Agilent Technologies

Advanced Design System 2011.01

Feburary 2011
Interactive Controls and Displays

Acknowledgments

SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

7-zip - 7-Zip Copyright: Copyright (C) 1999-2009 Igor Pavlov. Licenses for files are: 7z.dll: GNU LGPL + unRAR restriction, All other files: GNU LGPL. 7-zip License: This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version. This library is distributed in the hope that it will be useful,but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details. You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA. unRAR copyright: The decompression engine for RAR archives was developed using source code of unRAR program.All copyrights to original unRAR code are owned by Alexander Roshal. unRAR License: The unRAR sources cannot be used to re-create the RAR compression algorithm, which is proprietary. Distribution of modified unRAR sources in separate form or as a part of other software is permitted, provided that it is clearly stated in the documentation and source comments that the code may not be used to develop a RAR (WinRAR) compatible archiver. 7-zip Availability: http://www.7-zip.org/

AMD Version 2.2 - AMD Notice: The AMD code was modified. Used by permission. AMD copyright: AMD Version 2.2, Copyright © 2007 by Timothy A. Davis, Patrick R. Amestoy, and Iain S. Duff. All Rights Reserved. AMD License: Your use or distribution of AMD or any modified version of AMD implies that you agree to this License. This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version. This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details. You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA Permission is hereby granted to use or copy this program under the terms of the GNU LGPL, provided that the Copyright, this License, and the Availability of the original version is retained on all copies.User documentation of any code that uses this code or any modified version of this code must cite the Copyright, this License, the Availability note, and "Used by permission." Permission to modify the code and to distribute modified code is granted, provided the Copyright, this License, and the Availability note are retained, and a notice that the code was modified is included. AMD Availability: http://www.cise.ufl.edu/research/sparse/amd

UMFPACK 5.0.2 - UMFPACK Notice: The UMFPACK code was modified. Used by permission. UMFPACK Copyright: UMFPACK Copyright © 1995-2006 by Timothy A. Davis. All Rights Reserved. UMFPACK License: Your use or distribution of UMFPACK or any modified version of UMFPACK implies that you agree to this License. This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License

as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version. This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details. You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA Permission is hereby granted to use or copy this program under the terms of the GNU LGPL, provided that the Copyright, this License, and the Availability of the original version is retained on all copies. User documentation of any code that uses this code or any modified version of this code must cite the Copyright, this License, the Availability note, and "Used by permission." Permission to modify the code and to distribute modified code is granted, provided the Copyright, this License, and the Availability note are retained, and a notice that the code was modified is included. UMFPACK Availability: http://www.cise.ufl.edu/research/sparse/umfpack UMFPACK (including versions 2.2.1 and earlier, in FORTRAN) is available at http://www.cise.ufl.edu/research/sparse . MA38 is available in the Harwell Subroutine Library. This version of UMFPACK includes a modified form of COLAMD Version 2.0, originally released on Jan. 31, 2000, also available at http://www.cise.ufl.edu/research/sparse . COLAMD V2.0 is also incorporated as a built-in function in MATLAB version 6.1, by The MathWorks, Inc. http://www.mathworks.com . COLAMD V1.0 appears as a column-preordering in SuperLU (SuperLU is available at http://www.netlib.org). UMFPACK v4.0 is a built-in routine in MATLAB 6.5. UMFPACK v4.3 is a built-in routine in MATLAB 7.1.

Qt Version 4.6.3 - Qt Notice: The Qt code was modified. Used by permission. Qt copyright: Qt Version 4.6.3, Copyright (c) 2010 by Nokia Corporation. All Rights Reserved. Qt License: Your use or distribution of Qt or any modified version of Qt implies that you agree to this License. This library is free software; you can redistribute it and/or modify it under the
terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version. This library is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details. You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA Permission is hereby granted to use or copy this program under the terms of the GNU LGPL, provided that the Copyright, this License, and the Availability of the original version is retained on all copies.User
documentation of any code that uses this code or any modified version of this code must cite the Copyright, this License, the Availability note, and "Used by permission." Permission to modify the code and to distribute modified code is granted, provided the Copyright, this License, and the Availability note are retained, and a notice that the code was modified is included. Qt Availability: http://www.qtsoftware.com/downloads Patches Applied to Qt can be found in the installation at:
$HPEESOF_DIR/prod/licenses/thirdparty/qt/patches. You may also contact Brian Buchanan at Agilent Inc. at brian_buchanan@agilent.com for more information.

The HiSIM_HV source code, and all copyrights, trade secrets or other intellectual property rights in and to the source code, is owned by Hiroshima University and/or STARC.

Errata The ADS product may contain references to "HP" or "HPEESOF" such as in file names and directory names. The business entity formerly known as "HP EEsof" is now part of Agilent Technologies and is known as "Agilent EEsof". To avoid broken functionality and to maintain backward compatibility for our customers, we did not change all the names and labels that contain "HP" or "HPEESOF" references.

Warranty The material contained in this document is provided "as is", and is subject to being changed, without notice, in future editions. Further, to the maximum extent permitted by applicable law, Agilent disclaims all warranties, either express or implied, with regard to this documentation and any information contained herein, including but not limited to the implied warranties of merchantability and fitness for a particular purpose. Agilent shall not be liable for errors or for incidental or consequential damages in connection with the furnishing, use, or performance of this document or of any information contained herein. Should Agilent and the user have a separate written agreement with warranty terms covering the material in this document that conflict with these terms, the warranty terms in the separate agreement shall control.

Technology Licenses The hardware and/or software described in this document are furnished under a license and may be used or copied only in accordance with the terms of such license. Portions of this product include the SystemC software licensed under Open Source terms, which are available for download at http://systemc.org/ . This software is redistributed by Agilent. The Contributors of the SystemC software provide this software "as is" and offer no warranty of any kind, express or implied, including without limitation warranties or conditions or title and non-infringement, and implied warranties or conditions merchantability and fitness for a particular purpose. Contributors shall not be liable for any damages of any kind including without limitation direct, indirect, special, incidental and consequential damages, such as lost profits. Any provisions that differ from this disclaimer are offered by Agilent only.

