

**SigmaStudioHelp**

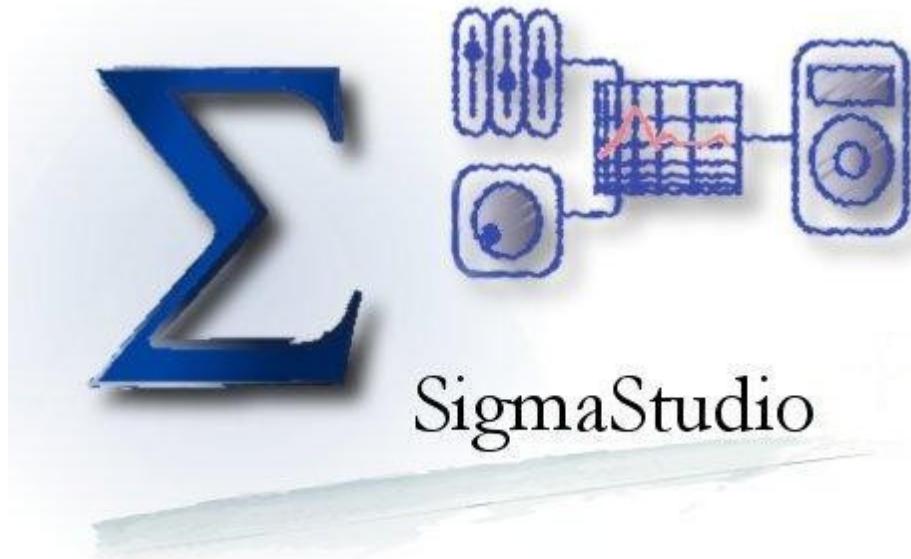
---

## Table Of Contents

Getting Started.....	2
Evaluation Board Setup Examples.....	17
USB Interfaces .....	26
Development Environment.....	36
Dialogs .....	57
Hardware Windows .....	63
Using SigmaStudio .....	78
Building Schematics .....	79
Schematic Actions and Commands.....	124
Toolbox .....	136
System (Schematic Design).....	137
ADI Algorithms .....	149
Basic DSP .....	171
Counters.....	194
Counter example: .....	195
Dynamics Processors.....	210
Envelope.....	211
Compressors .....	217
Filters .....	236
GPIO Conditioning .....	284
IO .....	289
Level Detectors / Lookup Tables.....	300
Licensed Algorithms .....	310
Analog Devices.....	311
External LED Array Driver.....	315
BBE .....	320
Dolby .....	323
SRS .....	328
Mixers / Splitters.....	334
Multiplexers / Demultiplexers .....	348
Sources .....	362
Volume Controls.....	375
Examples .....	388
Algorithm Information.....	421
SigmaDSP Architecture .....	440
Tutorials .....	452
Index.....	459

## Welcome To SigmaStudio

---



Welcome to the Analog Devices SigmaStudio Graphical Development Tool!

SigmaStudio was developed specifically for use with our SigmaDSP audio processors. The graphical development environment allows engineers with no DSP coding experience to add digital signal processing to their designs. A wide variety of signal processing blocks can be wired together, as in a schematic, and the SigmaStudio compiler generates production-ready code while providing a control interface for setting and tuning parameters.

This tool is tailored to engineers tasked with migrating analog sound processing systems to the digital domain, yet, it is powerful enough to satisfy the most experienced DSP designers. SigmaStudio can help you reduce development time and costs without sacrificing quality or performance.

- See what's new in this release of SigmaStudio
- To get started, follow the Quick Start guide
- To see what SigmaStudio can do, check out the Tutorials

If you are new to SigmaStudio, it is recommended that you first study the help documents to become familiar with the application and the terminology before beginning to work.

---

# Getting Started

## First Steps

---

Follow these steps to configure your system and to get started using SigmaStudio. Refer to the Quick Start guide for more information.

- Check the Installation Guide to ensure SigmaStudio is properly installed.
- Ensure that your hardware is setup correctly; see Evaluation Board Setup for some typical configurations.
- Connect the USB interface to your board and PC.
- Start the SigmaStudio software which brings you to the main Program Window.
- Create a new project and select a Processor in the Hardware Configuration Tab.
- Select the Schematic Tab and begin to construct your design!

For more information about SigmaDSP, please go to [www.analog.com/sigmadsp](http://www.analog.com/sigmadsp).

---

## Installation Procedure

This chapter describes how to install the SigmaStudio on a computer running the Microsoft Windows operating system.

### *□ System Requirements*

- Windows XP Professional or Home Edition with SP2; Windows 2000 with SP4; Windows ME; Windows Server 2003
- 128 MB of RAM (256 MB recommended)
- 50 MB of available hard disk space
- 1024 x 768 screen resolution
- USB 1.1/2.0 data port (Required for use with Evaluation hardware only)

Note: SigmaStudio can be installed under **Windows Vista**, however Windows Vista is not officially supported at this time and Analog Devices cannot offer any technical support regarding Vista installation or issues.

### **Step 1: Before You Install**

- Quit any applications you are running.
- Confirm that the *Microsoft .NET Framework v2.0 is installed*.

To install the Microsoft .Net Framework from the SigmaStudio CD, browse to the CD folder, and double-click the installer "**dotnetfx.exe**".

(The "Microsoft .NET Framework Version 2.0 Redistributable Package" installation and updates are also available for download directly from microsoft  
**[www.microsoft.com/downloads](http://www.microsoft.com/downloads)**)

### **Step 2: Application Setup**

- Run the setup application (Sigma Studio.msi).
- Review the contents of the license agreement.
- If installing SigmaStudio for the first time, restart your computer when the installation is complete.

### **Step 3: Driver Installation**

- Connect the USB cable to your evaluation hardware and to a spare USB port on your PC.
- The Windows "Found New Hardware Wizard" will launch.
- Choose "Install from a list or a specific location" and click "Next".
- Select "Search for the best driver in these locations" and Check the box for "Include this location in the search."
- Press the "Browse" button and locate the appropriate driver file in the SigmaStudio application folder (default folder is C:\Program Files\Analog Devices Inc\Sigma Studio\USB drivers):
  - For **USB serial converter** and Eval-Boards (FDTIxx.inf) select **ftd2xx.inf** file.
  - For **USBi** interface select **CyUSB.inf** file.
- Click "Continue Anyway" if you're prompted with "This software has not passed

Windows Logo testing."

**Step 3: Hardware Configuration**

- Setup the Evaluation hardware jumpers and i/o configurations, refer to Evaluation Board documentation on the on the SigmaStudio CD (EVAL\_ADxxx.pdf). Documentation is also available on the web at [www.analog.com/sigmadsp](http://www.analog.com/sigmadsp).
  - For additional help, see the Evaluation Board Setup Examples.
-

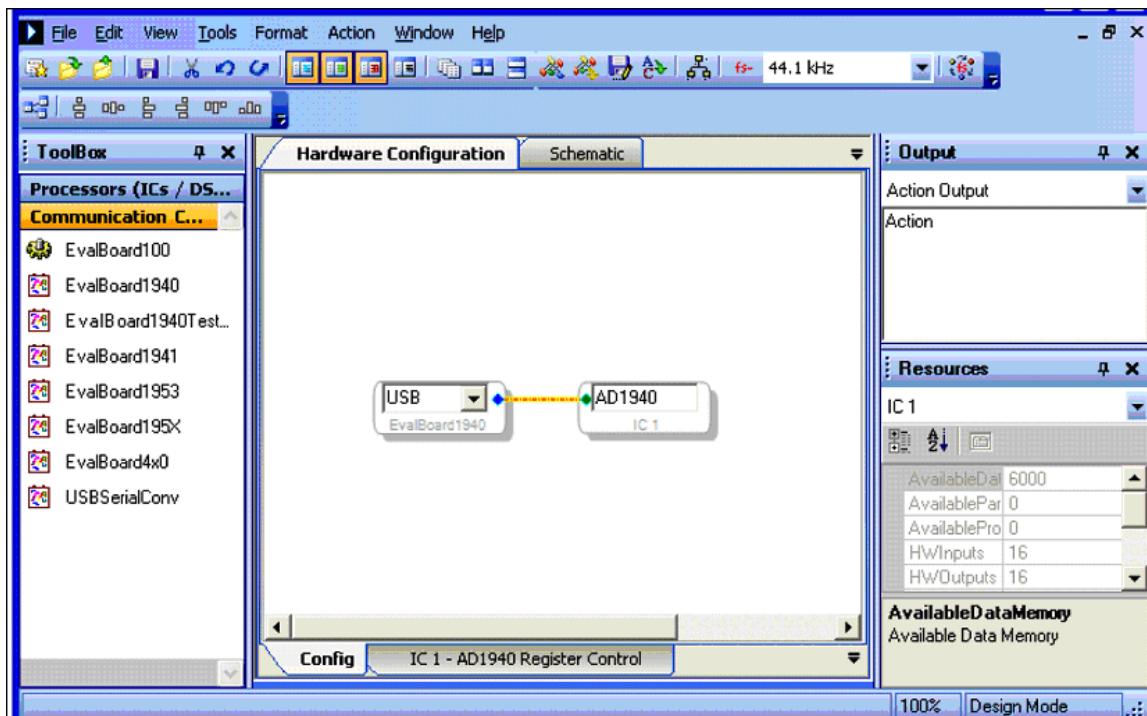
## Quick Start

These are abbreviated instructions to help get you started. Click here to take a tour of the application, and see the tutorials and sample schematics for more details and examples.

### Your First Project: Stereo Audio Input/Output with Volume Control

#### [Click Here for AD1940](#)

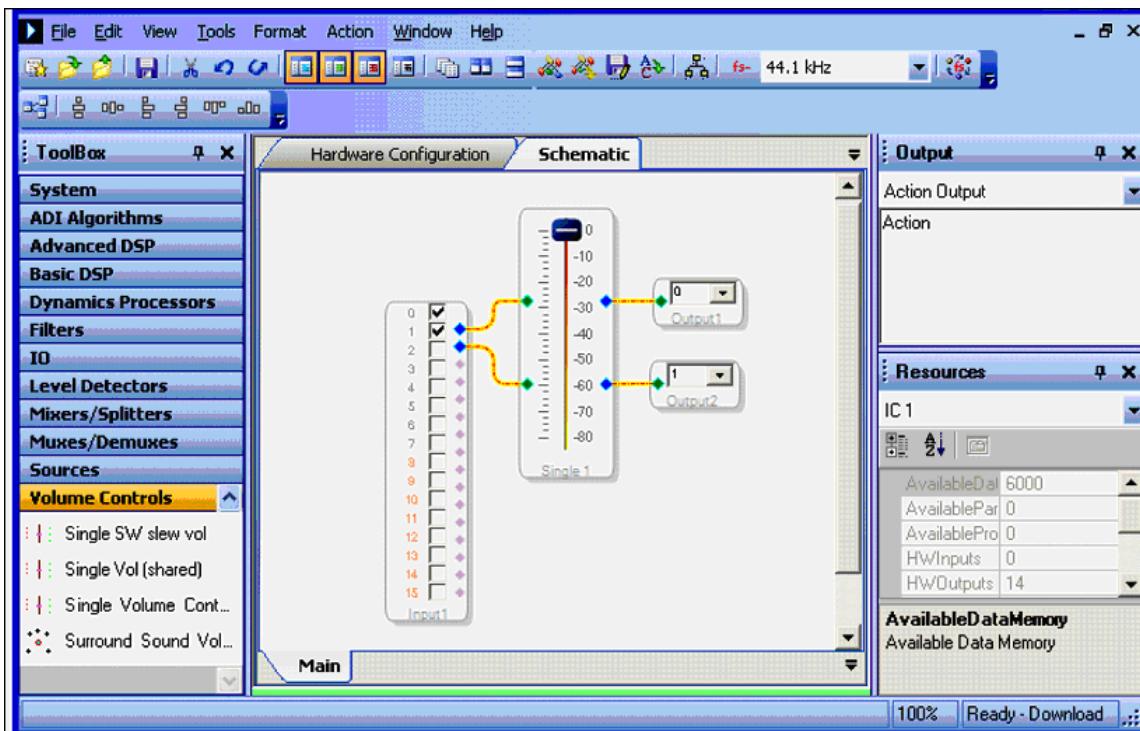
- 1) Launch SigmaStudio and create a project (**File, New Project**). The **Hardware Configuration** workspace will display.
- 2) Drag an **EvalBoard1940** and an **AD1940** block into the workspace.
- 3) On the **EvalBoard1940** block, select **USB**.
- 4) Connect the **EvalBoard1940** block to the **AD1940** block with a wire, by clicking and dragging from the blue output pin to the green input pin. It should look something like this:



**1940 screen**

- 5) Click **Schematic** (tab at the top of workspace).
- 6) Click **IO** in the **Toolbox** and drag an **Input** block to the workspace.
- 7) Click **Volume Controls** in the **Toolbox** and drag a **SingleVolume Control** to the workspace.
- 8) Right-click that block (called **Single 1**) and choose **Add Algorithm** → **IC1** → **Gain (slew)**. (This adds input / output.)
- 9) Repeat step 8 to give the **Volume Control** stereo inputs and outputs.
- 10) Connect the **Input** block's blue output pins to the green input pins on the **Volume Control**.

- 11) Click **IO** in the **Toolbox** and drag two **Output** blocks to the workspace.
- 12) If your speakers aren't connected to outputs 0/1, use the dropdown to change them to the correct outputs.
- 13) Connect the **Volume Control** blue output pins to the green input pins on the **Output** blocks.
- 14) After ensuring that your board is powered and connected to the PC, click **Link-Compile-Download** on the taskbar. Assuming the project compiled without error, you will be in **Ready-Download** mode and see something similar to this figure.



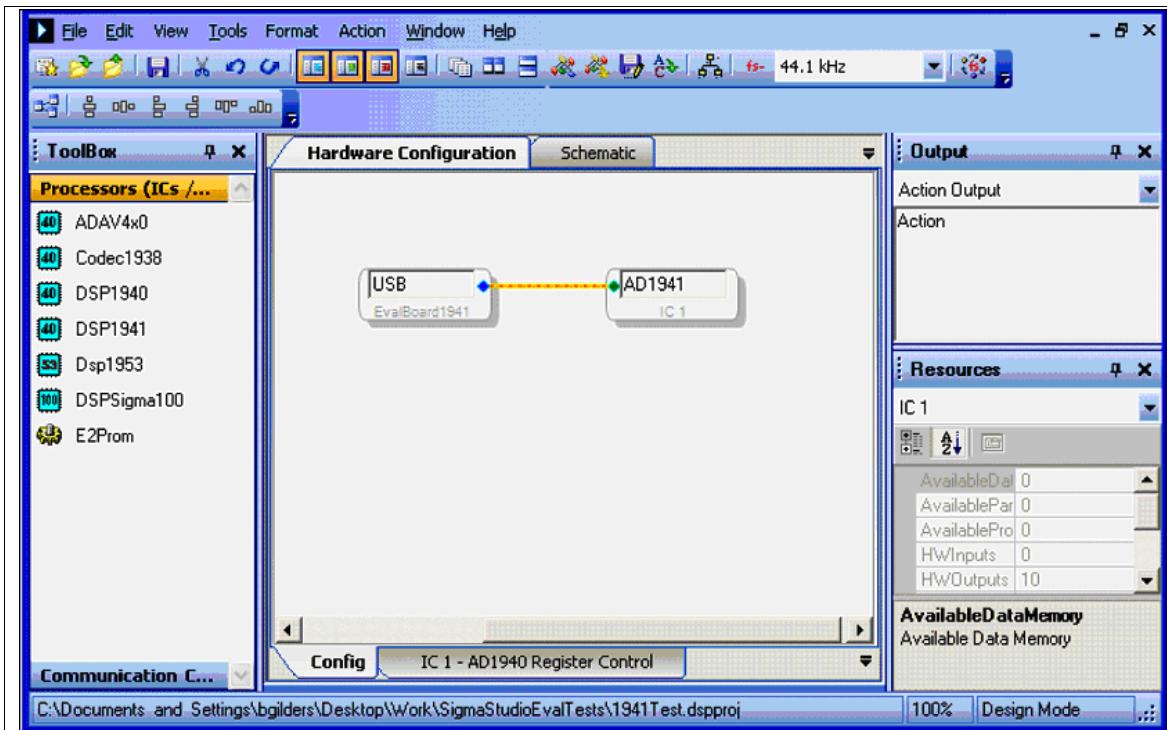
**1940 screen**

- 15) Start your source playing and you should hear audio. Move the volume slider to change the level.

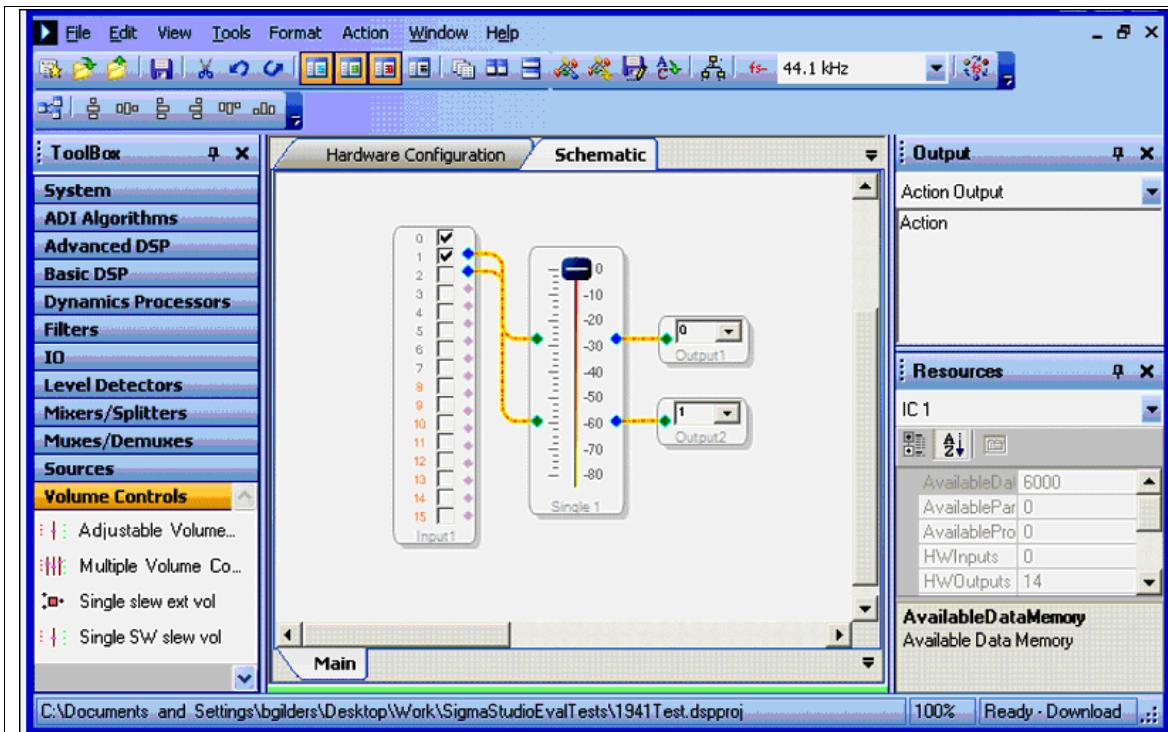
For more details on this and similar projects, see the first of the tutorials.

#### [Click Here for AD1941](#)

- 1) Launch SigmaStudio and create a new project (**File, New Project**). The **Hardware Configuration** workspace will display.
- 2) Drag an **EvalBoard1941** and an **AD1941** block into the workspace.
- 3) On the **EvalBoard1941** block, select **USB**.
- 4) Connect the **EvalBoard1941** block to the **AD1941** block with a wire, by clicking and dragging from the blue output pin to the green input pin. It should look something like this:

**1941 screen**

- 5) Click **Schematic** (tab at the top of workspace).
- 6) Click **IO** in the **Toolbox** and drag an **Input** block to the workspace.
- 7) Click **Volume Controls** in the **Toolbox** and drag a **SingleVolume Control** to the workspace.
- 8) Right-click that block (called **Single 1**) and choose **Add Algorithm** → **IC1** → **Gain (slew)**. (This adds input / output.)
- 9) Repeat step 8 to give the volume control stereo inputs and outputs.
- 10) Connect the **Input** block's blue output pins to the green input pins on the **Volume Control**.
- 11) Click **IO** in the **Toolbox** and drag two **Output** blocks to the workspace.
- 12) If your speakers aren't connected to outputs 0/1, use the dropdown to change them to the correct outputs.
- 13) Connect the **Volume Control** blue output pins to the green input pins on the **Output** blocks.
- 14) Connect the USB board to the PC.
- 15) Connect the power to the eval board (J14).
- 16) Press **Program Load** on the USB adapter board.
- 17) Now, connect the USB adapter to the eval board via the Control Interface (J17).
- 18) Click **Link-Compile-Download** on the taskbar. Assuming the project compiled without error, you will be in **Ready-Download** mode and see something similar to this figure:



## 1941 screen

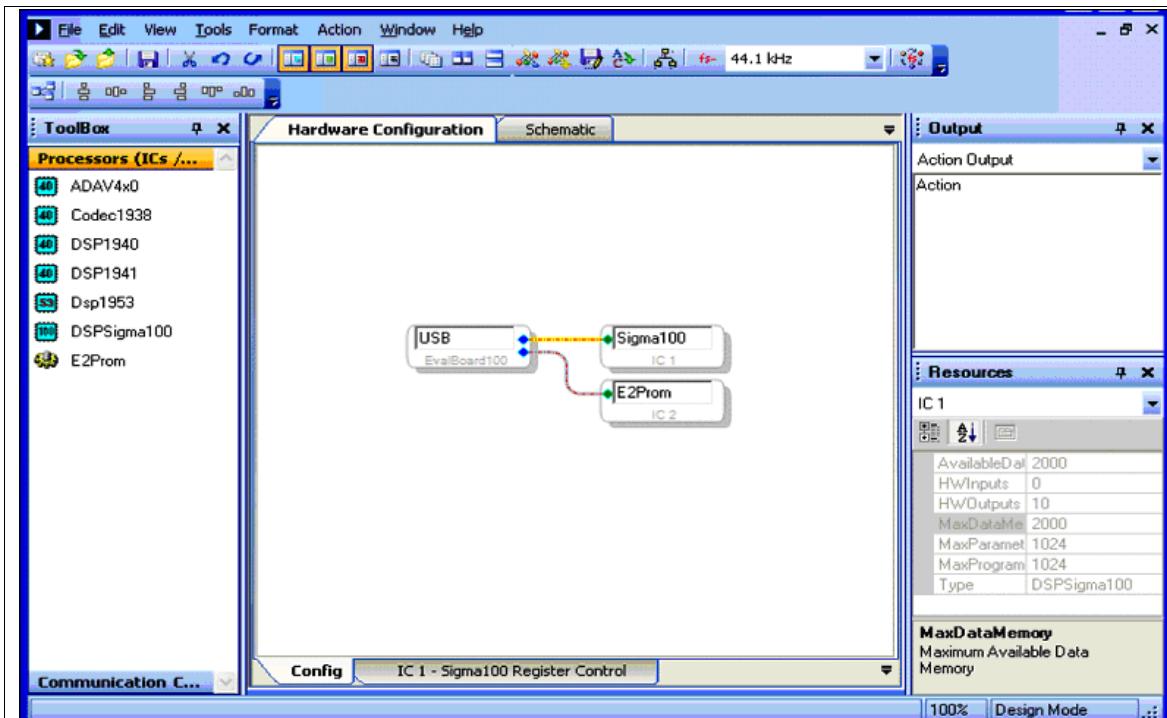
19) Start your source playing and you should hear audio. Move the volume slider to change the level.

For more details on this and similar projects, see the tutorials.

### Click here for ADAU1701

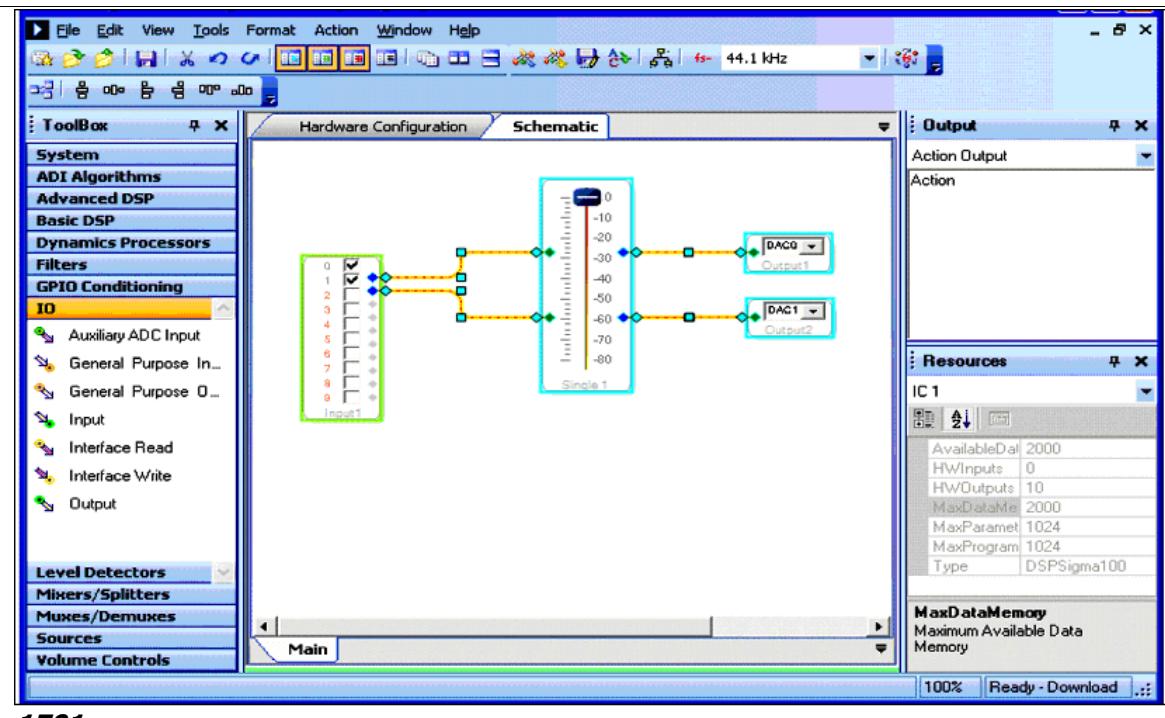
- 1) Launch SigmaStudio and create a new project. The **Hardware Configuration** tab and workspace will display.
- 2) Drag a **DSPSigma100**, **E2Prom**, and an **EvalBoard100** block into the workspace.
- 3) Connect the **EvalBoard100** to the **DSPSigma100** block by clicking and dragging from the top blue output pin to the green input pin.
- 4) Connect the **DSPSigma100** bottom blue output pin to the green input pin on the **E2Prom** block.

The result should look something like this:

**1701 screen**

- 5) Click **Schematic** (tab at the top of workspace).
- 6) Click **IO** in the **Toolbox** and drag an **Input** block to the workspace.
- 7) Click **Volume Controls** in the **Toolbox** and drag a **SingleVolume Control** to the workspace.
- 8) Right-click that block (called **Single 1**) and choose **Add Algorithm** → **IC1** → **Gain (no slew)**. (This adds input / output.)
- 9) Connect the **Input** block's blue output pins to the green input pins on the **Volume Control**.
- 10) Click **IO** in the **Toolbox** and drag two **Output** blocks to the workspace.
- 11) If your speakers aren't connected to outputs 0/1, use the dropdown to change them to the correct outputs.
- 12) Connect the **Volume Control** blue output pins to the green input pins on the **Output** blocks.
- 13) After ensuring that your board is powered and connected to the PC, click **Link-Compile-Download** on the taskbar.

Assuming the project compiled without error, you will be in **Ready-Download** mode and see something similar to this figure.



## 1701 screen

14) Start your source playing and you should hear audio. Move the volume slider to change the level.

For more details on this and similar projects, see the tutorials.

## New Features

---

### **Version 3:**

- “General 2nd Order Index Selectable” filter with graphical adjustment of filter response.
  - Sawtooth, Square, and Triangle wave source algorithms.
  - Mux and Mixer algorithms with software slew.
  - Project and block settings; State of all block/algorithm controls can be saved and recalled to/from “settings” files for a/b comparisons or re-using settings between projects.
  - “Tree Toolbox” window, design building blocks are functionally categorized for each IC in a tree view.
  - “Change IC”, change IC can be applied to a selection, assigning the IC for all selected block(s) and algorithm(s) in a single operation.
  - Hierarchy Files, Hierarchy boards can be saved as files and re-used in new designs.
  - Hierarchy Board “Hide”, Hierarchy board tabs can be hidden and optionally password protected to protect confidential design elements.
  - Hierarchy Board “Freeze”, Hierarchy boards can be individually frozen to prevent modifications.
  - Undo/Redo support, All design modification can be undone and re-done.
  - Cut/Copy/Paste support, Design blocks (blocks) can be copied and pasted, within a design and between project files.
  - Schematic Printing, schematic designs can be printed.
  - Enhanced capture window, communication capture window can be docked and customized.
-

## License Agreement

### SIGMA STUDIO END USER LICENSE AGREEMENT

**IMPORTANT!!!! READ THIS LICENSE AGREEMENT (THE "AGREEMENT") CAREFULLY. IT SETS FORTH THE TERMS AND CONDITIONS ON WHICH ANALOG DEVICES, INC. ("ANALOG DEVICES") IS WILLING TO LICENSE THE SIGMA STUDIO SOFTWARE (THE "SOFTWARE") TO ITS CUSTOMER ("YOU", EITHER AN INDIVIDUAL ACTING ON THEIR OWN BEHALF OR AN ENTITY ACTING THROUGH AN INDIVIDUAL). ANALOG DEVICES RECOMMENDS THAT YOU PRINT OUT THIS LICENSE AND RETAIN A COPY IN YOUR FILES.**

1. **GRANT OF LICENSE.** Subject to the terms of this Agreement, Analog Devices grants you the non-exclusive right to install and use a single object code copy of the Software on a single personal computer and to use the accompanying software documentation ("Documentation") in connection with your use of the Software **solely** for the Permitted Purpose described in Section 2 of this Agreement.
2. **THE PERMITTED PURPOSE.** The Software may **only** be used to program products purchased from Analog Devices ("ADI Products") for incorporation by you into audio products that are intended for resale to audio product end users (the "Permitted Purpose"). Code generated by use of the Software may be loaded onto the ADI Products as incorporated into your audio products but may **not** be distributed in any other form.
3. **ADDITIONAL RESTRICTIONS ON LICENSE.** You may use the Software **only** on the computer on which you originally install it, and you may not transfer it to other hardware, share it or run it concurrently on more than one computer, port it to other platforms or translate it into another computer language. You may **NOT**: (i) make any copies of all or any part of the Software or otherwise reproduce the Software except for archival copies as permitted by the United States Copyright Act; (ii) copy or modify all or any part of the Documentation or distribute it to third parties; (iii) rent, lease or sub-license the Software or use the Software in connection with a service-bureau, time sharing or fee-for-service arrangement with third parties; (iv) decompile, disassemble, reverse engineer or otherwise discover the source code for the Software, or attempt to disable or defeat any locking mechanism within the Software; (v) modify the Software; (vi) use any module of the Software independently of the Software as provided by Analog Devices; (vii) other than as specifically required to use the Software for the Permitted Purpose, incorporate the Software in whole or in part in any other product or create derivative works based on all or any part of the Software; (viii) remove any copyright, trademark, proprietary rights, disclaimer or warning notice included on or embedded in any part of the Software; or (ix) export the Software or use the Software in any country other than that in which it was obtained. You acknowledge that the Software is subject to United States export laws and regulations and you shall comply with all such laws and regulations in your use of the Software.
4. **SPECIAL RESTRICTIONS ON THE USE OF THIRD PARTY-OWNED PLUG-INS.** At your request and upon submission to Analog Devices of proof that you have entered into an

agreement with a third party that entitles you to incorporate certain of that third party's proprietary algorithms or other intellectual property into your audio products, you may apply to receive certain plug-ins from Analog Devices that include certain of that third party's algorithms or other intellectual property ("Plug-Ins"). Upon verification to Analog Devices' satisfaction that you are entitled to receive a particular Plug-In (which may include confirming your right to obtain the algorithm from the third party that owns it), Analog Devices will send you executable software that, when loaded onto your computer, will identify the computer on which the executable software is running (the "Identification Program"). You will then be required to send Analog Devices the file generated by the Identification Program and Analog Devices will use that file to generate a license key that will unlock the algorithm for use only on the computer identified by the Identification Program. Analog Devices will then send you the applicable Plug-In, the license key and the license file, which Plug-In, key and file will be governed by this Agreement, including the restrictions set forth in Sections 2 and 3 hereof. NOTE: You may **not** use the Plug-In for any purpose other than the Permitted Purpose. You may **not** use the Plug-In except in conjunction with your use of the integrated Software as provided by Analog Devices. You may **not** transfer the Plug-In or any product or derivative of the Plug-In to any other third party except as incorporated by you into audio products that are intended for resale to audio product end users. Upon termination of the agreement with the third party that entitles you to use a Plug-In, you agree to cease all use of such Plug-In and return it to Analog Devices or destroy it and certify such return or destruction in writing to Analog Devices.

**5. OWNERSHIP; NON-DISCLOSURE.** Analog Devices and its licensors own and retain all right, title and interest, including without limitation all copyright, trademark, trade secret, patent and other proprietary rights, in and to the Software, the Documentation and the Plug-Ins (the "Proprietary Materials"). Except as specifically authorized by this Agreement, you shall not disclose, sell, lease, transfer, sublicense, dispose of, or otherwise make available the Proprietary Materials or any portion thereof, in source or object code, to any third party. You agree that dissemination of the Proprietary Materials in breach of this Agreement would cause irreparable harm to Analog Devices and its licensors for which monetary compensation alone would be inadequate, and Analog Devices and/or such licensors are entitled to injunctive relief prohibiting any such dissemination without the necessity of posting bond, in addition to any other remedies available at law or in equity. This Agreement is NOT a sale of the Proprietary Materials or any copy of them. You obtain only such rights as are provided in this Agreement.

**6. TERM AND TERMINATION.** The license granted by this Agreement will continue indefinitely for as long as you purchase ADI Products for incorporation into your audio products unless earlier terminated as provided herein. With respect to Plug-Ins, the license will terminate upon the first to occur of (a) notice from the relevant third party that you are no longer entitled to incorporate that third party's proprietary algorithms or other intellectual property into your audio products or (b) termination of the agreement with the third party that entitles you to incorporate that third party's proprietary algorithms or other intellectual property into your audio products. This Agreement will terminate automatically if you fail to comply with any term or condition of this Agreement. Upon the termination of this Agreement, you will promptly return to Analog Devices or destroy all copies of the Proprietary Materials and certify such return or destruction in writing to Analog Devices. You may also terminate this Agreement voluntarily by destroying the Proprietary Materials or returning them to Analog Devices. Sections 3-5 and 7-18 shall survive any termination of this Agreement.

