

Avant!

**AvanWaves Command System
Application Note**

**1999.0
January 1999**

©1999 Avant! Corporation. All rights reserved.
Star-RC Reference Manual, Version 1997.3 Jan, 1998 for software Release 1997.3.
Not previously printed.

Star-RC software 1997.4 Copyright © 1997 by Avant! Corporation and Avant! subsidiary. All rights reserved. Unpublished-
rights reserved under the copyright laws of the United States.

Use of copyright notices is precautionary and does not imply publication or disclosure.

Disclaimer

AVANT! CORPORATION RESERVES THE RIGHT TO MAKE CHANGES WITHOUT FURTHER NOTICE TO ANY PRODUCTS DESCRIBED HEREIN. AVANT! CORPORATION MAKES NO WARRANTY, REPRESENTATION, OR GUARANTEE REGARDING THE SUITABILITY OF ITS PRODUCTS FOR ANY PARTICULAR PURPOSE, NOR DOES AVANT! CORPORATION ASSUME ANY LIABILITY ARISING OUT OF THE APPLICATION OR USE OF ANY PRODUCT, AND SPECIFICALLY DISCLAIMS ANY AND ALL LIABILITY, INCLUDING WITHOUT LIMITATION, CONSEQUENTIAL OR INCIDENTAL DAMAGES.

Proprietary Rights Notice

This document contains information of a proprietary nature. No part of this manual may be copied or distributed without the prior written consent of Avant! corporation. This document and the software described herein is only provided under a written license agreement or a type of written non-disclosure agreement with Avant! corporation or its subsidiaries. ALL INFORMATION CONTAINED HEREIN SHALL BE KEPT IN CONFIDENCE AND USED STRICTLY IN ACCORDANCE WITH THE TERMS OF THE WRITTEN NON-DISCLOSURE AGREEMENT OR WRITTEN LICENSE AGREEMENT WITH AVANT! CORPORATION OR ITS SUBSIDIARIES.

Trademark/Service-Mark Notice

ADM, Apollo, Apollo-GA, Apollo, Apollo-BV, Apollo-DP, Apollo-GA, Apollo-XO, ArcCell, ArcChip, ArcUtil, ATEM, Avan Testchip, AvanWaves, BaseLine, Baseline Software Accelerator, Cyclelink, Device Model Builder, DriveLine, Dynamic Model Switcher, EVaccess, Explorer, Hercules, HSPICE, HSPICE-Link, LTL, Mars-Rail, Master Toolbox, Milkyway, Planet, PlanetPL, PlanetRTL, Polaris, Polaris-CBS, Polaris-MT, Pure-Speed, SimLine, Sirius, Smart Extraction, Solar, SolarII, Star-DC, Star-Hspice, Star-HspiceLink, Star-MTB, Star-Power, Star-RC, Star-Sim, Star-Time, VeriCheck, VeriView, ΨChips, ΨCrunch, and ΨTime are trademarks of Avant! Corporation and its subsidiaries.

Avant! Corporation, Avant! logo, and AvanLabs are trademarks and service-marks of Avant! Corporation.

All other trademarks are the property of their respective owners.

Subsidiaries

ArcSys, Inc., ISS, Inc., Anagram, Inc., Meta-Software, Inc., Frontline Design Automation, Inc., and NexSyn, Inc. are subsidiaries of Avant! Corporation.

Contacting Avant! Corporation

Contact Avant! Corporation by:

Telephone:(510) 413-8000

FAX:(510) 413-8080

Avant! Corporation
46871 Bayside Parkway
Fremont, CA 94538



AvanWaves Command System

This application note describes the Avanwaves command system that lets you run AvanWaves in batch mode.

You can write a command input file using the command system. Avanwaves reads, interprets, and executes each command in the command input file.

You can also type and execute the commands interactively in the AvanWaves command user interface. To invoke the Awaves command user interface, enter on the command line:

```
awaves -i
```

or, select **Tools >Awaves Commands** after awaves starts up.

The .cfg file generated by AvanWaves gives you an example usage of the AvanWaves command system.

Lexical Rules

The rules are expressed using regular expressions. All literal "+", "-", "." are escaped using "\".