Restricted Rights Legend U.S. Government Restricted Rights. Software and technical data rights granted to the federal government include only those rights customarily provided to end user customers. Agilent provides this customary commercial license in Software and technical data pursuant to FAR 12.211 (Technical Data) and 12.212 (Computer Software) and, for the Department of Defense, DFARS 252.227-7015 (Technical Data - Commercial Items) and DFARS 227.7202-3 (Rights in Commercial Computer Software or Computer Software Documentation).

About Interactive Controls and Displays

The Interactive Controls and Displays library provides components that can interactively control the simulation and display simulation data.

Most components in this library are used to display simulation data in various forms, such as bar graphs (*TkBarGraph* (controls-displays), *TkHistogram* (controls-displays), *TkMeter* (controls-displays), *LMS_TkPlot* (controls-displays), and *LMS_CxTkPlot* (controls-displays)), signal level versus sample number plots (*TkPlot* (controls-displays)), X signal level versus Y signal level plots (*TkXYPlot* (controls-displays)), eye diagrams (*TkEye* (controls-displays)), constellation plots (*TkConstellation* (controls-displays)), or just numbers (*TkIQrms* (controls-displays), *TkPower* (controls-displays), *TkShowValues* (controls-displays), *TkText* (controls-displays), *TkShowBooleans* (controls-displays)).

TkButtons (controls-displays) and *TkSlider* (controls-displays) enable the designer to interactively change the signal level at the nodes where these are connected.

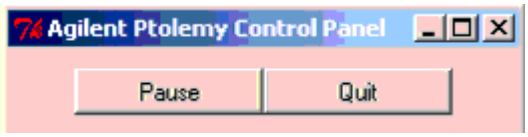
Other components provide forms of interactively controlling the simulation, such as running any Tcl script (*TclScript* (controls-displays)) or breaking the simulation when a certain condition is satisfied (*TkBreakPt* (controls-displays)).

Most components in this library are sinks (components with no output ports)-- sinks only consume data (these do not produce data). Like the non-interactive sinks, these components keep the simulation running as long as needed to consume data. The difference between the interactive and non-interactive sinks is that the non-interactive sinks control the amount of data consumed through parameters (typically called Start and Stop) that the designer must set before the simulation starts, whereas the interactive sinks keep consuming data until the designer interactively terminates the simulation. This process is explained further next.

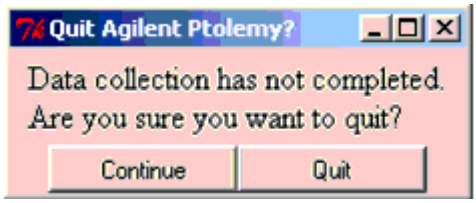
When a design that contains components from the interactive controls and displays library is simulated, a control panel window opens (Control Panel Window). The control panel has two buttons: *Pause* and *Quit*. Depending on what components from this library are used in the design, the control panel displays additional buttons, sliders, or text labels followed by numeric values that update as the simulation runs. Clicking **Pause** pauses the simulation and relabels the *Pause* button as *Continue*. Clicking **Continue** resumes the simulation. Clicking **Quit** stops the simulation.

If the simulated design has non-interactive sink components and the *Quit* button is clicked before the components have collected the data needed, the window shown in Quit Window displays a warning that the data collection is not complete. The designer can resume the simulation by clicking **Continue** or terminate by clicking the *Quit button.

Control Panel Window



Quit Window

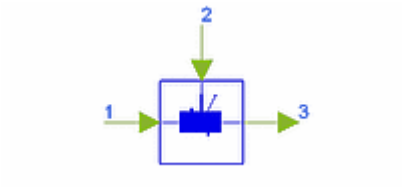


Interactive controls and displays components do not work in sweep simulations; these components are deactivated automatically before simulation starts. Since most interactive controls and displays components do not have output ports, deactivating these does not affect the remainder of the simulation. The components that have outputs behave as explained here.

- *TkSlider* (controls-displays) - The value at the output port is constant and equal to the value of the Value parameter; no slider is created in the control panel that allows interactively changing the output value.
- *TkButtons* (controls-displays) - The value at the output ports is 0; no buttons are created in the control panel that allow interactively changing the output value.
- *LMS_TkPlot* (controls-displays), *LMS_CxTkPlot* (controls-displays) - In the non-sweep mode these components filter the input signal using a set of adaptively varying coefficients; a bar graph display is created that shows how the filter coefficients vary throughout the simulation. In the sweep mode, these components still filter the input signal using a set of adaptively varying coefficients; the bar graph display does not display.

For more information about using interactive controls and displays components refer to *Interactive Controls and Displays for ADS Ptolemy Simulation* (ptolemy) in the *ADS Ptolemy Simulation* (ptolemy) documentation.

LMS_CxTkPlot



Description: Interactive Complex LMS Adaptive Filter
Library: Interactive Controls and Displays
Class: SDFLMS_CxTkPlot
Derived From: LMS_Cx
C++ Code: See *doc/sp_items/SDFLMS_CxTkPlot.html* under your installation directory.

Parameters

Name	Description	Default	Type	Range
Taps	filter tap values	(-.040609,0.0) (-.001628,0.0) (.17853,0.0) (.37665,0.0)(.37665,0.0) (.17853,0.0) (-.001628,0.0) (-.040609,0.0)	complex array	
Decimation	decimation ratio	1	int	[1, ∞)
DecimationPhase	decimation phase	0	int	[0, Decimation-1]
StepSize	adaptation step size	0.01	real	(0, ∞)
ErrorDelay	update loop delay	1	int	[1, ∞)
SaveTapsFile	filename in which to save final tap values		string	
StepSizeLow	low end of step size scale on interactive display	0.0	real	
StepSizeHigh	high end of step size scale on interactive display	0.1	real	
FullScale	full scale on tap display	1.0	real	
Geometry	location and size of window	+500+000	string	
Width	bar chart display width, in centimeters	10.0	real	
Height	bar chart display height, in centimeters	5.0	real	
Identifier	run-time display identifier	LMS_Cx filter taps: Real (red) & Imag (blue)	string	
UpdateInterval	number of invocations between display updates	10	int	

Pin Inputs

Pin	Name	Description	Signal Type
1	signalIn		complex
2	error		complex

Pin Outputs

Pin	Name	Description	Signal Type
3	signalOut		complex

Notes/Equations

1. LMS_CxTkPlot implements an adaptive filter using the least-mean square algorithm. The size of the LMS filter is determined by the number of coefficients in the Taps parameter; the default gives an 8th-order, linear phase lowpass filter. LMS supports decimation, but not interpolation.