**7. YOUR RESPONSIBILITIES.** You are responsible for the supervision, management and control of the use of the Software, including but not limited to: (i) complying with all laws in your use of the Software; (ii) selection of the Software to achieve your intended results; and (iii) establishing adequate test and backup procedures to verify accurate data and to prevent the loss of data in the event of a malfunction of the Software (iv) establishing adequate test and backup procedures to verify accurate data and to prevent the loss of data in the event of a version upgrade of the Software. You understand that the Software is a tool that can be used to create algorithms, processes and products that may infringe third party intellectual property rights. You assume full responsibility for the results of your use of the Software and agree to indemnify and hold harmless Analog Devices and its

licensors from and against any claims that algorithms, processes or products created by you using the Software infringe any third party intellectual property rights or other rights.

**8. DISCLAIMER OF ALL WARRANTIES.** You understand and agree that the Software, Documentation and Plug-Ins (if any) are provided free of charge to assist you in your development of audio products that incorporate the Sigma DSP products and, as such, are provided "AS IS" and without warranty, express or implied, of any kind. **WITH RESPECT TO THE SOFTWARE, DOCUMENTATION, AND PLUG-INS, ANALOG DEVICES AND ITS LICENSORS SPECIFICALLY DISCLAIM ALL IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF NONINFRINGEMENT, MERCHANTABILITY AND FITNESS FOR PARTICULAR PURPOSE.**

**9. EXCLUSION OF CERTAIN DAMAGES.** **TO THE MAXIMUM EXTENT PERMITTED BY LAW, ANALOG DEVICES AND ITS LICENSORS EXCLUDE ALL LIABILITY FOR DIRECT, SPECIAL, INCIDENTAL, INDIRECT, EXEMPLARY, PUNITIVE, CONSEQUENTIAL OR ANALOGOUS DAMAGES (INCLUDING, WITHOUT LIMITATION, ANY DAMAGES RESULTING FROM LOSS OF USE, LOSS OF DATA OR LOSS OF PROFITS) ARISING OUT OF OR IN CONNECTION WITH THE PERFORMANCE OF THE SOFTWARE, THE DOCUMENTATION AND THE PLUG-INS, WHETHER SUCH LIABILITY ARISES UNDER CONTRACT, TORT (INCLUDING NEGLIGENCE) OR STRICT LIABILITY, AND WHETHER ANALOG DEVICES OR ITS LICENSORS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.**

**10. LIMITATION OF AGGREGATE DAMAGES.** In the event that the complete exclusion of damages set forth in Section 9 is unenforceable, the aggregate liability of Analog Devices and its licensors in connection with this Agreement and/or the use of the Software, Documentation and Plug-Ins shall not exceed \$1,000, whether such liability arises by contract, tort (including negligence) or strict liability. An essential purpose of the limitation on damages provided in this Section is the allocation of risks between you and Analog Devices, and you acknowledge that Analog Devices and its licensors would not have been willing to license the Software and Plug-Ins to you in the absence of the limitation set forth in this Section 10 and the exclusions set forth in Section 9 above.

**11. USE IN LIFE SUPPORT APPLICATIONS.** Products sold by Analog Devices are not designed for use in life support and/or safety equipment where malfunction of the product can reasonably be expected to result in personal injury or death. You use and sell such products for use in life support and/or safety applications at your own risk and you agree to defend, indemnify and hold harmless Analog Devices and its licensors from any and all damages, claims, suits or expense resulting from such use or sale.

**12. U.S. GOVERNMENT RIGHTS.** The Software, Documentation and Plug-Ins were developed solely at private expense and contain proprietary data belonging to Analog Devices and its licensors. They constitute "commercial components," as that term is defined in 48 C.F.R. 2.101 (Oct. 2000), consisting of "commercial computer software" and "commercial computer software documentation," as such terms are used in 48 C.F.R. 12.212 (Sept. 1995). Consistent with 48 C.F.R. 12.212 and 48 C.F.R. 227.7202-1 through 227.7202-4 (June 1995), all U.S. Government entities acquire this product only with those rights set forth in the license agreement accompanying this product.

**13. FORCE MAJEURE.** Analog Devices shall not be liable for any failure to perform hereunder to the extent that such failure arises by factors outside Analog Devices's reasonable control, including but not limited to Acts of God, war, terrorism, natural disaster or third party communications failure.

**14. ASSIGNMENT.** You may not sell, license, assign, or otherwise transfer the Software, the Documentation or any Plug-Ins without the written permission of Analog Devices, which may be withheld in Analog Devices's sole discretion. Any such sale, license, sublicense, assignment, rental or

transfer in breach of this provision shall be void.

15. **AMENDMENT; WAIVER; SEVERABILITY.** No alteration, amendment, waiver, cancellation or other change to this Agreement shall be valid or binding unless agreed in writing by both parties. No waiver by any party of a breach of any covenant or condition of this Agreement by any other party shall be construed to be a waiver of any succeeding breach of the same or any other covenant or condition. If any provision of this Agreement, or the application thereof, shall for any reason and to any extent be determined by a court of competent jurisdiction to be invalid or unenforceable, the remaining provisions of this Agreement shall be interpreted so as best to reasonably effect the intent of the parties. The parties further agree to replace any such invalid or unenforceable provisions with valid and enforceable provisions designed to achieve, to the extent possible, the business purposes and intent of such invalid and unenforceable provisions.

16. **CHOICE OF LAW.** The terms of this Agreement shall be construed in accordance with the substantive laws of the Commonwealth of Massachusetts, without regard to its principles of conflicts of laws. Without limiting the generality of the foregoing, this Agreement shall not be governed by the United Nations Convention on Contracts for the International Sale of Goods even if such Convention would otherwise be applicable in some respect to this license of software.

17. **AUDIT RIGHTS.** Analog Devices may audit you and inspect your computer during regular business hours at your premises to verify your compliance with all terms and conditions of this Agreement.

18. **ACKNOWLEDGEMENT; ENTIRE AGREEMENT.** YOU ACKNOWLEDGE THAT YOU HAVE READ THIS AGREEMENT, UNDERSTAND IT AND AGREE TO BE BOUND BY ITS TERMS AND CONDITIONS. YOU FURTHER AGREE THAT IT IS THE COMPLETE AND EXCLUSIVE STATEMENT OF THE LICENSE AGREEMENT BETWEEN YOU AND ANALOG DEVICES WHICH SUPERSEDES ANY PROPOSAL, OR PRIOR AGREEMENT, ORAL OR WRITTEN, AND ANY OTHER COMMUNICATIONS BETWEEN YOU AND ANALOG DEVICES RELATING TO THE SUBJECT MATTER OF THIS LICENSE AGREEMENT,

---

## Getting Help with SigmaStudio

---

If you need more information about using the program, you can access the help documentation at any time by pressing F1 in the SigmaStudio application. In addition, help is available in PDF format (SigmaStudioHelp.pdf) in the application's installation directory.

You can Also:

- Check out the SigmaStudio FAQ page for answers to common questions.
- Step through the tutorial projects.
- Contact Analog Devices at [sigmadsp@analog.com](mailto:sigmadsp@analog.com).

**Note:** To locate SigmaStudio version information, select Help - About in the main application menu. A dialog is displayed identifying the current version of SigmaStudio installed on your system.

### □ Using the Help window

SigmaStudio's Help includes the following tabs:

- **Contents:** The Contents tab displays the table of contents - books and pages that represent the information in the Help system. Double click a closed book to open and display its contents. Double click an open book to hide its contents. Select a page to view the topic in the right-hand pane.
- **Index:** The Index tab displays a list of keywords and keyword phrases. These terms are associated with topics in the Help system and are intended to direct you to specific topics. Click a keyword to open the topic in the right-hand pane. Some keywords are associated with more than one topic, a dialog opens so you can select a specific topic to view.
- **Search:** The Search tab allows for finding related topics based on the entered keywords. Type some keyword(s) to located the desired topic.
- **Related Topics button:** When you click a Related Topics button, a popup menu opens that displays a list of related topics you can view. The topics are relevant to what you are currently reading or they reference related information. Choosing a topic in the popup menu will open in the right-hand pane.

## Evaluation Board Setup Examples

### Evaluation Board Quick Start Setup

---

For detailed information regarding Evaluation Board configuration, refer to the evaluation board data sheets on the on the SigmaStudio CD (EVAL\_ADxxx.pdf). Documentation is also available at [www.analog.com/sigmadsp](http://www.analog.com/sigmadsp).

## **AD1940/AD1941**

The three AD 194x boards are almost identical. The EB1940 has switches; the MINIB 1940 does not. Both boards use SPI (serial-peripheral interface) communication, while the 1941 uses the I<sup>2</sup>C (also written I<sup>2</sup>C) communication protocol. Read through the first sets of steps below for global instructions, then continue past them to see differences with the mini and the 1941 boards.

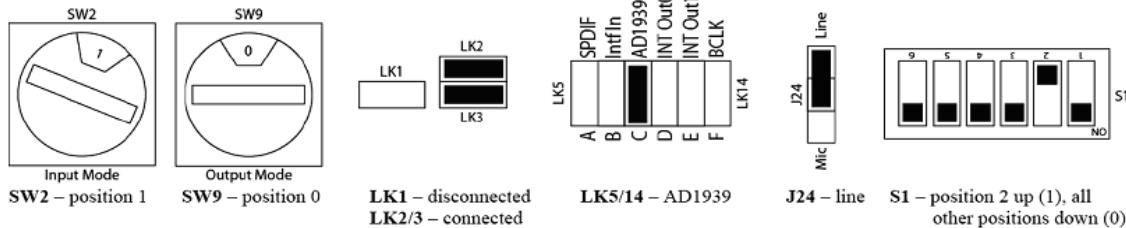
## **ADAU1701/ADAU1702**

The ADAU1701 board provides for both analog and digital communication.

---

**AD1940 (only) Evaluation Board Hardware Setup**, 6-channel Analog Input, Analog Output  
(*For further help finding switches/jumpers on the board, or for other setup i/o configurations, see the **EVAL-AD1941-AD1941EB\_Datasheet** on the SigmaStudio CD and the Evaluation Board Setup topic.*)

The switches and their settings:



Any setting not listed may be considered unimportant ("don't care").

### *Signal flow*

Input will come from the six-channel analog inputs to AD1940 SDATA\_IN0/1/2 and will output on whatever outputs (SDATA\_OUTx) are designed in the software; defaults are SDATA\_OUT0/1/2, or Analog Output 0-5. The AD1871 ADC is the master in this setup.

### *Orientation*

With the eval board's power connector in the upper-left corner:

- Input is connected to the stereo RCA connectors at the top of the board.
- Output is connected to any of the 4 stereo or 8 RCA connectors at the right side of the board.
- Signal path must be designed in Sigma Studio to send output to the desired port.

**AD1940 MINIB Evaluation Board Hardware Setup**, Stereo Analog Input, 6-Channel Analog Output

(To operate this board in standalone mode, connect the +6V power supply to the power connector, labeled J14.)

- 1) Attach the AD1940 MINIB board to the USB adapter board (J2 on both boards) and connect a USB cable between the USB board and the computer.
- 2) Connect your audio source to the 1/8" (mini) audio input jack (J1).
- 3) Connect your speakers or headphones to the 1/8" audio output 1 jack (J4).

- 4) Set switch S2 to match the maximum level of your input source (either 1 or 2 Vrms).
  - 5) Start SigmaStudio and follow steps 2-14 above.
- 

### **ADAU1701 Evaluation Board Setup**

#### *Powering up the board*

The board can be powered by either the USB connection or the included power supply. For the board to run independently from the computer, connect the power supply at J14 (colored red in the next figure). The power LED should be lit.

#### *Connecting the board to the computer*

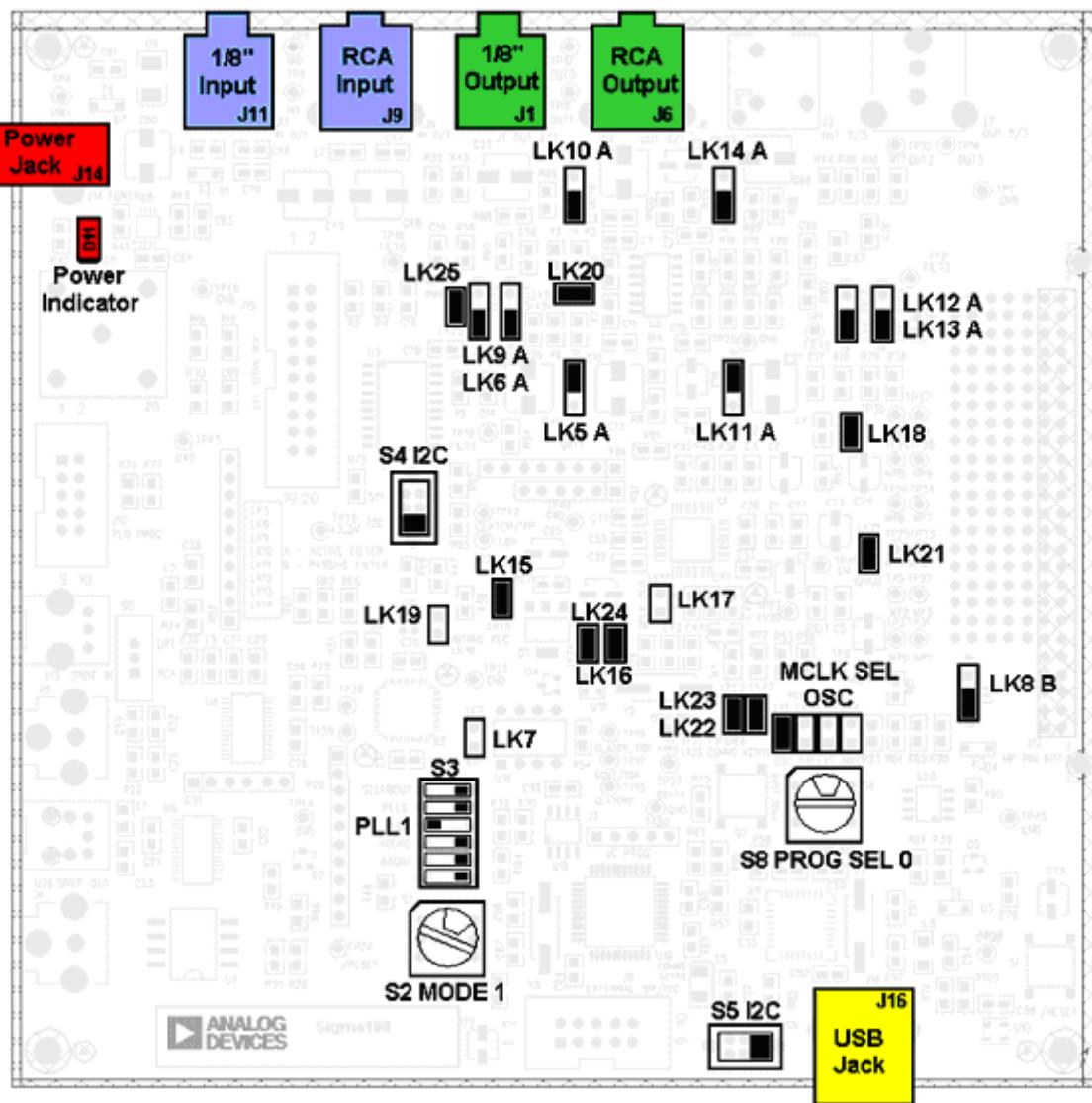
- 1) Use the included USB cable to make a connection between the evaluation board and an available USB port on your computer. The USB jack is yellow in the next figure.
- 2) Windows should recognize the device, and **Found New Hardware Wizard** should display.
- 3) Select "Install from a list or specific location (Advanced)" and then "Search for the best driver in these locations." Ensure that "Search removable media" is unchecked and "Include this location in the search" is checked.
- 4) Browse to find the USB drivers folder. (By default, it should be at **C:\Program Files\Analog Devices Inc\Sigma Studio\USB Drivers**.) Click OK and Next to begin the installation. (Click **Continue Anyway** if you're prompted with "the software hasn't passed Windows Logo testing.") Click Finish when done.

#### *Connecting the audio cables: Stereo analog inputs and outputs.*

- 1) Connect the audio source to the IN 0/1 jacks (blue in the below figure) on top of the board, using either RCA or 1/8" cables.
- 2) Connect the OUT 0/1 jacks (green below) to your speakers or headphones.

#### *Switch and jumper settings*

To configure the board for stereo analog in and out, make sure the switches and jumpers are set as shown below.





## AD1940/1 rev A Evaluation Board Setup

Depending on your application, different configurations can be set on this evaluation board. Following are examples of three typical input/output configurations:

### S/PDIF Input, Analog Output

- SW2 & SW9 – position 0
- LK6/14 – DIR
- SW4 – 1 and 7 left (1), 2-6 & 8 right (0)
- SW3 – 4 left (1), 1-3 & 5-6 right (0)
- SW10 – "Coax" or "Optical"; depends on source
- LK2 – left (0)
- LK1/3/10/11/13 – all in left jumper position
- The remaining switches' and jumpers' settings may be considered unimportant ("don't care").

Input will come from the S/PDIF receiver to AD1940 SDATA\_IN0 and will output on whichever outputs (SDATA\_OUTx) are designated in the software. Default output for the input is SDATA\_OUT0, or Analog Output 0/1.

### Analog Input, Analog Output

- SW2 – position 4
- SW9 – position 0
- LK6/14 – OSC
- LK12 – right (enable)
- SW4 – 1 and 7 left (1), 2-6 & 8 right (0)
- SW3 – 4 left (1), 1-3 & 5-6 right (0)
- LK2 – left (0)
- LK1/3/10/11/13 – all in left jumper position
- J21/22 – The bottom 2 of each column should be jumpered together.
- The remaining switches' and jumpers' settings may be considered unimportant ("don't care").

Input will come from the six-channel analog inputs to AD1940 SDATA\_IN0/1/2 and will output on whatever outputs (SDATA\_OUTx) are designed in the software. Default output for the input is SDATA\_OUT0/1/2, or Analog Output 0-5.

### I<sup>2</sup>S Input, I<sup>2</sup>S Output

- SW2 – position 2
- SW9 – position 0
- LK6/14 – INTF-In
- LK12 – right (enable)
- SW4 – 1 & 7 left (1), 2-6 & 8 right (0)
- SW3 – 4 left (1), 1-3 & 5-6 right (0)
- LK2 – left (0)
- LK1/3/10/11/13 – all in left jumper position
- SW6 – up (in) SW7 – right (out)
- SW8 – left (out)
- The remaining switches' and jumpers' settings may be considered unimportant ("don't care").

Input will come from the external interface inputs (J11) to AD1940 SDATA\_IN0/1/2/3 and will output on whatever outputs (SDATA\_OUTx) are designated in the software. Default output for the input is SDATA\_OUT0/1/2/3 (J7). If MCLK is to be output on J7, a second jumper needs to be placed on LK6/14 at INTF-Out0 in parallel with the jumper already on INTF-In.

## AD1940 rev B Evaluation Board Setup

Depending on your application, different configurations can be set on the evaluation board. Following are examples of three typical input/output configurations:

### Six-Channel Analog Input/Analog Output

Input will come from the six-channel analog inputs to AD1940/AD1941 pins SDATA\_IN0/1/2 and will output on whichever outputs (SDATA\_OUTx) are designated in the SigmaStudio software. Default output is SDATA\_OUT0/1/2 or analog output 0-5. In this setup, the master clock is generated by the AD1939's oscillator connected to a 12.288-MHz crystal. The AD1871 will be the serial-data clock master and the AD1939 and AD1940/AD1941's serial-data ports will be the slave.

The following list shows how the switches and jumpers need to be set up for this mode; any setting not listed may be considered unimportant ("don't care").

- SW2 - Position 1
- SW9 - Position 0
- LK1 - Disconnected
- LK2/3 - Connected
- LK5/14 - AD1939
- J24 - Line
- S1 - Position 2 up (1), all other positions down (0)

### S/PDIF Input/Analog & S/PDIF Output

Input will come from the stereo S/PDIF receiver to AD1940/AD1941 pin SDATA\_IN0 and will output on whichever outputs (SDATA\_OUTx) are designated in the SigmaStudio software.

Default output is SDATA\_OUT0 or analog outputs 0-1. SDATA\_OUT0 will also be sent to the S/PDIF transmitter. In this setup, the master clock and serial-data clocks are generated by the CS8414.

The following list shows how the switches and jumpers need to be set up for this mode; any setting not listed may be considered unimportant ("don't care").

- SW2 - Position 0
- SW9 - Position 0
- LK1 - Connected
- LK2/3 - Disconnected
- LK5/14 - S/PDIF
- SW10 - Selects S/PDIF source from either optical or RCA
- S1 - Position 2 up (1), all other positions down (0)

### I<sup>2</sup>S Input/ I<sup>2</sup>S Output

This mode is intended to be used to connect such external devices as DACs/ADCs or bit-reduced-audio decoders (e.g., AC-3). Input will come from the external input header (J11) and will feed AD1940/AD1941 pins SDATA\_IN0/1/2/3. Output will be on headers J7 and J8 on whichever outputs (SDATA\_OUTx) are designated in the SigmaStudio software. Default output is SDATA\_OUT0/1/2/3 or analog outputs 0-5. If the master clock is to be output on J7, then a second jumper needs to be placed on LK5/14 at the Intf-Out0 position. For the master clock to be output on J8, an additional jumper must be placed in position Intf-Out. In order to avoid multiple clocks driving the MCLK line, ensure that SW6-8 are set properly before placing these jumpers. In this setup, the master clock and serial-data clocks are

generated by the external source.

The following list shows how the switches and jumpers need to be set up for this mode; any setting not listed may be considered unimportant ("don't care").

- SW2 - Position 5
  - SW9 - Position 0
  - LK1 - Connected
  - LK2/3 - Disconnected
  - LK5/14 - Intf In
  - SW6 - In (down)
  - SW7 - Out (right) if the master clock is to be output on the I2S headers
  - SW8 - Out (left) if the master clock is to be output on the I2S headers
  - S1 - Position 2 up (1), all other positions down (0)
-

## **USB Interfaces**

### **USB Interfaces**

---

SigmaStudio can communicate with hardware in real over a USB connection. DSP Program Data, Register Settings, and Parameter control data is sent to a USB device, converted to either SPI (serial peripheral interface) data or I<sup>2</sup>C data, and sent to the SigmaDSP hardware. It is also possible to read hardware register values via USB.

When using an evaluation board with integrated USB connectors, locate the Communication Channel block with the corresponding SigmaDSP part number. For example, **EvalBoard170x** should be used with the EVAL-ADAU1701EB and EVAL-ADAU1702EB platforms. Connect a USB cable directly to the evaluation hardware and to the PC.

There are also two generic communication channels available:

1. USB Serial Converter (EVAL-ADUSB1)
2. USBi (EVAL-ADUSB2)

See your evaluation hardware's data sheet for more information, [analog.com/sigmadsp](http://analog.com/sigmadsp).

---

### **USB Driver Installation:**

The proper USB drivers must be installed to communicate with SigmaStudio. To setup the USB hardware:

1. Connect the USB cable to your evaluation hardware and to a spare USB port on your PC.
2. The Windows "Found New Hardware Wizard" will launch.
3. Choose "Install from a list or a specific location" and click "Next".
4. Select "Search for the best driver in these locations" and Check the box for "Include this location in the search."
5. Press the "Browse" button and locate the appropriate driver file in the SigmaStudio application folder (default folder is C:\Program Files\Analog Devices Inc\Sigma Studio\USB drivers):
  1. For **USB serial converter** and Eval-Boards (FTDIxx.inf) select **ftd2xx.inf** file.
  1. For **USBi** interface select the **CyUSB.inf** file.
6. Click "Continue Anyway" if you're prompted with "This software has not passed Windows Logo testing."

---

### **To enable USB communication in SigmaStudio:**

1. Select the Hardware Configuration Tab.
2. Click the **Communication Channels** category (at the bottom of the **ToolBox** window).
3. Select your evaluation board, USBSerialConv or USBi and drag and drop it into the Configuration window.
4. Connect the Communication Channel to the DSP IC processor block(s).



The color of the USB label on the communication block indicates if a USB connection has been established. If the USB hardware is properly configured, the background color will be light orange or white. If the communication is not properly initialized the background will be red. Note that this only indicates that a USB connection is active, it does not guarantee communication with the SigmaDSP IC or that the SigmaDSP hardware is properly configured.



The "Connected" background color is white when all board ICs are connected, and orange when only some ICs are connected, but not all. For example the ADAU1701 evaluation board includes both an ADAU1701 IC and an E2Prom IC. When connecting only the ADAU1701 the background will be light orange, but it will be white when connecting both an ADAU1701 and an E2Prom IC.



## USB Serial Converter (EVAL-ADUSB1)

---

The Analog Devices USB Serial Converter (EVAL-ADUSB1) board is the interface between your PC's USB port and the evaluation board control-port connections. In the case of the AD1953, -1954, -1940, and -1941 boards, the USB serial converter connects directly to the DB25 connector on the board.

The USB adapter is typically powered from the computer's USB port. To use the adapter in standalone mode, power it by connecting +5Vdc and ground to test points TP1 and TP2.

---

### To Install the USB Serial Converter:

1. Plug the adapter into the evaluation board's control port (DB25 connector).
2. Connect the USB cable to your PC and to the USB adapter.
3. When prompted for drivers:

#### Windows 2000

- Choose "Search for a suitable driver" and click next.
- Check "Specify a location" and uncheck the other options. Click Next.

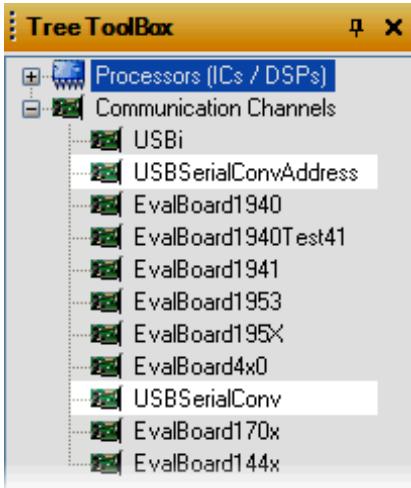
#### Windows XP

- Choose "Install from a list or a specific location."
- Choose "Search for the best driver in these locations."
- Check the box for "Include this location in the search."

4. Click Browse and find the **ftd2xx.inf** file in the USB driver folder in the SigmaStudio installation directory. The default location is **C:\Program Files\Analog Devices Inc\SigmaStudio\USB drivers**.
  5. In Windows XP, click *Continue Anyway* if you're presented with a dialog regarding the software not passing Windows Logo testing.
- 

### Using the USB Serial Converter:

To communication between SigmaStudio and the USB Serial Converter board, a communication channel block must be added to the Schematic design. To locate these blocks, select the Hardware Configuration Tab and in the ToolBox or Tree ToolBox window choose the "Communication Channels" category.



There are 2 communication channel blocks that can be used with the USB Serial Converter board: USBSerialConv and USBSerialConvAddress. Both blocks offer connections for multiple processors, allowing you to use multiple processors in a single design.

**Note:** Only one of the USBSerialConv/USBSerialConvAddress block output pins must be connected. It is not necessary to connect or terminate all of the output pins. Also, you connect the processor to any of the pins, there is no specific requirement for order of pin connection.

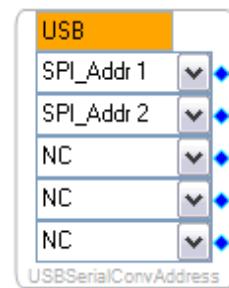
### 1. USBSerialConv

Allows connection of multiple DSP processors blocks and the E2Prom block. Addresses are assigned sequentially from top to bottom (e.g. SPI communication). The pins are associated with clatch0 - 4 from top to bottom).



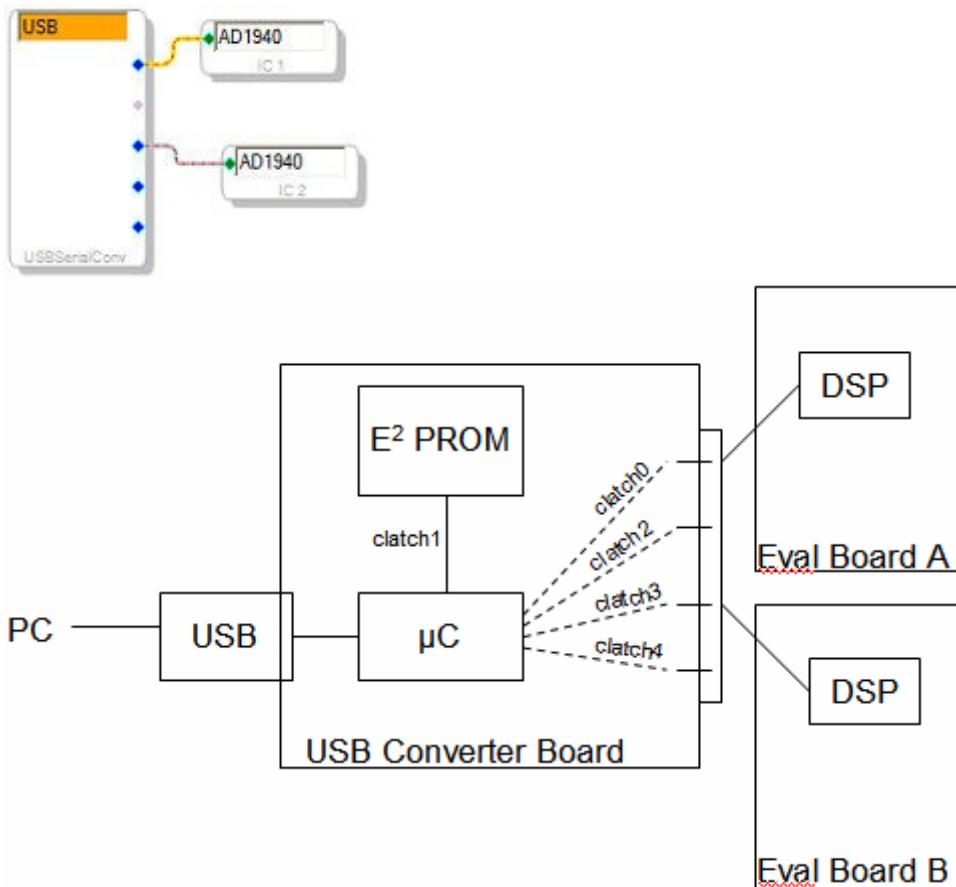
### 2. USBSerialConvAddress

This block allows you to explicitly specify the part addresses, I<sup>2</sup>C address or SPI clatch line (see below). Note that the DSP hardware's address must match the block's selected address for



communication to function. See the parts data sheet for more information about addressing (e.g. ADR\_SEL, ADDR0, or ADDR1 pins).

Depending on the SigmaDSP part, connections made to this block will be for either SPI (serial peripheral interface) data or I<sup>2</sup>C data. For SPI, this channel lets you connect multiple boards at once, with the option of connecting a particular board to any clatch line. (The only exception is clatch1, which is reserved for the converter EPROM, represented on the USBSerialConv block by a grayed pin.) The pins are clatch0 - 4 from top to bottom. The diagram below shows one possible configuration of connecting boards to pins.



#### EVAL-ADUSB1 board switches and jumpers:

- S1 - Load a program saved in flash to the SigmaDSP.
- S2 - Select which of eight SigmaDSP programs will be loaded when S1 is pressed.
- S3 - Reset the USB adapter board.
- TP1 - +5Vdc connection for cases where the adapter isn't powered through the PC's

USB port.

- TP2 - Ground connection for cases where the adapter isn't powered through the PC's USB port.
  - J3 - When a jumper is present on this header, writing is enabled to the flash memory (for storing SigmaDSP programs).
- D4 - This LED indicates the USB board is powered.

The SigmaDSP program and parameter files can be saved to flash memory on the USB board using the Flash Downloader tool in SigmaStudio. Each program can be loaded to the SigmaDSP by setting S2 to the appropriate setting and then pressing the program load button, S1. Refer to the Flash Downloader page.

See the EVAL-ADUSB1 data sheet for more information, [analog.com/sigmadsp](http://analog.com/sigmadsp).

---

## USBi (EVAL-ADUSB2)

---

The Analog Devices USBi "USB Interface" (EVAL-ADUSB2) board is the interface between your PC's USB port and SigmaDSP hardware's data input pins. This interface can be used with any evaluation board which includes an External SPI/I<sup>2</sup>C header (Aardvark Header). The EVAL-ADUSB2 board is capable of both SPI and I<sup>2</sup>C communication (which is user selectable) and can supply IOVDD of either 3.3V or 1.8V.

The USBi interface is powered from the computer's USB port.

---

### To Install the USBi Board:

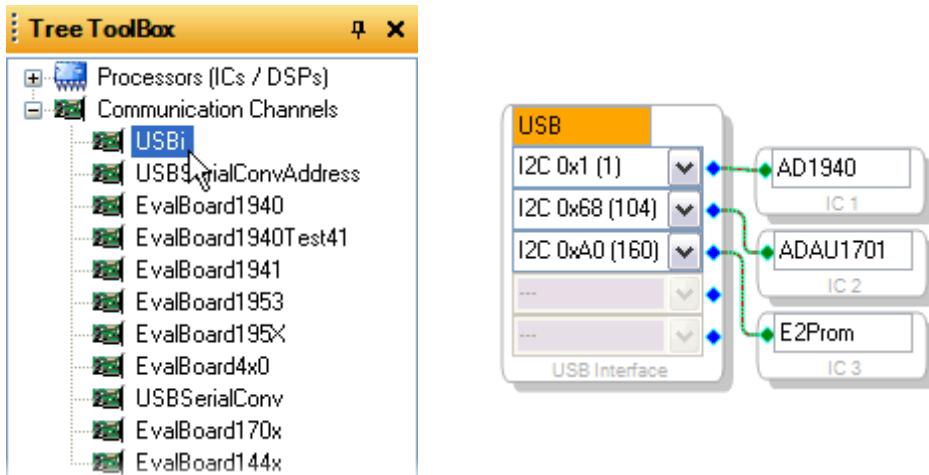
1. Connect the USB cable to the USBi Board a spare USB port on your PC.
2. The Windows "Found New Hardware Wizard" will launch.
3. Choose "Install from a list or a specific location" and click "Next".
4. Select "Search for the best driver in these locations" and Check the box for "Include this location in the search."
5. Press the "Browse" button and locate the appropriate driver file in the SigmaStudio application folder (default folder is C:\Program Files\Analog Devices Inc\Sigma Studio\USB drivers): Select the **CyUSB.inf** file.
6. Click "Continue Anyway" if you're prompted with "This software has not passed Windows Logo testing."

**Note:** The first time the USBi board drivers are installed on a PC, you will be prompted to repeat the above steps a second time, for the "Analog Devices USBi (unloaded)" device. This is normal and occurs because the USBi board's firmware is updated during the first driver installation operation.

---

### Using USBi in SigmaStudio:

To communication between SigmaStudio and the USBi board, a communication channel block must be added to the Schematic design. To locate the block, select the Hardware Configuration Tab, int the ToolBox or Tree ToolBox window choose the "Communication Channels" category, and select the USBi block.



**Note:** Only one of this block's output pins must be connected. It is not necessary to connect or terminate all of the output pins. Also, you connect a processor to any of the pins, there is no specific requirement for pin connection order.