Variable: [a-zA-Z][a-zA-Z0-9]*

Int_Const: [0-9]+

Char_Const: Any double quoted string that contains a number of characters except double quote new line

Float_Const: [0-9]*\.[0-9]+
 [0-9]*\.[0-9]+e[\+|-][0-9]+
 [0-9]+e[0-9]+
 [0-9]+e[\+|-][0-9]+
 [0-9]*\.[0-9]+e[0-9]+

Syntax

Program: StatementList
 epsilon(i.e. empty)

StatementList: Statement
 Statement StatementList

Statement: Expression;
 if (Expression) Statement else Statement
 while (Expression) Statement
 { StatementList }
 Function { StatementList }

Expression: Prefix_Expression
 Infix_Expression

Prefix_Expression: Prefix_Operator Operand

Infix_Expression: Operand
 Variable = Expression
 Expression + Expression
 Expression - Expression
 Expression && Expression
 Expression / Expression
 Expression % Expression
 Expression | Expression
 Expression & Expression

```
Expression == Expression
Expression <= Expression
Expression >= Expression
Expression < Expression
Expression > Expression
Expression != Expression
Expression ** Expression
Expression || Expression
Expression ? Expression : Expression

Prefix_Operator: !
-
Operand: Variable
Function
Constant
( Expression )

Function: Variable ( Expression_List )
Variable ( )

Constant: Int_Const
Char_Const
Float_Const
```

Functions Definitions

Following are the Avanwaves command system functions and their descriptions.

Use	To
designOpen (string <i>designPath</i>)	Open the design at <i>designPath</i> .
designClose (int <i>designNumber</i>)	Close the design <i>designNumber</i> . The first design is design 0, the second design is design 1 and so on. For example, to close the third design you loaded execute designClose(2);.
designSelect (int <i>designNumber</i>)	Select design <i>designNumber</i> to be the current design. The number is used the same way as in the <i>designClose</i> function. For example, to select the first design you loaded as the current design, execute designSelect(0);.
designGetCurrent()	Return the current design.
designFind(symbol, string <i>designPath</i>)	Return the design whose path is the <i>designPath</i> . If <i>designPath</i> is "", then use this function to return the current design. For example, designFind(D0, "");.
node(symbol/string design, string analysis, string nodename)	Compose a node to be used in other functions, such as pnlAddCurve. For example, node(D0, "A0", "TIME") creates a node from Design 0, Analysis 0 called TIME.
fruiSelect(int <i>analysisNumber</i>)	Select the analysis number <i>analysisNumber</i> in the current design.

Use	To
<p>pnlLabel(int <i>pane</i>, yNode, xNode, float xcoord, float ycoord, string <i>labelstr</i>)</p>	<p>Create a label, whose title is <i>label str</i>. This functions places the label at a coordinate (xcoord, ycoord) according to the curve(xNode, yNode). The <i>xNode</i> and <i>yNode</i> value are supplied with the <i>node</i> function. For example: pnlLabel(1, node (D0,"A0", "v(clk)"), node(D0,"A0", "TIME"), 2e-08, 1.8e-01, "My Label") adds a label <i>My Label</i> at (2e-08, 1.8e-01) according to the coordinates of curve D0:A0:v(clk).</p>
<p>pnlAddPane()</p>	<p>Add a new panel.</p>
<p>pnlDeletePane(int <i>pane</i>, int <i>atomic</i>)</p>	<p>Delete <i>pane</i>, if <i>atomic</i> is 0. In this case, the AvanWaves window is not updated; otherwise, the window is redrawn. Atomic is usually nonzero. But, in some situations, run "pnlDelete(<i>pane</i>, 0)". For example use this command if you want to delete many panels, and you don't want any redraw until the last one is deleted.</p>

Use	To
<p><code>pnlAddCurve(int pane, yNode, xNode, int preference)</code></p>	<p>Add a curve composed of <i>xNode</i>, <i>yNode</i> to <i>pane</i> number <i>pane</i>. <i>Preference</i> is one of the following values: [-1, 0, 1, 2, 3, 4, 5, 6, 7] If the value is between [0, 7], it corresponds to the curve preferences [1, 8]. If the value is -1, then the next available preference is automatically assigned.</p> <p>For example, <code>pnlAddCurve(1, node(D0,"A0","v(clk)"), node(D0,"A0","TIME"), 4)</code> adds a curve D0:A0:v(clk) to panel 1 using curve preference #5 as specified in the preference dialog.</p> <p>Invoke the dialog by selecting Tools >Preferences >Analysis >Curves. Similar to the function <code>pnlLabel</code>, the <i>xNode</i>/<i>yNode</i> values are usually supplied with the <i>node</i> function.</p>
<p><code>pnlHidePane(int pane, int atomic)</code></p>	<p>Hide panel number <i>pane</i>. See the function <code>pnlDeletePane()</code> for the uses of the atomic parameter. For example, <code>pnlHidePane(2, 1)</code> hides the second panel.</p>
<p><code>pnlShowPane(int pane, int atomic)</code></p>	<p>Show panel number <i>pane</i> if it is hidden, it is the opposite of <code>pnlHidePane()</code>.</p>
<p><code>pnlSelectPane(int pane)</code></p>	<p>Select panel number <i>pane</i>. This selection is not exclusive that is, select <i>pane</i> does not deselect other panels that are currently selected.</p>
<p><code>pnlUnselectPane(int pane, int atomic)</code></p>	<p>Unselect panel number <i>pane</i>. See <code>pnlDeletePane()</code> for the uses of the second parameter <i>atomic</i>.</p>