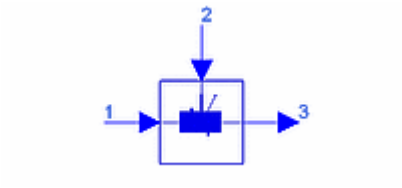
The filter coefficients can be specified directly or read from a file. To load filter coefficients from a file, replace the default coefficients with the string <*filename*>. Use an absolute path name for the filename to allow the filter to work as expected regardless of the directory where the simulation process actually runs.

2. When used correctly, this LMS adaptive filter adapts to try to minimize the mean-squared error of the signal at its error input [1]. The output of the filter should be compared to (subtracted from) some reference signal to produce an error signal. That error signal should be fed back to the error input. The ErrorDelay parameter must equal the total number of delays in the path from the output of the filter back to the error input. This action ensures correct alignment of the adaptation algorithm. The number of delays must be greater than 0 or the simulation deadlocks.
   The adaptation algorithm is the well-known LMS, or stochastic-gradient algorithm.
3. If the SaveTapsFile string is non-null, a file is created with the name given by that string, and the final tap values are stored there after the run has completed.
4. When the Decimation ratio is ≥ 1, the filter behaves exactly as if it were followed by a DownSample component.
5. The DecimationPhase parameter is equivalent to the Phase parameter of the DownSample component. When decimating, samples are conceptually discarded. For example, to decimate by a factor of 3, one of every 3 outputs is selected. The DecimationPhase parameter determines which of these is selected. If DecimationPhase is 0 (default), the most recent samples are selected.
6. See also: *FIR* (numeric) and *LMS* (numeric).
7. For general information regarding the control and display of simulation data, refer to the *About Interactive Controls and Displays* (controls-displays).

References

1. S. Haykin, *Adaptive Filter Theory*, Prentice Hall: Englewood Cliffs, NJ. 1991. 2nd ed.

LMS_TkPlot



Description: Interactive LMS Adaptive Filter
Library: Interactive Controls and Displays
Class: SDFLMS_TkPlot
Derived From: LMS
C++ Code: See *doc/sp_items/SDFLMS_TkPlot.html* under your installation directory.

Parameters

Name	Description	Default	Type	Range
Taps	filter tap values	-.040609 -.001628 .17853 .37665 .37665 .17853 -.001628 -.040609	real array	
Decimation	decimation ratio	1	int	[1, ∞)
DecimationPhase	decimation phase	0	int	[0, Decimation-1]
StepSize	adaptation step size	0.01	real	(0, ∞)
ErrorDelay	update loop delay	1	int	[1, ∞)
SaveTapsFile	filename in which to save final tap values		string	
StepSizeLow	low end of step size scale on interactive display	0.0	real	
StepSizeHigh	high end of step size scale on interactive display	0.1	real	
FullScale	full scale on tap display	1.0	real	
Geometry	location and size of window	+500+000	string	
Width	bar chart display width, in centimeters	10.0	real	
Height	bar chart display height, in centimeters	5.0	real	
Identifier	run-time display identifier	LMS filter taps	string	
UpdateInterval	number of invocations between display updates	10	int	

Pin Inputs

Pin	Name	Description	Signal Type
1	signalIn		real
2	error		real

Pin Inputs

Pin	Name	Description	Signal Type
3	signalOut		real

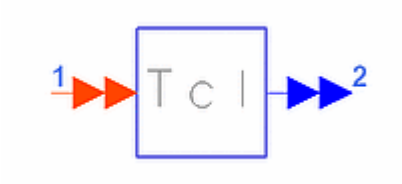
Notes/Equations

1. LMS_TkPlot is an adaptive filter using the least-mean square algorithm. The initial filter coefficients are given by the Taps parameter. The default initial coefficients give an 8th-order, linear phase lowpass filter. To read initial coefficients from a file, replace the default coefficients with <*fileName*>, preferably specifying a complete path. LMS supports decimation, but not interpolation.
2. When used correctly, this LMS adaptive filter adapts to try to minimize the mean-squared error of the signal at its error input [1]. The output of the filter should be compared to (subtracted from) some reference signal to produce an error signal. That error signal should be fed back to the error input. The ErrorDelay parameter must equal the total number of delays in the path from the output of the filter back to the error input. This action ensures correct alignment of the adaptation algorithm. The number of delays must be greater than 0 or the simulation deadlocks.
   The adaptation algorithm is the well-known LMS, or stochastic-gradient, algorithm.
3. If the SaveTapsFile string is non-null, a file is created with the name given by that string, and the final tap values are stored there after the run has completed.
4. When the Decimation ratio is ≥ 1, the filter behaves exactly as if it were followed by a DownSample component.
5. The DecimationPhase parameter is equivalent to the Phase parameter of the DownSample component. When decimating, samples are conceptually discarded. For example, to decimate by a factor of 3, one of every 3 outputs is selected. The DecimationPhase parameter determines which of these is selected. If DecimationPhase is 0 (default), the most recent samples are selected.
6. See also: *FIR* (numeric).
7. For general information regarding the control and display of simulation data, refer to the *About Interactive Controls and Displays* (controls-displays).

