the control design. The mode (ideal or real) should be the same in the design and in the real-time process, otherwise the design is not optimal.
An additional output of this window is the disturbance gain, which is the inverse of the controllers DC-gain. In some cases this is interesting (disturbance gain of controllers with I-processes). If this value is not zero, disturbance signals in front of the processes can't be controlled to zero error.
After design (if the last line of the window dis-



plays:) you can activate a printout with the printer button, then the results are copied to the main report memo field. There you get a list of all controller coefficients.
All parameters automatically transferred to the real time control window and can directly be tested there at the relevant process or your hardware.

Menu: Controller Design→K,T → p,q

(No tool button)

Here you can define the parameters of a digital PIDT1-controller by changing the K and T-values in a more comfortable way. This is a way to get a free choice of stepdepth up to a maximum value. The different design methods are described in more detail in my workbook page D19ff. You can convert parallel into bode PID-form. PI is with st=∞, PD is with T_N=∞.

Ideal algorithm: delay time $T_t=T_0/2+T_r$

$$x_R(nT_0) = p_1 x_R((n-1)T_0) + p_2 x_R((n-2)T_0) + q_0 x_d((n)T_0) + q_1 x_d((n-1)T_0) + q_2 x_d((n-2)T_0)$$

real algorithm: delay time $T_t=3T_0/2$

$$x_R(nT_0) = p_1 x_R((n-1)T_0) + p_2 x_R((n-2)T_0) + q_0 x_d((n-1)T_0) + q_1 x_d((n-2)T_0) + q_2 x_d((n-3)T_0)$$

Type	F(p)	q_i, p_i, all missing q_i,p_i=0	remarks
PI	$K_R(1+pT_N)/pT_N$	$q_0=K_R(1+T_0/T_N)$, $q_1=-K_R$, $p_1=1$	recursive
PD	$K(1+pT_V)$	$q_0=K(1+T_V/T_0)$, $q_1=-K(2T_V/T_0+1)$, $q_2=KT_V/T_0$, $p_1=1$, $st\approx 1+T_V/T_0$	recursive
PID	$K(1+pT_D+1/pT_i)=$ $\frac{K_R(1+pT_N)(1+pT_V)}{pT_N}$	$q_0=K(1+T_D/T_0+T_0/T_i)$, $q_1=-K(2T_D/T_0+1)$, $q_2=KT_D/T_0$, $p_1=1$, $st\approx 1+T_V/T_0$	recursive design I
PIDT1	$K(1+pT_D+1/pT_i)/(1+pT_1)$, $\frac{K_R(1+pT_N)(1+pT_V)}{pT_N(1+pT_1)}$ $st_{max}=(T_N+T_V)/T_N+T_V/T_0$, $st_{analog}=T_V/T_1$	$q_0=\frac{K}{1+T_1/T_0}\left(1+\frac{T_D}{T_0}+\frac{T_0}{T_i}\right)$, $q_1=\frac{K}{1+T_1/T_0}\left(-1-2\frac{T_D}{T_0}\right)$ and $q_2=\frac{K}{1+T_1/T_0}\left(\frac{T_D}{T_0}\right)$ $p_1=\frac{1+2T_1/T_0}{1+T_1/T_0}$ and $p_2=\frac{-T_1/T_0}{1+T_1/T_0}$	recursive design II rectangular
PIDT1	$K(1+pT_D+1/pT_i)/(1+pT_1)$, $\frac{K_R(1+pT_N)(1+pT_V)}{pT_N(1+pT_1)}$ $st_{max}=(T_N+T_V)/T_N+T_V/T_0$, $st_{analog}=T_V/T_1$	$q_0=Ka\left(1+\frac{T_D}{T_0}+\frac{T_0}{T_i}\right)$, $q_1=Ka\left(-1-2\frac{T_D}{T_0}\right)$ $q_2=Ka\left(\frac{T_D}{T_0}\right)$, $p_1=2-a$, $p_2=a-1$ $a=\frac{T_D/T_1}{1+T_D/T_0}$	recursive design III rectangular, starting impulses are equal
PIDT1	$K(1+pT_D+1/pT_i)/(1+pT_1)$, $\frac{K_R(1+pT_N)(1+pT_V)}{pT_N(1+pT_1)}$ $st_{max}=(T_N+T_V)/T_N+T_V/T_0$, $st_{analog}=T_V/T_1$	$q_0=b(1+2\frac{T_N}{T_0})(1+2\frac{T_V}{T_0})$, $b=\frac{K_RT_0}{2T_N(1+2T_1/T_0)}$ $q_1=b\left((1-2\frac{T_N}{T_0})(1+2\frac{T_V}{T_0})+(1+2\frac{T_N}{T_0})(1-2\frac{T_V}{T_0})\right)$ $q_2=b(1-2\frac{T_N}{T_0})(1-2\frac{T_V}{T_0})$, $p_1=\frac{4T_1/T_0}{(1+2T_1/T_0)}$ and $p_2=1-p_1$	recursive design IV trapezoidal HF-gains are equal