Use	To
<code>pnlClearPane(int pane)</code>	Clear(delete) all curves in panel number <i>pane</i> .
<code>pnlClearSelectedPanels()</code>	Clear(delete) all curves in selected panels only.
<code>pnlSetTitle(int pane, string titleStr)</code>	Set the title of panel number <i>pane</i> to be <i>titleStr</i> . For example, <code>pnlSetTitle(1, "my title")</code> sets the panel number 1's title to be "my title".
<code>pnlGridEnable(int pane, int on)</code>	Turn on or turn off the grid in panel number <i>pane</i> depending on the value of the parameter "on". TODO:: redraw
<code>pnlStackedMode(int pane, int stacked)</code>	Set panel number <i>pane</i> in stacked mode if the value of <i>stacked</i> is not zero; otherwise, the panel is set in overlay mode.
<code>pnlSetXScaleRange(int pane, int logMode, float min, float max)</code>	Set the X axis range of panel number <i>pane</i> as follows: if the value of <i>logMode</i> is 0, the scale is linear, otherwise the scale is logarithmic. The range is from [min, max].
<code>pnlSetY1ScaleRange(int pane, int logMode, float min, float max)</code>	Set the left Y axis range of panel number <i>pane</i> as follows: if the value of <i>logMode</i> is 0, the scale is linear, otherwise the scale is logarithmic. The range is from [min, max].
<code>pnlSetY2ScaleRange(int pane, int logMode, float min, float max)</code>	Set the right Y axis range of panel number <i>pane</i> as follows: if the value of <i>logMode</i> is 0, the scale is linear, otherwise the scale is logarithmic. The range is from [min, max].

Use	To
<code>pnlSetStackedScaleRange(int pane, nodeY, int logMode, float min, float max)</code>	Set the stacked range of Y axis of <i>nodeY</i> as follows: if the value of <i>logMode</i> is 0, the scale is linear, otherwise the scale is logarithmic. The range is from [<i>min</i> , <i>max</i>].
<code>planOpen(string planName)</code>	Open a saved plan (a configuration). The corresponding design must be already opened, and the parameter <i>planName</i> should be just the plan file name without the ".cfg" extension. For example, <code>planOpen("111")</code> opens the 111.cfg plan in the current design.
<code>planSave(string planName)</code>	Save the current display to the plan specified by <i>planName</i> , which is just the filename without .cfg extension, and without any absolute or relative path prefix. For example, <code>planSave("222")</code> saves the current display to plan 222.cfg in the current design.
<code>planDelete(string planName)</code>	Delete the plan whose name is <i>planName</i> in the current design, note that again, <i>planName</i> shouldn't have any .cfg suffix or path prefix.

Use	To
<p>setMeasurePreferences(int <i>pane</i>, int <i>lock</i>, string <i>precision</i>, 0, float <i>lockXVal</i>, float <i>lockYVal</i>, int <i>snapToDataPt</i>)</p>	<p>Set the measure preferences for panel <i>pane</i>. Since the whole panel shares the same measure preferences , this function sets the measure preferences for the entire panel.</p>
<p>The function parameters are:</p>	
<p><i>lock</i></p>	<p>Valid values are 23, 24, or 25, where:</p>
<p>23</p>	<p>No lock</p>
<p>24</p>	<p>Lock X value</p>
<p>25</p>	<p>Lock Y value</p>
<p><i>.precision</i></p>	<p>Takes any values in { 1, 6}. which represents the number of digits after the decimal.</p>
<p><i>lock X value</i></p>	<p>Signifies that the value of <i>lock</i> is 24 and the X value is locked at <i>lockXVal</i>.</p>
<p><i>lock Y value</i></p>	<p>Signifies that the value of <i>lock</i> is 25, and the Y value is locked at <i>lockYVal</i>.</p>
<p><i>snapToDataPt</i></p>	<p>Moves (snaps) the measure points with respect to data points. If the value is 0, it does not snap measure points to data points.</p>