References

1. S. Haykin, *Adaptive Filter Theory*, Prentice Hall: Englewood Cliffs, NJ. 1991. 2nd ed.

TclScript



Description: Invoke Tcl Script
Library: Interactive Controls and Displays
Class: SDFTclScript
C++ Code: See *doc/sp_items/SDFTclScript.html* under your installation directory.

Parameters

Name	Description	Default	Type
TclFile	file from which to read the tcl script	$HPTOLEMY/src/controls-displays/tcltk/stars/tkScript.tcl	filename

Pin Inputs

Pin	Name	Description	Signal Type
1	input	Any number of inputs to feed to Tcl	multiple anytype

Pin Outputs

Pin	Name	Description	Signal Type
2	output	Any outputs obtained from Tcl	multiple real

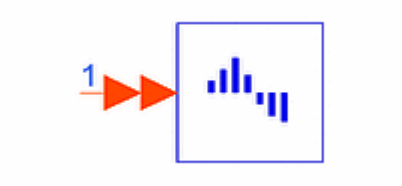
Notes/Equations

1. TclScript reads a file containing Tcl commands. The component can be used in a variety of ways, including using Tk to animate or control a simulation. A number of procedures and global variables are defined for use by the Tcl script by the time the script is sourced. These enable the script to read the inputs to the component or set output values. Optionally, the Tcl script can define a procedure to be called by ADS Ptolemy for every simulation of the component.
2. Much of the complexity in using TclScript is due to the need to use unique names for each instance of TclScript. These unique names are constructed using a unique string defined by TclScript. That string is made available to the Tcl script in the form of a global Tcl variable *$starID*.
   The procedure used by the Tcl script to set output values is *setOutputs_$starID*, while the procedure used to read input values is *grabInputs_$starID*. The *setOutputs* procedure takes as many arguments as there are output ports. The *grabInputs* procedure returns a string with as many values as there are inputs, separated by spaces. The Tcl script is sourced during startup, so you do not need to read inputs at that time. However, you can set (initialize) output values.
   Optionally, the Tcl script can define a Tcl procedure called *goTcl_$starID*. If this

procedure is defined in the script, the procedure is invoked every time the component simulates. The procedure takes one argument, the *starID*, and returns no values. If the *goTcl* procedure is defined, the communication with Tcl is said to be synchronous; that is, it is synchronized to the simulation of the component. Otherwise, the communication is asynchronous; that is, the Tcl script is responsible for setting up procedures that interact with the component only when Tcl invokes them.

3. For asynchronous operation, typically X events are bound to Tcl/Tk commands that read or write data to the component. These Tcl commands use *grabInputs_$starID*, which returns a list containing the current value of each of the inputs, and *setOutputs_$starID*, which sets the value of the outputs. The argument list for *setOutputs_$starID* should contain a floating-point (real) value for each output of the component. Inputs can be of any type. The *print()* method of the particle is used to construct a string passed to Tcl. This mechanism is entirely asynchronous, in that the Tcl/Tk script decides when these actions should be performed on the basis of X events.
4. For synchronous operation, the Tcl procedure *goTcl_$starID* is called by the component every time the component simulates. The procedure could, for example, grab input values and calculate output values, although it can do anything you want, even ignoring the input and output values.
5. An example Tcl script is located at: *$HPEESOF_DIR/adsptolemy/src/controls-displays/tcltk/stars/tkScript.tcl*
6. For general information regarding the control and display of simulation data, refer to *About Interactive Controls and Displays* (controls-displays).

TkBarGraph



Description: Bar Graph Display
Library: Interactive Controls and Displays
Class: SDFTkBarGraph
C++ Code: See *doc/sp_items/SDFTkBarGraph.html* under your installation directory.

Parameters

Name	Description	Default	Type
Label	Bar graph title	bar chart display	string
Top	Y-axis upper limit	1.0	real
Bottom	Y-axis lower limit	-1.0	real
NumberOfBars	Number of bars to display	16	int
BarGraphHeight	Bar graph height, in centimeters	5	real
BarGraphWidth	Bar graph width, in centimeters	10	real
Position	Bar graph window position offset from upper left corner, in pixels	+0+0	string
UpdateSize	Number of bars drawn per graph update	1	int

Pin Input

Pin	Name	Description	Signal Type
1	input	Any number of inputs to plot	multiple anytype

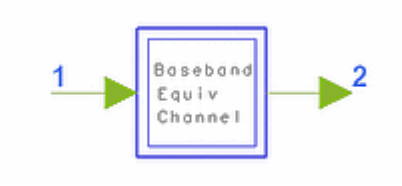
Notes/Equations

1. TkBarGraph dynamically displays the value of any number of input signals in bar-graph form. The first 12 input signals are assigned distinct colors; after that, colors are repeated. Colors can be controlled using X resources.
2. The UpdateSize parameter controls the number of bars drawn on the graph per graph update.
   The update occurs as described next; this description refers to one input but applies to each input if more than one signal is connected to the TkBarGraph multi-input port.
   If the total number of samples read is less than NumberOfBars, then the most recent UpdateSize samples are used to draw new bars in the graph. The first time the total number of samples read exceeds NumberOfBars, for example by N ($1 \le N \le$ UpdateSize), the oldest N bars are deleted before new UpdateSize bars are drawn, so that the total number of bars displayed is equal to NumberOfBars. After this point, each time UpdateSize new samples are read, the oldest UpdateSize bars are deleted

from the graph and the most recent UpdateSize samples read are used to draw new bars in the graph.

3. For general information regarding the control and display of simulation data, refer to *About Interactive Controls and Displays* (controls-displays).