Menu: Controller Design→Flash Design 2PT1

This design can only be used, if the delay time T_z is zero. If $T_z \neq 0$ there will be a warning and T_z will be ignored. The name of this method is created by myself, the design is done with some simple equations, so you get your result as fast as a flash! Pole compensation is used (compensation of the largest process time constants). Then the closed loop behaviour is equal to a PT2 with the DC gain 1 (no control error), the damping d and the natural frequency ω_0. All parameters are correlated by simple equations. In the window you can change all numbers in the white fields. Then the other related numbers are calculated and actualized. The relation between damping and phase margin should be repeated here:

$d = 0.5 \Big/ \sqrt{\sqrt{(\tan^2(90° - \varphi_R) + 0.5)^2 - 0.25}}$, and vice versa

$\varphi_R = 90° - \arctan\sqrt{-0.5 + \sqrt{(4d^4 + 1)/16d^4}}$. With st=1 you get a PI-controller. The value Kr*st is a good indication for the maximum controller output amplitude after a reference step. Additionally you can directly get a proposal of an operational amplifier circuit with choice of either the R_1 (often 100kOhm) or at large time constants the capacitor C_2. If you type in the parameters of the controller, then the design parameters are hidden and you can directly calculate the Rs and Cs of an Operational amplifier circuit.

The main equations of p.WB 46 and PT2-relations on WB 17-19 are:

$$F_R = K_R \frac{(1+pT_N)(1+pT_v)}{pT_N(1+p\frac{T_v}{st})};\ F_S = \frac{K}{(1+pT_1)(1+pT_2)} \Rightarrow K_R = \frac{stT_1}{4T_2 d^2 K};\ \omega_o = {st}\big/{2dT_2}$$

with T_N=T_1 and T_V=T_2, if T_1 is the largest process time constant.

$$\ddot{u} = \frac{\Delta U_1}{K} = \exp\left[\frac{-d\pi}{\sqrt{(1-d^2)}}\right] or\ d = \frac{-\ln(\ddot{u})}{\sqrt{\pi^2 + (\ln \ddot{u})^2}}\ (Eq.6.5)$$

$$t_m = \pi / \omega_E = \pi / \omega_o \sqrt{(1-d^2)}$$

Menu: Controller Design→FRA 2PT1with delay

If the process has a delay time $Tz \neq 0$ the flash design cannot be used. In this case the standard FRA-design (frequency response approach) should be used to get a good PIDT1-controller. Also pole compensation defines the time constants of the controller T_N and T_V equal to the process time constants T_1 and T_2 (with T_N>T_V) and the phase curve of the open loop bode plot with the equation $\varphi = -\omega T_z - 90° - \arctan(\omega T_2 / st)$ is investigated for the crossover frequency ω_d with the condition $\varphi(\omega_d) = -180° + \varphi_R$ with φ_R as the phase margin. This search is done by nested intervals. K_R is the compensation gain to get unity gain. The relevant equation is

$K_R = \frac{\omega_d T_1}{K}\sqrt{1 + \left(\frac{\omega_d T_2}{st}\right)^2}$. The damping relations are similar to that of the flash-design, so the above relations are used, too. For Tz=0 the results are identical to that of the flash controller design.

Menu: Some Tools→Nichols equations

In this menu point you can calculate the closed loop bode plot value from the open loop bode plot value by using the following Nichols equations:

$$|F| = \frac{|F_0|}{\sqrt{1+2|F_0|\cos\varphi_0 + |F_0|^2}} \quad \textit{and} \quad \varphi = \arctan\frac{\sin\varphi_0}{\cos\varphi_0 + |F_0|}$$

The phase value is corrected so negative input give negative output values

Menu: Some Tools→PT2- equations

In this item you can calculate the relative overshoot ü and the time of the first maximum dependent of the d and ω_0 of the step response of a PT2-block and vice versa. The equations are:

$d = -\ln(\ddot{u}) \big/ \sqrt{\pi^2 + (\ln(\ddot{u}))^2}$ and $\omega_0 = \pi/t_{\max}(1-d^2)$,

and the correspondent equations :

$\ddot{u} = \exp(-d\pi/\sqrt{1-d^2})$ and $t_{\max} = \pi/\omega_0(1-d^2)$

After input of one new number and leaving the field all other three values will be calculated. The change from one field to the next is optimally done with the TAB-key.

Menu: Some Tools→2-RC-Block

Here you can calculate the both time constants T_1 and T_2 and DC-gain K of a series circuit of two 2 RC-blocks (see picture in the window).

Menu: Some Tools→Curve plotter

This module can create some simple curves on the graphic display to demonstrate the graphic possibilities.