This block provides connections for multiple processors (ICs / DSPs), allowing you to use multiple processors in a single design. This block allows you to explicitly specify both the communication protocol\*\* (SPI or I<sup>2</sup>C) and the part's address, I<sup>2</sup>C address or SPI clatch line. The DSP hardware's address must match the block's selected address for communication to function properly. See the parts data sheet for more information about addressing (e.g. ADR\_SEL, ADDR0, or ADDR1 pins).

\*\*The default USBi protocol is I<sup>2</sup>C, to enable SPI, you must first select an SPI address in the block's drop down menu and then perform three write operations (e.g. press the write button the Register Read/Write Window 3 times). The USBi hardware has LEDs which indicate the current mode, I<sup>2</sup>C or SPI.

See the EVAL-ADUSB2 data sheet for more information, [analog.com/sigmadsp](http://analog.com/sigmadsp).

# Development Environment

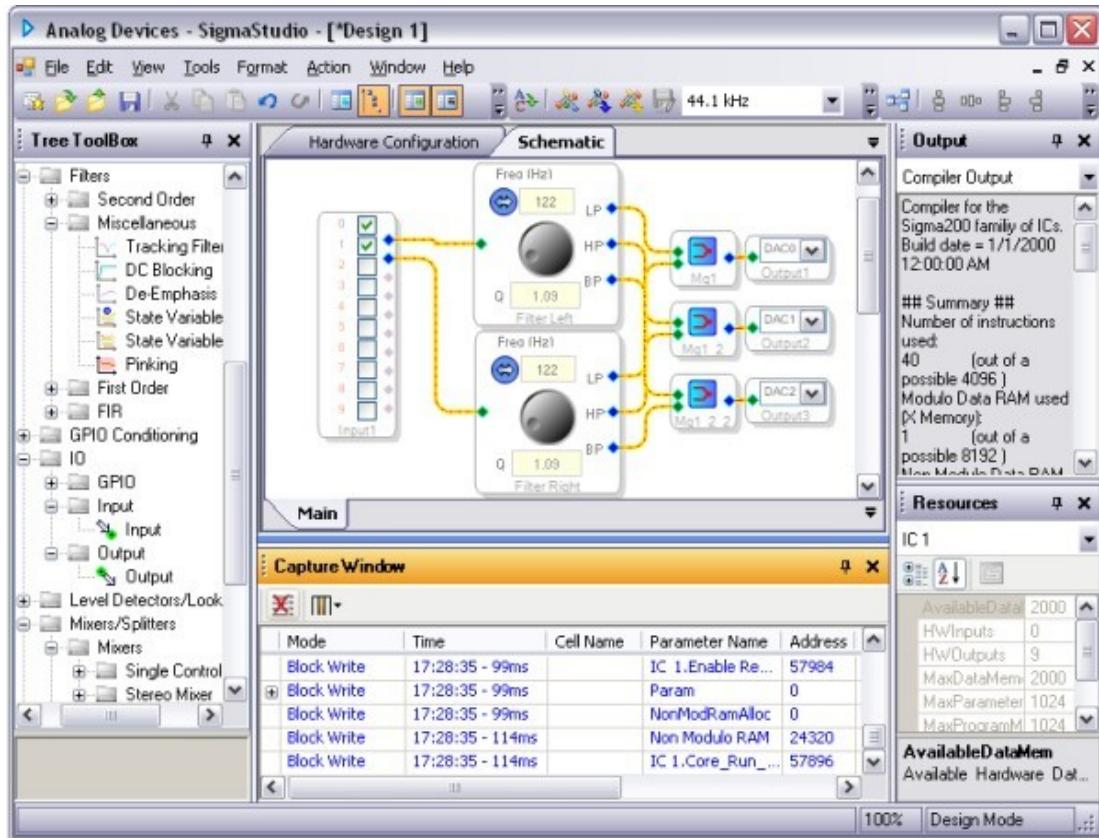
## Development Environment

The SigmaStudio development environment provides a graphical user interface to help you build and deploy signal processing systems quickly and efficiently. Refer to the following sections if you have any problems or questions while working with the application.

Program Window	Summary of the Main Screen and its tools.
Hardware Configuration Tab	Choose a DSP IC for your design or setup USB communication.
Schematic Tab	Use the Schematic workspace to create your DSP design.
Toolbox	The basic building blocks to construct your design.
Workspace Windows	Workspace windows display important project information.
Toolbars	Tool buttons for invoking application commands.
Dialogs	Summary of the SigmaStudio dialogs.
Hardware Windows	Access hardware register settings and download data.

**Note:** The first time you start SigmaStudio, it displays the Program Window with the default layout. You can change the desktop arrangement to meet your needs and preferences, including resizing, moving, and hiding windows and toolbars. For more details, see Arranging the Workspace.

## Development Environment

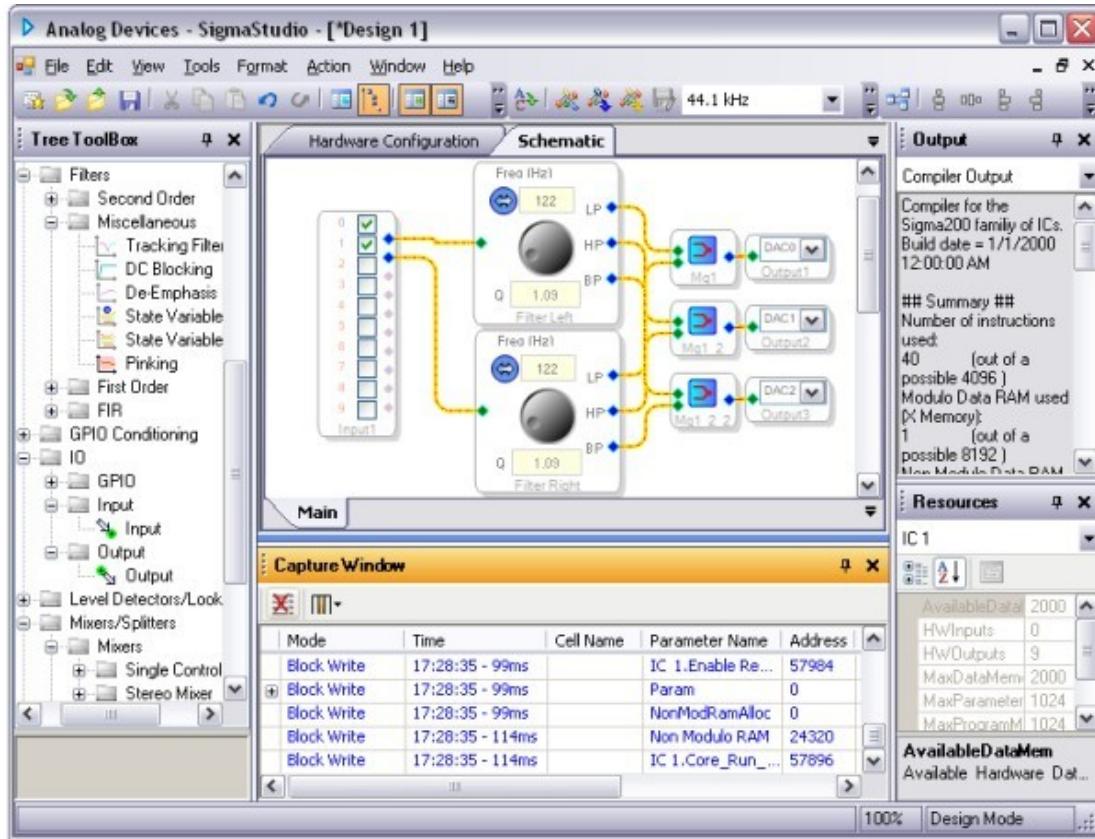


© 2006-2007 Analog Devices, Inc. All rights reserved.

## Program Window

When you start SigmaStudio, you will see the main Program Window. To get started, create a New Project by selecting **File - New Project** from the main menu or by pressing the new project button on the toolbar, or open an existing project file.

The following screen will appear for a new project:



The Program Window contains the main menu, toolbars, and the design workspace. The default design window is the Hardware Configuration tab and its toolbox to the left. This workspace lets you choose which DSP or IC to use.

SigmaStudio is an MDI application (meaning more than one project file can be open at a time), but only a single instance of the SigmaStudio application can be running at any time. To toggle between open projects, select **Window** in the application main menu or press **ALT + TAB**.

**Note:** You must select a "Processor" and insert it into the Hardware Configuration tab before you begin a design. See the Hardware Configuration Tab topic for more information.

### Project Files

SigmaStudio projects are stored in files with a **.dspproj** extension. These files can be stored anywhere on the hard drive as well as removable or network media. In addition to project files,

folders are automatically created for the compiled output of a project and for a projects settings. For more information see the settings topic and the Link/Compile/Download topic.

### Arranging The Workspace

It is possible to modify the workspace configuration to best meet your needs.

1. Workspace windows can be docked around the perimeter of the main window, grouped together, or undocked/floated and moved anywhere on the screen. To toggle the docked state of a window, double-click the window's title bar or right click in the title bar. When a window is undocked/floating, dragging the window within the main program will display an outline of the location where it will be docked with dropped.  
If you wish to position a floating window within the workspace without docking, click and hold the left mouse button in the title bar and then hold down the CTRL key.
2. Windows can be configured to automatically hide or "shrink" to the perimeter of the application until needed. Press the auto-hide button  to toggle this mode.
3. Each pane can be resized using the splitters around the perimeter of each pane.
4. Windows can be closed by clicking the close button displayed in the upper right corner, or by right-clicking the title bar and selecting "Hide". To display a closed window, select in the View menu or use the buttons in the Standard Toolbar.

**Note:** Workspace settings are stored in the "ToolbarLayout.dat" file which is located in the application's directory. Close SigmaStudio and delete this file and restart to restore the default layout.

## **Hardware Configuration Tab**

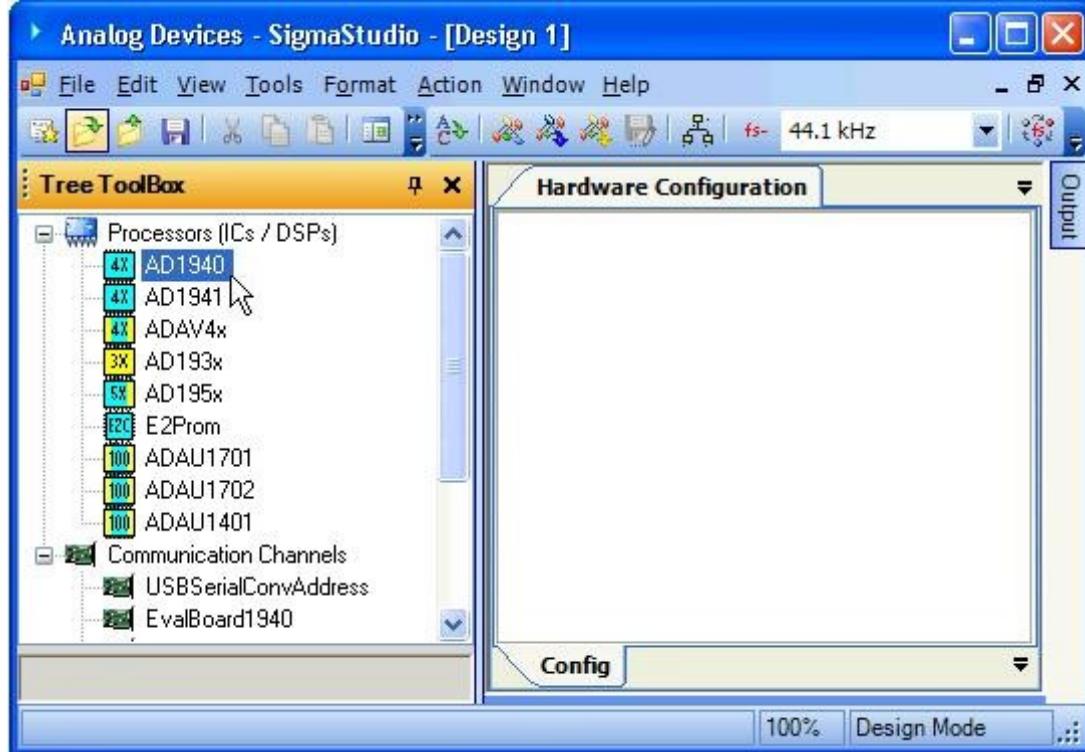
---

The Hardware Configuration workspace allows you to choose a Processor or Processors for your design. It also allows you to set up communication between SigmaStudio and the hardware.

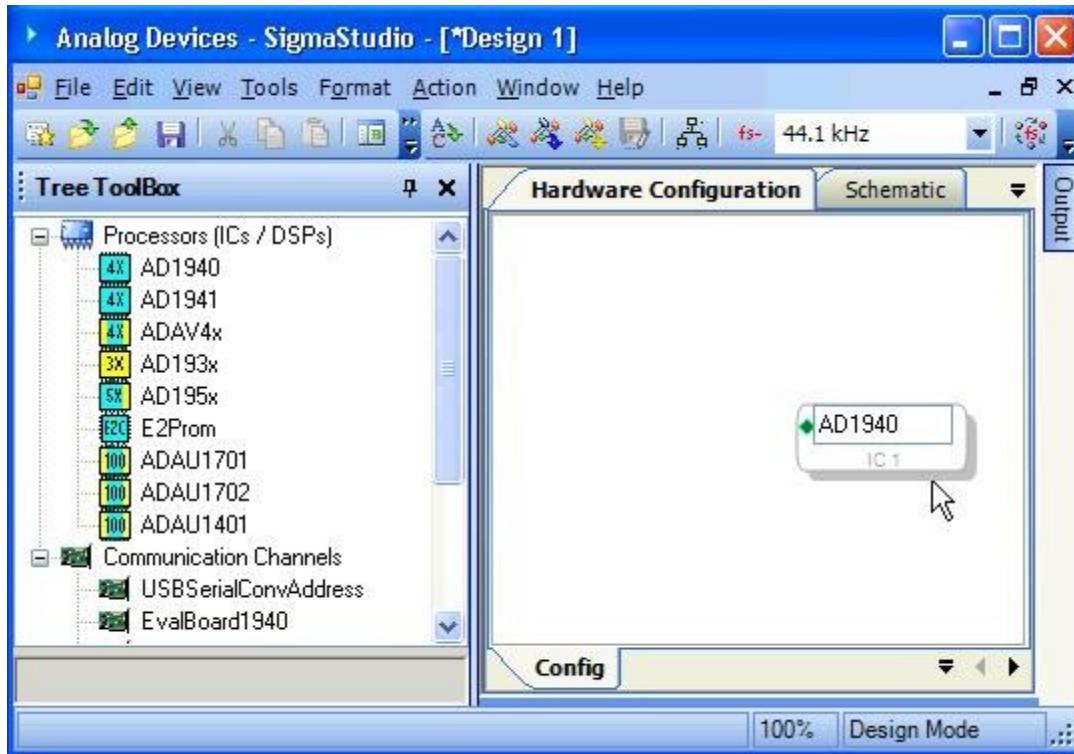
To learn about the advanced operations that can be accessed in the Hardware Configuration tab (including Output Capture, Flash/E2Prom download, Register Control Window, and the Register Read/Write Window), refer to the topics within the Hardware Windows section.

**To insert a processor (IC/DSP) into your design:**

1. Select a "Processor" from the Toolbox with the left mouse button:



2. Drag and Drop the processor block into the Hardware Configuration Window on the right:



**Note:** Notice that the Schematic Tab appears once you have inserted a DSP in the Hardware Configuration window.

#### To establish a connection between your DSP processor and hardware:

1. Click the **Communication Channels** category (at the bottom of the **Toolbox** column).
2. Select your evaluation board or USB device from the list and drag and drop it into the workspace.

**Note:** The **Communication Channels** menu lists the names with the prefix **EvalBoard** and the board number. These communication channels will work for the same IC types on platforms that don't use the evaluation-board setup. There's also generic communication channels, USBSerialConv and USBi. (See the USB Serial Converter Communication Channel or USBi for more information.)

3. Connect the two blocks by drawing a wire between the communication channel and the processor block, blue colored diamond to green diamond. For a USB connection using AD1940, the cells in your workspace should look something like this:



The color of the USB label on the communication block indicates whether a USB communication channel has been established. If you have properly configured the USB hardware, the background color will be light orange or white. If the communication is not initialized the background will be red. Note that this only indicates that a USB connection is active, it does not guarantee communication.

## Development Environment

with the SigmaDSP IC or that the SigmaDSP hardware is properly configured.



The "Connected" background color is white when all board ICs are connected, and orange when only some ICs are connected, but not all. For example the ADAU1701 evaluation board includes both an ADAU1701 IC and an E2Prom IC. When connecting only the ADAU1701 the background will be light orange, but it will be white when connecting both an ADAU1701 and an E2Prom IC.



---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Schematic Tab

Once the hardware configuration is set, you are ready to construct a DSP design. Designs are created in the Schematic tab workspace (Figure 1). The Toolbox at the left gives access to various libraries containing building blocks that can be dragged and dropped in the schematic workspace on the right.

**Note:** The Schematic Tab is not displayed until a programmable DSP IC is inserted in the Hardware Configuration tab.

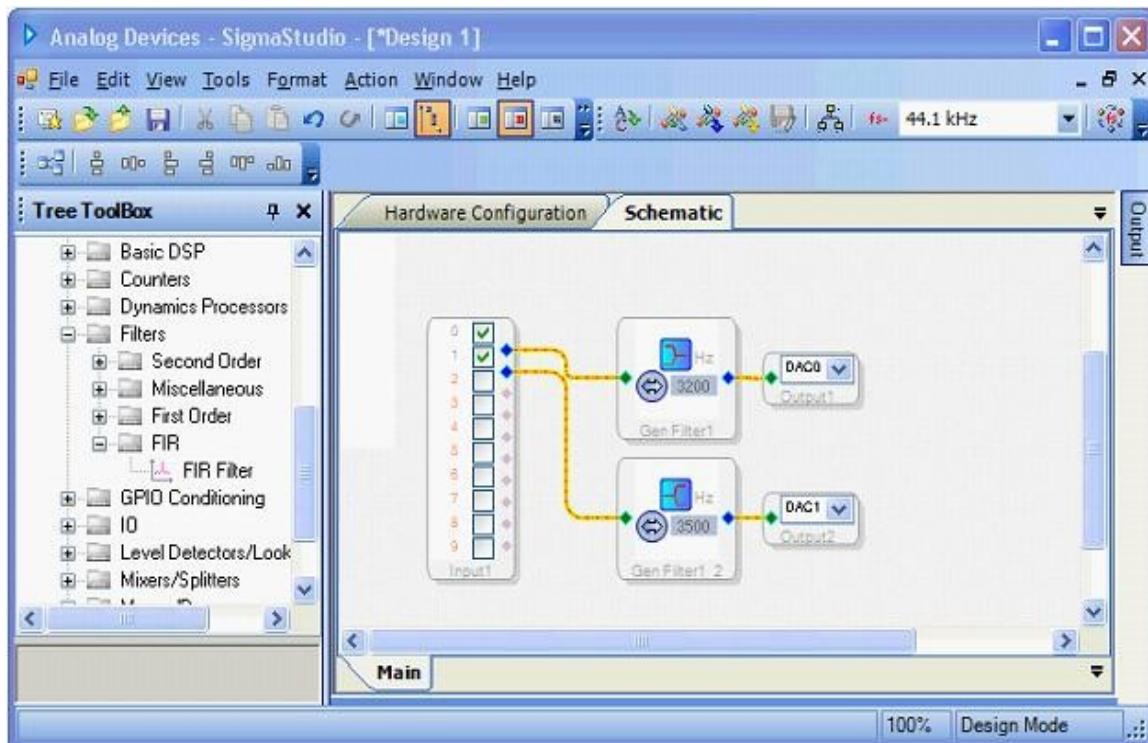


Figure1

Schematic designs can be organized into multiple layers called **Hierarchy Boards**. The design example in Figure 1 has a single hierarchy layer called 'Main', as shown in the tab at the bottom of the screen. 'Main' is the default layer created for all designs and cannot be renamed. For more information on Hierarchy Boards and how to organize your program flow into board layers see the Hierarchy Board topic.

### Where to go now

If you're ready to begin designing, refer to the following sections for more information:

- To set up a simple application, skip to Tutorials and follow the examples for Stereo Audio with Volume Control and 5-band EQ and Probe and Stimulus Blocks.

- Learn about individual blocks and their functions in the Schematic Toolbox section, example schematics, and get detailed information about the algorithms represented by the blocks.
- For advanced operations and control of the hardware, take a look at the Hardware Windows section.
- Learn more about general SigmaStudio functions in Using SigmaStudio, and see detailed information about microcontroller considerations in Basic SigmaDSP Architecture.

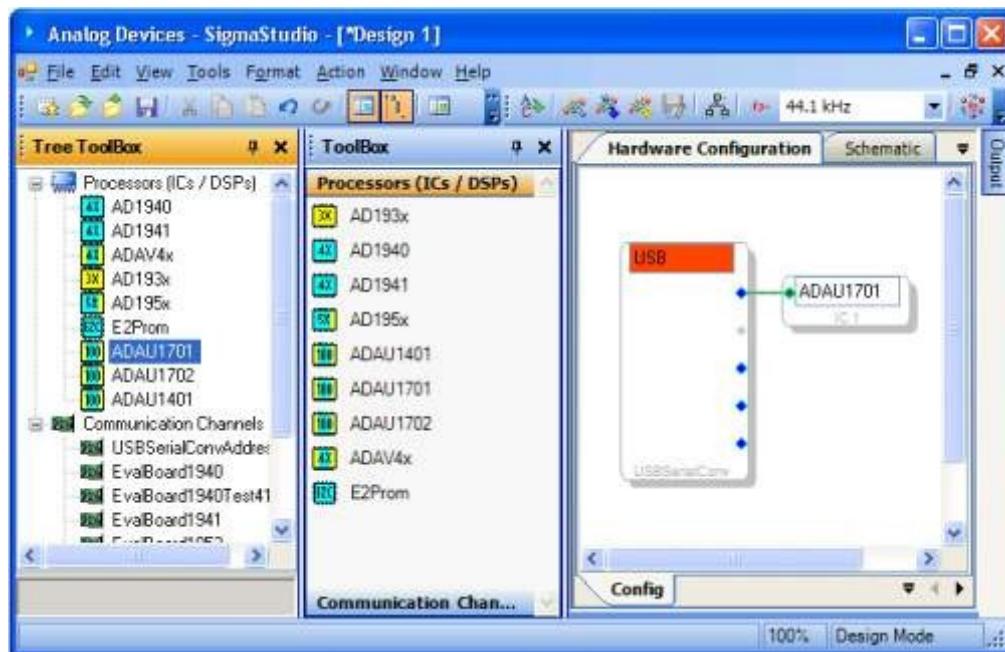
---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Tree Toolbox and Toolbox Windows

The Tree Toolbox and Toolbox windows contain the building blocks for constructing your design. For more information about the building blocks available in SigmaStudio and their usage, see the ToolBox topic.

To show or hide the Toolbox windows select "Tree ToolBox" or "ToolBox" in the main application's **View** menu or by clicking the buttons on the Standard Toolbar. By default these windows are docked on the left side of the program window.



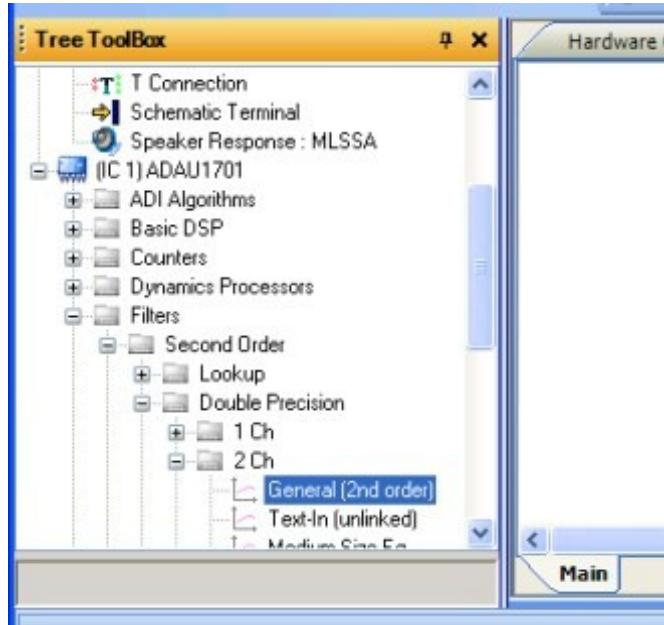
Select the Hardware Configuration tab to display Processor blocks (ICs and DSPs) and USB communication blocks. Selecting the Schematic tab will display system and DSP building blocks.

**Note:** The "Tree ToolBox" and "ToolBox" windows offer the same functionality and building blocks, but present the data differently. While it is recommended that you first try the Tree Toolbox, as it is designed to simplify the design process, you should choose the toolbox window that best meets your needs and preferences.

---

### Tree ToolBox

The Tree Toolbox window displays the available building blocks in a tree view. In the example below, the project contains is an *ADAU1701* DSP named *IC1*. System blocks are arranged in a hierarchy of folders according to function and algorithm. Navigate the folders to find a desired algorithm block and drag and drop it into the schematic window.



---

### Toolbox

The Toolbox window is organized into a series or Tabs representing the different algorithm categories. Click a library topic to expand the list and see the blocks you can drop into your schematic. The available blocks depend on which IC(s) or DSP(s) are inserted in the Hardware Configuration tab.



Unlike the "Tree Toolbox", Toolbox blocks can represent more than one algorithm at a time (for example single or double precision filters) requiring an additional step of algorithm selection after dropping the block in the schematic. See the Adding and Growing Algorithms for more information.

---

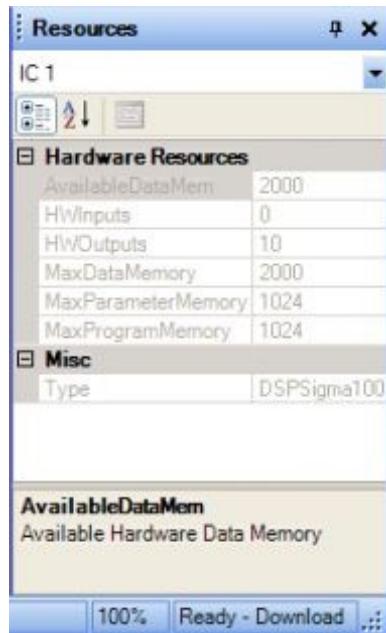
© 2006-2007 Analog Devices, Inc. All rights reserved.

## Workspace Windows

The Workspace windows display important project information including available resources, compiler output, and hardware communication. To show or hide the workspace windows select them in the application's main **View** menu or by clicking the buttons on the Standard Toolbar.

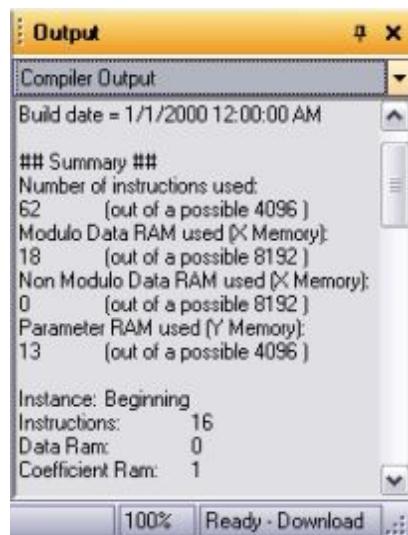
- **DSP Resources**
- **Output**
- **Capture Window**

**DSP Resources:** The DSP Resources window lets you view the resources available in your design, depending on your project complexity. You are able to monitor program RAM, parameter RAM, and other hardware-specific resources. By default this window is docked on the right side of the program window.



**Output:** The Output window is a quick way to view the files generated upon linking, compiling, and other actions in the Schematic workspace. By default this window is docked on the right side of the program window.

Note, you can also find this information by locating the project's output folder and reading the compiler output files. For more information about those files, see [Compile/Downloading Your Project](#).



**Note:** The DSP Resources and Output windows are not fully functional for all SigmaDSP IC types.

---

**Capture Window:** The capture window lets you see, in real time, the exact data that SigmaStudio is sending to the hardware. This window is only active when a USB communication channel is inserted in the project and the project has been compiled and downloaded. This window is un-docked/floating by default.

The window is empty when it opens initially, but after you press Link-Compile-Download you will see the data that gets downloaded to the board: coefficients, parameter addresses, and parameter data. Whenever you make adjustments to sliders or control in a compiled and downloaded schematic, you can see the data that is sent.

Mode	Time	Cell Name	Parameter Name	Ad...	Value	Data	Bytes
Block Write	15:57:5 - 587ms		IC 1.CoreRegister	2076		0X00 ,0X18	2
+ Block Write	15:57:5 - 603ms		Program Data	1024		0X00 ,0X00 ,0X00 ,0X00 ...	5120
+ Block Write	15:57:5 - 634ms		Param	0		0X00 ,0X80 ,0X00 ,0X00	4096
+ Block Write	15:57:5 - 665ms		IC 1.HWConFig ...	2076		0X00 ,0X18 ,0X08 ,0X00	24
						0X00 ,0X00 ,0X00 ,0X00	
						0X00 ,0X00 ,0X00 ,0X00	
						0X00 ,0X00 ,0X00 ,0X00	
						0X80 ,0X00 ,0X00 ,0X00	
						0X00 ,0X00 ,0X00 ,0X01	
Block Write	15:57:5 - 665ms		IC 1.CoreRegister	2076		0X00 ,0X1C	2
Safeload Write	15:57:40 - 509ms	Gen Filter1	EQ1940Dual10B1	0	1	0X00 ,0X80 ,0X00 ,0X00	4
Safeload Write	15:57:40 - 509ms	Gen Filter1	EQ1940Dual11B1	1	-1.88495...	0xFF ,0XE ,0XB9 ,0XB9	4
Safeload Write	15:57:40 - 509ms	Gen Filter1	EQ1940Dual12B1	2	0.90421...	0X00 ,0X73 ,0XBD ,0X42	4

- ① "Clear All Output Data" button deletes the current content of the Capture Window.
- ② "Show Columns" button allows you to customize which capture data is displayed.
- ③ The column headers can be dragged or re-ordered as you prefer. Also, right click on the column names to access the show and hide context menu.
- ④ The plus/minus buttons allow you to show and hide the Data for a particular read or write message.
- ⑤ **Right-Click** in the capture window to display the context menu, you can copy the currently selected data to the clipboard or save the selected data to a file.

See Capture Output Data for a detailed explanation of the data displayed in the Capture Window.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

---

## Toolbars

---

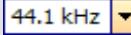
The application toolbars provide shortcut buttons for common commands. Three toolbars are displayed in the main program window by default. To hide or show a toolbar, right-click on the menu bar, then select the toolbar to be shown or hidden.

**Standard Toolbar:**

	<b>New Project</b> , Creates a new project file
	<b>Open</b> , Open an existing project file (*.dspproj)
	<b>Close Project</b> , Close the current project
	<b>Save</b> , Save the current project
	<b>Cut</b> , Cuts the selection (to the Windows Clipboard)
	<b>Copy</b> , Copy the selection (to the Windows Clipboard)
	<b>Paste</b> , Paste data from Windows Clipboard
	<b>Undo</b> , Reverses the last action, each click reverses one more action
	<b>Redo</b> , Reverses the last undo action
	<b>ToolBox</b> , Show or hide the Toolbox window
	<b>Tree ToolBox</b> , Show or hide the Tree Toolbox window
	<b>DSP Resources</b> , Show or hide the DSP Resources workspace window
	<b>Output</b> , Show or hide the Output workspace window
	<b>Capture Window</b> , Show or hide the Capture workspace window
	<b>Cascade</b> , Organize the project windows in a cascade with each project title bar visible
	<b>Tile Vertical Centers</b> , Organize the project window vertically so they do not overlap
	<b>Tile Horizontal Centers</b> , Organize the project window horizontally so they do not overlap

**Schematic Control Toolbar:**

	<b>Allow Realtime AB Testing</b> , enabled toggle among any open and compiled projects without having to recompile.
	<b>Link Project</b> , Link the project (without compiling) and display link information and errors
	<b>Link Compile Connect</b> , Link and compile the current project without hardware download
	<b>Link Compile Download</b> , Link and compile the current project and send your compiled program data to the DSP hardware
	<b>Export System Files</b> , Export program, address, and register data for system integration
	<b>Freeze Schematic</b> , Prevent changes or edits to the current project's schematic

	<b>Set System Sample Rate</b> , Set sampling rate for all inputs and sources in current project
	<b>New Item Sample Rate</b> , Sets the default sampling rate for new input and source blocks
	<b>Propagate Sampling Rate</b> , Apply any input and source sample rate changes

**Schematic Layout Toolbar:**

	
Button:	Function:
	<b>Select All</b> , Select all blocks in the schematic tab
	<b>Align Vertical Centers</b> , Align the vertical center of all selected blocks in the schematic
	<b>Align Horizontal Centers</b> , Align the horizontal center of all selected blocks in the schematic
	<b>Align Left Sides</b> , Align the left edges of all selected blocks in the schematic
	<b>Align Right Sides</b> , Align the right edges of all selected blocks in the schematic
	<b>Align Top</b> , Align the top edges of all selected blocks in the schematic
	<b>Align Bottoms</b> , Align the bottom edges of all selected blocks in the schematic

**Note:** You can create your own custom Toolbars, customize the default toolbars, customize the application menus, and change keyboard shortcuts by right-clicking in the menu bar or toolbar area and selecting **Customize...**

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Dialogs

### About Dialog

---

The About dialog is accessed by choosing **Help - About** from the main application menu. It displays the program version and copyright information. If you are reporting SigmaStudio issues to Analog Devices support, please include this version information with any written communication.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Add-Ins Browser Dialog

---

Additional Schematic blocks including both Algorithms and ICs can be added to the SigmaStudio environment through the Add-Ins Browser Dialog. Go to **Tools - Add-Ins Browser...** in the main application menu to access this dialog.

### To add a block:

- Click the **Add DLL** button on the Add-Ins dialog toolbar or select **File - Add DLL** in the dialog's menu.
- Navigate to the directory containing the SigmaStudio AddIn library file or files (\*.dll)
- Select the files to be added and press the **Open** button.
- Press the **Save** toolbar button to apply the changes.

### To enable/disable a block:

- Click the check box next to the file name.
- Press the **Save** toolbar button to apply the changes.

### To delete a block:

- Select the file to delete in the Add-Ins file list.
- Press the **Delete Item** toolbar button or choose **File - Delete Item** in the menu.
- Press the **Save** toolbar button to apply the changes.

Added blocks are immediately available for use, you do not need to restart SigmaStudio. You should see the additional block(s) in the SigmaStudio Toolbox after saving.

**Note:** Add-In information is saved in the "AddIns.xml" file located in the application's installation directory. You may want to back-up this file when installing SigmaStudio updates. Also, you can view/edit this file a text editor if you prefer.