TkBasebandEquivChannel



Description: Baseband Equivalent Channel
Library: Interactive Controls and Displays
Class: SDFTkBasebandEquivChannel

Parameters

Name	Description	Default	Type
SymbolRateHz	symbol rate through the channel,in Hertz	600	real
PhaseJitterFrequencyHz	frequency of phase jitter distortion to add the signal,in Hertz	60	real
LinearDistortionTaps	taps value of FIR filter	(1.0, 0.0)	complex array

Pin Inputs

Pin	Name	Description	Signal Type
1	input		complex

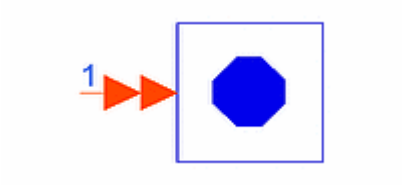
Pin Outputs

Pin	Name	Description	Signal Type
2	output	input signal plus directions	complex

Notes/Equations

1. TkBasebandEquivChannel is a subnetwork that models a baseband equivalent channel with linear distortion, frequency offset, phase jitter, and additive white Gaussian noise. Many of the channel model parameters can be set dynamically by the designer.
2. To model linear distortion, such as intersymbol interference, the input signal is passed through a complex FIR filter with the taps set by LinearDistortionTaps. The frequency offset distortion is set by the Freq. Offset slider control. Similarly, the phase jitter amplitude (peak-to-peak, in degrees) is set by the Phase Jitter slider control while the phase jitter frequency is set by the PhaseJitterFrequencyHz parameter. The phase of both the frequency offset and the phase jitter can be reset with the Reset Phase control button. The amplitude of the added complex white Gaussian noise is set by the Noise Power slider control.
3. See also: *FIR* (numeric), *FreqPhase* (numeric), *AWGN_Channel* (numeric), *TelephoneChannel* (numeric).

TkBreakPt



Description: Conditional Breakpoint
Library: Interactive Controls and Displays
Class: SDFTkBreakPt
Derived From: TclScript
C++ Code: See *doc/sp_items/SDFTkBreakPt.html* under your installation directory.

Parameters

Name	Description	Default	Type
Condition	Condition on which to pause the run	$input(1) < 0	string
OptionalAlternateScript	TclTk commands to run instead of default behavior		string

Pin Inputs

Pin	Name	Description	Signal Type
1	input	Any number of inputs to feed to Tcl	multiple anytype

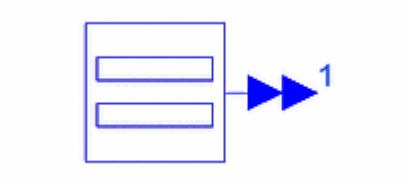
Notes/Equations

1. TkBreakPt evaluates a Tcl expression each time the component is simulated. If the expression returns a value of true, then the run is paused. The expression can be any valid Tcl expression. For details of valid Tcl expressions, refer to the documentation for the Tcl command expr.
   TkBreakPt takes multiple inputs. To write a conditional expression easily, the inputs are numbered: input(1), input(2), input(3), and so on. Therefore, if only one input is connected to a TkBreakPt, use input(1) to refer to that input.
   Because Tcl uses a dollar sign ($) to reference the value of a variable, the expression
   `$input(1) < 0`
   is true if the value of the first input is negative. Similarly, the expression
   `$input(1) < $input(2)`
   is true whenever the value of the first input is less than the value of the second input.
   It can be difficult to distinguish the inputs if these are all connected directly to the multiport input. One solution is to use the BusMerge component. Connect the bus merge component of your choice to the input of TkBreakPt. Then connect the inputs to the bus merge component. The top input is input(1), the next is input(2), and so on.
   The OptionalAlternateScript parameter is the script to source if the condition is true.

If this parameter is blank, then the default script is executed, which pauses the run and displays a message in the Control panel. This should be fine for most applications of TkBreakPt.

2. For general information regarding the control and display of simulation data, refer to *About Interactive Controls and Displays* (controls-displays).

TkButtons



Description: Interactive Buttons
Library: Interactive Controls and Displays
Class: SDFTkButtons
Derived From: TclScript
C++ Code: See *doc/sp_items/SDFTkButtons.html* under your installation directory.

Parameters

Name	Description	Default	Type
Label	buttons label	Buttons you can push:	string
Identifiers	list of strings that identify each button; use a space between strings to create multiple buttons	BUTTON1	string array
Value	value produced when a button is pushed	1.0	real
SimultaneousEvents	produce simultaneous output values: do not allow, allow	do not allow	enum
ButtonControl	return to simulation when button is pushed: Not synchronous, Synchronous	Not synchronous	enum
PutInControlPanel	put slide in control panel (versus its own window): NO, YES	YES	enum
OutputType	Pushes produce impulses or levels: Impulse, Level	Impulse	enum

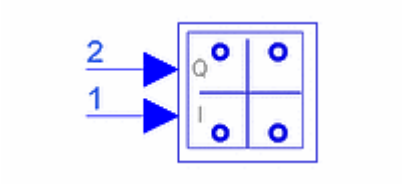
Pin Output

Pin	Name	Description	Signal Type
1	output	Any outputs obtained from Tcl	multiple real

Notes/Equations

1. TkButtons outputs 0.0 value on all outputs unless the corresponding button is pushed. When the button is pushed, the output takes the value given by the parameter Value. If ButtonControl is Synchronous, outputs are produced only when some button is pushed-the component waits for a button to be pushed before simulation proceeds. If SimultaneousEvents is allowed, button pushes are registered only when PUSH TO PRODUCE OUTPUTS is pushed. If synchronous is no, TkButtons is nondeterminate.
2. For general information regarding the control and display of simulation data, refer to *About Interactive Controls and Displays* (controls-displays).