Menu: Some Tools→Freq- Calculator

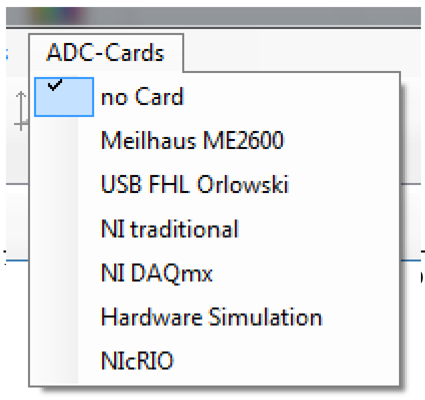
With this tool you can simply convert in bode plot displays the horizontal values x into frequency and radian frequency. With change of one of this three values both other values are calculated. Same is possible with vertical values. You can convert dB – values in factor and vice versa. If the coordinate function of display is activated with popup menu of the graphic, the values are transferred automatically with a left mouse click into this window.

Menu: Some Tools→PID Mag Phase

This module calculates the magnitude and phase values of single points of a PIDT1- function.

Menu: ADC- Cards

This function gives a quick overview over the state of the connected AD-Cards , at this time possible are:



NI traditional is only possible for old NI- cards in Win XP, NI stopped the support.

You can see all voltages of all inputs and you can change all DA- outputs in one volt steps.

“Hardware Simulation” opens a window where you can select a model defined either by selection of PT1- IT1- PT2 parameters or by loading an F(z). With this module you can test controllers in the Realtime functions without connected processes over AD- cards.

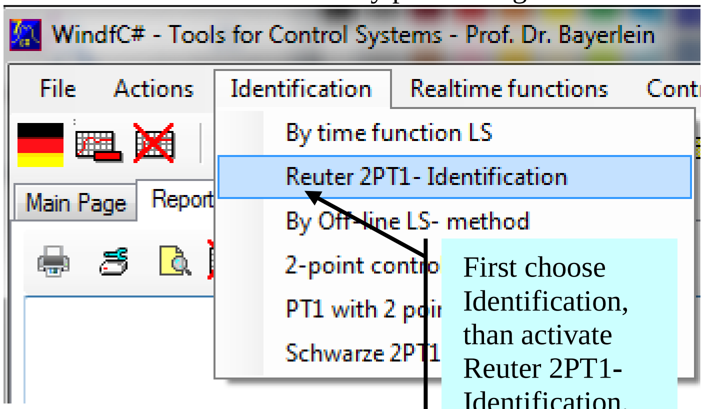
Next steps in updating the program Windfc#:

Still missing is the 3D- Display of mesh plots of analog filters and the module of xy- inverted pendulum.
At this time we have a Meilhaus ME4600 which we want to implement. Further AD- Cards are in work. We are waiting for new Orlowski USB. 2.0 card which can reach 1 ms for 1 AD-DA step.
New ideas to complete this program with useful functions are welcome.
Good Luck in testing the program
Prof. Dr. Bayerlein
FH- Lubeck

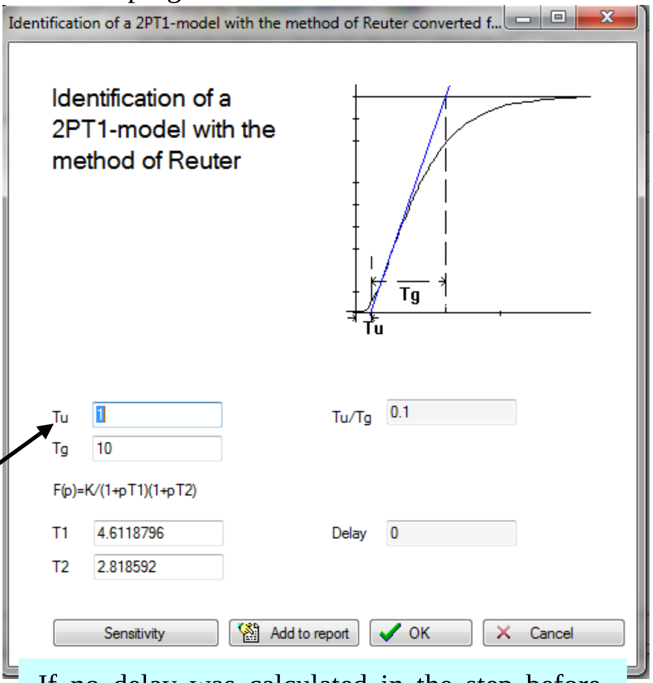
Tutorials for the use of this program for our American students written by Mr. Clasen.

Controller design with Reuter method supported by Windfc#

With the determined process parameters T_u, T_g and Ks from your step response plot, you can design a PI- and PIDT1- control with any phase margin under assistance of the program Windfc#.