## Filter Table Generator

---

The Filter Table Generator dialog is used to calculate filter coefficients and coefficient tables. Go to **Tools - Fixed-Point Filter Table Generator...** in the main application menu to access this dialog.

To use the generator, select a filter type tab on the left, set the filter design parameters and press the **Generate** button.

The table generator provides a variety of design option including filter type, center frequency, Q, gain and number of table steps. A graph of the filter frequency response is displayed as well as a list of filter coefficients, in both decimal and hex (parameter RAM-ready hexadecimal) format. The hex values are sign-extended from the fourth bit. To see the coefficient values in decimal format check the **Show Values** box in the **Value Table** tab and press the **Generate** button.

This tool is most useful for designing a set of filter coefficients with stepped gain response (as in the above image) for such applications as an graphic equalizer.

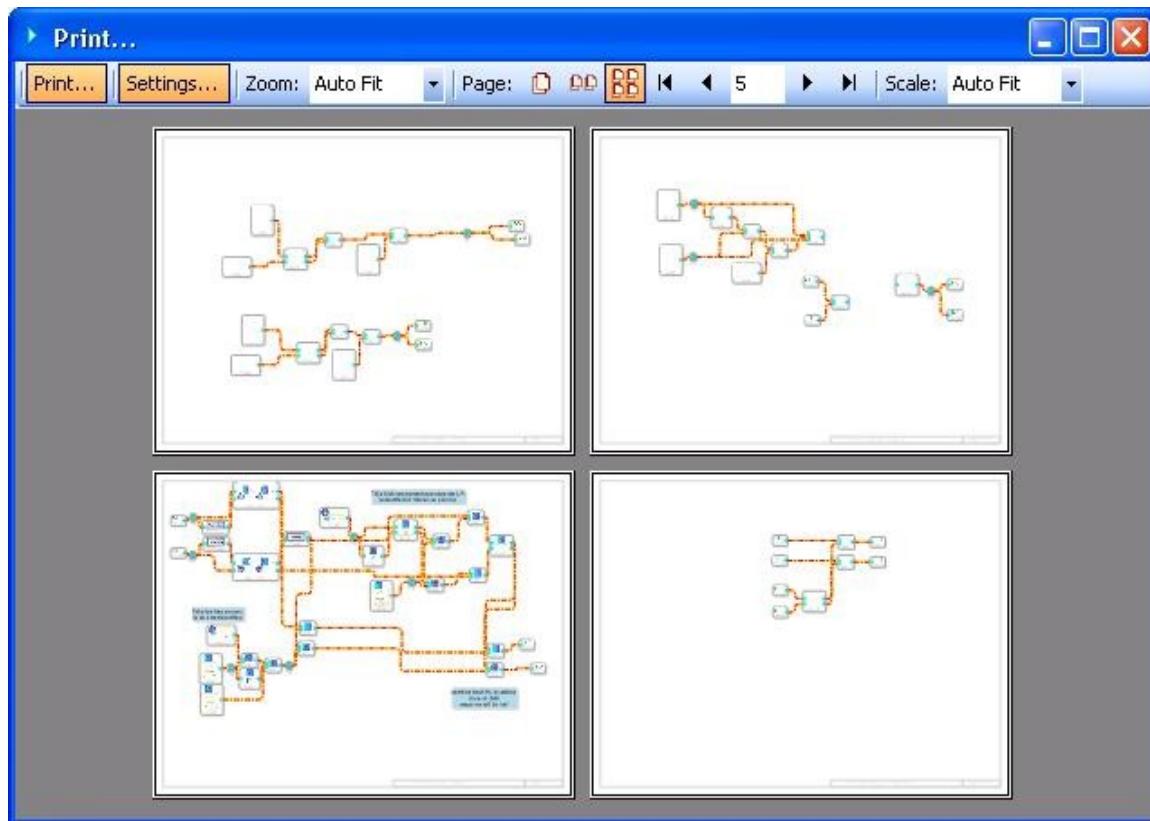
---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Print and Print Setup Dialogs

Printing and Print preview are provided for SigmaStudio projects (Schematic window only).

These functions are accessed from **File - Print...** on the main application menu or by pressing CTRL + P. Also, the printer settings can be configured in the print setup dialog (**File - Print Setup...**).



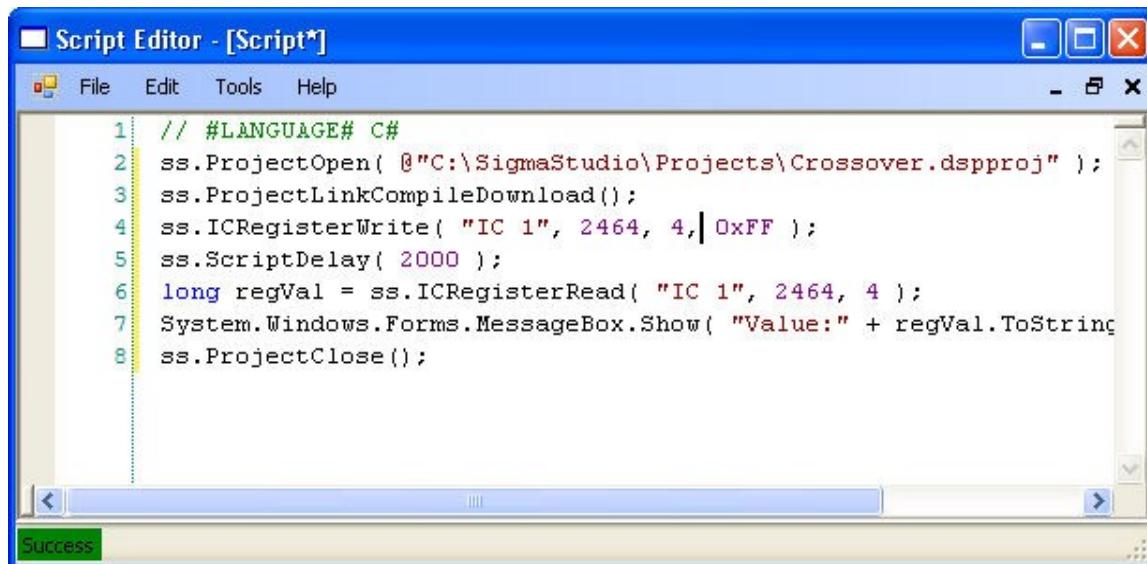
### Print Scale:

By default SigmaStudio scales each page of the schematic to fit all design blocks into the printed page (**Auto Fit**). If you prefer, you can select a different scale factor to enlarge or reduce the printed image size. However, note that any blocks lying outside of the printer's margins after scaling will be cropped.

## Script Dialog

SigmaStudio includes a script-able automation engine allowing control over most common application operations. To access the Script dialog, select **Tools - Script...** in the main application menu or press F12.

Scripting in both the C# and Visual Basic languages is supported. For more information about the SigmaStudio Script engine or for detailed documentation please contact Analog Devices, [sigmadsp@analog.com](mailto:sigmadsp@analog.com).



The screenshot shows the "Script Editor - [Script\*]" window. The menu bar includes File, Edit, Tools, and Help. The code editor displays the following C# script:

```
1 // #LANGUAGE# C#
2 ss.ProjectOpen( @"C:\SigmaStudio\Projects\Crossover.dspproj" );
3 ss.ProjectLinkCompileDownload();
4 ss.ICRegisterWrite( "IC 1", 2464, 4, | 0xFF );
5 ss.ScriptDelay( 2000 );
6 long regVal = ss.ICRegisterRead( "IC 1", 2464, 4 );
7 System.Windows.Forms.MessageBox.Show( "Value:" + regVal.ToString() );
8 ss.ProjectClose();
```

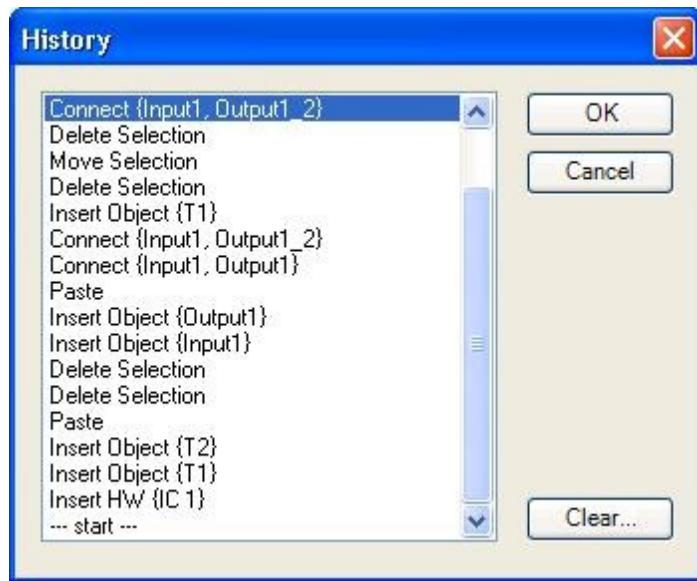
The status bar at the bottom shows "Success".

### SigmaStudioServer:

SigmaStudioServer is an automation interface for SigmaStudio that allows external software applications to control or automate SigmaStudio functions. (For example, National Instruments LabVIEW) The server interface library `Analog.SigmaStudioServer.dll` implements Microsoft .Net and ActiveX server interfaces. For more information or detailed documentation please contact Analog Devices, [sigmadsp@analog.com](mailto:sigmadsp@analog.com).

## Undo History Dialog

The Undo History dialog shows you a list of the actions you have performed on a schematic in the current editing session. To access this dialog go the **Edit - History...** in the main application menu. You can revert the schematic design to any point in its Undo History simply by selecting an entry in the list and pressing the OK button.



SigmaStudio's undo mechanism uses system memory resources. You can empty the history and free these resources by pressing the **Clear...** button.

### **Undo:**

Most actions that you perform in the Schematic and Hardware Configuration tabs can be undone. For example, inserting or deleting a block, creating a wire, re-naming a block, or moving the screen location of a block. You can undo the most recent action by choosing **Edit - Undo** from the main menu or by using the keyboard shortcut, **Ctrl-Z**.

### **Redo:**

Undoing can also be undone. After undoing an action, you can redo it by choosing **Edit - Redo** from the main menu or by using the keyboard shortcut, **Ctrl-Y** or **Ctrl-Shift-Z**.

# Hardware Windows

## Hardware Configuration and Communication Windows

---

The Hardware Configuration tab provides access to advanced hardware configuration, communication, and control. These tools are described in the following pages:

- **Register Read/Write Window**
- **Register Control Window**
- **Flash Downloader**
- **E2Prom Window**
  
- **Imploader Window**

If you have not set up the communication channel for your board, please refer to the Evaluation Board Setup Examples.

---

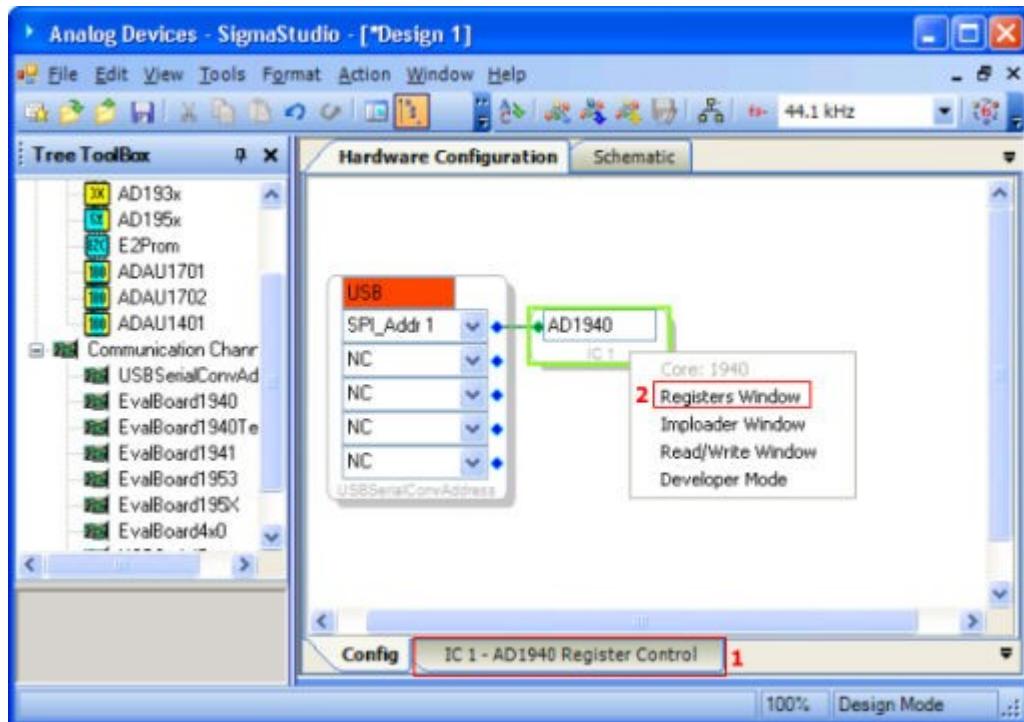
© 2006-2007 Analog Devices, Inc. All rights reserved.

## Register Control Window

The Register Control windows provides access to the internal registers and core settings for the DSP and IC blocks inserted in the Hardware configuration window.

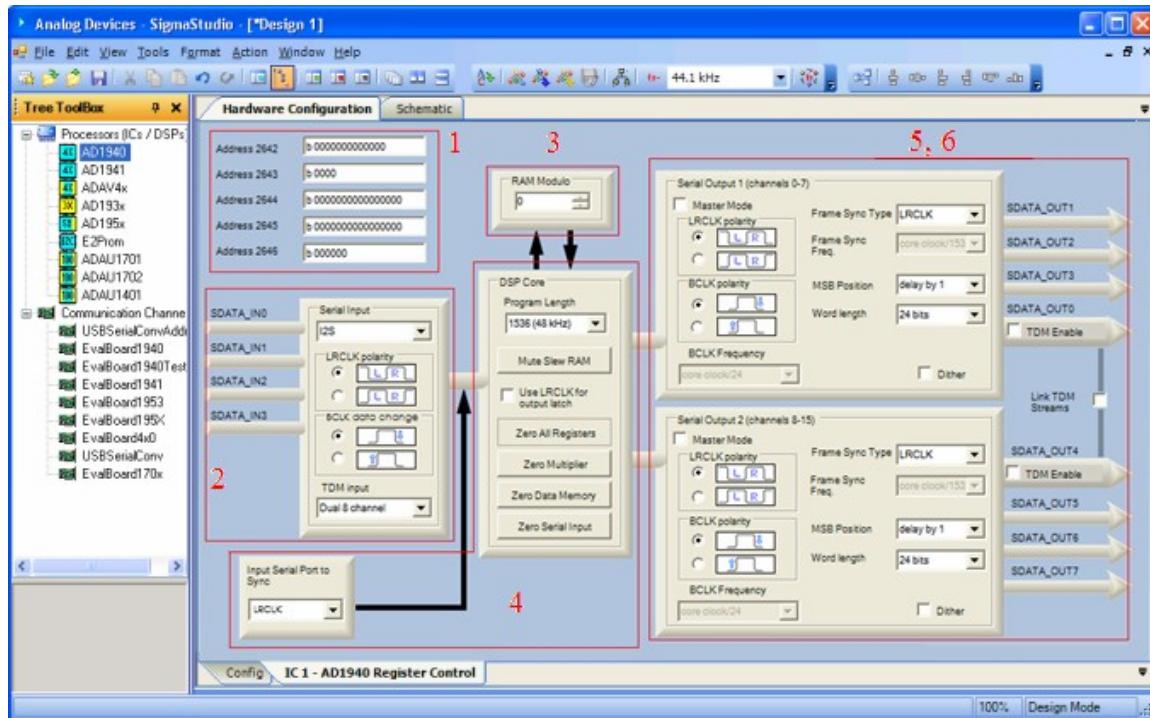
To access the Register Control window:

1. Click the Hardware Configuration tab at the top of the workspace.
2. Click the **Register Control** tab (1) at the bottom of the Hardware Configuration window or Right-click on the DSP processor block and select **Register Window** (2) from the menu.



The controls and settings available in the Register Control window are different for each Processor (IC/DSP) block, depending on the hardware's capability. Any changes made in the register control window are immediately sent to the hardware when there is an active USB communication channel.

Following is a description of the AD1940 Register Control window. For information about these or other IC register settings, please see the refer to your part's datasheet:  
[www.analog.com/sigmadsp](http://www.analog.com/sigmadsp)



## 1. Internal Registers

This area lists addresses 2642 - 2646 and their status. As you make changes in the other areas of the window, you'll see the results here. If you know the register bit locations you wish to change, you can change them here as well, and the buttons in the other areas of the window will be updated accordingly.

- 2642 - DSP Core Control Register
- 2643 - Ram Configuration Register
- 2644 - Serial Output Control Register 1
- 2645 - Serial Output Control Register 2
- 2646 - Serial Input Control Register

## 2. Serial Input - affects the serial-input control register (address 2646), controlling clock polarity and data input modes.

- **Serial Input Mode (Bits 2:0)** - These two bits control the data format that the input port expects to receive. For settings and clock diagrams, see the AD1940/-41 datasheet pp 27-29.
- **BCLK Polarity (Bit 3)** - This bit controls on which edge of the bit clock the input data are clocked: the falling edge of BCLK\_IN when this bit is set to 0, and the rising edge when it's set to 1.
- **LRCLK Polarity (Bit 4)** - If bit 4 set to 0, the left-channel data on the SDATA\_Inx pins are clocked when LRCLK\_IN is low and the right data clocked when LRCLK\_IN is high. When bit 4 set to 1, this is reversed. In TDM mode, when this bit is set to 0, data are clocked in starting with the next appropriate BCLK edge following a falling edge on the LRCLK\_IN pin. When the bit is set to 1 (in TDM mode), the input data are valid on the BCLK edge following a rising edge on LRCLK\_IN.
- **TDM Input (Bit 5)** - Setting this bit to 0 puts the AD1940/-41 into dual 8-channel TDM input mode, with the two streams coming in on SDATA\_IN2 and SDATA\_IN3. Setting it to 1 puts the part into 16-channel TDM input mode, input on pin SDATA\_IN2.

**3. Ram Modulo** - affects the ram-configuration register (address 2643)

The AD1940/1941 uses a modulo RAM-addressing scheme to allow filters and other blocks to be coded easily without requiring filter data to be explicitly moved during the filtering operation. The default value is 12 where the entire 6144 (6k) RAM is treated as modulo memory with auto-incrementing address-offset registers. Each LSB of this register corresponds to 512 RAM locations. A modulo value of 11 would give you 5632 datawords of modulo memory and 512 in a non-modulo portion.

**4. DSP Core** - affects DSP core control register (address 2642)

The controls in this register set the operation of the AD1940/-41's DSP core.

- **Program Length (Bits 1:0)** - These bits set the length of the internal program. The default length is 1536 instructions, but it can be shortened by factors of 2 to accommodate sample rates higher than 48 kHz.
- **Input Serial Port to Sequencer Synch (Bits 3:2)** - Normally the internal sequencer is synchronized to the incoming audio frame rate by comparing the internal program counter with the edge of the LRCLK input signal. In some cases the AD1940/-41 may be used to decimate an incoming signal by some integer factor. In this case, it is desirable to synch the sequencer to a submultiple of the incoming LRCLK rate so more than one audio input sample is available to the program during a single audio output frame. (Operation in this mode may require custom assembly-language coding not currently supported by ADI graphical tools.)
- **Zero Serial Input (Bit 6)** - When this bit is set to 1, the eight serial input channels are forced to all zeros.
- **Zero Data Memory (Bit 7)** - Setting this bit to 1 initializes all data memory locations to 0. This bit is cleared to 0 after the operation is complete. This bit should be asserted after a complete program/parameter download has occurred to ensure clickfree operation.
- **Zero Multiplier (Bit 8)** - When this bit is set to 1, the input to the DSP multiplier is set to 0, which results in the multiplier output being 0. This control bit is included for maximum flexibility and normally is not used.
- **Zero All Registers (Bit 9)** - Active low. Setting this bit to 0 sets the contents of the accumulators and serial output registers to 0. Like the other register bits, this one powers up to 0. That means the AD1940/-41 will not pass a signal until a 1 is written to this bit. This is intended to prevent inadvertent noises from accruing during the powerup sequence.
- **Use LRCLK for Output Latch (Bit 10)** - Normally, data are transferred from the DSP core to the serial output registers at the end of each program cycle. In some cases (e.g., when the output sample rate is set to some multiple of the input sampling rate), it is desirable to transfer the internal core data multiple times during a single input audio sample period. Setting this bit to 1 allows the output LRCLK signal to control the data transfer rather than the internal end-of-sequence signal. (Operation in this mode may require custom assembly-language coding not currently supported by ADI graphical tools.)
- **Mute Slew Ram (Bit 12)** - Setting this bit to 1 initiates a mute of all 64 slew RAM locations. When reset to 0, all RAM locations return to their previous state. This bit is functional only if slew RAM locations are used in the custom program design.

**5, 6. Serial Outputs 1 (address 2644) and 2 (address 2645)**

The output control registers give the user control of clock polarities, clock frequencies and types, and data format. In all modes except for the right-justified ones (MSB delayed by 8, 12, or 16), the serial port accepts an arbitrary number of bits up to 24. Extra bits will not cause an error but will be truncated internally. Proper operation of the right-justified modes requires the LSB to align

with the edge of the LRCLK.

- **Wordlength (Bits 1:0)** - These bits set the length of the output dataword. Options are 16, 20, or 24 bits.
- **MSB Position (Bits 4:2)** - These three bits set the position of the MSB of data with respect to the LRCLK edge. The data output of the AD1940/-41 is always MSB first.
- **Frame Synch Type (Bit 6)** - This bit sets the type of signal on the LRCLK\_OUTx pins. When set to 0, the signal is a clock with a 50% duty cycle; when set to 1, the signal is a pulse with a duration of one bit clock at the beginning of the data frame.
- **Frame Synch Frequency (Bits 8:7)** - When the output port is used as a clock master, these bits set the frequency of the output LRCLK, which is divided down from the internal 73.728 MHz core clock.
- **BCLK frequency (Bits 10:9)** - When the output port is being used as a clock master, these bits set the frequency of the output bit clock, which is divided down from the internal 73.728 MHz core clock.
- **Master/Slave (Bit 11)** - This bit sets whether the output port is a clock master or slave. The default setting is slave; on powerup, Pins BCLK\_OUTx and LRCLK\_OUTx are set as inputs until this bit is set to 1, at which time they become clock outputs.
- **BCLK Polarity (Bit 12)** - This bit controls on which edge of the bit clock the output data are clocked: on the falling edge of BCLK\_OUTx when this bit is set to 0, and on the rising edge when it's set at 1.
- **LRCLK Polarity (Bit 13)** - When set to 0, the left-channel data are clocked when LRCLK is low, and the right data clocked when LRCLK is high. When set to 1, this is reversed.
- **Link TDM Streams (Bit 14, Serial Output Control Register 1)** - When this bit is set to 1, the TDM output streams a linked to output a single 16-channel stream on SDATA\_OUT0. When set to 0, TDM data are output on two independent 8-channel streams, on pins SDATA\_OUT0 and SDATA\_OUT4.
- **Dither-Enable (Bit 15)** - Setting this bit to 1 enables dither on the appropriate channels.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

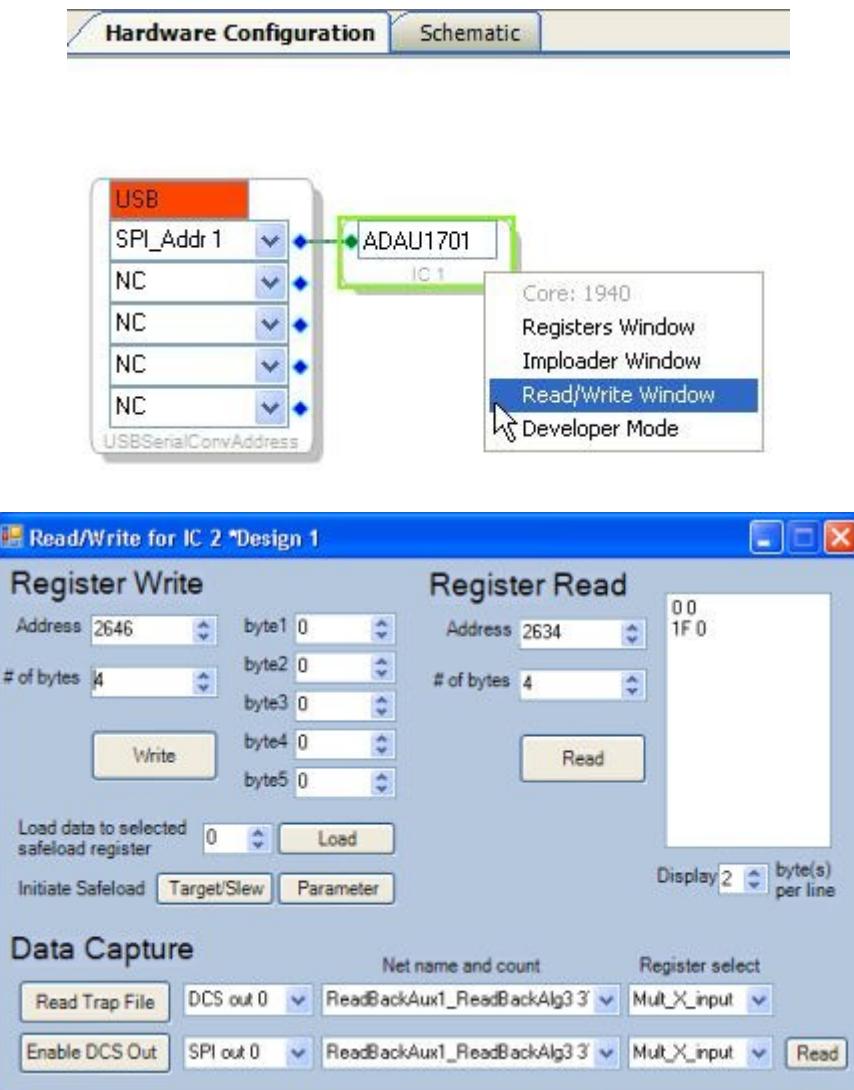
## **Register Read/Write Window**

---

The Register Read/Write window allows you to read and write bytes of data from and to a DSP IC, for some DSPs it also performs program file loading and data capture.

To access the Register Control window:

1. Click the Hardware Configuration tab at the top of the workspace.
2. Right-click on the DSP processor block, (click outside of the part number field, e.g. AD1701) to display the context menu.
3. Select **Read/Write Window** from the menu.



### Register Read/Write

This window allows you to read and write bytes of data directly to Parameter RAM, Program RAM, Target/Slew RAM, and available registers, which is useful for sending specific parameters directly to the DSP and also checking parameter values that have been sent.

- For writing, set the address, number of bytes, the data to be written, and click **Write**.
- For reading, set the address and number of bytes, and click **Read**. These values will be displayed.

**Note:** Only 28 bits of the 32 available are used, so the 4 MSBs of the first hex value are zero-padded from the 4th bit. (This is different from the **Capture Output Data** window and **params** file, which are sign-extended from the fourth bit.) It also is important to keep in mind that **Register Read** is actually reading parameters back from the DSP, whereas **Capture Output Data** and **params** file are reading the parameters being sent.

### Data Capture

These functions allow any node in the signal-processing flow to be sent to control-port-readable registers or to a digital output pin, provided it's supported by your particular hardware settings. This can be used to monitor and display information about internal signal levels or compressor / limiter activity. The digital output registers are output on SDATA\_OUT7 when the data-capture serial-out enable bit (bit 14) is set in serial output control register 2. (This is AD1940-specific). Clicking **Enable DCS Out** sets this bit for you.

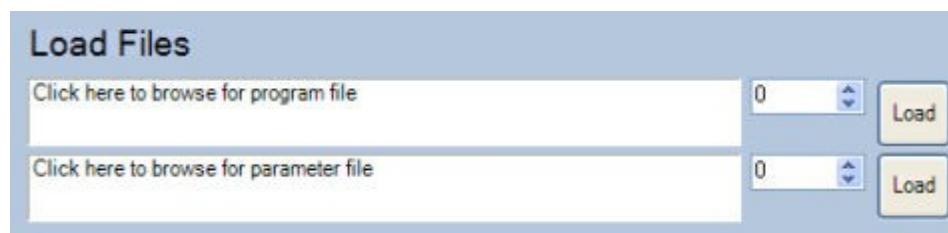
- Read the generated **trap.dat** file by clicking **Read Trap File**, which will populate the lists with the available TRAP addresses.
- Select one of the 6 independent control-port-readable data-capture registers (SPI 0-5) or one of two digital output capture registers (DCS 0/1).
- Select the **Net Name** and its count that you wish to capture. The count corresponds to the program step number where the capture will occur.
- Select the register for output. This register will be transferred to the data-capture register when the program counter equals the capture count. Options are:
  - Mult\_X\_input - multiplier X input
  - Mult\_Y\_input - multiplier Y input
  - MAC\_out - multiplier-accumulator output
  - Accum\_fbck - accumulator feedback
- Click **Read** to see the data. The captured data are in 5.19 twos-complement data format, the 4 LSBs truncated from the internal 5.23 dataword.

### Load Files

These functions allow loading a previously compiled program or parameter file to the hardware, provided this feature is support by your particular DSP part. This can be used to quickly compare project settings.

To load a file:

- Click the "Click here to browse for program/parameter file" control.
- Locate the \*.dat file you wish to load (e.g. hex\_program\_data.dat) and click open, the numerical control will display the size of the program bytes.
- Press **Load** to send data to the hardware.



File load functionality for the AD1940/AD1940 is available through the Imloader.

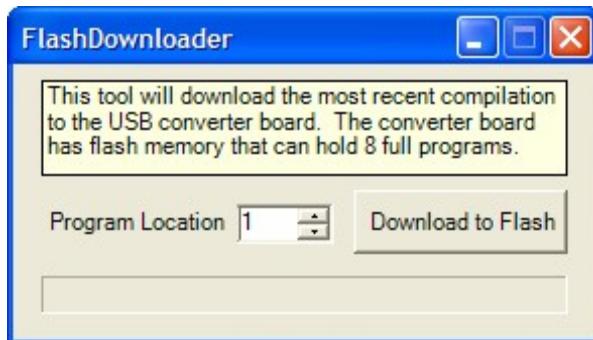
© 2006-2007 Analog Devices, Inc. All rights reserved.

## Flash Downloader

The **Flash Downloader** lets you store your compiled projects to the flash RAM on the USB Serial Converter board. Eight slots are available. This allows quick comparisons among projects; you can Link-Compile-Download each project to RAM and then switch easily among them.

### Steps

1. Compile the project you wish to download.
2. Click the Hardware Configuration tab at the top of the workspace.
3. Right-click the Communication Channel block.
4. Select **Open Flash Downloader** from the menu.
5. Enter the Program Location to be stored. (**Note:** The Program Location number listed on the popup window begins with 1, whereas the rotary switch on the USB converter board begins with 0; please adjust accordingly).
6. Click **Download to Flash**.



Now your project is accessible in the Program Location you entered. Select it on the USB board and push **Program Load**.

---

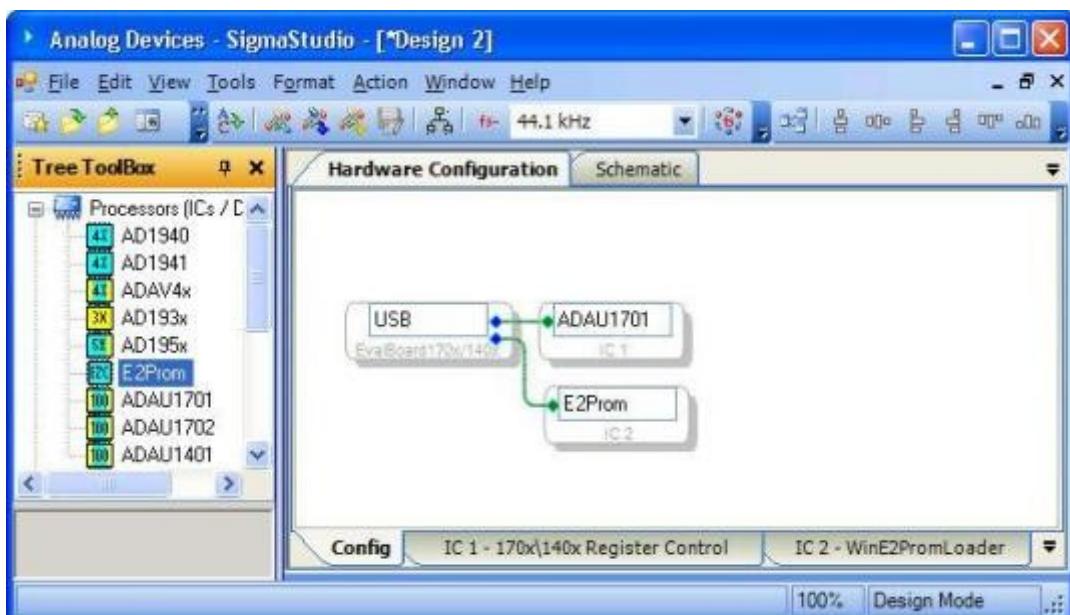
© 2006-2007 Analog Devices, Inc. All rights reserved.

## E2Prom Read/Write Window

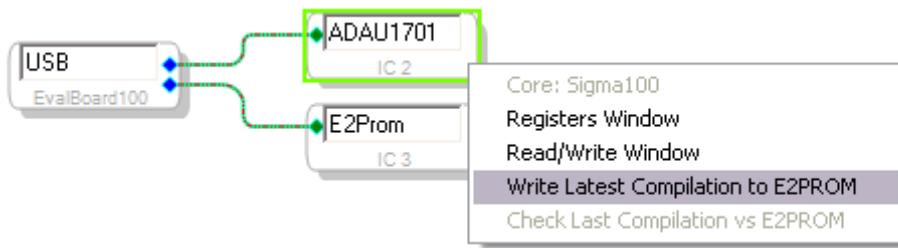
The E2Prom Read/Write window allows you to read and write program data to an E2Prom IC. This tool can be used with your own hardware systems or with the E2Prom available on some SigmaDSP evaluation boards. It also supports saving and loading the program data to files.

### E2Prom Configuration:

1. To use the E2Prom, insert the E2Prom IC block into Hardware Configuration tab and connect it to the USB communication channel block as show in the figure below.



2. Compile/Link/Download the project and verify that "Ready - Download" is displayed on the status bar.
3. To download the compiled project's program data to the E2Prom, Right-Click on the DSP block in the Hardware Configuration window and select **Write Latest Compilation to E2PROM** from the menu. Note if your E2Prom's size differs from the default 16kB size, you should first adjust the size setting in the Read/Write window (see below) before writing .



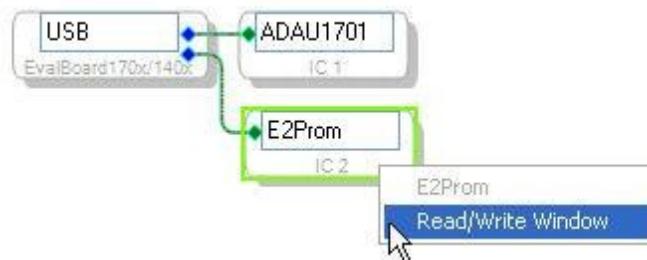
4. If your hardware is setup correctly, you should see the Download E2Prom progress dialog.