Use	To
<pre>createOnePointMeasure(int pane, yNode, xNode, float x, float y, float sx, float sy, float ex, float ey, int showTitle, int showStartX, int showStartY, int showEndX, int showEndY, int showDeltaX, int showDeltaY,int showSlope, int showDerivative, int hide, int orientation, int transparency, string titleStr, string startXLabel, string endXLabel, string XLabel, string YLabel, string DeltaXLabel, string DeltaYLabel, string SlopeLabel, string DerivativeLabel)</pre>	<p>The syntax for this function is very complex. We recommend that you do not use the command interface to create measures, and use the mouse instead :-).</p>
<pre>createTwoPointMeasure (int pane, yNode1, xNode1, yNode1, xNode2)</pre>	<p>The syntax for this function is also complex. It has 17 to 59 parameters. Contact the Star-Hspice support team for more information.</p>
<pre>createMacro(string macroName, string macroBody, string comments);</pre>	<p>Create a macro called <i>\$macroName</i>. You can use the macro in expressions as an ordinary function.</p>

Use	To
<p>Print(string <i>printName</i>, string <i>whichPanel</i>, string <i>panelsPerPage</i>, bool <i>stepXView</i>, string <i>paperSize</i>, string <i>orientation</i>, string <i>color</i>, string <i>legendStyle</i>, bool <i>printDateTime</i>);</p>	<p>Print the panels specified in <i>\$printName</i>.</p>
<p>The function parameters are:</p>	
<p><i>whichpanel</i></p>	<p>Valid values are <i>all</i> or <i>selected</i>, specifying printing all or selected panels respectively.</p>
<p><i>panelsPerPage</i></p>	<p>Valid values are <i>one</i> or <i>multi</i>, specifying printing one panel per page or multiple panels per page respectively.</p>
<p><i>stepXView</i></p>	<p>Specifies whether to step X view when printing, an int value of 0 means false, all other int value mean true.</p>
<p><i>paperSize</i></p>	<p>Specifies the size of the printer paper, valid values are <i>a4</i>, <i>legal</i>, or <i>letter</i>.</p>
<p><i>orientation</i></p>	<p>Specifies the paper orientation, valid values are <i>portrait</i> or <i>landscape</i>.</p>
<p><i>color</i></p>	<p>Valid values are <i>color</i> or <i>mono</i>, specifying color or monochrome printing respectively.</p>
<p><i>legendStyle</i></p>	<p>Specifies the printed legend style, valid values are <i>hidden</i>, <i>regular</i>, or <i>verbose</i>.</p>
<p><i>printDateTime</i></p>	<p>Specifies whether to tag each page with the current date and time, which is printed on the lower left corner of the page. An int value of "0" means false, all other int values mean true.</p>

Use	To
<code>createExpr(string <i>exprName</i>, string <i>exprBody</i>);</code>	Create an expression named <i>\$exprName</i> . You can lot the expression by using <i>pnlAddCurve()</i> .
<code>updateAll()</code>	Update the display.
<code>quit()</code>	Quit AvanWaves.

Command Examples

Before running the following example, check that you have access to the `$installdir/demo/awaves/tutorial` directory and the file `cpath.sp` is available. If you cannot access the tutorial directory or the `cpath.sp` file, enter on the command line:

```
$installdir/demo/awaves/tutorial/tran/cpath.sp to your
/dir/to/cpath.sp.
```

To list the available awaves options, enter on the command line:

```
% awaves -help
```

Avanwave command line arguments are:

<code>-d <design path></code>	Specify the design path.
<code>-c <config></code>	Specify the configuration to open (must appear immediately after the design).
<code>-e <script></code>	Execute a Meta Waves script file.
<code>-b <script></code>	Execute a Meta Waves script file in background mode (no windows).
<code>-laf <motif openlook windows></code>	Specify the look and feel.
<code>-i</code>	Invoke awaves command UI at start up.
<code>-h</code>	Help (this message).

Example 1

In this example, two waveforms, v(clk) and v(data), are selected for display in a panel with user-specified plotting ranges, title and label.