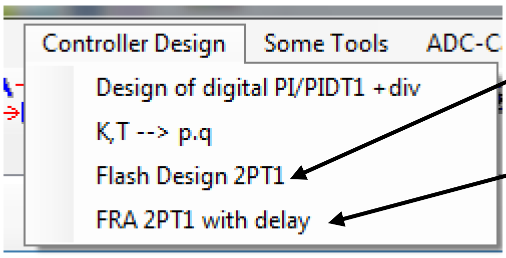


First choose Identification, than activate Reuter 2PT1-Identification.


Identification of a 2PT1-model with the method of Reuter converted f...
Identification of a 2PT1-model with the method of Reuter
Tg
Tu
Sensitivity
Add to report
OK
Cancel

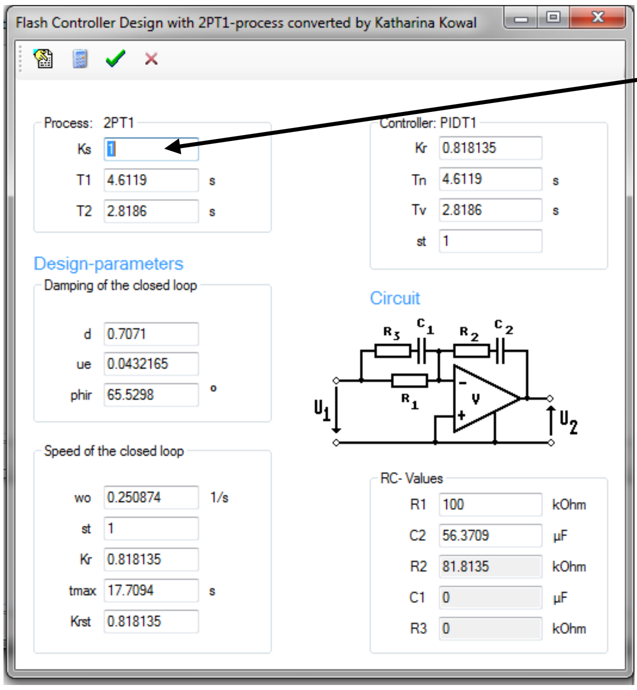
Tu 1 Tu/Tg 0.1
Tg 10
F(p)=K/(1+pT1)(1+pT2)
T1 4.6118796 Delay 0
T2 2.818592

Type in your determined values of T_u and T_g. Press the Sensivity-Button. T1, T2 and perhaps a delay will be computed. Close the window with green arrow and run the controller design program



If no delay was calculated in the step before, you can use the flash PIDT1-design. If only a very small delay was detected, set it down to zero, T1 will change itself.

If the detected delay is too big, that means if the changing of T1 is significant, if you set delay to zero, you have to use the FRA PIDT1-design for 2 PT1 and additional delay time.


Flash Controller Design with 2PT1-process converted by Katharina Kowal
Circuit
R3
C1
R2
C2
R1
V
U1
U2