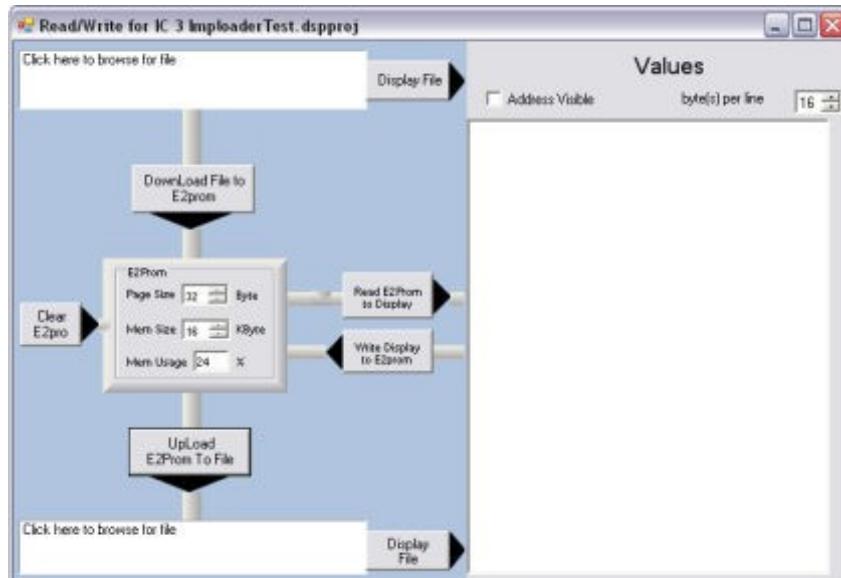


---

#### Using the E2Prom Read/Write window:

To open the E2Prom Read/Write window, Click the WinE2PromLoader tab at the bottom of the Hardware Configuration window or Right-click on the E2Prom IC block and select Read/Write Window from the menu.



**Settings:**

**Address Visible** Check this box to see the memory address values in the right-hand display.  
**byte(s) per line** Adjusts the number of bytes shown on each line of the right-hand display.

**Page Size** Sets the page size for reading/writing to the E2Prom.  
**Mem Size** The size of the E2Prom memory, verify this setting **before** attempting to read or write.  
**Mem Usage** Displays how much memory is used by the most recently downloaded/uploaded program.

**Buttons:**

**Download File to E2Prom** Downloads a previously uploaded program file to the E2Prom hardware (note this will overwrite the current E2Prom data). Click in the control at the top of the window to select a file prior to loading.

**Upload E2Prom To File** Save the current E2Prom data to the file. Click in the bottom control to select a save filename before pressing this button.

**Clear E2Prom** Write '0' to the entire E2Prom memory overwriting any program data previously downloaded.

**Display File** This button will show the program data of the selected download or upload file in the right-hand display.

**Read E2Prom to Display** Reads the current E2Prom data and shows it in the right-hand display.

**Write Display to E2Prom** Writes all the data bytes shown in the right-hand display to the E2Prom.

**Values Display:**

In addition to displaying the data, the right-hand values display allows you to edit the program data bytes directly. Value changes can be written to the E2Prom using the "Write Display to E2Prom" button.

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Imploader Window

---

The Imploder window (**AD1940 and AD1941 only**) allows you to compile and download previously compiled projects without the need to load the project files. This is useful for comparing among different settings or flows.

To access the Imploder window:

1. Click the Hardware Configuration tab at the top of the workspace.
2. Right-click on the DSP processor block to display the context menu.
3. Select **Imploader Window** from the menu.

To load a program or parameters:

1. Select **File - Open** from the menu or click the Open toolbar button.
2. Locate the compiled netlist file for the project you wish to load (\*.cir, \*.cir2) and click open.
3. Press the "Download Program and Parameters" button.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

# Using SigmaStudio

## SigmaStudio System

---

These pages describe the basic functions of the SigmaStudio software:

- Building Schematics (IC, Input/Output, Blocks, Algorithms and Wires)
- Link/Compile/Download
- Capture Output Data
- Export Program and Parameter Data
- System Implementation
- Sampling Rate Considerations
- Numeric Formats
- Hierarchy Boards
- Schematic Actions and Commands:
  - A/B Comparison Between Projects
  - Copy/Paste
  - Change IC
  - Freeze
  - Schematic Settings (Presets)

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Building Schematics

### Building Schematics

---

This section present the fundamental concepts of the SigmaStudio System. For a brief overview of creating a SigmaStudio project, refer to the Quick Start. You can also find more information in the tutorials and examples.

- **Schematic Blocks**
- **Algorithms**
- **Wires and Aliases**

---

Following are the steps required to create a new SigmaStudio schematic:

### **Step 1: Create a New Project File**

SigmaStudio starts with no project files loaded. To create a new project, select **File - New Project** from the application's main menu or click the New Project button on the application toolbar. A new project is created, with the default project name "Design 1". New projects are blank and the Hardware Configuration tab is displayed.

### **Step 2: Choose a Processor (ICs/DSPs)**

- Processors (ICs and DSPs) are managed in the Hardware Configuration tab.

Before you can begin a system design, you must select at least one IC block in the Toolbox's **Processors (ICs/DSPs)** category and drag and drop it into the Hardware Configuration tab. The available algorithm blocks differ for each SigmaDSP product, so it is important to select the DSP IC that you intend to use in your final design from the beginning. Note that It is possible to create designs with more than one processor.

To communicate with evaluation hardware, you must also insert a USB block from the Toolbox's **Communication Channels** category into the Hardware Configuration tab and connect it to the processor block. Refer to the USB Communication Channels section for more information.

### **Step 3: Inputs and Sources**

- Inputs and Sources are located in the Schematic tab

Once a Processor block is added to the project, the Schematic tab is displayed. Select the schematic tab and add Inputs or Sources by dragging and dropping blocks from the Toolbox's **IO** and **Sources** categories.

**Note:** Every schematic MUST contain either an Input block or a Source block. If no inputs are present in the schematic design, you will receive the compiler error: **Error - No Inputs are Defined for IC.**

### **Step 4: Create a Signal Processing Design**

- Algorithm/Function blocks are managed in the Schematic tab

Drag and drop blocks from the ToolBox or Tree ToolBox pane into the schematic tab to create your design. Note that in addition to advanced signal processing blocks, there are a variety or low level building blocks available (delay, multiply, and addition, feedback) allowing you to implement custom algorithms to fulfill your specific design requirements.

### **Step 5: Outputs**

To output processed signals from the SigmaDSP hardware, you will need to insert an Output block into the schematic design. Drag and drop output blocks from the ToolBox or Tree ToolBox pane into the schematic tab. The output blocks represent the hardware's physical analog and digital output pins.

### Step 6: Wire Inputs to Outputs

Each schematic block has input and output pins which can be wired together to create the system's signal flow. All block's input pins must be connected to a source or another block's output pin and all block's output pins must be connected or terminated.

**Note:** If there are unconnected pins in the schematic you will receive the compiler error: **Fatal Error: Unconnected pins found in cell.**

### Step 7: Link and Compile the Project

Once the schematic is complete and all blocks are correctly wired together, you can link and compile the project. If any errors are encountered during compilation, the Link Window will open and display the error information. If compilation is successful you will see a green bar directly below the schematic tab window and **Ready - Download** is displayed in the application's status bar.

### Step 8: Adjust Controls in Real-time

Once your schematic is compiled and downloaded to the hardware, you can adjust the schematic's controls (knobs, sliders, and spin boxes) to change algorithm parameters in real time. This allows you to tune your design's settings before implementing the final system. See System Implementation for more information about how to integrate a schematic design into your end system.

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Schematic Blocks

---

Schematic Blocks are used to build a SigmaStudio design. The blocks available for each processor are displayed in the ToolBox and Tree ToolBox windows and can be dragged and dropped into the schematic.

### Pins:

Each schematic block can contain one or more pins used to connect the blocks together. Pins are used to route audio and control data in the schematic flow. There are 3 different types of pins:

1. Input Pin (**Green**)
2. Output Pin (**Blue**)
3. Control Data Pin (**Orange**)



**Note:** Input pins can only connect to Output pins and Output pins can only connect to Input pins. Typically, control data pins are only connected to other control data pins, but it is possible to connect an audio data pin to a control data pin and a control data pin to an audio data pin when necessary.

Hover over a block's pin with the cursor to display the pin's tool-tip which includes the pin number, unique name, and signal data type.



### Algorithms:

Each block represents one or more algorithms. You can add algorithms to blocks or modify them to meet your specific requirements by selecting Add Algorithm or Grow Algorithm from the block's right-click context menu. A block that contains no algorithms will have no controls or input/output pins. See Algorithms (Adding and Growing) for more information.

### Naming:

To change the name of a block, double click on the block label and type a new name. Note that within each hierarchy board all block names must be unique. You will see an error dialog if you attempt to use a name that is already present in the current window.



### Selecting:

To select a block, click its border or label. To select multiple blocks, hold down the Shift or Ctrl key while clicking. You also can click in the schematic window and drag a box around blocks to select the, this will select any blocks that are in the selected screen area. To select all block you can press **Ctrl + A** or use the **Select All** button in the Schematic Layout toolbar. Selection is indicated by a light-green outline.

**Deleting:**

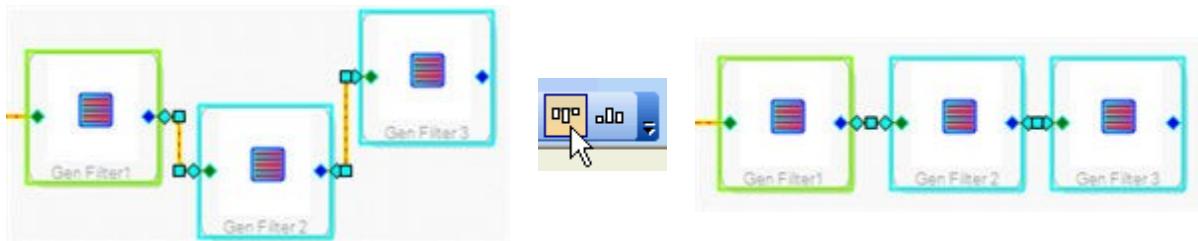
To remove a block from the schematic select the block and then press the **Delete** key. You can also remove selected blocks by choosing **Edit - Cut** from the main application menu or by pressing **Ctrl + X**.

**Action Menu:**

Right-click on a block to bring up the block's popup menu. This menu includes items for control Settings, Add/Remove Algorithm, Grow/Reduce Algorithm, Changing IC, and Cut/Copy/Paste editing.

**Layout:**

You can change the position of a block by selecting it with the mouse and dragging it to a new screen location. Use the Schematic Layout toolbar buttons to align groups of blocks in your schematic. Alignment is applied to all selected blocks. The blocks with a blue outline will be aligned to the block with a green outline.

**Controls:**

Blocks can contain a variety of controls for editing algorithm parameters.

**Spin Controls:**

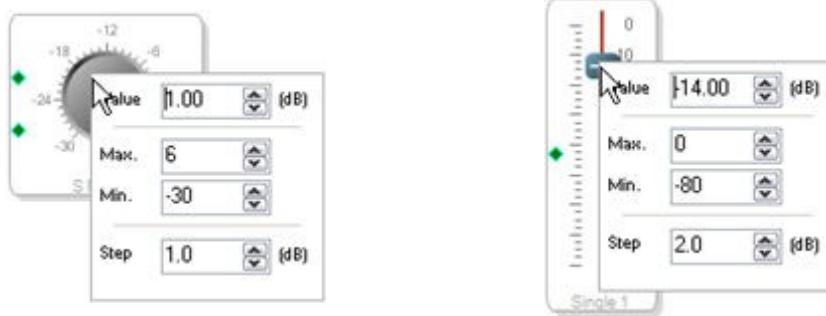
Spin controls allow you to change values either by entering the value directly in the edit-box or by clicking the left/right or up/down arrows. You can left-click and hold an arrow to make large adjustments or click-hold-drag to increase the rate of change. A grey control (for example the top control shown below) means the control is disabled and the value cannot be changed for the current algorithm.



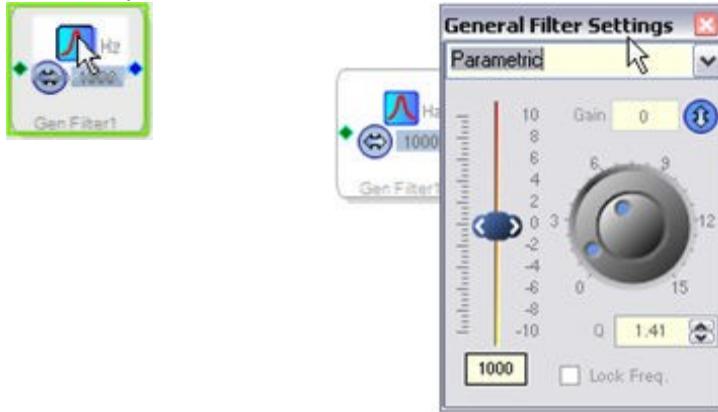
**Knobs & Sliders:**

To change the value of knobs or slider control, click and hold the left mouse button on the control and drag to adjust the value. You can also change the knob or slider control's value, range, and step size. To modify the knob or slider settings, **right-click** on the control which displays the control pop-up window (shown below).

- **Value** Set the control's value.
- **Min/Max** Adjust the control's value range (within the limits of the block's algorithm).
- **Step** Set the control's step size or granularity.

**Pop-up Control Windows:**

Some blocks include pop-up control windows providing advanced functionality, for example the General (2nd Order) filter. To open a block's advanced control window, click on the icon button.



© 2006-2007 Analog Devices, Inc. All rights reserved.

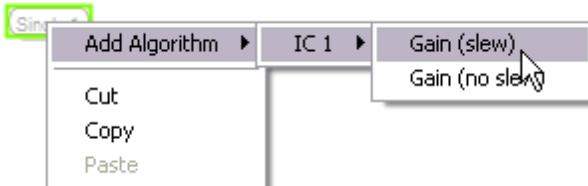
## **Algorithms: Add/Remove, Grow/Reduce**

---

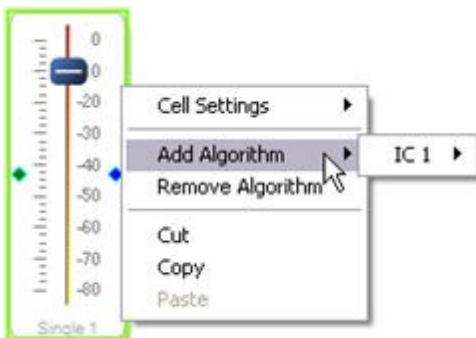
Each schematic block represents one or more signal processing algorithms. Algorithms range from very simple, like Signal Add, to advanced system components like Dynamic Bass Boost. You can add or remove algorithms from blocks to meet your specific requirements. Algorithms can also be grown as explained below.

### Add Algorithm:

You must add an algorithm -- typically associated with set of i/o pins -- for a block to function. To Add an algorithm to a block, **right-click** the block and select **Add Algorithm > IC #**, selecting the DSP IC for the algorithm. Note, if there are multiple processors (ICs/DSPs) in your project, you can select which DSP will run an algorithm, see below for more information.



At this point, right-click the block on its border or label to add another algorithm if desired. (It's important to right-click the border or label; if you right-click the center, you may display the pop-up window for entering parameters values.)

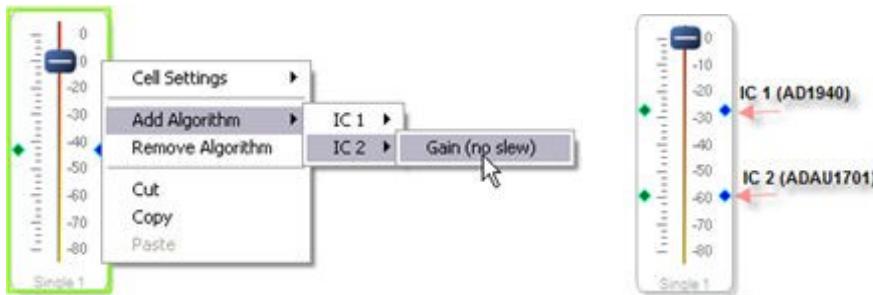


A block that contains no algorithms will have no controls or pins and you will only see the block's name (as shown in the figure below). You must add an algorithm to empty blocks before they can be used in the schematic design.

**Gen Filter1**

**Note:** When using the Tree ToolBox window, an algorithm is always created for each inserted block. However, if you use the traditional Toolbox window to insert blocks, you may additionally have to add an algorithm.

When adding an algorithm, not only are you choosing the method of computation for the block, you're also selecting a particular DSP association for the algorithm. This is important if you're connect multiple DSP processors: by adding algorithms to blocks, you can share blocks and controls and communicate with multiple DSPs simultaneously. For example: the Single Volume Control block has a single slider control for all algorithms. If you create a project with 2 DSP processors, IC 1 (AD1940) and IC 2 (ADAU1701), and select Add Algorithm, you will be prompted for which chip you want to add this algorithm to.



It is possible to have the first algorithm assigned to IC 1, and a second algorithm assigned to IC 2, but they share the volume block and a single slider control. Note that you cannot create a wire between different processor, see the Wires topic for more information.

#### **Remove Algorithm:**

It is also possible to remove algorithms from blocks. To remove an algorithm, **right-click** the block and select **Remove Algorithm**. This will remove the last algorithm (bottom pins) which is the algorithm that was added most recently. If the block contains only one algorithm, removing the algorithm will result in an empty block

#### **Grow Algorithm:**

Growing algorithms, means building upon the existing algorithm of the block, keeping the same algorithm in place (expanding upon it) and the same DSP association (Adding algorithms does neither). To grow an algorithm, **right-click** the block and select **Grow Algorithm > (algorithm name) > (grow amount)**. Note that growing is not available for all algorithms.

The easiest way to understand the distinction between adding and growing is with a mixer block. Drag a Cross Mixer (2 Inputs) into the workspace. Right-click and select Grow Algorithm, See the example below.

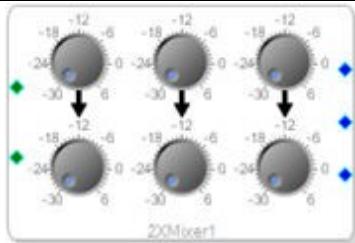
- Growing the mixer creates more mixer output pins, essentially you are creating extra mixer output channels that share a common input. There is still only a single algorithm in the block.
- Adding an algorithm creates separate algorithms that share the control window, but do not share inputs/outputs pins or resources. In the mixer example, an additional input pin, oupput pin, and a cross-mixer control are added to the block.

#### **Reduce Algorithm:**

Reducing is the opposite of growing, it removes the extra controls and pins that are created by the Grow Algorithm operation. Like remove algorithm, reduce will remove controls/pins starting at the bottom or the most recently grown item.

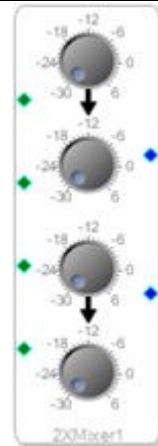
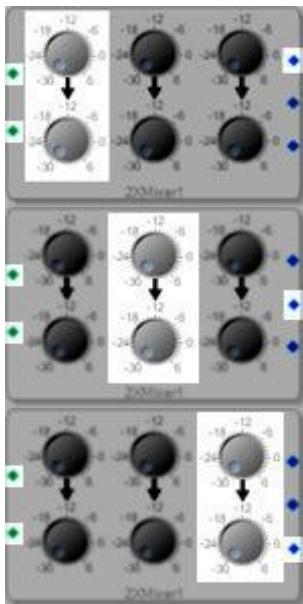
#### **Examples:**

Grow Algorithm	Add Algorithm
----------------	---------------



To produce this block:  
Drag a Cross Mixer (2 Inputs) into the schematic, Right-click the block, and select **Grow Algorithm > 1. 2-Channel X Mixer > 2**.

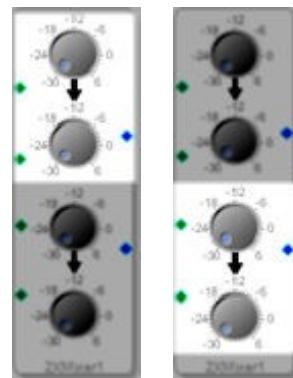
The mixer has 2 inputs (stereo) and 1 output (mono). When you grow it, you are growing only the number of outputs. Here, the top output pin is controlled from the left-most mixer (vertical knob pair), the middle output pin from the middle mixer, and the bottom output pin from the right (last) mixer.



To produce this block:  
Drag a Cross Mixer (2 Inputs) into the schematic, Right-click the block, and select **Add Algorithm > IC 1 > 2-Channel X Mixer**.

In this example, you have mixer a 2-Channel X Mixer algorithm on the top, and a second mixer algorithm at the bottom -- as added.

The top knob pair (two knobs) control the first algorithm, the second knob pair control the second algorithm. No input or output signal is common between the 2 algorithms.



Similarly, if you were to add a third algorithm, there would be six inputs and three outputs.

© 2006-2007 Analog Devices, Inc. All rights reserved.

## **Wires and Aliases**

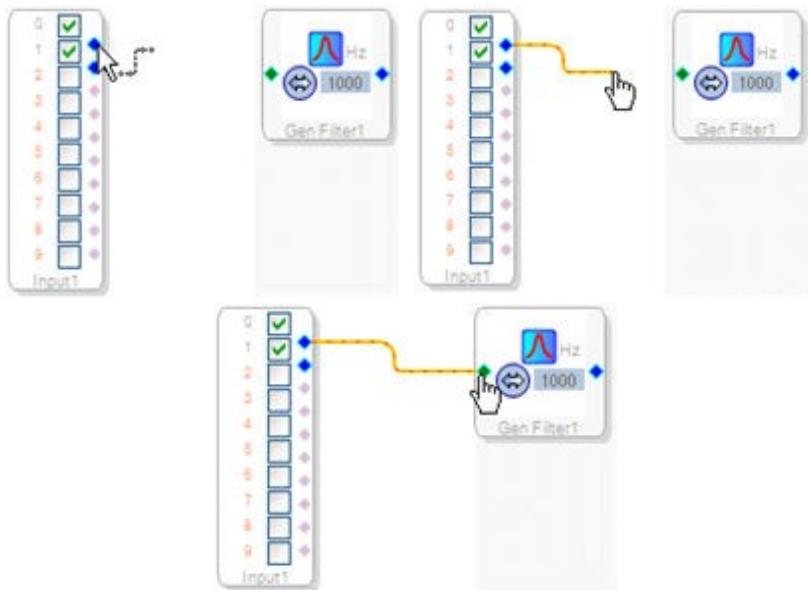
---

## SigmaStudioHelp

SigmaStudio schematic design are built from blocks which are connected together with "wires". Wires define the system's signal flow.

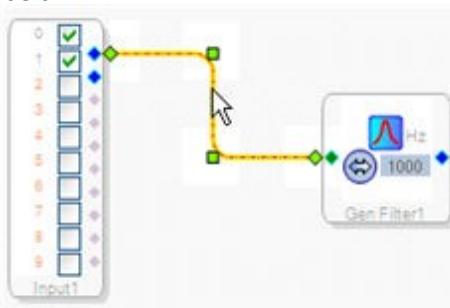
### To create a schematic wire:

Move the mouse cursor over a block's pin so the wire icon, , is displayed. Next, Left-click on a block pin, and while holding the mouse button, drag the cursor to another block's corresponding pin. Input pins can only connect to Output pins and Output pins can only connect to Input pins.



### Selecting:

To select a wire, click on it with the left mouse button. Selected wires are indicated by green squares (points) as shown below.



### Position:

To change the position of a wire, place the mouse cursor over a point. Next, Left-click on the wire point, and while holding the mouse button, drag the cursor to reposition the wire.



### Menu:

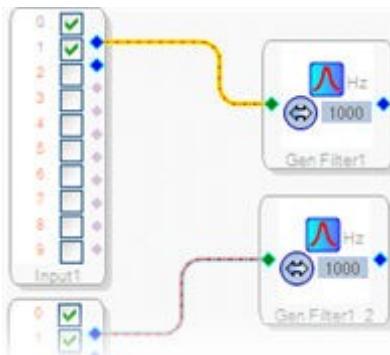
Right-click on a wire to bring up the wire menu. This menu includes following commands:



<b>Delete</b>	Removes the wire.
<b>Insert Point</b>	Adds an additional layout point to the wire at the current position.
<b>Remove Segment</b>	Remove a line segment created by inserting a point.
<b>Enable Auto Layout</b>	Resets any custom positioning allowing the wire to layout automatically.
<b>Align to Source / Destination Pin</b>	Moves the Source or Destination block so the Input and output pins are aligned.

### Wire colors:

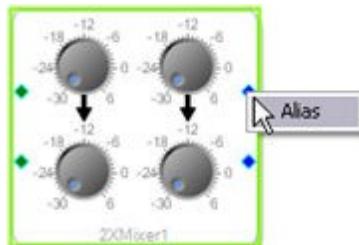
Wires are colored according to their associated DSP processor. The input and output pins of the wire must be associated with the same DSP or you will not be able to create a wire between the pins. Projects with more than one processor IC will have distinct colors for each IC.



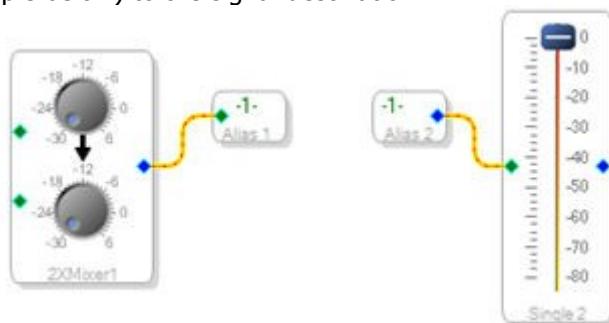
**Alias:**

To organize your project visually, it often helps to create an alias to provide a clear connective reference (a "jump") in the signal flow in the schematic. An **Alias** consists of a pair of alias input and alias output blocks. Using alias blocks can reduce the visual clutter in the schematic window created by long wire connections.

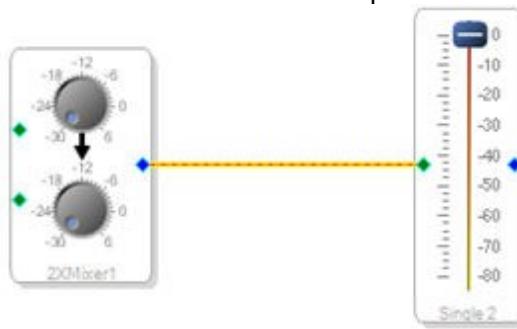
To create an alias, **Right-Click** on a block's output pin (blue pin) and select **Alias** from the menu.



When you click **Alias**, two blocks appear, input and output. The alias input is automatically connected to source block's output pin. Next, create a wire from the alias output block (Alias 2 in the example below) to the signal destination.



Note: Using an alias is functionally equivalent to connecting two pins with a wire. The signal flow in the example below is identical to the alias example above.

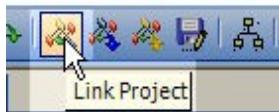


## **Link/Compile/Download**

---

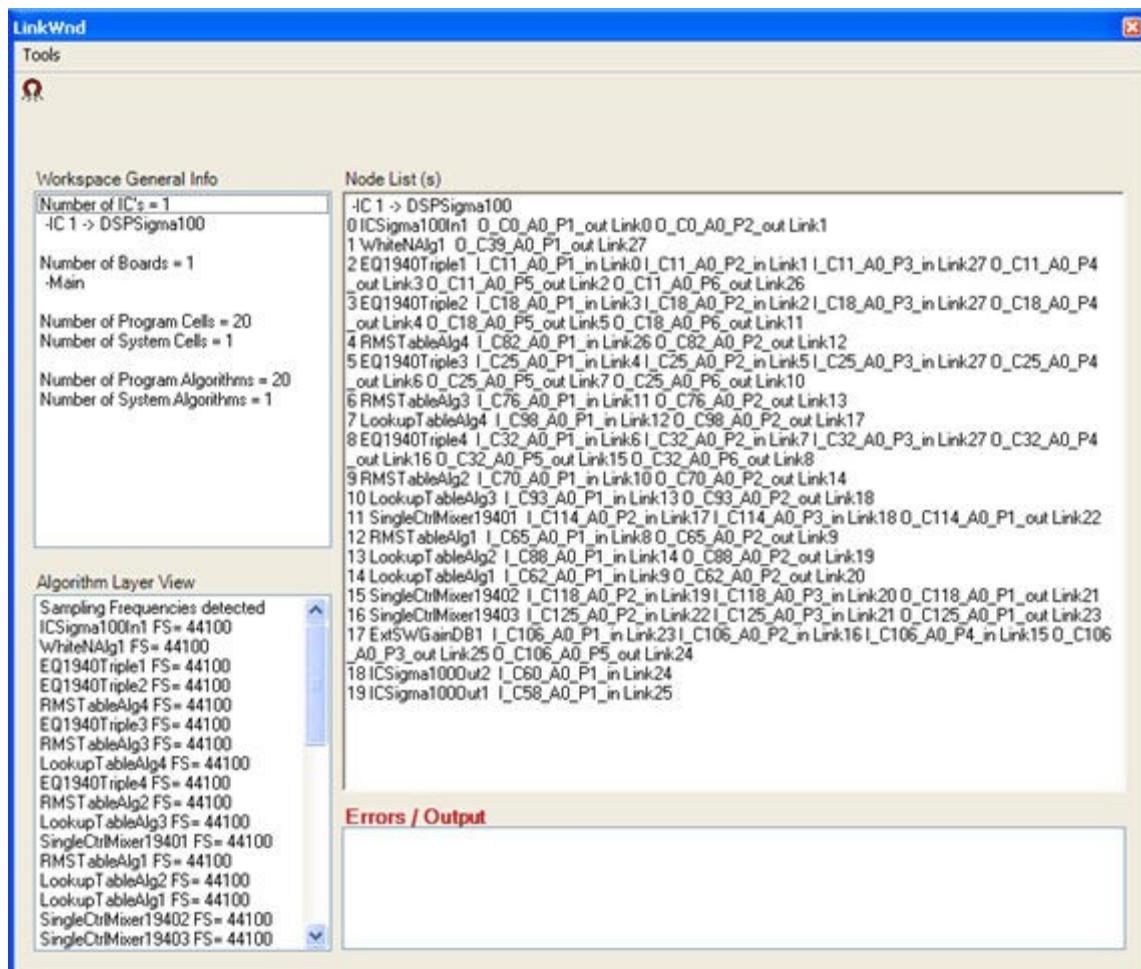
Before a complete design can be evaluated, it must be linked, compiled, and downloaded to the hardware. To perform all these steps in a single operation, use the Link Compile Download command described below.

## Link:



The link operation analyzes the schematic's signal flow (blocks and wires), checks for any in design errors, generates the program flow, and sets algorithm sampling rates and DSP associations. To link a project, press the **Link Project** toolbar button, select **Action - Link Project** from the main menu, or press **Ctrl+I**.

When linking is complete, the Link Window opens, displaying the project's algorithm and node list information.



Any errors encountered during the link operation are shown in the **Errors / Output** section of the Link Window. You cannot compile a project until all link errors are resolved. Note that all schematic block's pins must be connected or terminated or you will receive link errors.

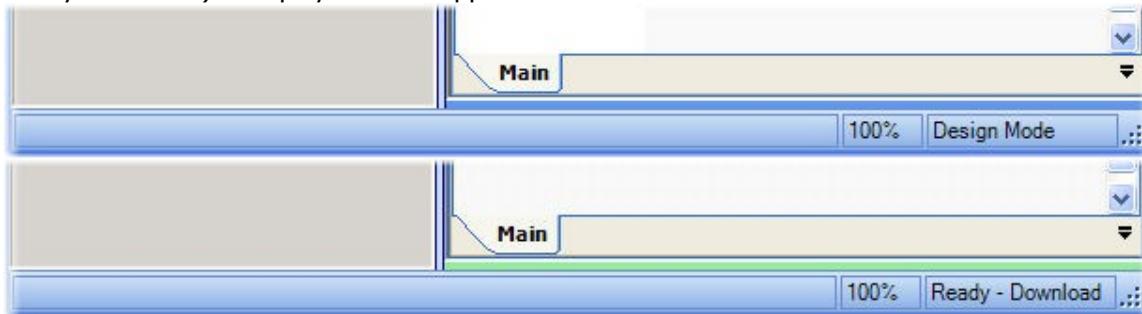
**Errors / Output**

Fatal Error: Unconnected pins found in cell: Input1  
 Fatal Error: Unconnected pins found in cell: Mid EQ1  
 Fatal Error: At least one of the Algorithm: RMSTableAlg4's input pin is not connected but its output is.  
 Fatal Error: Unconnected pins found in cell: RMS Table4

The linker generates a net list file, **net\_list.cir2**, which contains the same information displayed in the Link Window's Node List pane.

**Compile:**

Following the link step, compilation is performed. Compile generates the DSP program code and parameter data from the graphical schematic design. To compile a project, press the **Link Compile Connect** toolbar button. This will first perform the link step and then compile the project. The link window will be displayed if there are link errors. If compilation is successful, it is indicated by the schematic status bar turning green and (100% Ready-Download) is displayed in the application status bar.



Upon project compilation, several files are generated:

- **hex\_program\_data.dat** - load file for downloading code using microcontroller in hex format
- **program\_data.dat** - load file for downloading code using ADI loader
- **hex\_program\_simdata.dat** - load file for downloading code using microcontroller, with no commas, X'es, or spaces; this is a useful format for Verilog simulations
- **spi\_map.dat** - parameter RAM locations for each schematic instance
- **compiler\_output.txt** - SigmaStudio compiler output file
- **trap.dat** - lists the values to enter in the trap registers to output a signal to the data-capture output pin.

These files are written into the project file's directory (\*.dspproj), or the active directory if the project has not been saved. The files are placed in an automatically generated sub-folder or folders named with the IC name followed by the project file name, for example **IC 1\_Design 1**.

Awareness of these files, both their format and their contents, will prove useful. Also, it is helpful to archive and version these files during system design. SigmaStudio can minimize development time by avoiding the need to program in assembly code. Typically once the graphical SigmaStudio design is complete, these generated SigmaDSP code files must be integrated with

an external microcontroller. See System Implementation for more information.

---

### Download:



Once a project is compiled, the program and parameter data can be downloaded to the evaluation hardware for testing. To download your project, press the **Link Compile Download** toolbar button, select **Action - Link Compile Download** from the main menu, or press **F7**.

This performs the link step, compile the project, and send your design code from the schematic to the DSP hardware.

When complete, as indicated by the green bar below the workspace and (100% Ready-Download) in the schematic status bar, your program will be responsive to all real-time changes made to control, sliders, knobs, etc., as you actively change the design parameters.