TkConstellation



Description: Tk IQ Constellation
Library: Interactive Controls and Displays
Class: SDFTkConstellation

Parameters

Name	Description	Default	Type
Label	Plot title	Tk Constellation	string
NumSamplesPerSymbol	Number of samples per symbol	10	int
Amplitude	Y-axis amplitude	2	real
SampleDelay	Input delay before sampling	2	int
Persistence	Number of points displayed on the plot at any one time	200	int
UpdateSize	Number of new points plotted per plot update	10	int
Style	Plot style: dot, connect	connect	enum

Pin Inputs

Pin	Name	Description	Signal Type
1	X	X input	real
2	Y	Y input	real

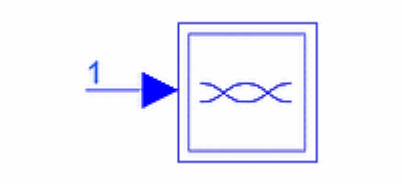
Notes/Equations

1. The TkConstellation subcircuit displays an IQ constellation of your data. Set NumSamplesPerSymbol to the number of samples per symbol. The X and Y axes of the graph are drawn from +Amplitude to −Amplitude.
2. By setting the Style parameter to connect, your constellation shows the trajectories. Setting the Style parameter to dot displays the constellation as dots. Dot style constellations must be sampled at the appropriate delay. When simulated, a TkSlider component appears to control this delay. After you determine a good sampling delay, set the SampleDelay parameter to remember it.
3. The Persistence parameter controls how many points are displayed on the constellation plot at any one time. Only the most recent Persistent number of points are plotted.
4. The UpdateSize parameter controls the number of points drawn on the plot per plot update. The update occurs as described next.
   If the total number of samples read (for each input) is less than Persistence, then the most recent UpdateSize samples from each input are used to draw new points in the

plot. The first time the total number of samples read exceeds Persistence, for example by N ($1 \leq N \leq$ UpdateSize), the oldest N points are deleted before the new UpdateSize points are drawn, so that the total number of points displayed is equal to Persistence. After this point, every time UpdateSize new samples are read (from each of the two inputs), the oldest UpdateSize points are deleted from the plot and the most recent UpdateSize samples read are used to draw new points in the plot.

5. For general information regarding the control and display of simulation data, refer to *About Interactive Controls and Displays* (controls-displays).

TkEye



Description: Tk Eye
Library: Interactive Controls and Displays
Class: SDFTkEye

Parameters

Name	Description	Default	Type
Label	Plot title	Tk Eye	string
NumSamplesPerSymbol	Number of samples per symbol	10	real
NumSymbols	Number of symbols on X axis	2	real
Amplitude	Y axis amplitude	2	real
Persistence	Number of points displayed on the plot at any one time	200	real
UpdateSize	Number of new points plotted per plot update	10	real
Style	Plot style: dot, connect	connect	enum

Pin Input

Pin	Name	Description	Signal Type
1	X	input signal	real

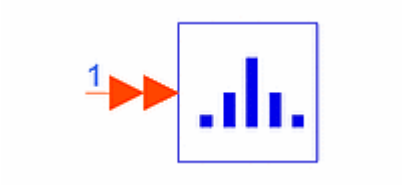
Notes/Equations

1. The TkEye subcircuit displays an eye diagram of your data. Set NumSamplesPerSymbol to the number of samples per symbol, and NumSymbols to the number of symbols, or eyes, in the resultant diagram. The Y axis of the graph is drawn from +Amplitude to −Amplitude.
2. When simulated, a TkSlider component appears to add a variable sample delay to the eye diagram. Using the slider, you can center the eye on your graph.
3. The Persistence parameter controls how many points are displayed on the eye diagram plot at any one time. Only the most recent Persistent number of points are plotted. Using a higher Persistence results in more signal traces being overlaid on top of each other, giving a more accurate eye diagram.
4. The UpdateSize parameter controls the number of points drawn on the plot per plot update. The update happens as described next.
   If the total number of samples read is less than Persistence, then the most recent UpdateSize samples are used to draw new points in the plot. The first time the total number of samples read exceeds Persistence, for example by N ($1 \le N \le$ UpdateSize), the oldest N points are deleted before the new UpdateSize points are drawn, so that the total number of points displayed is equal to Persistence. After this

point, every time UpdateSize new samples are read, the oldest UpdateSize points are deleted from the plot and the most recent UpdateSize samples read are used to draw new points in the plot.

5. For general information regarding the control and display of simulation data, refer to *About Interactive Controls and Displays* (controls-displays).

TkHistogram



Description: Display histogram of inputs
Library: Interactive Controls and Displays
Class: SDFTkHistogram
Derived From: TkBarGraph
C++ Code: See *doc/sp_items/SDFTkHistogram.html* under your installation directory.

Parameters

Name	Description	Default	Type	Range
Label	Histogram title	Histogram Display	string	
Top	X-axis upper limit	1.0	real	
Bottom	X-axis lower limit	-1.0	real	
NumberOfBars	Number of buckets	16	int	
BarGraphHeight	Bar graph height, in centimeters	5	real	
BarGraphWidth	Bar graph width, in centimeters	10	real	
Position	Bar graph window position offset from upper left corner, in pixels	+0+0	string	
UpdateSize	Number of input samples read per graph update	1	int	
DataPoints	Number of data points to analyze, 0 for all	100	int	[0, ∞)

Pin Input

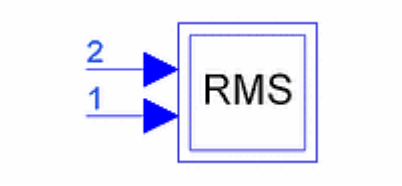
Pin	Name	Description	Signal Type
1	input	Inputs to make histograms of	multiple anytype

Notes/Equations/

1. TkHistogram dynamically displays histograms of any number of inputs. NumberOfBars controls the number of bars; Top and Bottom control the ends of the histogram display. Data values less than Bottom and greater than Top are not plotted.
2. TkHistogram gathers statistics on the last DataPoints values as if it were a sliding histogram. When DataPoints is set to 0, the model gathers statistics on all data.
3. The UpdateSize parameter controls how often to update the histogram. Setting the parameter to a higher value makes the model run faster, but histogram updates won't be as smooth.
4. For general information regarding the control and display of simulation data, refer to

About Interactive Controls and Displays (controls-displays).