Process: 2PT1
Ks 1
T1 4.6119 s
T2 2.8186 s

Controller: PIDT1
Kr 0.818135
Tn 4.6119 s
Tv 2.8186 s
st 1

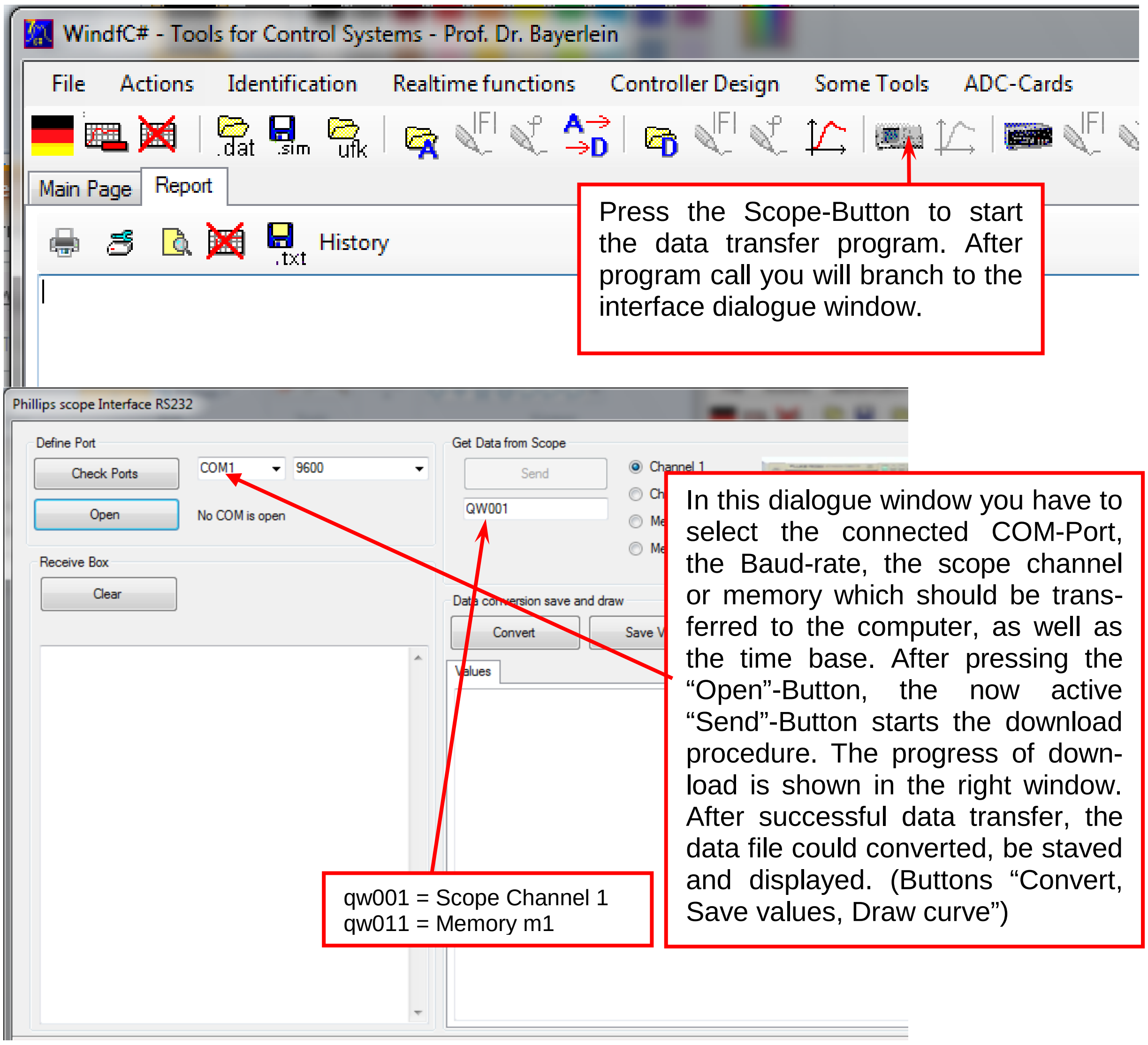
Design-parameters
Damping of the closed loop
d 0.7071
ue 0.0432165
phir 65.5298 °

Speed of the closed loop
wo 0.250874 1/s
st 1
Kr 0.818135
tmax 17.7094 s
Krst 0.818135

RC- Values
R1 100 kOhm
C2 56.3709 µF
R2 81.8135 kOhm
C1 0 µF
R3 0 kOhm

Type in your calculated Ks and set the phase margin to 60°. The step depth st has to be set to 1 (PI), 2, 5 and 10 (PIDT1). Write down the calculated values for the R & C-components, as well as the values for t_{max} and ue into the table given below. If you have to use the FRA-Controller design, because of the delay time, the procedure is similar. Only t_{max} will not be calculated. For documentation you click on "Add to Report". In the main report results will be added.

Data transfer from Fluke-Scope via RS232 interface to computer by program WINDFC#