**Note:** Any time you make edits in the schematic (e.g. add/remove blocks, add/remove wires, add/remove algorithms) you must recompile and download the project. The hardware program sync state is indicated by the blue or green color of the schematic status bar. If the bar is blue the hardware program and schematic are out of sync.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Capture Output Data

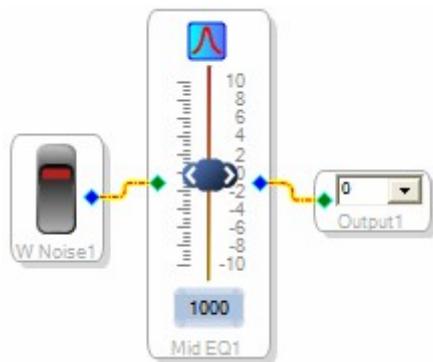
---

The Capture Window shows, in real time, the data SigmaStudio is sending to the hardware. This includes data sent during compile/link/download, register read/write operations, and changes to schematic block controls. The following example provides an explanation of the data displayed in the capture window.

### Output Capture Example :

This example uses the AD1940 DSP and the sample schematic shown at the right: utilizing a White-Noise Source, Medium Size EQ Filter (single precision), and Output block.

Display the Capture Window by selecting **View - Capture Window** in the application's main menu or by clicking the button on the Standard Toolbar.



Select the Schematic tab and compile/link/download the project. The compiled program data is downloaded to the hardware and displayed in the Capture Window.

Capture Window							
Mode	Time	Cell Name	Parameter Name	Address	Value	Data	Bytes
Block Write	22:5:24 - 243ms		CoreReg	2642		0X00 ,0X00	2
Block Write	22:5:24 - 243ms		Program Data	1024		0xFF ,0XF2 ,0X01 ,0X20 ,0X01	7680
Block Write	22:5:24 - 352ms		Parameter Data	0		0X00 ,0XA ,0X45 ,0X07	4096
Block Write	22:5:24 - 399ms		IC 1-1	2642		0X00 ,0X00 ,0X1C ,0X00	8
Safeload Write	22:5:24 - 399ms		Slew Ram Data	2560		0X00 ,0X00 ,0X00 ,0X00 ,0X00	320
Block Write	22:5:24 - 399ms		CoreReg	2642		0X02 ,0X00	2

1. **Mode** column denotes whether an entry represents a read or write message and whether or not writing utilizes Safeload. Mode is also represented by different text colors.
2. **Cell Name** is the name of the schematic block sending a message.
3. **Parameter Name** represents the name of the algorithms variable, register name, or memory location depending on the message type.
4. **Address** is the target hardware address.
5. **Value** displays the floating point representation of the Data column.
6. **Data** is the raw hexadecimal byte values written to the hardware.
7. **Bytes** shows the total number of bytes written or read.

- For the AD1940, the first data sent is to mute the board. You can see that the parameter address for muting is 2642 and the values are 0x00 0x00.

Mode	Time	Cell Name	Parameter Name	Address	Value	Data	Bytes
Block Write	22:5:24 - 243ms		CoreReg	2642		0X00 ,0X00	2

- Next is the program code block, starting at address 1024, and utilizing 165 of the available 7680 byte program memory. Click the plus button in the left-hand column to expand this row and see all the data values.

Block Write	22:27:15 - 31 ...	Program Data	1024		0xFF ,0XF2 ,0X01 ,0X20 ,0X01	7680
					0X00 ,0X08 ,0X00 ,0XE2 ,0X01	
					0X00 ,0X08 ,0X00 ,0XC0 ,0X01	
					0X00 ,0X00 ,0X00 ,0X00 ,0X01	
					0X00 ,0X12 ,0X00 ,0X20 ...	

- The following row is the parameter data, address 0, which is updated in real time as you can see once you make changes to sliders, control knobs, etc. on a compiled schematic.

## SigmaStudioHelp

Block Write	23:0:11 - 589ms		Parameter Data	0		0X00 ,0X88 ,0X02 ,0XCC 0X00 ,0X00 ,0X00 ,0X00 0X00 ,0X83 ,0X53 ,0X2B 0X0F ,0X0C ,0X05 ,0X5D 0X00 ,0X73 ,0X26 ...	4096

- The next write message is across 5 registers, addresses 2642-2646 (8 total bytes), configuring the RAM, serial output, and Serial Inputs.

Block Write	23:21:14 - 11...	IC 1.-1	2642	0X14 ,0X00 ,0X1C ,0X00 0X00 ,0X00 ,0X00 ,0X00	8

- This block of code is to configure the AD1940's slew ram, at address 2560. Note that this is shown in green because it is a Safeload write.

Safeload Write	23:21:14 - 11...	Slew Ram Data	2560	0X00 ,0X00 ,0X00 ,0X00 ,0X00	320

- The final row is the un-mute message. Observe that the address is again 2642, as with the mute command, but now the data values have changed to un-mute the hardware.

Block Write	23:21:14 - 13...	CoreReg	2642	0X16 ,0X00	2

At this point you can click **Clear All Output Data** button to empty the capture window. Now try moving the EQ slider to see that as the values are changed they are sent to the DSP.

Capture Window							
Mode	Time	Cell Name	Parameter Name	Address	Value	Data	Bytes
Safeload Write	23:42:45 - 22...	Mid EQ1	EQ1940SingleS1081	2	1.024810...	0X00 ,0X83 ,0X2D ,0X01	4
Safeload Write	23:42:45 - 22...	Mid EQ1	EQ1940SingleS1181	3	-1.90526...	0xFF ,0X0C ,0X20 ,0X42	4
Safeload Write	23:42:45 - 22...	Mid EQ1	EQ1940SingleS1281	4	0.899957...	0X00 ,0X73 ,0X31 ,0XCF	4
Safeload Write	23:42:45 - 22...	Mid EQ1	EQ1940SingleS11A1	5	1.905265...	0X00 ,0XF3 ,0XDF ,0XBE	4
Safeload Write	23:42:45 - 22...	Mid EQ1	EQ1940SingleS12A1	6	-0.92476...	0xFF ,0X89 ,0XA1 ,0X30	4
Safeload Write	23:42:45 - 27...	Mid EQ1	EQ1940SingleS1081	2	1.023650...	0X00 ,0X83 ,0X06 ,0XA	4

**Cell Name** - Same name as appears on the visual block in the SigmaStudio schematic tab.

**Parameter Name** - Describes the algorithm, IC type, number of inputs, and for this example the filter coefficient.

**Parameter Address** - The address where data are being sent.

**Parameter Value** - The decimal equivalent value for filter coefficient.

**Parameter Data** - 4-byte hex representation of the filter coefficient. The DSP takes data in 5.23 format, 5 bits to the left of the decimal point and 23 bits to the right. The hex values listed here are the 4 bytes necessary to store the 28 bits. Only 28 bits of the 32 available are used, so the 4 most significant bits of the first hex value are sign-extended from the 4th bit.

© 2006-2007 Analog Devices, Inc. All rights reserved.

## **Export Program and Parameters**

---

Once a project is compiled you can export the data that is sent to the DSP out to disk files. These files, which can be opened in any text editor for analysis, are essential for finding parameters and addresses to integrate into microcontroller code. These files include C/C++ compatible header files defining the projects parameters and registers.

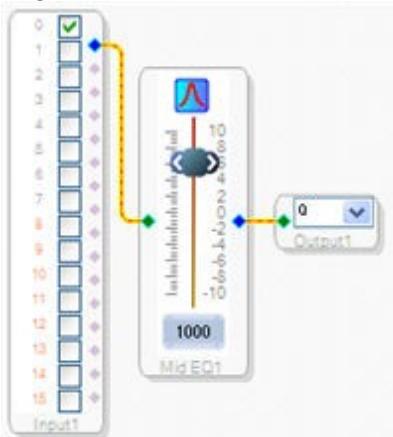


Following project compilation, click the **Export System Files** toolbar button or choose **Action - Export System Files** circled below. (Be sure to re-compile your project each time you wish to generate the parameter file, so you have up-to-date information.)

You will be prompted to enter a location for the files, choose a name and press the **Save** button. Several files will be generated, \*.params and \*.hex files and 2 header files for each IC in the project.

- \*.params** lists all schematic algorithm parameters names, memory addresses, and values.
- \*.hex** lists the parameter memory raw byte values in hexadecimal format, this data is also reflected in the *Parameter Data* field of the params file.
- \*.h** header file which defines all schematic parameter's names, addresses and values.
- \_REG.h** header file contains definitions for all hardware registers and the settings defined in the register control window.

#### Example - for a schematic like this one:



You will see the following in the **\*.params** file:

```

Cell Name      = Mid_EQ1
Parameter Name    = EQ1940SingleS10B1
Parameter Address = 0
Parameter Value     = 1.03666877746582
Parameter Data :
0X00 , 0X84 , 0XB1 , 0X90 ,

Cell Name      = Mid_EQ1
Parameter Name    = EQ1940SingleS11B1
Parameter Address = 1

```

## SigmaStudioHelp

```
Parameter Value    = -1.91309142112732
Parameter Data :
0X0F , 0X0B , 0X1F , 0XD2 ,

Cell Name    = Mid EQ1
Parameter Name    = EQ1940SingleS12B1
Parameter Address = 2
Parameter Value    = 0.896005511283875
Parameter Data :
0X00 , 0X72 , 0XB0 , 0X4F ,

Cell Name    = Mid EQ1
Parameter Name    = EQ1940SingleS11A1
Parameter Address = 3
Parameter Value    = 1.91309142112732
Parameter Data :
0X00 , 0XF4 , 0XE0 , 0X2E ,

Cell Name    = Mid EQ1
Parameter Name    = EQ1940SingleS12A1
Parameter Address = 4
Parameter Value    = -0.932674407958984
Parameter Data :
0X0F , 0X88 , 0X9E , 0X20 ,

Parameter data for: IC 1 (Hexadecimal format starting at parameter
address 0)
See also C:\Design 1.hex file
0X00 , 0X84 , 0XB1 , 0X90 ,
0X0F , 0X0B , 0X1F , 0XD2 ,
0X00 , 0X72 , 0XB0 , 0X4F ,
0X00 , 0XF4 , 0XE0 , 0X2E ,
0X0F , 0X88 , 0X9E , 0X20 ,

Parameter data for: IC 1 (Binary format starting at parameter
address 0)
00000000 10000100 10110001 10010000
00001111 00001011 00011111 11010010
00000000 01110010 10110000 01001111
00000000 11110100 11100000 00101110
00001111 10001000 10011110 00100000
```

**Note:** The **hex (\*.hex)** file contains only the following:

```
0X00 , 0X84 , 0XB1 , 0X90 ,
0X0F , 0X0B , 0X1F , 0XD2 ,
0X00 , 0X72 , 0XB0 , 0X4F ,
0X00 , 0XF4 , 0XE0 , 0X2E ,
0X0F , 0X88 , 0X9E , 0X20 ,
```

© 2006-2007 Analog Devices, Inc. All rights reserved.

## System Implementation

---

When you have finished designing your schematic and have compiled your project, you are ready to translate your design into microcontroller code, following these basic steps:

1. Obtain parameters and addresses from SigmaStudio (export the .params, .hex, and .h files).
2. Parse the file with parameter information.
3. Incorporate (integrate) into microcontroller code.

To access the parameters and addresses from SigmaStudio, you must have already compiled your project and exported the parameter files. After doing that, you'll have two parameter files:

- *yourfilename.hex* - hex values of parameters to be written to DSP.
- *yourfilename.params* - detailed file with addresses and values of all parameters.

You can view sample \*.params and \*.hex files in the Export Program and Parameters topic.

Following is an example of how you can integrate the parameters into microcontroller code (AD1940). The highlighted functions in the main function are also defined:

```
#include "prog_data.h"
#include "param_data.h"
#include "Design_IC_1.h" // *** This file is generated by SigmaStudio ***
#include "Design_IC_1_REG.h" // *** This file is generated by SigmaStudio ***
#include "comms.h"
#define byte unsigned char

// Microcontroller Code MAIN program
int main(void)
{
    // Setup SPI port !! this is hardware-specific
    SpiInit();
    Mute();

    // Write a program to DSP
    SpiWrite(PROG_START_ADDR, PROGRAM_DATA, PROG_LENGTH*6);

    // Wait some time
    Wait(...);

    // Write a parameter file
    SpiWrite(PARAM_START_ADDR, PARAM_DATA, PARAM_LENGTH*4);
    Wait(...);
    UnMute();

    // Write a single value (0.242) to specified address
    // The address, MODULE_Main_Single1_VALUE, is defined in the
    // SigmaStudio generated header file (Design_IC_1.h).
    ParamWrite(MODULE_Main_Single1_VALUE, 0.242);

    // Example safeload of single parameter, will safeload 1.999 in 5.23
    // format to address 2
    // This function includes the initiate safeload to parameter RAM
    SafeloadSingleParamWrite(2, 1.999);

    // Example of a multiple parameter safeload (i.e. filter coefficients)
    // - loads the data to each of the five safeload locations,
```

```

// - then initiates the safeload
SafeloadMultipleParamWrite(MODULE_Main_MidEQ1_Algo_Stage0_B0_ADDRESS, 1.999, 0);
SafeloadMultipleParamWrite(MODULE_Main_MidEQ1_Algo_Stage0_B1_ADDRESS, -.988, 1);
SafeloadMultipleParamWrite(MODULE_Main_MidEQ1_Algo_Stage0_B2_ADDRESS, 1.999, 2);
SafeloadMultipleParamWrite(MODULE_Main_MidEQ1_Algo_Stage0_A1_ADDRESS, -.999888, 3);
SafeloadMultipleParamWrite(MODULE_Main_MidEQ1_Algo_Stage0_A2_ADDRESS, 1.999, 4);
InitSafeloadParam();

    return 0;
}

// Microcontroller Code Wait function
void Wait(float time) //time to wait in milliseconds
{
    long cycles;
    cycles = time / 1.422e-3; //assuming 45 MHz uC core clock speed
    while (cycles != 0)
    {
        --cycles;
    }
}

// Microcontroller Code Mute function
void Mute()
{
    byte corecontrol[2] = {0x00, 0x00};
    //Write core control register
    SpiWrite(CORE_CONTROL_REG, corecontrol, 2);
    //Wait 20us
    Wait(20);
}

// Microcontroller Code UnMute function
void UnMute()
{
    byte corecontrol[2] = {0x02, 0x00};
    //Write core control register
    SpiWrite(CORE_CONTROL_REG, corecontrol, 2);
    //Wait 20us
    Wait(20);
}

// Microcontroller SpiWrite function
// The function takes the start address to be written to, an array of
// data, and the length of the array (in bytes).
void SpiWrite(short address, unsigned char* data, int length)
{
    int i = 0;
    byte address_hi = 0; // register/RAM address high byte
    byte address_lo = 0; // register/RAM address lo byte
    address_lo = (byte)address; // get low byte of register/RAM address
    address_hi = (byte)(address>>8); // get high byte of address shift left by 8 bits

    // Fill in your write function here
    SPI_TX_REG = 0x00; // Write to DSP at chip address 0
    SPI_TX_REG = address_hi; // Write high address byte
    SPI_TX_REG = address_lo; // Write lo address byte
    for (i=0; i<length; i++) // Write specified length of data
    {
        SPI_TX_REG = data[i];
    }

    // Note: After each write to SPI_TX_REG, you may need to verify that the buffer
    // has been emptied (meaning the write has completed). This is hardware-specific.
}

// Microcontroller Code - functions, write a single parameter

```

## SigmaStudioHelp

```
void ParamWrite(short address, float param)
{
    byte param_hex[4] = {0x00, 0x00, 0x00, 0x00} ;
    // Convert decimal parameter value to 5.23 format
    To523(param, param_hex);
    SpiWrite(address, param_hex, 4);
}

// Microcontroller Code functions, safeload: single parameter write
void SafeloadSingleParamWrite(short address, float param)
{
    byte param_hex[4] = {0x00, 0x00, 0x00, 0x00};

    //convert decimal paramater value to 5.23 format
    To523(param, param_hex);
    byte paramex_hex[5];
    paramex_hex[0]=0x00;
    paramex_hex[1]=param_hex[0];
    paramex_hex[2]=param_hex[1];
    paramex_hex[3]=param_hex[2];
    paramex_hex[4]=param_hex[3];

    SafeParam(SAFELOAD_DATA_0, paramex_hex);
    SafeAddr(SAFELOAD_ADDRESS_0, address);
    InitSafeloadParam();
}

// Microcontroller Code functions, safeload: multiple parameter write
// Note: After you call this function, call InitSafeloadParam ONCE.
void SafeloadMultipleParamWrite(short address, float param, int location)
{
    byte param_hex[4] = {0x00, 0x00, 0x00, 0x00};

    //convert decimal paramater value to 5.23 format
    To523(param, param_hex);
    byte paramex_hex[5];
    paramex_hex[0]=0x00;
    paramex_hex[1]=param_hex[0];
    paramex_hex[2]=param_hex[1];
    paramex_hex[3]=param_hex[2];
    paramex_hex[4]=param_hex[3];

    SafeParam(SAFELOAD_DATA_0+location, paramex_hex);
    SafeAddr(SAFELOAD_ADDRESS_0+location, address);
}

// Microcontroller Code - functions, safeload: Initiate the safeload
// write to parameter RAM
void InitSafeloadParam()
{
    byte corecontrol[2] = {2, 0};

    //Read contents of core control register
    SpiRead(CORE_CONTROL_REG, corecontrol, 2);

    //Set Initiate Safe Transfer to Param RAM bit
    corecontrol[1] = corecontrol[1] | 0x10;

    //Write back to core control register
    SpiWrite(CORE_CONTROL_REG, corecontrol, 2);
    Wait(20); //Wait 20us
}

//Microcontroller Code - functions, SpiRead
void SpiRead(short address, unsigned char* data, int length)
```

```

{
    int i = 0;
    byte address_hi = 0; //DSP register/RAM address high byte
    byte address_lo = 0; //DSP register/RAM address lo byte

    //get low byte of register/RAM address
    address_lo = (byte)address;
    //get high byte of address shift, left by 8 bits
    address_hi = (byte)(address>>8);

    //Fill in your write function here
    SPI_RX_REG = 0x01;           //Write to DSP and set RW bit to 1
    SPI_RX_REG = address_hi;    //Write high address byte
    SPI_RX_REG = address_lo;    //Write lo address byte

    for (i=0; i<length; i++) //read specified length of data
    {
        SPI_RX_REG = data[i];
    }

    // Note: After each write to SPI_TX_REG, you may need to verify that the buffer has
    // been emptied (meaning the write has completed). This is hardware-specific.
}

//Microcontroller Code - functions, Safeload: SafeAddr
void SafeAddr(short safe_addr, short param_addr)
{
    byte safe_addr_lo;
    byte safe_addr_hi;

    //get low byte of register/RAM address
    safe_addr_lo=(byte)safe_addr;
    //get high byte of address shift left by 8 bits
    safe_addr_hi=(byte)(safe_addr>>8);

    byte param_addr_lo;
    byte param_addr_hi;

    //get low byte of parameter address
    param_addr_lo=(byte)param_addr;
    //get high byte of address shift left by 8 bits
    param_addr_hi=(byte)(param_addr>>8);

    SPI_TX_REG = 0x00; // Write to DSP at chip address 0
    SPI_TX_REG = safe_addr_hi; //Write high byte of safeload address
    SPI_TX_REG = safe_addr_lo; //Write low byte of safeload address

    SPI_TX_REG = param_addr_hi; //Write high byte of parameter address
    SPI_TX_REG = param_addr_lo; //Write low byte of parameter address
}

//Microcontroller Code - functions, Safeload: SafeParam
void SafeParam(short safe_param, unsigned char* param_data)
{
    byte safe_param_lo;
    byte safe_param_hi;

    //get low byte of register/RAM address
    safe_param_lo=(byte)safe_param;
    //get high byte of address shift left by 8 bits
    safe_param_hi=(byte)(safe_param>>8);

    SPI_TX_REG = 0x00; //Write to DSP at chip address 0
    SPI_TX_REG = safe_param_hi; //Write high byte of Safeload address
}

```

## SigmaStudioHelp

```
SPI_TX_REG = safe_param_lo; //Write low byte of Safeload address

SPI_TX_REG = param_data[4]; //Write parameter byte 4 (MSBs)
SPI_TX_REG = param_data[3]; //Write parameter byte 3
SPI_TX_REG = param_data[2]; //Write parameter byte 2
SPI_TX_REG = param_data[1]; //Write parameter byte 1
SPI_TX_REG = param_data[0]; //Write parameter byte 1 (LSBs)
}

// Note: SafeParam and SafeAddr must be executed together.

// Microcontroller code - functions: convert a float number to 5.23 format
void To523(float param_dec, byte* param_hex)
{
    long param223;
    long param227;

    //multiply decimal number by 2^23
    param223 = param_dec * (1 << 23);
    //convert to positive binary
    param227 = param223 + (1 << 27);

    param_hex[3]=(byte)param227; //get byte 3 (LSBs) of parameter value
    param_hex[2]=(byte)(param227>>8); //get byte 2 of parameter value
    param_hex[1]=(byte)(param227>>16); //get byte 1 of parameter value
    //get byte 0 (MSBs) of parameter value
    param_hex[0]=(byte)(param227>>24);
    //invert sign bit to get correct sign
    param_hex[0] = param_hex[0] ^ 0x08;
}
```

---

The program and parameter header files can be defined using the data generated from the Export System Files command.

You might define something like the following:

```
// For the AD1940, the length of the parameter data array is 1024 coefficients x 4 bytes
const byte PARAM_DATA[4096] =
{
    0x00 , 0x80 , 0x00 , 0x00 ,
    0x00 , 0x00 , 0x00 , 0x00 ,
    0x00 , 0x00 , 0x90 , 0xce ,
    0x00 , 0x7f , 0xda , 0xa4 ,
    0x00 , 0x07 , 0x1b , 0x35 ,
    ...
}

// For the ADAV4xx, the length of the program data array is 2560 coefficients x 6 bytes
const byte PROGRAM_DATA[12336] =
{
    0x00, 0x00 , 0x00 , 0x00 , 0x00 , 0x01 ,
    0x00, 0x00 , 0x00 , 0x00 , 0xe8 , 0x01 ,
    0x00, 0x00 , 0x00 , 0x00 , 0x00 , 0x01 ,
    0x00, 0x00 , 0x08 , 0x00 , 0xe8 , 0x01 ,
    0x00, 0x00 , 0x00 , 0x00 , 0x00 , 0x01 ,
    0x00, 0x30 , 0xa8 , 0x00 , 0xe8 , 0x01 ,
    ...
}
```

The program header file should also define the program and parameter start addresses and lengths, and for

the AD1940 the safeload addresses. Refer to the your part's data sheet for this information.

```
#define PROG_START_ADR 1024
#define PROG_LENGTH 2560
#define PARAM_START_ADR 0
#define PARAM_LENGTH 1024

#define SAFELOAD_DATA_0 0x1040
#define SAFELOAD_DATA_1 0x1041
#define SAFELOAD_DATA_2 0x1042
#define SAFELOAD_DATA_3 0x1043
#define SAFELOAD_DATA_4 0x1044

#define SAFELOAD_ADDRESS_0 0x1045
#define SAFELOAD_ADDRESS_1 0x1046
#define SAFELOAD_ADDRESS_2 0x1047
#define SAFELOAD_ADDRESS_3 0x1048
#define SAFELOAD_ADDRESS_4 0x1049
```

Following is an example of a typical function to change a parameter. It may be a simple function, using ParamWrite(&), or you can write a higher-level function to do the same thing:

```
//toggle the block to turn ON (no clicks) using target/slew RAM
static MyBlockEnable
{
    SafeloadTargetRAM(MYBLOCK_DISABLE, {0x00,0x40,0x00,0x00,0x00});
    SafeloadTargetRAM(MYBLOCK_ENABLE, {0x00,0x40,0x80,0x00,0x00});
}
```

Note: MYBLOCK\_DISABLE, MYBLOCK\_ENABLE would be addresses of the target RAM parameters defined int the .h file generated by SigmaStudio !! for example:

```
// Module Single 1 - Single Volume Control
#define MODULE_Main_Single1_COUNT      1
#define MODULE_Main_Single1_DEVICE     IC1
#define MODULE_Main_Single1_ADDRESS   2560
#define MODULE_Main_Single1_TYPE      SIGMASTUDIO_SLEW
#define MODULE_Main_Single1_VALUES    SIGMASTUDIO_SLEW(0x0080144961)
```

A example function to write to target RAM via **safeload**:

## SigmaStudioHelp

```
static SafeloadTargetRAM(TargetRamAddress, TargetRamData)
{
    SpiWrite(SAFELOAD_ADDR_0, TargetRamAddress); //send target ram address
    SpiWrite(SAFELOAD_DATA_0, TargetRamData);

    //send target ram data
    byte corecontrol[2] = {2, 0};
    SpiRead(CORE_CONTROL_REG, corecontrol, 2);

    //Read contents of core control register
    corecontrol[1] = corecontrol[1] | 0x20;

    //Set Initiate Safe Transfer to Target RAM bit
    SpiWrite(CORE_CONTROL_REG, corecontrol, 2);

    //Write back to core control register
    Wait(&); // Wait & cycles
}
```

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Sampling Rate Considerations

The number of instructions available on the DSP (MIPS) depends on the sampling frequency selected. AD1940/1941 DSPs can perform:

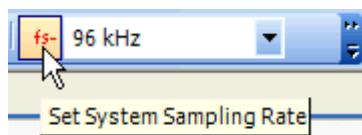
- 1536 instructions at 32 kHz
- 1536 instructions at 44.1 kHz
- 1536 instructions at 48 kHz
- 768 instructions at 96 kHz
- 384 instructions at 192 kHz

The sample rate should be set by the program-length bits in the core control register. SigmaStudio defaults to 44.1-kHz sampling, the CD red-book standard. If you want to use another rate, select it prior to your design from the **New Item Sample Rate** drop-down list in the top toolbar. Now any blocks added to your schematic will be set to that sampling rate.

You also can change the sampling rate mid-design by either setting the sample rate for all blocks in the design, or you can change the sample rate of each input individually.

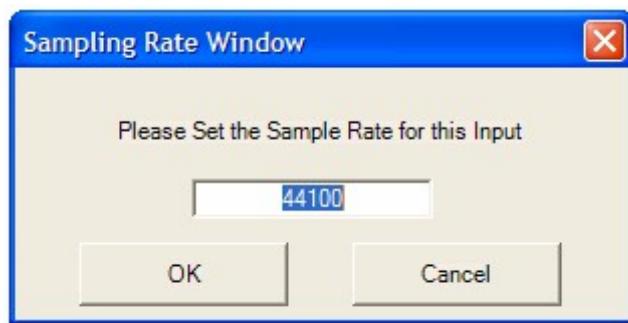
### 1. To change the sampling rate for all schematic blocks:

Select the desired sampling rate from the **New Item Sample Rate** drop-down list, then press the **Set System Sampling Rate**.



### 2. To change sampling rate individually for each input:

Right-click on the input or source blocks and select **Set Sample Rate** from the menu which will open the sampling rate window shown below. Enter the desired sampling rate and press OK.



At this point only the input or source blocks rate has been changed. To apply this change to all down-stream blocks, click the **Propagate Sampling Rate** button, to the right of the

toolbar drop-down list:



## Numeric Formats

DSP systems use a standardized numeric format. Fixed-point numbers are formatted A.B, where A is the number of bits to the left of the decimal point (the integer part) and B is the number of bits to the right of the decimal point (the fractional part).

The AD1940/AD1941 use the same numeric format for both the coefficient values (stored in the parameter RAM) and the signal-data values, as follows:

Numerical Format: 5.23

Range: -16.0 to (+16.0 - 1 LSB)

### Examples

1000 0000 0000 0000 0000 0000 = -16.0

1110 0000 0000 0000 0000 0000 = -4.0

1111 1000 0000 0000 0000 0000 = -1.0

1111 1110 0000 0000 0000 0000 = -0.25

1111 1111 1111 1111 1111 1111 = (1 LSB below 0.0)

0000 0000 0000 0000 0000 0000 = 0.0

0000 0010 0000 0000 0000 0000 = 0.25

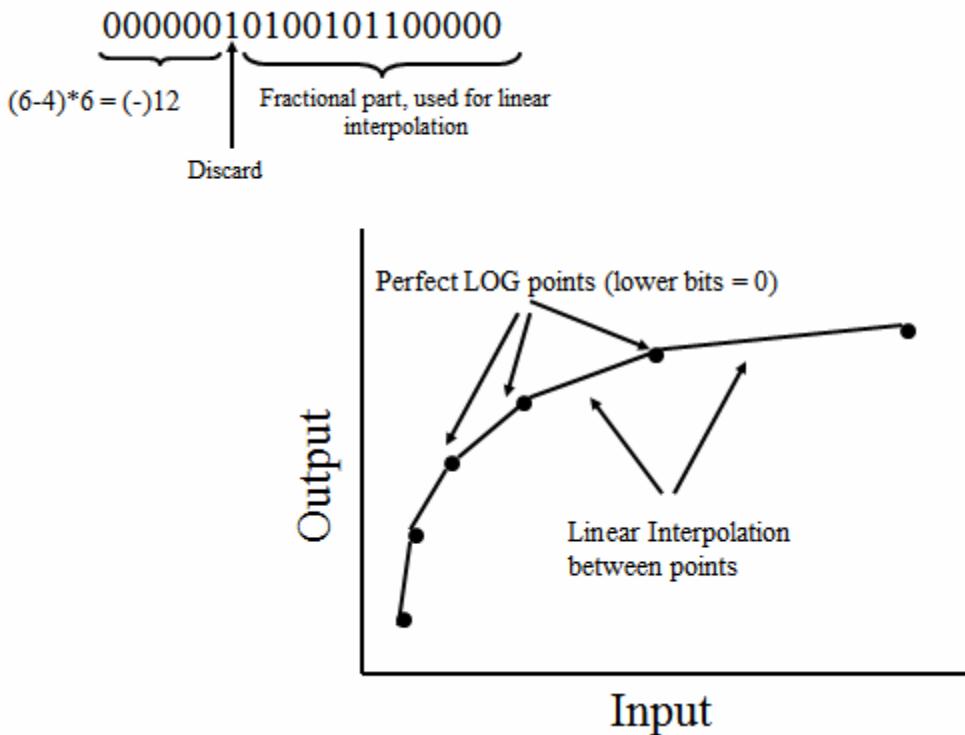
0000 1000 0000 0000 0000 0000 = 1.0 (0 dB full scale)

0010 0000 0000 0000 0000 0000 = 4.0

0111 1111 1111 1111 1111 1111 = (16.0 - 1 LSB)

Concept for dB conversion:

1. Count the number of leading 0s (minus 4) and multiply by 6; this is the integer part of the dB scale.
2. Use the bits after the leading 1 to linearly interpolate between the 6 dB points.



In order to get a linear value into a hex/binary number that can be written to the SigmaDSP, and vice versa, it is important to understand number conversions.

The actual level parameters that you need to write are only the last 28 bits, and this value is simply a linear gain value. Using -40 dB as an example, convert this value to a linear value using the standard dB equation:

$$20\log_{10}(x/1) = -40\text{dB} \quad x = .01$$

Take this linear value and multiply by  $2^{23}$  in order to get the decimal representation of the value in hex, because it is a 5.23 value.

$$.01 * 2^{23} = 83886.08.$$

Now take the integer part of this result and convert 83886 to hex, and you will get 0x00147AE.

The output of some blocks is a 5.19 number. The formula to determine the dB-output value from the readback value is the following:

$$\text{dB\_value} = 96.32959861 * (\text{readback\_value} / 2^{19} - 1)$$

## Hierarchy Boards

---

For complex system designs, it can become difficult to manage a large number schematic block's in a single design window and it is desirable to have additional schematic workspace and the ability to organize design components. Schematic Hierarchy Boards allows you to create multiple layers within the schematic design window expanding the schematic workspace.

- **Create Hierarchy Boards**
- **Hide, Freeze, and Navigation**
- **Hierarchy Board Files**

A schematic Hierarchy Board consists of a Hierarchy Board Block and an associated schematic tab (or layer), located at the bottom of the schematic window. Hierarchy boards support Copy/Paste, schematic settings, and Undo operations. All projects include the **Main** tab by default. The Main hierarchy board is the root layer for any additional user defined Hierarchy boards and cannot be renamed.

---

### Create a Hierarchy Board:

Select the *Schematic Design* category in the Tree ToolBox window (or the *System* Category in the ToolBox), and drag-and-drop a **Hierarchy Board** block into the schematic workspace.

Along with the hierarchy board block, a hierarchy tab is created, located at the bottom of the Schematic design window. The default name is *Board1* and the corresponding tab has the same name as the board block.

Note: It is possible to create nested hierarchy boards (a hierarchy board with-in another hierarchy board's tab). To create a nested board, first select a hierarchy tab at the bottom of the Schematic window, then drag-and-drop a hierarchy board block. If you hover the mouse button over a board tab, the tool-tip will display the "path" of the board (the location of the board block relative to Main). This is helpful when several board's have the same name, which is common when using board files.

---

See the Hierarchy Board, Hierarchy Input, and Hierarchy Output topics for more information.

### **Hide, Freeze, and Navigation**

To locate the hierarchy board block for a particular hierarchy tab, select the hierarchy tab at the bottom of the Schematic window, then double click the tab. This will display the design layer that contains the associated Hierarchy board block. Similarly, you can double click a hierarchy board block to display its design layer.

Right-click the hierarchy block to display the block's context menu. This menu includes commands for view the board layer, hiding the board's tab, and freezing the board design.

<b>View</b>	The view command will select the blocks hierarchy tab, displaying the hierarchy layer in the schematic window. Note: you can also view the hierarchy layer by <b>double-clicking</b> the hierarchy board block or by clicking on the hierarchy tab directly.
<b>Hide/Show</b>	<b>Hide</b> - Hides a Hierarchy board's tab with an optional password. This allows you to create proprietary design components in a board and hide the implementation from other users of the project file. <b>Show</b> - When a board's tab is hidden the Show menu item will un-hide the board's tab, prompting for a password if one was entered when hiding. Note: Hide uses an un-encrypted password and offers only a minimal level of security. This feature is not guaranteed to protect your IP from unscrupulous individuals.
<b>Freeze</b>	Freezing a hierarchy board secures the design -- no additional changes or edits can be made to the hierarchy layer. This can safeguard your design against accidental changes. When freezing you can enter a password (optional) which will prevent un-authorized changes to the design.
<b>Copy/Paste</b>	Cut/Copy of a Hierarchy block will copy the entire hierarchy layer to the clipboard, including any nested hierarchy boards. Pasting a hierarchy block will insert a new tab or tabs.

---

### **Hierarchy Board Files**

The entire contents of a Hierarchy Board layer (inputs, outputs, blocks, wires, and nested hierarchy boards) along with the board's Hide/Freeze state can be saved to a Hierarchy Board File (\*.dspbrd). This allows you to create reusable design components and import them into new SigmaStudio projects.

#### **To Create a Board File:**

Right-Click on the hierarchy board block and select **File - Save Board** from the pop-up menu. SigmaStudio board files use the \*.dspbrd file extension.