TkIQrms



Description: Display RMS Value of Input IQ Signal
Library: Interactive Controls and Displays
Class: SDFTkIQrms

Parameters

Name	Description	Default	Type
Label	label to put on the display	input signal rms value	string

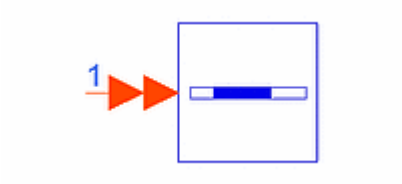
Pin Inputs

Pin	Name	Description	Signal Type
1	I_in	I phase input	real
2	Q_in	Q phase input	real

Notes/Equations

1. This component displays the rms value of the IQ signal connected at its input. The value displayed is the rms value over the period starting at sample 0 and ending at the current sample.
2. For general information regarding the control and display of simulation data, refer to *About Interactive Controls and Displays* (controls-displays).

TkMeter



Description: Bar Meters Display
Library: Interactive Controls and Displays
Class" SDFTkMeter
Derived From: TclScript
C++ Code: See *doc/sp_items/SDFTkMeter.html* under your installation directory.

Parameters

Name	Description	Default	Type	Range
Label	display label	sliding scale display	string	
Low	low end of scale for displayed bars	-1.0	string	(-∞, ∞)
High	high end of scale	1.0	string	(Low, ∞)
PutInControlPanel	put bars in control panel (versus their own window): NO, YES	YES	enum	

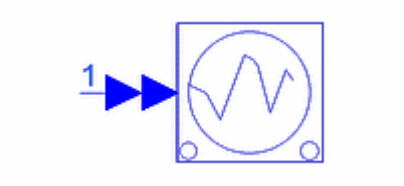
Pin Inputs

Pin	Name	Description	Signal Type
1	input	Any number of inputs to feed to Tcl	multiple anytype

Notes/Equations

1. TkMeter dynamically displays the value of any number of input signals on a set of bar meters.
2. For general information regarding the control and display of simulation data, refer to *About Interactive Controls and Displays* (controls-displays).

TkPlot



Description: Plot Inputs versus Time
Library: Interactive Controls and Displays
Class: SDFTkPlot
Derived From: TkXYPlot
C++ Code: See *doc/sp_items/SDFTkPlot.html* under your installation directory.

Parameters

Name	Description	Default	Type
Label	Plot title		string
Geometry	Window size and location (width x height + horizontal offset + vertical offset), in pixels	720x400+0+0	string
xTitle	X-axis label	n	string
yTitle	Y-axis label	y	string
xRange	X-axis values range	0 100	real array
yRange	Y-axis values range	-1.5 1.5	real array
Persistence	Number of points displayed on the plot at any one time	100	int
Style	Plot style: dot, connect	connect	enum
UpdateSize	Number of new points plotted per plot update	10	int
RepeatBorderPoints	Repeat rightmost border point on left border: NO, YES	YES	enum

Pin Inputs

Pin	Name	Description	Signal Type
1	Y	Vertical coordinate	multiple real

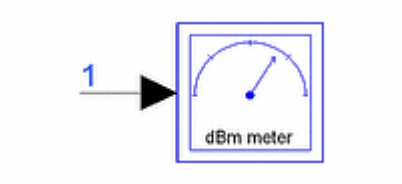
Notes/Equations

1. Plot Y input(s) versus sample number with dynamic updating. Two styles are available: dot causes individual points to be plotted; connect causes connected lines to be plotted. Drawing a box in the plot resets the plot area to that outlined by the box. Buttons allow for zooming in and out, and for resizing the box to just fit the data in view.
2. The Persistence parameter controls how many points are displayed on the plot at any one time. Only the most recent Persistent number of points are plotted.
3. The UpdateSize parameter controls the number of points drawn on the plot per plot update. The update occurs as described next.

If the total number of samples read is less than Persistence, then the most recent UpdateSize samples are used to draw new points in the plot. The first time the total number of samples read exceeds Persistence, for example by N ($1 \leq N \leq$ UpdateSize), the oldest N points are deleted before the new UpdateSize points are drawn, so that the total number of points displayed is equal to Persistence. After this point, every time UpdateSize new samples are read, the oldest UpdateSize points are deleted from the plot and the most recent UpdateSize samples read are used to draw new points in the plot.

4. For general information regarding the control and display of simulation data, refer to *About Interactive Controls and Displays* (controls-displays).

TkPower



Description: Display Signal Power in dBm
Library: Interactive Controls and Displays
Class: TSDFTkPower

Parameters

Name	Description	Default	Unit	Type	Range
Label	label to put on the display	Power in dBm		string	
SignalType	input signal type: Baseband, RF	RF		enum	
RefR	reference resistance	50	Ohm	real	(0, ∞)
StartUpDelay	number of initial input samples to be ignored	0		int	[0, ∞)
IntegrationSamples	period in number of samples for integrator reset signal (0 never resets the integrator)	0		int	[0, ∞)

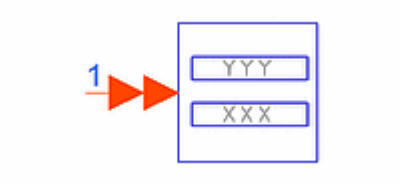
Pin Input

Pin	Name	Description	Signal Type
1	input	input signal	timed

Notes/Equations

1. This component accepts a timed signal at its input and displays its average power in dBm.
   - If IntegrationSamples = 0, then the value displayed is the average power over the time period starting at time zero and ending at the current time.
   - If IntegrationSamples > 0, then the integrator used to calculate the average power is reset periodically every IntegrationSamples samples.
     If the resulting average power value is less than −300 dBm, TkPower displays −300 dBm as the average power.
2. The StartUpDelay parameter can be used to exclude an initial number of input samples from the average power calculation. This can be useful when trying to exclude transients or other initial signal delays. For the initial period of StartUpDelay samples the value displayed by TkPower is −300 dBm.
3. If RefR < 0 or = 0, it is reset to 50 Ohms.
   If StartUpDelay < 0, it is reset to 0.
   If IntegrationSamples < 0, it is reset to 0.
4. For general information regarding the control and display of simulation data, refer to *About Interactive Controls and Displays* (controls-displays).

TkShowBooleans



Description: Booleans Display
Library: Interactive Controls and Displays
Class: SDFTkShowBooleans
Derived From: TclScript
C++ Code: See *doc/sp_items/SDFTkShowBooleans.html* under your installation directory.