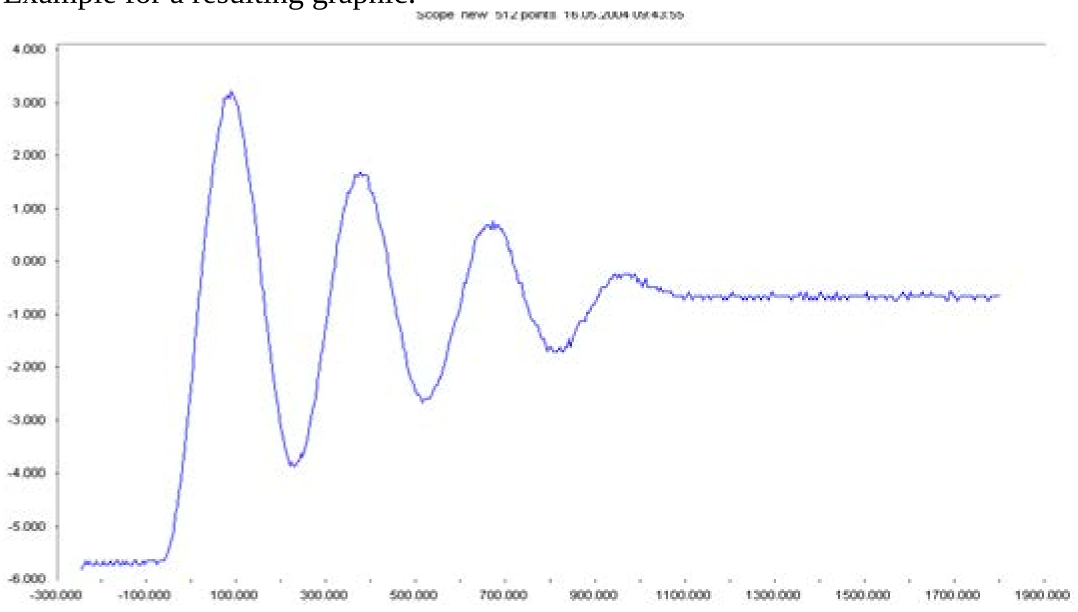


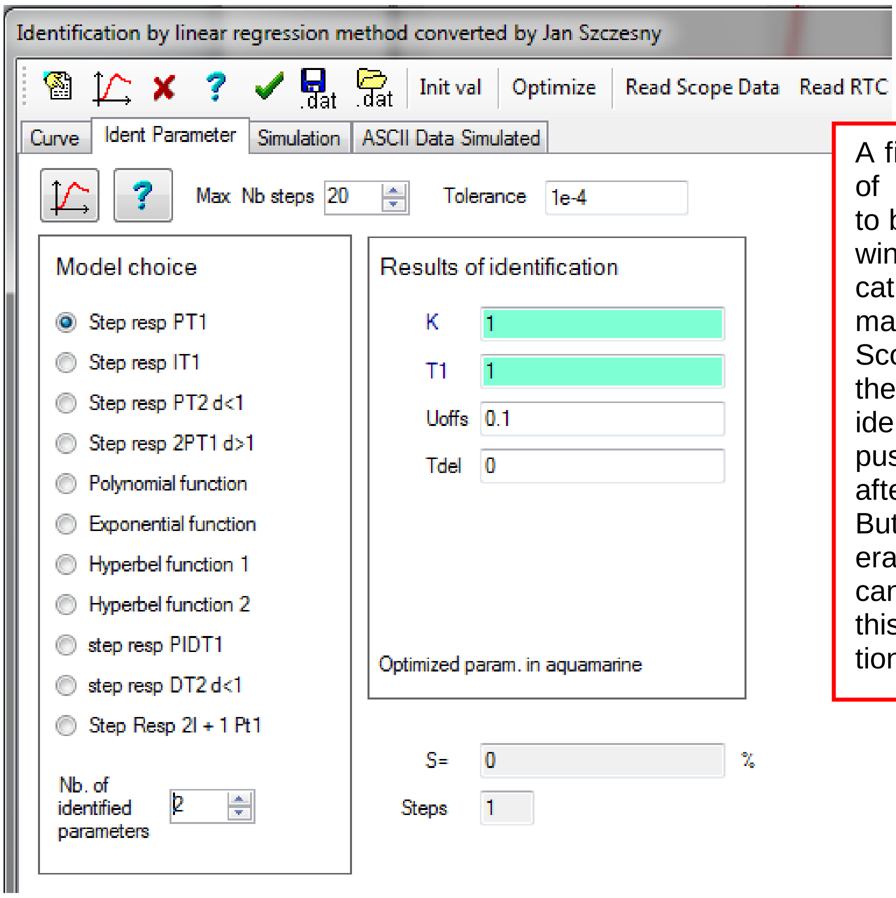
Press the Scope-Button to start the data transfer program. After program call you will branch to the interface dialogue window.

In this dialogue window you have to select the connected COM-Port, the Baud-rate, the scope channel or memory which should be trans-ferred to the computer, as well as the time base. After pressing the “Open”-Button, the now active “Send”-Button starts the download procedure. The progress of down-load is shown in the right window. After successful data transfer, the data file could converted, be staved and displayed. (Buttons “Convert, Save values, Draw curve”)

qw001 = Scope Channel 1
qw011 = Memory m1

Example for a resulting graphic:





Identification by linear regression method converted by Jan Szczesny

Max Nb steps 20   Tolerance 1e-4

Model choice

- [✓ Step resp PT1]
- Step resp IT1
- Step resp PT2 d<1
- Step resp 2PT1 d>1
- Polynomial function
- Exponential function
- Hyperbel function 1
- Hyperbel function 2
- step resp PIDT1
- step resp DT2 d<1
- Step Resp 2I + 1 Pt1