#### **To Load a Board File:**

1. Insert a new hierarchy board block into the Schematic window.
2. Right-Click the hierarchy board block and select **File - Load Board...** from the menu.
3. Navigate to a previously saved board file (\*.dspbrd) and press **Open**.

The loaded board will be name is changed to match the saved board name, and all the blocks contained in the hierarchy board's layer are loaded into a new tab.  
In the example below, the Mixer block was saved to a file and then loaded into a new hierarchy board block, Mixer\_2.

Note, the *Mixer* and *Mixer\_2* schematic layers are identical.

---

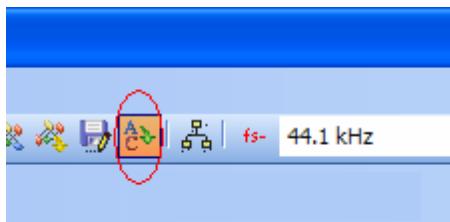
© 2006-2007 Analog Devices, Inc. All rights reserved.

## Schematic Actions and Commands

### Allow Real-Time A/B Testing Between Projects

Suppose you are viewing your design A and listening to its audio flow, and you wonder how your design B (also open) sounds in comparison.

Press the real-time A/B-testing icon and when you switch between the 2 projects, SigmaStudio automatically downloads the design B or design A program to the DSP without needing to recompile.



In other words, clicking the icon lets you switch among any number of open compiled projects without having to recompile each. (The function is very useful: without it, you must recompile a project every time you bring it to the front even if you haven't made any changes to the schematic.)

To switch between open projects, press **Ctrl+Tab** or select a project name from the main **Window** menu.

## Copy/Paste

SigmaStudio supports clipboard operations in the Schematic Tab window:

### **Copy:**

To Copy to the clipboard, select a block or blocks in the schematic tab, and press **Ctrl+C** or select **Edit - Copy** from the application menu. You can also right-click on a block and select copy from the context menu. Selected wires will only be copied if both the source and destination blocks are included in the selection.

It is also possible copy blocks within the schematic window by holding down the **Ctrl** key, clicking on a block in the selection, and then dragging the selection (while holding down the left mouse button and Ctrl key).

### **Cut:**

The Cut operation removes objects from the schematic design and copies them to the clipboard. To Cut the schematic selection, press **Ctrl+X** or select **Edit - Cut** from the application menu.

### **Paste:**

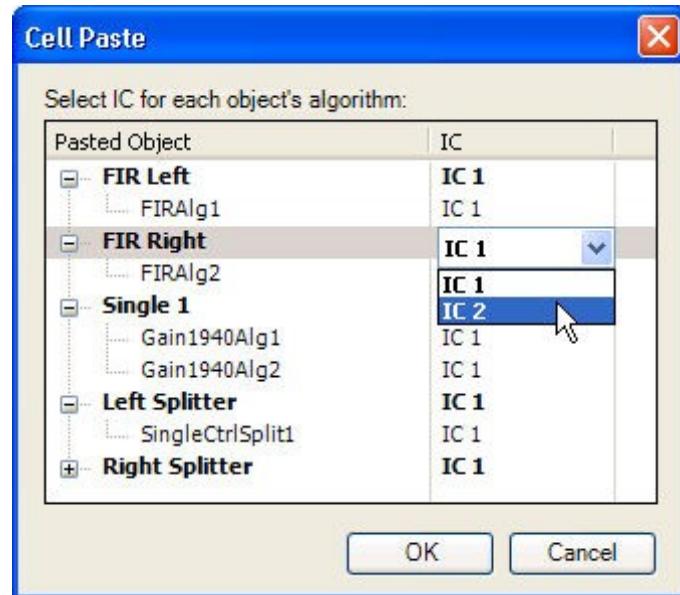
Blocks and wires on the can be pasted from the clipboard within the same schematic window or to another project's schematic. It is also possible to paste objects from one Hierarchy board to another. To paste the contents of the clipboard either: press **Ctrl+V**, select **Edit - Paste** from the application menu, or right-click in the Schematic tab window and select **Paste** from the menu.

**Note:** The names of the pasted blocks may be different, blocks within each hierarchy board must have unique names.

### **Paste Special:**

By default, a pasted block's algorithm(s) are assigned to the same IC as the source block.

For designs that use multiple processors it is possible to specify the target DSP IC by using the Paste Special operation. The Paste Special command can be accessed in the main application menu, **Edit - Paste Special...** This will open the Paste Special dialog where each block and algorithm in the clipboard is listed in the left column. In the right column, select the desired DSP IC to associate with each block or algorithm.



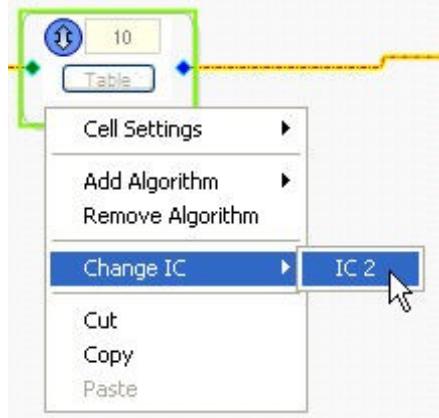
**Note:** this dialog will be opened automatically when pasting between projects that have different IC types or names, which requires the DSP associations to be defined.

## Change IC

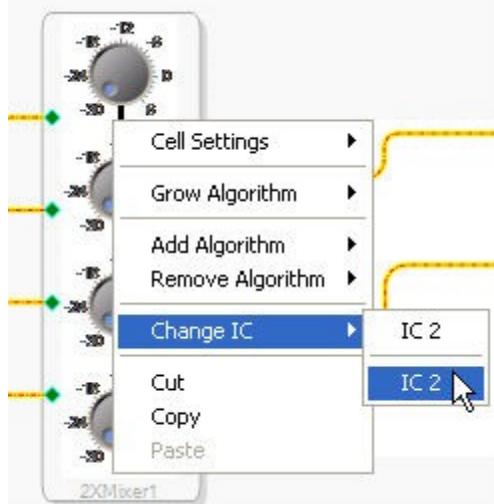
Each Schematic block contains one or more algorithms, and each algorithm is associated with a particular DSP processor in the Hardware Configuration Tab. In designs that contain more than one processor, it is possible to change the IC assignment for existing schematic algorithm.

### To Change IC assignment:

Right-click on a block and select **Change IC** from the pop-up menu, the available processors that support the algorithm will be listed in the menu:



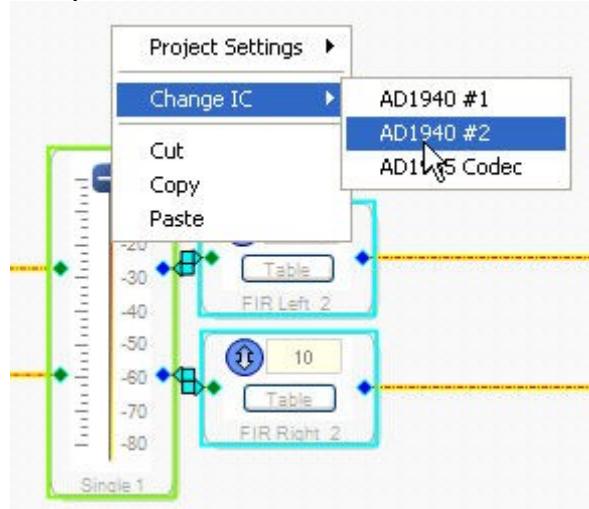
If a block contains multiple algorithms, there will be a separate Change IC menu choice for each block algorithm. If you want to change the IC assignment for all of the block's algorithms in a single operation see below.



### To Change IC assignment for all selected blocks:

It is possible to change the IC for all selected blocks and algorithms in a single operation. First, select multiple blocks that require IC re-assignment. Secondly, right-click in the

Schematic tab window and select **Change IC** from the menu (all processor ICs will be listed).

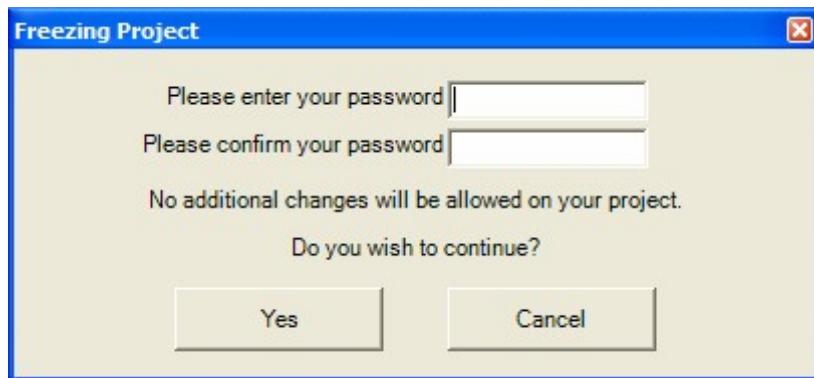


**Note:** If you select an IC that does not support all of the selected blocks' algorithms, an error dialog will be displayed and only the supported algorithms will be re-assigned to the selected IC. Also note, it is not possible to perform the Change IC command for any Input or Output blocks.

## Freeze Schematic

Freezing your project lets you to secure the current design -- no additional changes or edits can occur. This can safeguard your design against accidental changes and also provide access control.

Click the **Freeze Selected Schematic** toolbar button or select **Action - Freeze Schematic** from the main menu. The project freeze window will open, allowing you to enter an optional password:



If you want to secure this design, enter a password and click Yes, otherwise leave the password field blank and just hit Yes.

To make further changes to your design, you will have to select **Freeze Selected Schematic** again and enter the password to un-freeze it.

## Schematic Settings (Presets)

Settings are a snapshot of all control values (knobs, sliders, spin-controls, and combo-boxes) which are saved to a settings file (\*.bin). Settings include: the block(s) name(s), all control values, and the min/max and step size for any knobs or sliders. Settings files can be saved and recalled within a project and imported or exported between projects.

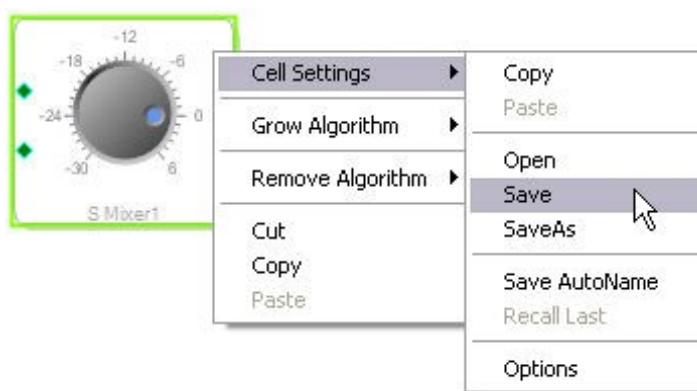
It is possible to copy/paste settings among blocks, hierarchy boards, and projects. When pasting settings between projects or hierarchy boards, the blocks names are used for identification; pasted settings are applied to blocks that have the same name in both the source and destination schematics.

Settings can be saved for:

- **Individual Schematic Blocks (Cell Settings)**
- **Hierarchy Boards (Board Settings)**
- **Complete Project Schematics (Project Settings)**

### Cell Settings:

To manage settings for an individual block, right click on the block's border or label and select **Cell Settings** from the pop-up menu:



- |               |  |
|---------------|--|
| <b>Copy</b>   | Copy all the values from this block's controls to the clipboard.   |
| <b>Paste</b>  | Paste the control values from the clipboard to the selected block. Note that it is possible to copy/paste control settings between different types of blocks if they have common controls.   |
| <b>Open</b>   | Load the settings from a previously saved settings file and update this block's controls values.   |
| <b>Save</b>   | Save all control values from this block into a settings file (*.bin file).<br><b>NOTE:</b> If a settings were previously saved or loaded from a file, choosing save will overwrite the existing settings with the current control values; use SaveAs or Save AutoName to generate a new settings file. |
| <b>SaveAs</b> | Save the current settings to a specific file name and/or directory location.   |

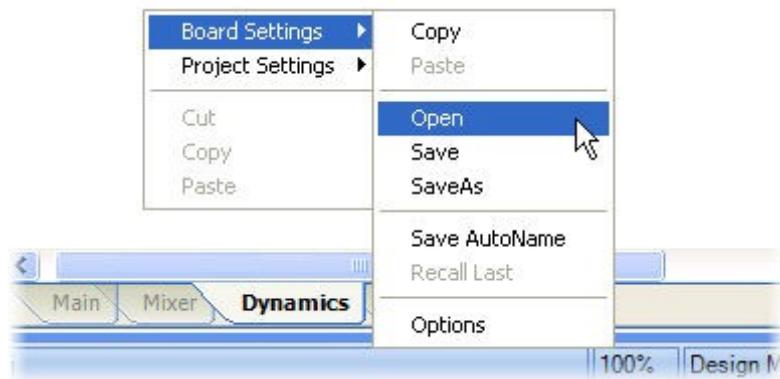
---

<b>Save AutoName</b>	Save the current settings to a new settings file in the default settings directory and automatically generate a unique name for the file.
<b>Recall Last</b>	Recall the most recently saved settings for this block.
<b>Options</b>	Opens the settings options dialog.

---

**Board Settings:**

To manage the settings for a hierarchy board (the settings of all blocks contained in the hierarchy board tab), select the hierarchy tab at the bottom of the schematic window, then right click in the schematic window and select **Board Settings** from the pop-up menu:

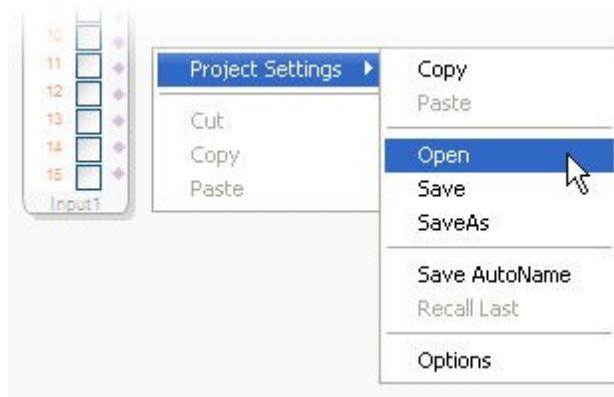


<b>Copy</b>	Copy the values of all controls of all blocks in this hierarchy layer to the clipboard.
<b>Paste</b>	Paste the control values from the clipboard to the blocks in this hierarchy layer. Note that it is possible to copy/paste control settings between different boards.
<b>Open</b>	Load the settings from a previously saved settings file and update the blocks control values.
<b>Save</b>	Save all control values from all the blocks in this hierarchy layer into a settings file (*.bin file). <b>NOTE:</b> If a settings were previously saved or loaded from a file, choosing save will overwrite the existing settings with the current control values; use SaveAs or Save AutoName to generate a new settings file.
<b>SaveAs</b>	Save the current settings to a specific file name and/or directory location.
<b>Save AutoName</b>	Save the current settings to a new settings file in the default settings directory and automatically generate a unique name for the file.
<b>Recall Last</b>	Recall the most recently saved settings for this hierarchy board.
<b>Options</b>	Opens the settings options dialog.

---

**Project Settings:**

To manage the settings for an entire project, right click in the schematic window and select Project Settings from the pop-up menu:

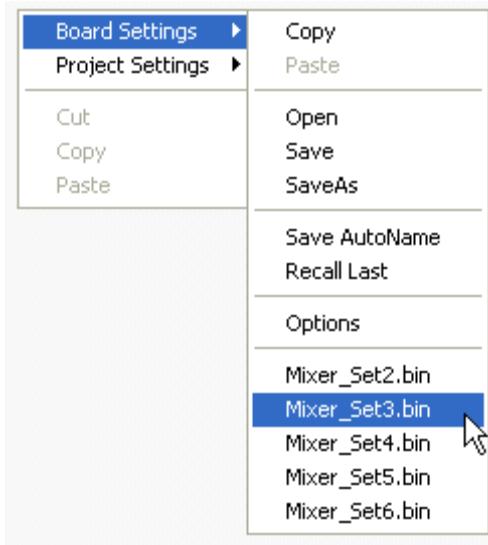


- Copy** Copy the values of all controls of all blocks in the project to the clipboard.
- Paste** Paste the control values from the clipboard to the project's blocks. Note that it is possible to copy/paste control settings between different projects for any blocks with the name and hierarchy layer.
- Open** Load the settings from a previously saved settings file.
- Save** Save all control values from all blocks to a settings file (\*.bin file).  
**NOTE:** If a settings were previously saved or loaded from a file, choosing save will overwrite the existing settings with the current control values; use SaveAs or Save AutoName to generate a new settings file.
- SaveAs** Save the current settings to a specific file name and/or directory location.
- Save AutoName** Save the current settings to a new settings file in the default settings directory and automatically generate a unique name for the file.
- Recall Last** Recall the most recently saved settings for this project.
- Options** Opens the settings options dialog.

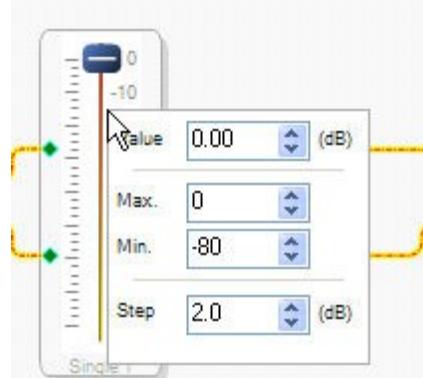
---

### Setting History:

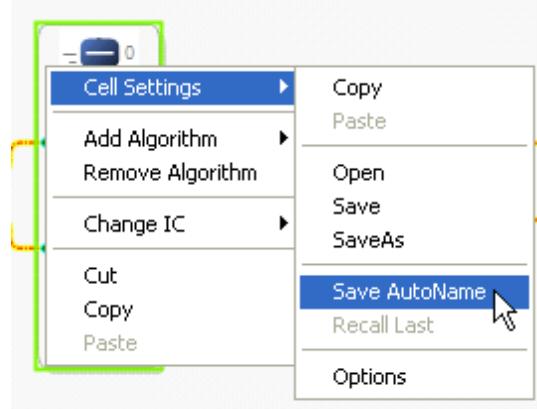
The most recent settings that are saved or opened will be displayed at the bottom of the settings menu as shown below. Note, you can clear the settings history or customize the history size in the settings options dialog.

**Block Settings Example:**

1. Examine the Volume block's values shown below:

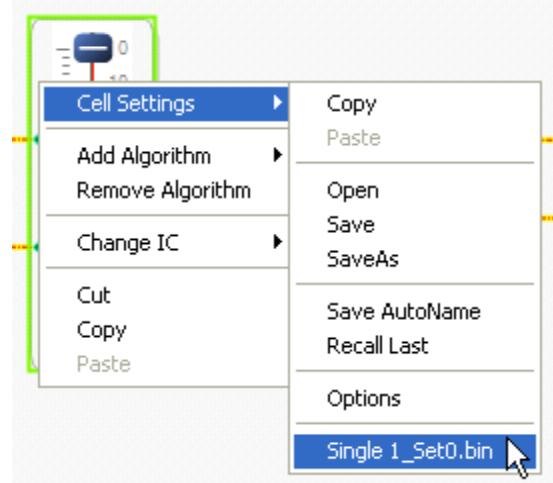


2. Save the block's settings to a settings file:

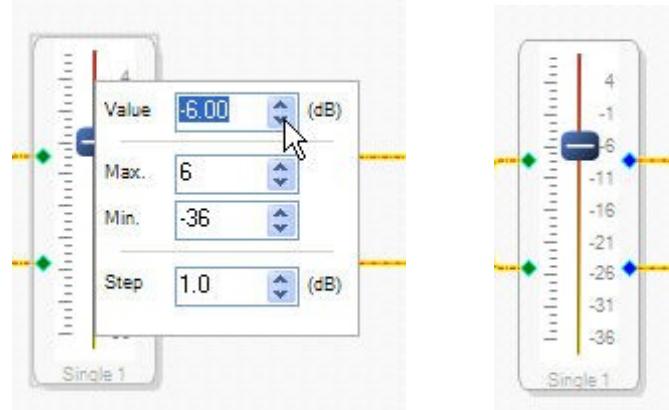


This generates a file in the application's settings directory, in this example the file created is named "*Single 1\_Set0.bin*" and located in the directory C:\Program Files\Analog Devices Inc\SigmaStudio 3.0\Settings\Design 1\.

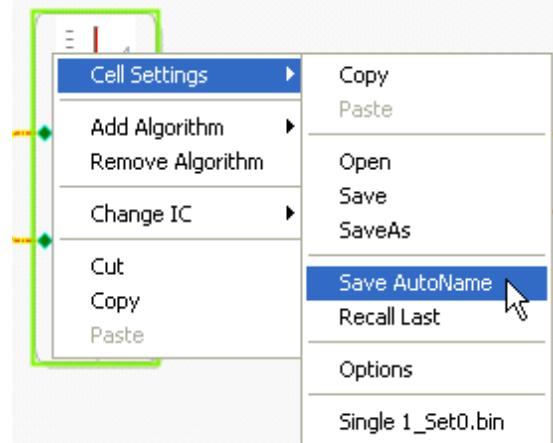
3. If you right click the block again you will see the newly setting file listed:



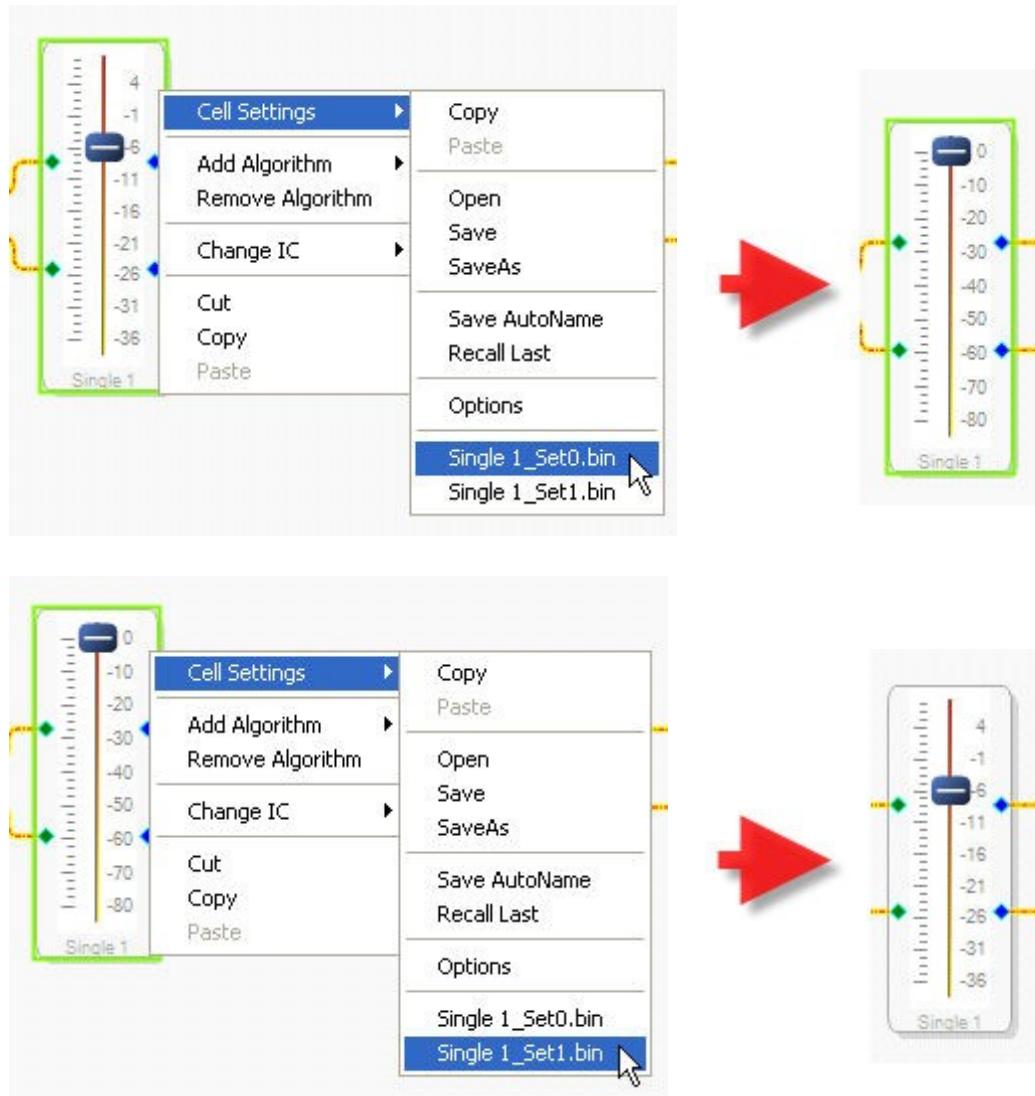
4. Now, change the slider control's values:



5. Next, save the blocks values to another settings file:



6. Now there are two settings files which can be recalled at any time:



7.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

# Toolbox

## Toolbox Introduction

---

The Toolbox contains the building blocks for constructing a system design. Click a category in the *ToolBox* or *Tree ToolBox* window to see the variety of schematic blocks that are available. The available blocks will depend on the DSP processor(s) used in the project.

These Toolbox library contains the following categories:

- System
- ADI Algorithms
- Basic DSP
- Counters
- Dynamics Processors
- Filters
- GPIO Conditioning
- Input / Output
- Level Detectors / Lookup Tables
- Licensed Algorithms
- Mixers / Splitters
- Multiplexers / Demultiplexers
- Sources
- Volume Controls

For tips on getting the most out of your design, see Building Schematics.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

# System (Schematic Design)

## System

---

The **System** library in the ToolBox lists blocks that give you basic control of design layout and response. These are helper blocks, which don't own a DSP; their functions are not driven by DSP algorithms and they don't use any instructions. But they're useful in organizing your designs.

The algorithms for these blocks are built in, don't require that you add an algorithm, and the blocks are not associated with specific DSPs. That association is made at link time, after you've made your wire connections.

The following blocks are available:

- Hierarchy Board
- Hierarchy Input
- Hierarchy Output
- Schematic Terminal
- Simulation Probe
- Simulation Stimuli
- Speaker Response: MLSSA
- T Connection
- User Comment
- User Image

These pages contain information about the individual blocks, but be sure also to take a look at the System Example to see a sample schematic.

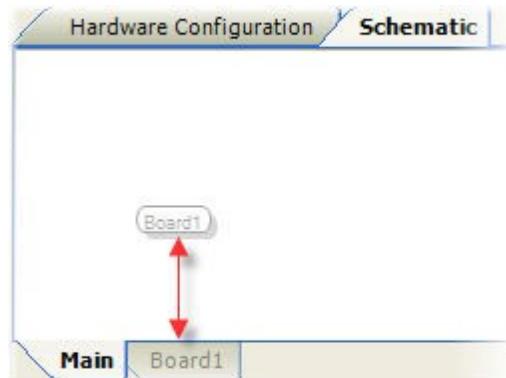
## Hierarchy Board

Schematic Hierarchy Boards allows you to create multiple layers within the schematic design window, providing additional schematic workspace. Hierarchy Boards are helpful for large schematic designs and for organizing design components. Hierarchy boards support Copy/Paste, schematic settings, and Undo operations. In addition Hierarchy boards can be saved to files to create reusable design components.

- For more information about Hierarchy Boards, see the [Hierarchy Boards topic](#).

### To create a Hierarchy Board:

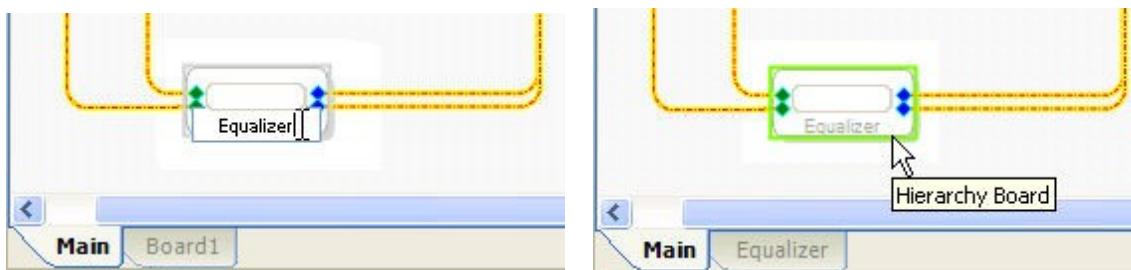
- Drag a Hierarchy Board block into the workspace. It will have no input or output pins.
- The board block is associated with a Hierarchy Tab located at the bottom of the Schematic design window. The default name is Board1, the corresponding tab will have the same name as the board block.



- The Hierarchy Board block provides an interface to route signals in and out of a Hierarchy board layer. To route signals into or out of the board, input and output pins must be created. See [Hierarchy Input](#) and [Hierarchy Output](#) for more information.

### Hierarchy Board Name:

To change the name of hierarchy board block, double click on the block's label and type a new name. This will also change the name displayed in the associated hierarchy tab.



See the Hierarchy Board System Example for a sample hierarchy board design.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Hierarchy Input

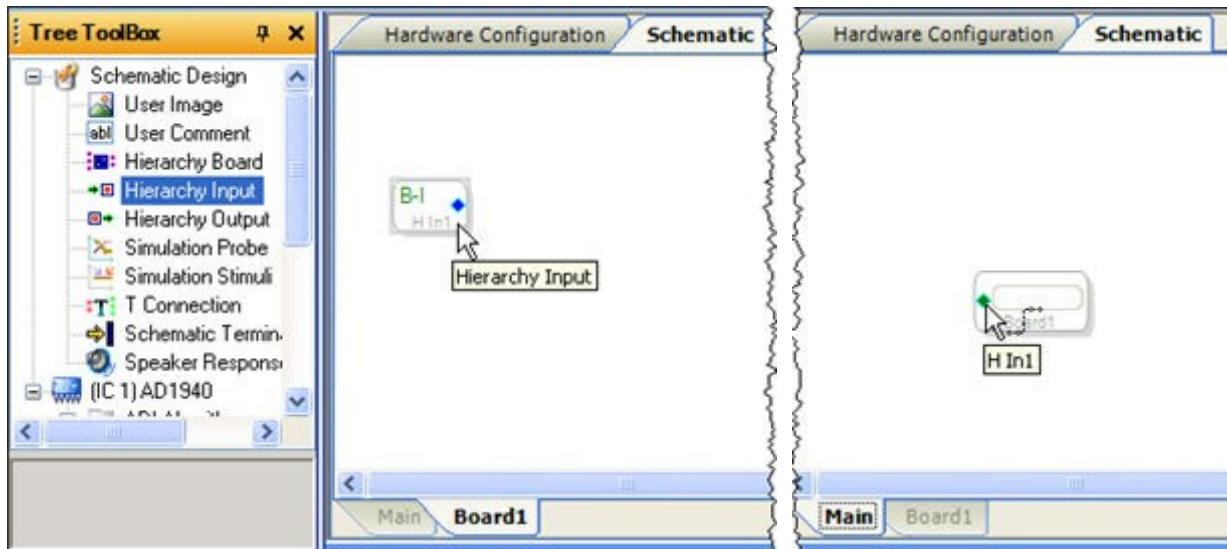
The Hierarchy input block is used to create signal inputs into a Schematic Hierarchy Tab. Hierarchy Boards are helpful for large schematic designs and for organizing design components.



- For more information about Hierarchy Boards, see the Hierarchy Boards topic.

To create a Hierarchy Input:

- Select a hierarchy board tab at the bottom of the Schematic window.
- Drag a **Hierarchy Input** block into the schematic.
- This will create a Hierarchy Input block in the schematic and a corresponding input pin on the Hierarchy Board block.



**Note:** To see determine which Hierarchy input block belongs to a Hierarchy board input pin, hover the mouse pointer above the Hierarchy board pin and the input block's name will be displayed in the tool-tip (as in the image above).

See the Hierarchy Board System Example for a sample hierarchy board design.

## Hierarchy Output

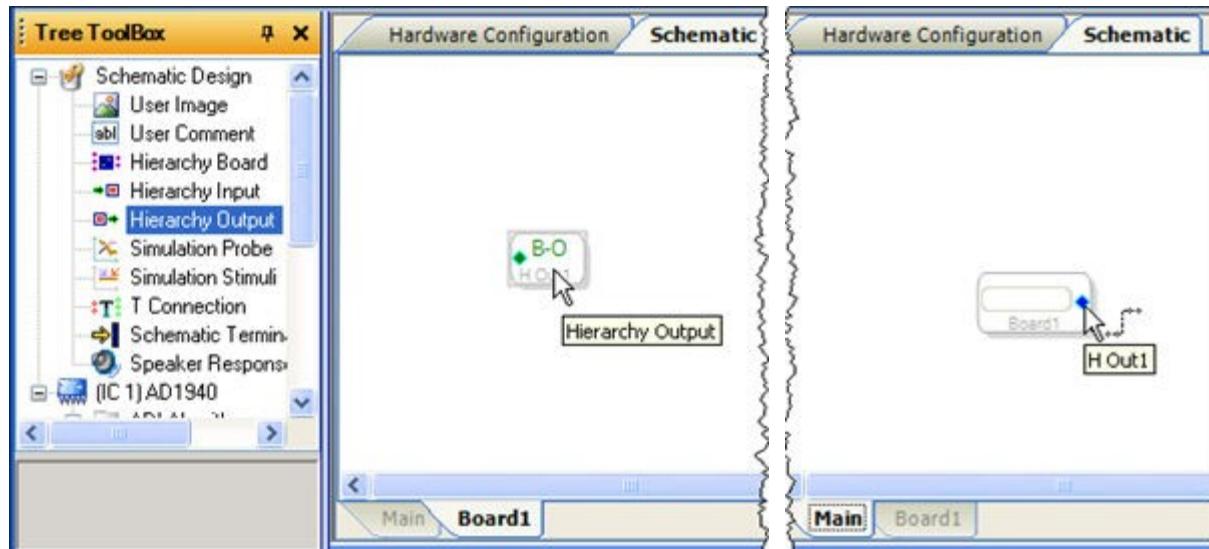
The Hierarchy output block is used to create signal outputs from a Schematic Hierarchy Tab. Hierarchy Boards are helpful for large schematic designs and for organizing design components.



- For more information about Hierarchy Boards, see the Hierarchy Boards topic.

To create a Hierarchy Output:

- Select a hierarchy board tab at the bottom of the Schematic window.
- Drag a **Hierarchy Output** block into the schematic.
- This will create a Hierarchy Output block in the schematic and a corresponding output pin on the Hierarchy Board block.

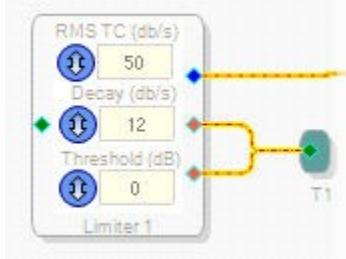


**Note:** To see determine which Hierarchy output block belongs to a Hierarchy board output pin, hover the mouse pointer above the Hierarchy board pin and the output block's name will be displayed in the tool-tip (as in the image above).

See the Hierarchy Board System Example for a sample hierarchy board design.

## Schematic Terminal

Whenever your schematic has output pins that are not necessary in your design, the Schematic Terminal block can function as a sink for the signal flow. Drag it to the workspace for connecting signal(s) from multiple blocks that are not actually sent to an output (see the Example).