To run this example, save the following to a file called *example1* and then type: `awaves -e example1` to execute it.

```
// filename: "example1", this line is comments
// start

// definte variable
myVar = "$installdir/demo/awaves/tutorial/tran/cpath.sp";

designOpen(myVar);
designFind(D0, "");
pnlAddCurve(1, node(D0,"A0","v(clk)"), node(D0,"A0","TIME"), 0);
pnlAddCurve(1, node(D0,"A0","v(data)"), node(D0,"A0","TIME"), 1);
pnlShowPane(1, 1);
pnlSelectPane(1);
pnlSetTitle(1, "*** My title via executable script");
pnlSetXScaleRange(1, 0, 5e-09, 2.2500000613520112e-08);
pnlSetYlScaleRange(1, 0, 1e00, 3.46499994993209848e+00);
pnlLabel(1, node(D0,"A0","v(clk)"), node(D0,"A0","TIME"), 2.1e-08, 1.9,
"myLabel");
fruiSelect(0);
// end of example1
```

Example 2

This example shows how to select two waveforms, v(clk) and c(data), and display them in stack mode. Save the following file as *example2* and execute it using the command `awaves -e example2`.

```
// filename: example2
// begin
designOpen("$installdir/demo/awaves/tutorial/tran/cpath.sp");
designFind(D0, "");
pnlAddCurve(1, node(D0,"A0","v(clk)"), node(D0,"A0","TIME"), 0);
pnlAddCurve(1, node(D0,"A0","v(data)"), node(D0,"A0","TIME"), 1);
pnlShowPane(1, 1);
pnlSelectPane(1);
pnlSetTitle(1, "Example 2: Display waveforms in stack mode");
pnlStackedMode(1, 1);
fruiSelect(0);
// end of example2
```

Example 3

This example shows how to build expression, add a panel, and print the waveforms to designed printer. Modify the names of `myDefaultPrinter` and `myColorPrinter` to be your printers before you execute the file.

```
// Definte variables
myDesign = "$installdir/demo/awaves/tutorial/tran/cpath.sp";
myDefaultPrinter = "laser2";
myColorPrinter = "color3paper";

// open my design
designOpen( myDesign );

// add two curves and a label
pnlAddCurve(1, node(D0,"A0","v(clk)"), node(D0,"A0","TIME"), 7);
pnlAddCurve(1, node(D0,"A0","v(data)"), node(D0,"A0","TIME"), 5);

pnlLabel(1, node(D0,"A0","v(clk)"), node(D0,"A0","TIME"),
2.5772727951789420e-08, 1.8657691776752472e-01, "myLabel");

pnlGridEnable(1,0);

// create a macro,

myMacroName = "abc(x,y)";
```

AvanWaves Command System

```
myMacroBody = "sqrt(abs(x*x+y*y))";
myMacroComments = "blah blah blah";

createMacro( myMacroName, myMacroBody, myMacroComments );

// create a simple expression using abs()
// note that " needed to be quoted if it is inside a string
createExpr("aaa", "abs(node(D0,\"A2\", \"v(data)\"))");

// create another expression using macro "abc(x,y)"
createExpr("bbb",
"abc(node(D0,\"A1\", \"v(clk)\"), node(D0,\"A2\", \"v(q4)\"))");

// add a panel and plot "aaa" and "bbb"

pnlAddPane();
pnlAddCurve(2, node(D0,"A2", "bbb"), node(D0,"A2", "TIME"), 1);
pnlAddCurve(2, node(D0,"A2", "aaa"), node(D0,"A2", "TIME"), 4);

// select the second panel we just created
pnlSelectPane(2);
pnlSetTitle(2, "A panel of expression \"aaa\" & \"bbb\" ");

// print a few panels
Print( myDefaultPrinter, "selected", "one", 0, "letter",
"port", "mono", "verb", 1);
Print( myColorPrinter, "all", "multi", 0, "lett", "port", "color",
"reg", 1);

// end of example 3
```

Example 4

This is an example of using the Avanwaves command system interactively.

To invoke the AvanWaves command user interface, enter on the command line: `awaves -i`, or select **Tools > Awaves Commands** after `awaves` start up.

The online help for the command UI is not available yet.

When the command interface window opens, type each of the following commands in the command input area, then press the <Return> key.

```
$designOpen("$installdir/demo/awaves/tutorial/tran/cpath.sp");
$pnlAddCurve(1, node(D0,"A1","v(q5)", node(D0,"A1","TIME"), 1);
$pnlAddPane( );
$pnlAddCurve(2, node(D0,"A1","v(qn)", node(D0,"A1","TIME"), 2);
```

Two panels with a curve appear.

```
$quit();
```

AvanWaves quits.