Nb. of identified parameters 2

Results of identification

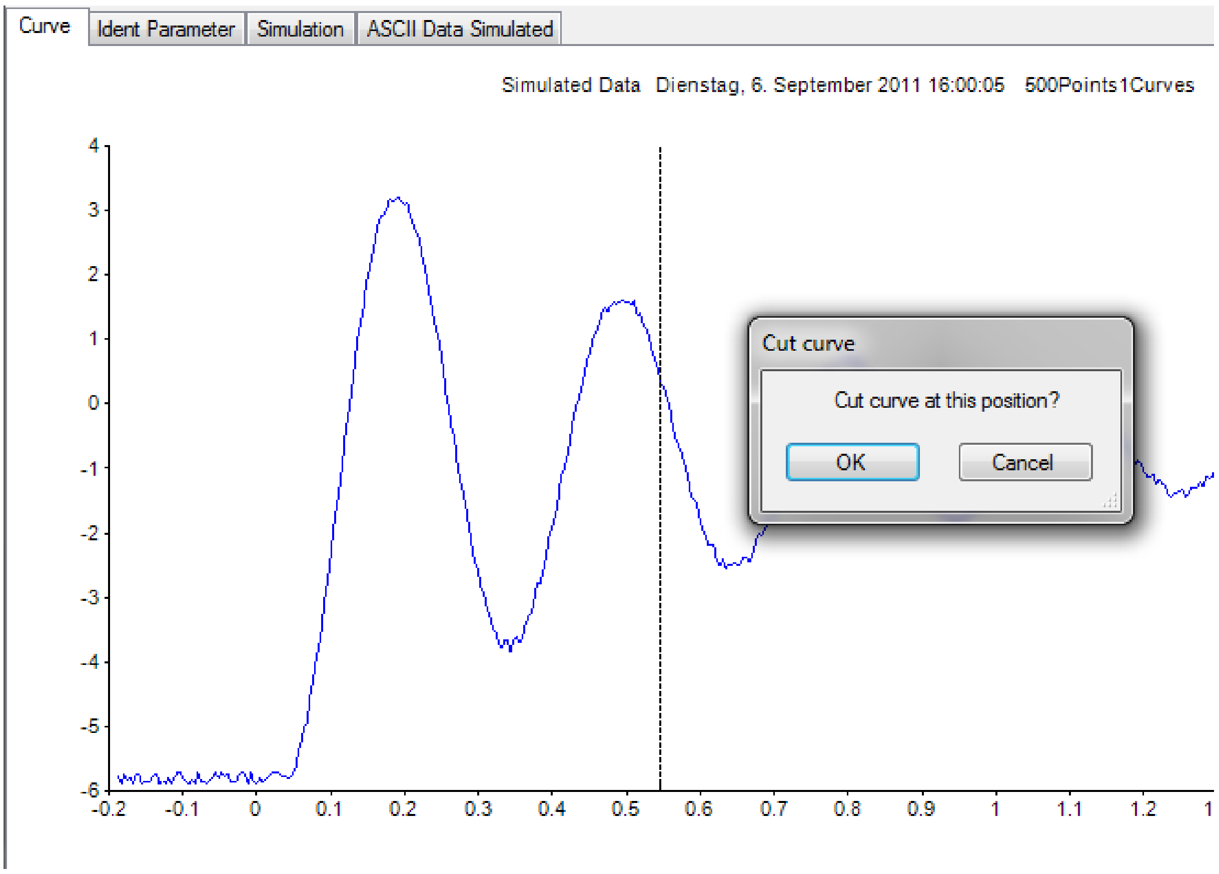
K	1
T1	1
Uoffs	0.1
Tdel	0

Optimized param. in aquamarine

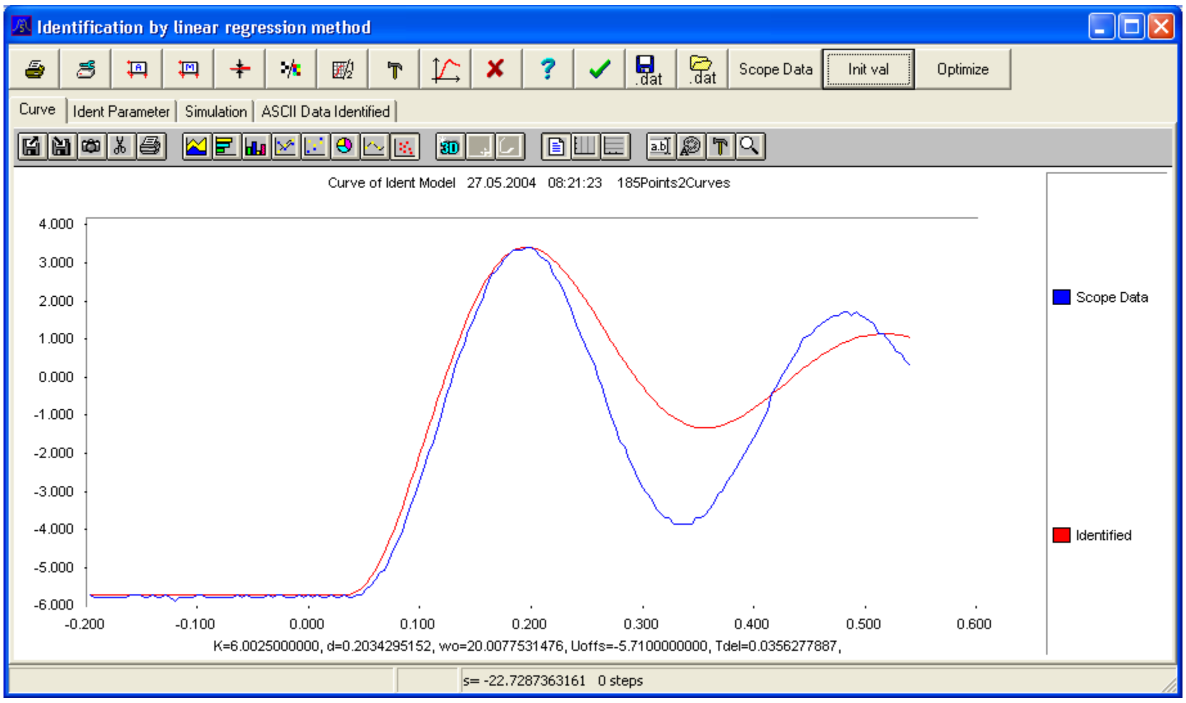
S= 0 %

Steps 1

A fitting model and the number of identified parameters have to be chosen in this parameter-window. The results of identification process could be varied manually. Activating the „Read Scope Data“-Button transfers the downloaded data into the identification window. Now push the “Init val“-Button, and after that the „Optimize“-Button. If it is necessary, several variations of parameters can be done manually. After this you will get the identification result.



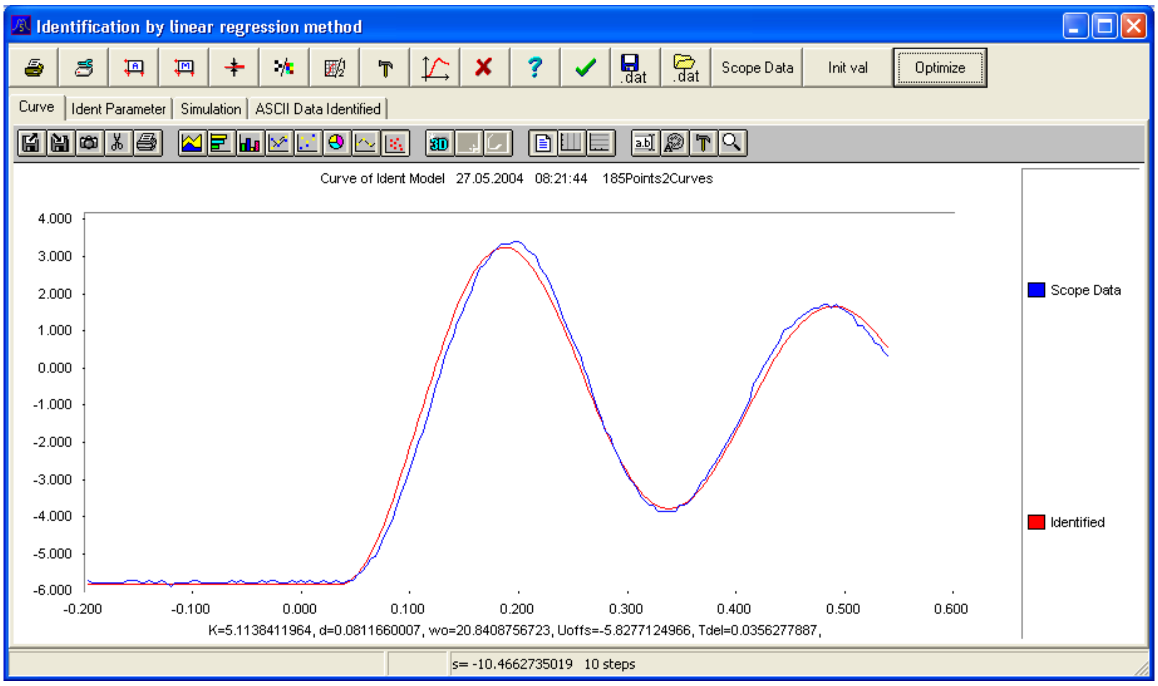
Because measured curves sometimes do not match to the selected model in all parts, it is possible to cut off not matching parts on the right of the curve by popup menu → cut curve . Select the x value where you want to cut and click left. In our example of a real spring-mass-damping system, the second part of the measured curve does not match to the PT2 theory (because of friction), therefore it will be cut off.
Result: Not matching parts of the curve were removed by this method:
Note : Screen shot are done with an old version, but the curves in Windfc# are the same.
Activating the “Init val“-Button produces this first identification.



K=6.0025000000, d=0.2034295152, wo=20.0077531476, Uoffs=-5.7100000000, Tdel=0.0356277887,

s= -22.7287363161 0 steps

Activating the ”Optimize“-Button leads to a better harmony between measured and modeled values. The detected model parameters will be displayed below the time base and could be printed out together with the curves.



K=5.1138411964, d=0.0811660007, wo=20.8408756723, Uoffs=-5.8277124966, Tdel=0.0356277887,

s= -10.4662735019 10 steps