Parameters

Name	Description	Default	Type
Label	title for set of Boolean displays	Inputs to TkShowBooleans:	string
Identifiers	identifier for each input for display	BOOL	string array
PutInControlPanel	put bars in control panel (versus their own window): NO, YES	YES	enum

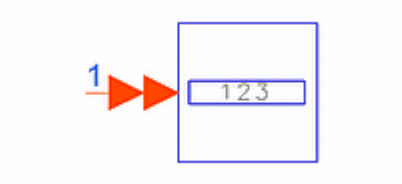
Pin Inputs

Pin	Name	Description	Signal Type
1	input	Any number of inputs to feed to Tcl	multiple anytype

Notes/Equations

1. TkShowBooleans displays input Booleans using color to highlight their values.
2. Zero is false; non-zero is true.
3. For general information regarding the control and display of simulation data, refer to *About Interactive Controls and Displays* (controls-displays).

TkShowValues



Description: Input Values Display
Library: Interactive Controls and Displays
Class: SDFTkShowValues
Derived From: TclScript
C++ Code: See *doc/sp_items/SDFTkShowValues.html* under your installation directory.

Parameters

Name	Description	Default	Type
Label	display label	Inputs to TkShowValues:	string
PutInControlPanel	put bars in control panel (versus their own window): NO, YES	YES	enum

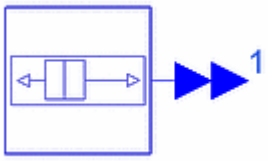
Pin Inputs

Pin	Name	Description	Signal Type
1	input	Any number of inputs to feed to Tcl	multiple anytype

Notes/Equations

1. TkShowValues displays the input values in textual form. The print() method of the input particles is used, so any data type can be handled, although the space allocated on the screen may need to be adjusted. The width of the display window is appropriate for integer, floating-point (real), and complex particles.
2. For general information regarding the control and display of simulation data, refer to *About Interactive Controls and Displays* (controls-displays).

TkSlider



Description: Interactive Slider
Library: Interactive Controls and Displays
Class: SDFTkSlider
Derived From: TclScript
C++ Code: See *doc/sp_items/SDFTkSlider.html* under your installation directory.

Parameters

Name	Description	Default	Type
Low	low end of interactive scale	0.0	real
High	high end of interactive scale	1.0	real
Value	initial value to send to output	0.0	real
Identifier	identifier for control panel slider	Scale	string
PutInControlPanel	put bars in control panel (versus their own window): NO, YES	YES	enum
Granularity	number of intervals in slider	100	int

Pin Outputs

Pin	Name	Description	Signal Type
1	output	Any outputs obtained from Tcl	multiple real

Notes/Equations

1. TkSlider outputs a value determined by an interactive on-screen scale slider.
2. For general information regarding the control and display of simulation data, refer to *About Interactive Controls and Displays* (controls-displays).

TkText

Description: Display History of Input Values
Library: Interactive Controls and Displays
Class: SDFTkText
Derived From: TkShowValues
C++ Code: See *doc/sp_items/SDFTkText.html* under your installation directory.

Parameters

Name	Description	Default	Type
Label	display label	Inputs to TkText:	string
NumberOfPastValues	number of past values to save	100	int

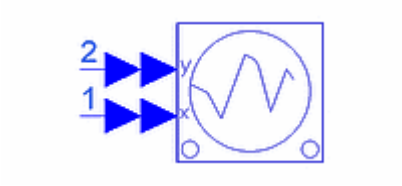
Pin Inputs

Pin	Name	Description	Signal Type
1	input	Any number of inputs to feed to Tcl	multiple anytype

Notes/Equations

1. TkText displays input values in a separate window and keeps a specified number of past values. The print() method of the input particles is used, so any data type can be handled. The default width of the display window is appropriate for integer, floating-point (real), and complex particles.
2. For general information regarding the control and display of simulation data, refer to *About Interactive Controls and Displays* (controls-displays).

TkXYPlot



Description: Plot Y versus X Inputs
Library: Interactive Controls and Displays
Class: SDFTkXYPlot
C++ Code: See *doc/sp_items/SDFTkXYPlot.html* under your installation directory.

Parameters

Name	Description	Default	Type
Label	Plot title		string
Geometry	Window size and location (width x height + horizontal offset + vertical offset), in pixels	720x400+0+0	string
xTitle	X-axis label	x	string
yTitle	Y-axis label	y	string
xRange	X-axis values range	-1.5 1.5	real array
yRange	Y-axis values range	-1.5 1.5	real array
Persistence	Number of points displayed on the plot at any one time	100	int
Style	Plot style: dot, connect	dot	enum
UpdateSize	Number of new points plotted per plot update	10	int

Pin Inputs

Pin	Name	Description	Signal Type
1	X	Horizontal coordinate	multiple real
2	Y	Vertical coordinate	multiple real

Notes/Equations

1. Plot Y input(s) versus X input(s) with dynamic updating. Two styles are available: dot causes points to be plotted; connect causes connected lines to be plotted. Drawing a box in the plot resets the plot area to that outlined by the box. Buttons allow for zooming in and out, and for resizing the box to just fit the data in view.
2. The Persistence parameter controls how many points are displayed on the plot at any one time. Only the most recent Persistent number of points are plotted.
3. The UpdateSize parameter controls the number of points drawn on a plot per plot update. The update occurs as described next.
   If the total number of samples read (for each input) is less than Persistence, then the most recent UpdateSize samples from each input are used to draw new points in the

plot. The first time the total number of samples read exceeds Persistence, for example by N ($1 \leq N \leq$ UpdateSize), the oldest N points are deleted before the new UpdateSize points are drawn, so that the total number of points displayed is equal to Persistence. After this point, every time UpdateSize new samples are read (from each of the two inputs), the oldest UpdateSize points are deleted from the plot and the most recent UpdateSize samples read are used to draw new points in the plot.

4. For general information regarding the control and display of simulation data, refer to *About Interactive Controls and Displays* (controls-displays).