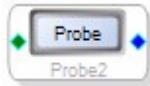
If you do not terminate unused block output pins, the compiler will display an unconnected pin error.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Simulation Probe

The Simulation Probe is used in conjunction with the Simulation Stimuli to plot the frequency response of the system you configure. Click the Probe **block** to bring up the graph (blank), then the Stimulus block to show the curve for your system. See the System Example and second tutorial for more information.

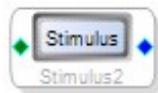


---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Simulation Stimuli

The Simulation Stimuli is used with the Simulation Probe to plot the frequency response of your design. Click the Probe block to bring up the graph (blank), then the Stimulus window to plot your curve. See the System Example and second tutorial for more information.



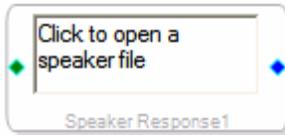
---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Speaker Response : MLSSA

The Speaker Response algorithm uses the MLSSA (maximum-length-sequence system analyzer, an industry standard) loudspeaker measurement system ([www.mlssa.com](http://www.mlssa.com)). To employ this block:

1. Drag it into the workspace.
2. Click the text area.
3. Select the appropriate file and load your speaker data.



**Note:** Speaker files are ASCII text in the MLSSA format, with header information in quotes on the first two lines. Data are delimited by commas and

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## T Connection

The T Connection splits a signal, as many times as necessary.



This block is different from the Mixers/Splitters splitter algorithms: it does not contain an algorithm or use any DSP resources, the T connections are established during the link/compile operation



---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## User Comment

---

SigmaStudio lets you write text in the Schematic workspace; no DSP code or function is involved.

1. Drag the selection into the workspace.
2. Click in the displayed text area.
3. Enter your desired comment(s).

Enter your comment here

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## User Image

---

SigmaStudio lets you add an image to the Schematic workspace; no DSP code or function is involved.

1. Drag the cell into the workspace.
2. Right-click the image displayed.
3. Browse and find your desired image.



---

© 2006-2007 Analog Devices, Inc. All rights reserved.

# ADI Algorithms

## ADI Algorithms

---

The ToolBox's library of proprietary ADI algorithms lets you access unique signal processing tools for specific audio applications.

Following is a list of the cells available in this release:

- Dynamic Bass Boost
- Dynamic Enhancement
- Flanger
- Loudness (Low and High)
- Loudness (Low and High) External Control
- Loudness (Lower End)
- Midnight Mode
- Phat-Stereo
- Reverb
- Vocal Chorus

---

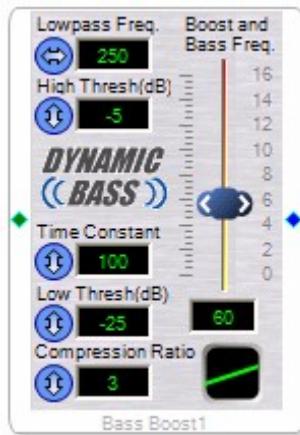
© 2006-2007 Analog Devices, Inc. All rights reserved.

## Dynamic Bass Boost

The Dynamic Bass Boost block provides boost that varies with input-signal level: lower levels require, and receive, more bass than higher levels. Using a variable-Q filter, this block dynamically adjusts the amount of boost.

The filter calculates its bass boost between the Threshold and the Minimum Gain settings. A fixed maximum is applied to inputs above minimum gain and below threshold.

Seven parameters, described below, control this block's performance. Enter their values in the appropriate field or use the arrows. While it is important for you to know how these parameters work and precisely what they do, nothing surpasses playing with each one and getting a feel for how it affects, singly and in combination, the sound you are trying to achieve.



**Lowpass Freq** - The lowpass frequency ranges from 20Hz to 250Hz; frequencies below the selected point are used by the detector for determining the boost amount.

**High Threshold (dB)** - The high threshold value, ranging from -20 to 10dB, sets the upper point for detector action. Signals higher than the minimum gain will not influence the boost calculation; they're boosted a fixed amount.

**Time Constant** - Ranging from 0 to 500 ms (milliseconds), this controls the rms time constant for the detector, changing attack and release rates.

**Low Threshold (dB)** - This value, ranging from -100 to -20dB, is the lower threshold of the detector. Any signal coming into the detector below this threshold will not influence the boost calculation; it receives a fixed boost.

**Compression Ratio** - Ranging from 1 to 15, the compression ratio -- perhaps more readily understood as a dynamic-boost ratio -- controls the rate at which the bass boost changes from the low to the high threshold.

**Boost** - This slider, ranging from 0 to 16dB, controls the maximum gain that gets dynamically applied to the algorithm. Also see next.

**Bass Freq.** - This field below the boost slider ranges from 20 to 300Hz and designates the center frequency for the boosting filter. Enter your desired value directly into the field or use the two horizontal arrows at the sides of the slider. **Note:** When the slider is all the way down at 0dB, the block is still not truly bypassed.

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Dynamic Enhancement

The Dynamic Enhancement block provides variable bass enhancement as a function of input-signal level. Lower levels require more bass than higher levels. The filter dynamically adjusts the amount of bass enhancement depending on the volume of the input signal, by using a variable-Q filter.

A fixed enhancement is applied to input levels below the threshold while a dynamic gain is applied to input levels above the threshold.



There are four parameters available for control on the block, which are described below:

**Time Constant** – The value for the Time Constant will be entered in the edit box or use the updown arrow to enter the value. The time constant ranges from 0 to 500 milliseconds. It controls the RMS time constant of the detector. Attack and release time will get affected by this parameter.

**Threshold (dB)** - The value for the Threshold will be entered in the edit box or use the updown arrows to enter the value. Threshold value ranges from -24 to -20dB . Any signal into the detector below Threshold will not influence the boost calculation and receives a fixed Enhancement.

**Boost** – The Boost will be controlled using the boost slider. The boost ranges from -24dB to 20 dB and it controls the maximum dynamic Enhancement gain applied to the algorithm.

**Bass Freq.** – The Bass Frequency value will be entered in the edit box below the boost slider or using the horizontal arrows of the boost slider control. The Bass Frequency ranges from 20 to 300Hz and it designates the center frequency for the boosting filter.

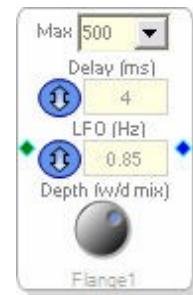
**Note:** This is not true By-pass of input block when Boost is set to 0dB.

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Flanger

Flanging is a classic audio phasing effect which generates a swirling frequency (moving comb filter) effect. Flanging is produced by mixing the input signal with a delayed copy of itself where the delay time is continuously varied at a low frequency. The Flanger algorithm supports growing by one to support stereo processing.

Flanging is typically used for guitar but can be utilized as a special vocal effect.



### Controls:

#### Max

Max delay value determines the amount of DSP memory allocated for this block during compilation. The maximum delay depends on the particular DSP's available data RAM. Changing the Max value requires a recompile and program download.

#### Delay

Delay adjusts the maximum delay amount up to the Max delay. The delay amount affects the density and spacing of frequency notches in the comb filter response. Higher delay amounts increase the intensity of the flanging effect.

#### LFO Rate (0.01Hz – 2Hz)

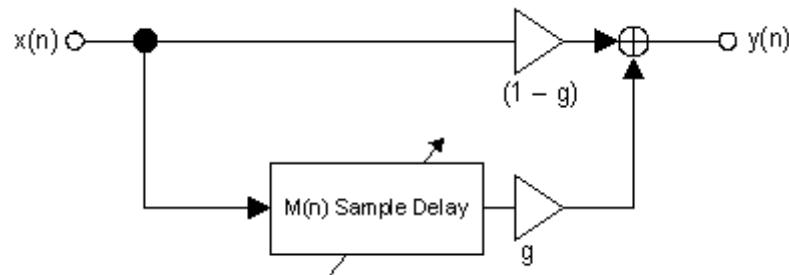
LFO rate sets the frequency of the LFO waveform used to modulate delay time. Perceptually this adjusts the number of times per second the frequency sweeps up and down.

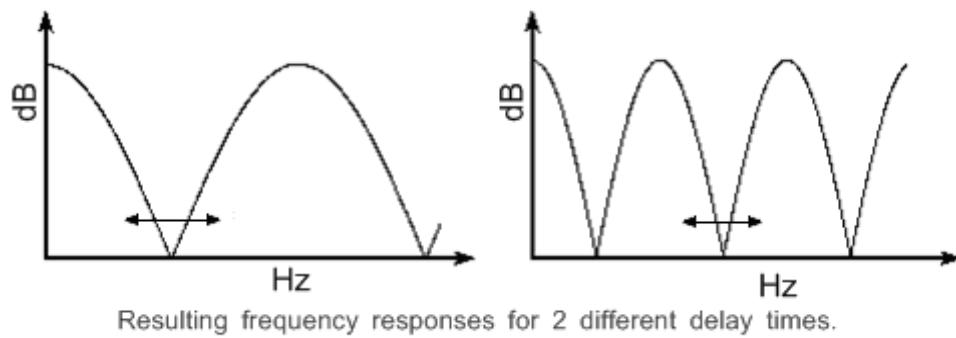
#### Depth (Wet/Dry Mix) (0% – 100%)

Depth determines the intensity of the flange effect. This adjusts the mixture of the "dry" input signal with the "wet" flanged signal.

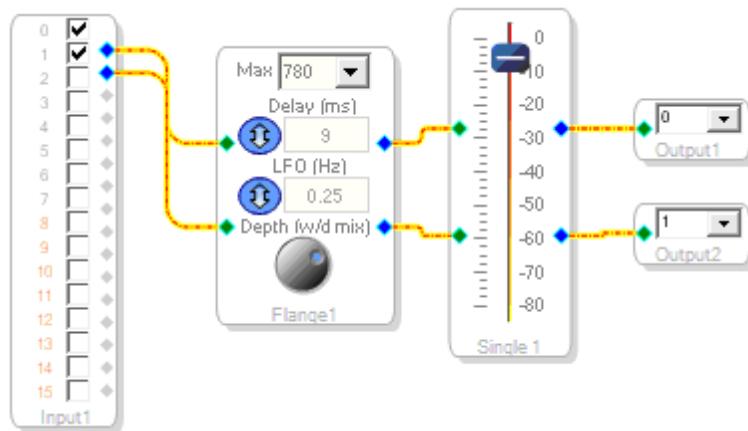
### Algorithm:

Flanger utilizes an FIR comb filter with a short delay time (typically <15ms). The delay time is continuously modulated by a low-frequency-oscillator creating the characteristic flanging sound when the filter notches sweep up and down in frequency over time.



**Example:**

The following example shows a stereo Flange effect using a 2 input Flanger and Volume Control.




---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Loudness (Low and High)

For low levels, the Loudness (Low and High) block raises bass <60 Hz and also the treble >7kHz (but see below about the control knobs).

The boost values are derived from the well-known equal-loudness curves of Fletcher and Munson and others. This research revealed that at low levels, lows and highs need to be considerably louder in order for the tonal balance to sound correctly proportioned and the overall sound to have the same apparent loudness to the human ear.

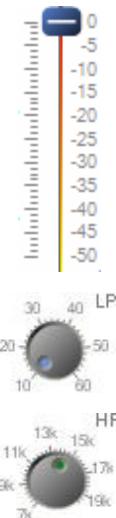
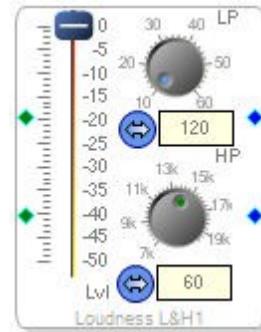
Note that this algorithm is fixed, not dynamic: it assumes the input level is constant.

This block's parameters include the level slider, LPF and HPF knobs, and Level.

1) The level slider controls the output volume of the entire signal, but, more important, it is also the control for the loudness algorithm. At low levels, the loudness algorithm boosts the low frequencies somewhat more than it does the high frequencies. At 0dB, no matter what the input level is, there is no boost, LF or HF.

2) The LP knob lets you change the cutoff of the lowpass filter. The default value approximates the Fletcher-Munson curve. Higher-frequency values provide greater bass-*bandwidth* gain.

3) The HF knob let you change the cutoff frequency of the highpass filter. The default value of 7kHz approximates the Fletcher-Munson curve.



## Loudness (Low and High) External Control

---

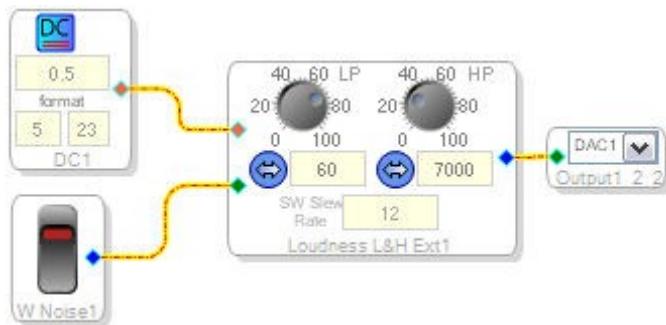
The Loudness (Low and High) External Control block like the Loundess (Low and High) block enhances the perceived loudness of a signal by raising the bass (<60 Hz) and the treble (>7kHz) level. Unlike the *Loundess (Low and High)* block, the loudness level parameter of the *Loudness (Low and High) External Control* block is controlled by an external signal instead of a volume slider.

The boost values are derived from the well-known equal-loudness curves of Fletcher and Munson and others. This research revealed that at low levels, lows and highs need to be considerably louder in order for the tonal balance to sound correctly proportioned and the overall sound to have the same apparent loudness to the human ear. Note that this algorithm is fixed, not dynamic: it assumes the input level is constant.

**External Control Pin:**

The external control pin is used to set the loudness volume parameter for the algorithm. This should be a 5.23 value between 0.0 and 1.0 which represents the desired volume (for example 0.5 for a level of -6dB). This controls the output volume of the entire block, and more importantly, it is also sets the threshold for the loudness algorithm. At low levels, the loudness algorithm boosts the low frequencies somewhat more than it does the high frequencies. For a control input of 1.0 (0dB), no matter what the input level is, there is no boost, LF or HF (See the Loundess (Low and High) level slider as it performs a similar function).

**Note:** The example below can be used to test the algorithm's functionality, but it is for demonstration purposes only. See below for a practical design example.

**Controls:****LP Knob**

The LP gain knob controls the relative amount of low frequency boost that is applied 0% - 100%.

**LP Frequency**

The LP frequency spin control lets you change the cutoff frequency of the lowpass filter. The default value approximates the Fletcher-Munson curve. Higher-frequency values provide greater bass-*bandwidth* gain.

**HF Knob**

The LP gain knob controls the relative amount of high frequency boost that is applied 0% - 100%.

**HP Frequency**

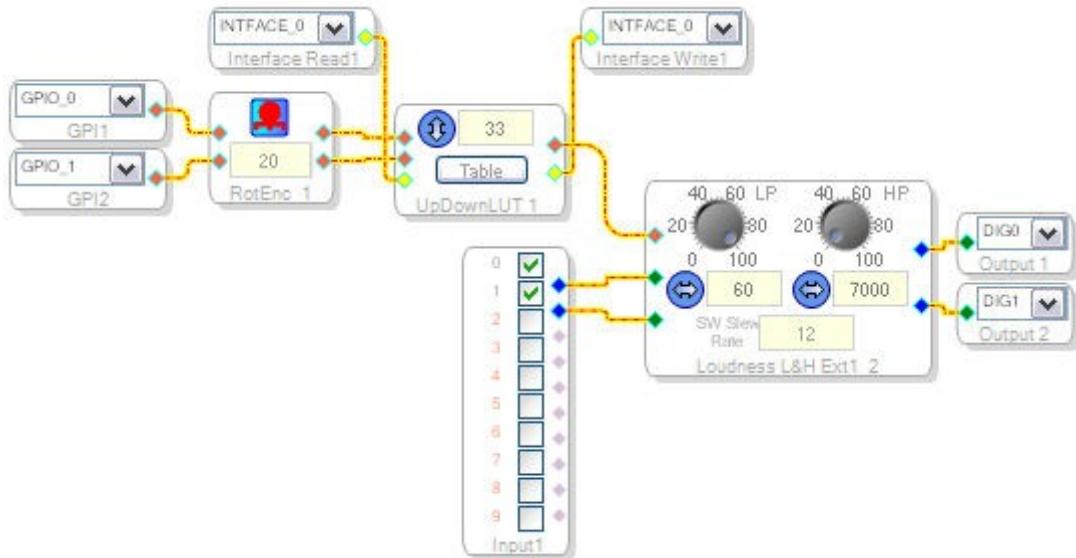
The HF frequency spin control lets you change the cutoff frequency of the highpass filter. The default value of 7kHz approximates the Fletcher-Munson curve.

**SW Slew Rate**

Adjusts the external volume parameter's slew rate (controls how quickly the algorithm responds to changes in the control signal).

**Example:**

The following example shows how a rotary encoder could be used to control the loudness level.



---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Loudness (Lower End)

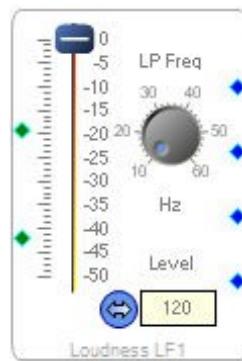
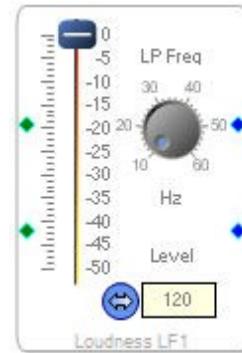
The Loudness (Lower End) algorithm raises the amplitude of bass frequencies for low volume levels. The Gain values are derived from the Fletcher- Munson equal loudness curve. Note: Only the low frequency end of the Fletcher Munson curves are used for this algorithm. These sets of curves reveal that at low levels, lower frequencies need to be boosted relative to higher frequencies, in order to have the same apparent loudness to the human ear. It is also important to note that this algorithm is not dynamic and assumes that the input level is constant.

There are two default algorithms that can be used with this block:

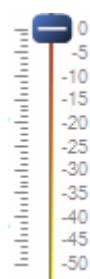
- **LF Loudness** (figure: top right)
- **LF Loudness Control w/ 2 bypassed outputs** (figure: bottom right)

They both function the same, except the latter algorithm contains another pair of stereo outputs that allow the signal to pass through unaffected by the low-end loudness boost, but it will be affected by the overall volume level slider. The figure in the bottom right shows the two extra output pins below the standard output pins used for the bypassed connection.

The parameters available for control on this block include the volume slider, LP Frequency, and Level.



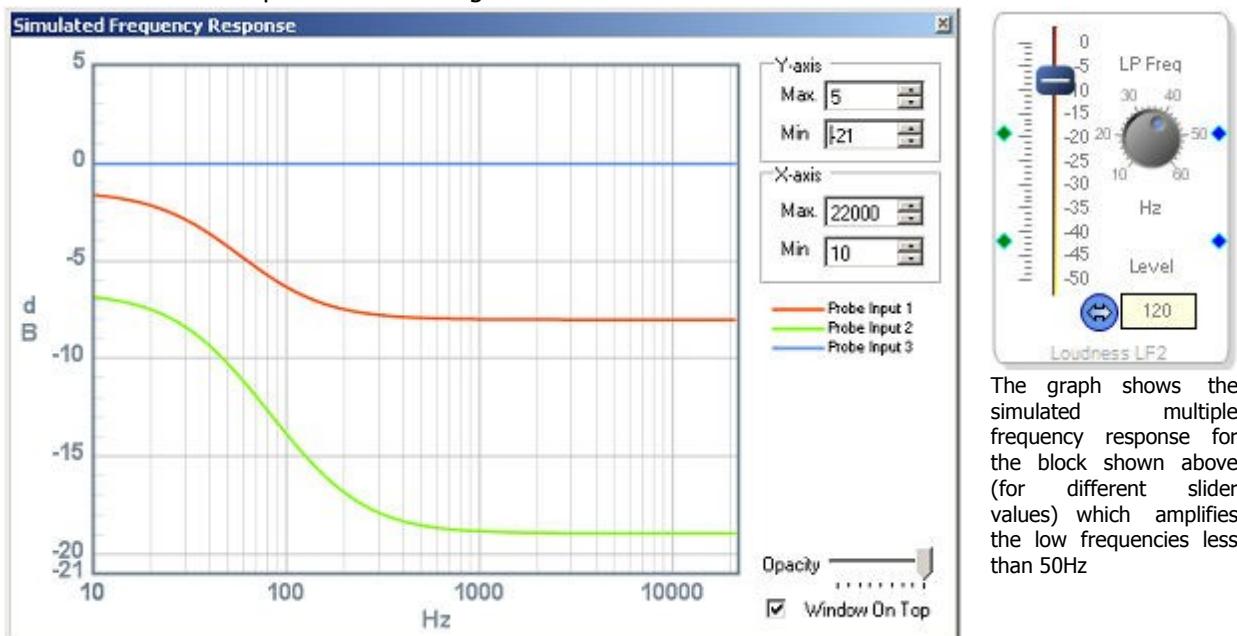
1) The volume slider controls the output gain of the entire signal and is also the control factor for the loudness algorithm. At low levels, the loudness algorithm needs to boost the low frequencies more than at higher frequencies. Note that at 0dB despite what the Level for the loudness algorithm may be, there will be no additional low-frequency boost.



2) The LP Frequency control knob allows you to edit the cutoff frequency of the low-pass filter. The default value of 10 Hz approximates the Fletcher-Munson curve. Increased frequency values will provide greater bass frequency bandwidth gain.



- 3) The Level edit box allows you to control the bass frequency boost. The higher value entered, the larger bass frequency gain there will be. The value for level is correlated with the volume slider value because the amount of boost is dependant on the signal level.



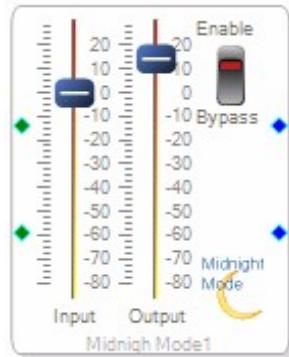

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Midnight Mode

Our Midnight Mode block is a dynamic-processing algorithm designed primarily for handling explosive movie soundtracks or similar material (video games, e.g.) to make them less obtrusive. A typical example would be quiet night-time environments where loud transients and similar sounds would disturb people (not to mention sleeping children and babies) in a room adjacent to or near the home theater or gaming area.

With Midnight Mode, high-level passages are attenuated and very quiet levels are readily raised into audibility, while dialog remains unchanged and intelligible. This permits higher overall levels with better clarity for the intended active listeners, with much less chance of disturbing nearby non-listeners.



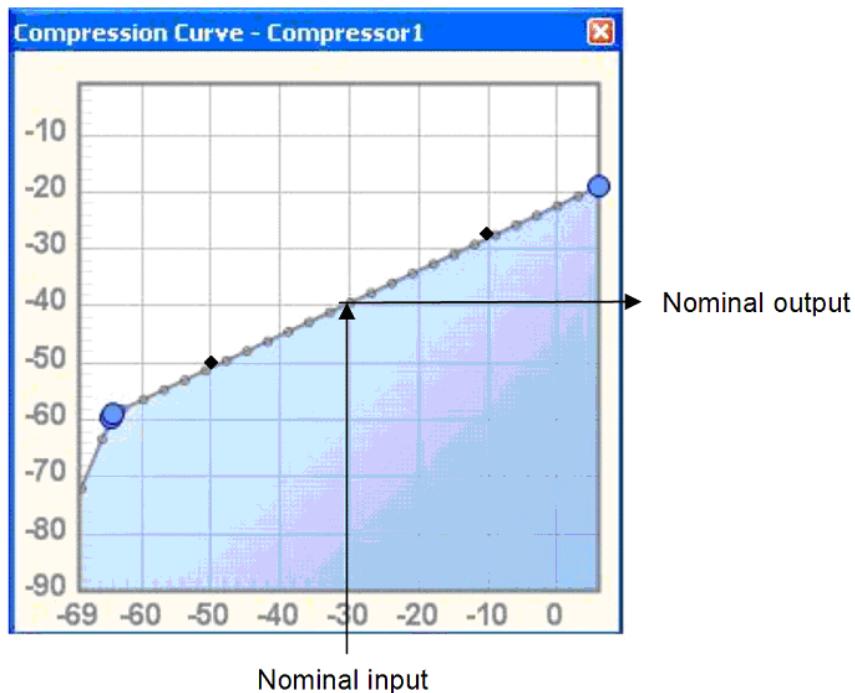
**Notes**

- This compressor block defaults to a bypassed state. Click the toggle switch to enable Midnight Mode.
- The Input slider works as a volume control, changing the level into the Midnight Mode algorithm. If the block is bypassed, the slider is throughput volume control.
- The Output slider adjusts the output level post-compressor.

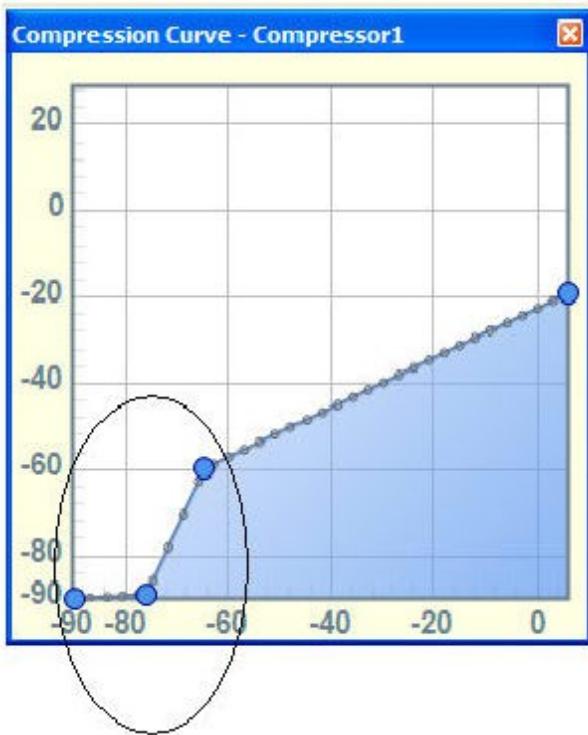
**How It Works**

Source material with a very wide dynamic range -- the "distance" from the very quietest to the very loudest levels -- must have its dynamics reduced for comfortable domestic playback, especially at night or similar times when high movie (or music) levels are inappropriate and undesirable. If the source material has a dynamic range of 60dB, say, the output dynamic range will be compressed to more comfortable 40dB.

The processing is centered on a nominal source level. Signals louder than this normal-level transition point get reduced in level, or made quieter, and the signals below this point get boosted in level, made louder. Refer to the figure below, which shows, along the horizontal axis, the input range from -10 (20 above the nominal -30dB input) to 20 below it (= -50dB) — with the compressor action demonstrated by the vertical axis, which shows outputs at -28, -40, and -50dB levels.

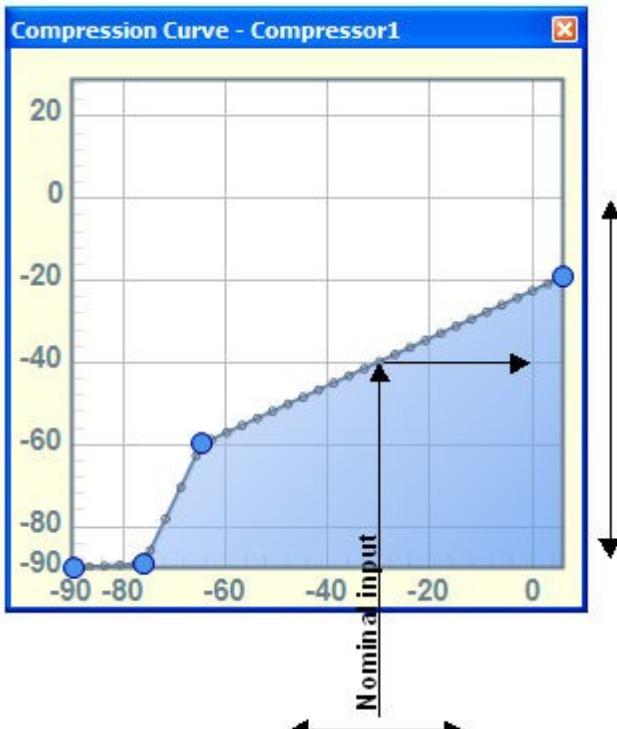
**Gating**

Now, very quiet sound is treated differently; see the figure below. The block is able to avoid such side effects as can occur with advanced compression, like modulation of the noise floor by very loud signals. The oval portion shows a noise gate (a downward expander), a function that sharply reduces low-level hum and other noise.



*Gain structure*

The enabled input slider works as a volume control, changing the input gain to the Midnight Mode processing, and moves the point where the nominal signal lands on the curve, up or down. The output slider moves the entire curve up or down.



The "Output" slider moves the entire curve up or down.

This basically adjusts the general volume level of the system.

The top-point of the curve should be adjusted to the maximum volume level of the system (clipping)

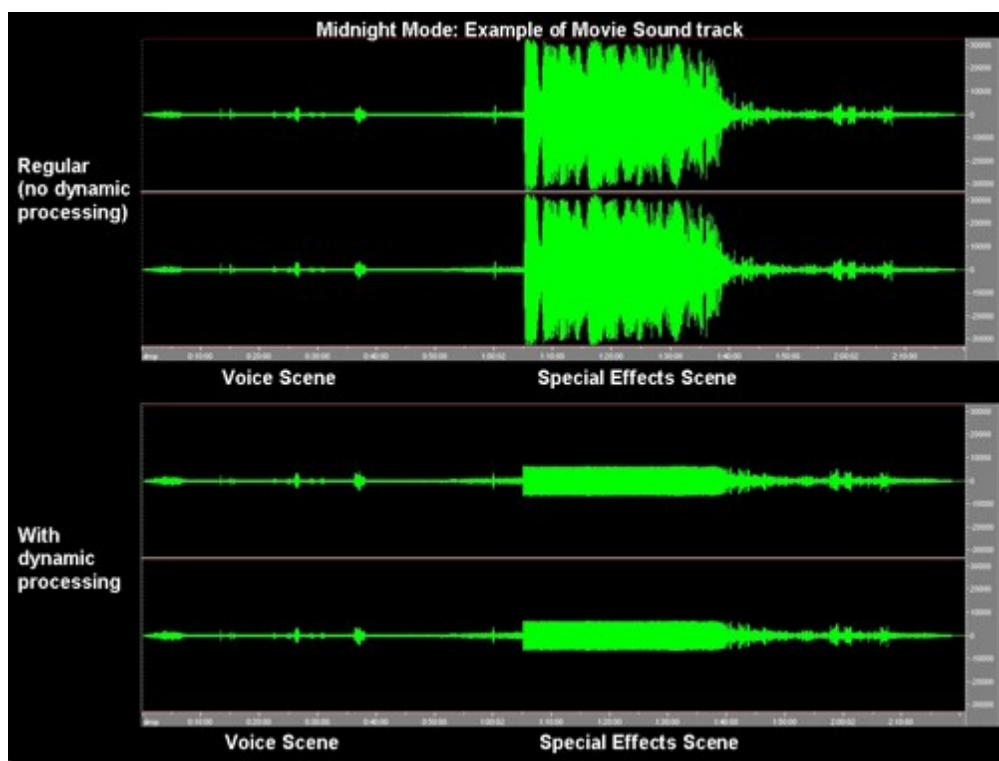
The "Input" slider moves the point where a nominal signal "lands" on the curve up or down

*Recommendation*

Input set to -6dB

Output set to +14dB

Below is a figure showing a clip from a movie soundtrack, with Midnight Mode bypassed (top) and enabled (bottom). Notice not only the control of the FX scene but the unchanged processing (volume) of the low-level voice signal.

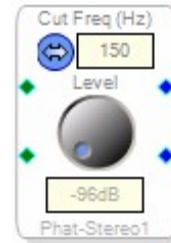


© 2006-2007 Analog Devices, Inc. All rights reserved.

## Phat-Stereo

Phat-Stereo is a spreading algorithm that uses stereo cross-coupling to simulate surround sound in stereo speakers and other 2-channel situations. The ear is most responsive to interaural phase shifts below 2 kHz, so this increase in phase shift results in a widening of the stereo image.

A 3D enhancement, it yields an enriched surround field both for headphones and for stereo speakers.



Two parameters control the block:

**Cut Freq (Hz)** - Controls the cutoff frequency of the first-order lowpass filter. Determines the frequency range of the added out-of-phase signals. For best results, the range should be 500 Hz to 2 kHz.

**Level** - Controls the output level of the filter; -80dB would basically be a passthrough with no signal modification.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Reverb

The Reverb algorithm simulates the natural reverberation of an echoic space, such as a performance hall, and mixes it back into the original signal. Technically, reverberation is the total soundfield remaining in an enclosed space after initial sources are not active.

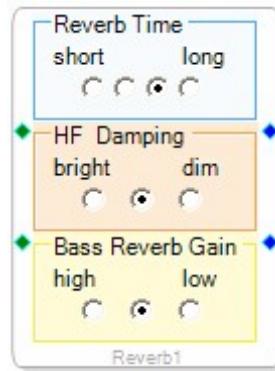
This block lets you control the following three parameters:

**Reverb Time** - These radio buttons let you control the Reverb Time settings, short to long. The Reverb time is the amount of time it takes for the reverberations to decay. (Technically this is specified by measuring how long it takes the signal SPL [sound-pressure level] to decay 60dB, to one-millionth its original value.)

**HF Damping** - These radio buttons let you control the HF Damping settings, bright to dim. HF damping refers to the brightness of the reverb reflections.

**Bass Reverb Gain** - These radio buttons let you control the Bass Reverb Gain settings, high to low. Bass reverb refers to the level and richness of the LF reflections.

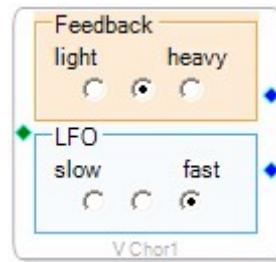
After the default algorithm has been established, this block is able to have other algorithms added to it. Right-click the block and select **Add Algorithm > IC N > Reverb**, which will add another pair of stereo inputs/outputs.



## Vocal Chorus

Vocal Chorus produces an effect where the audio signal is given multiple delays to enrich and thicken the sound, like several voices (or instruments) playing at once. The delay time is modulated by a low-frequency oscillator (LFO) to achieve a shimmering, spacious effect due to a combination of beat frequencies and the slight pitch-bending that occurs as the delay time is changed.

This cell is intended chiefly for voice.



- The Feedback settings, light to heavy, determine how much of the delayed signal to mix back in with the original, undelayed signal.
- The LFO settings, slow to fast, determine the delay times that the LFO modulates to achieve the "chorus" effect.

After the default algorithm has been established, this cell is able to have other algorithms added to it. Right-click the cell and select **Add Algorithm > IC N > Reverb**, which will add another pair of stereo inputs/outputs. (If you are using more than one DSP processor, you will need to add an algorithm for the desired IC.)

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

# Basic DSP

## Basic DSP

---

The Basic DSP library of the ToolBox lets you access basic building blocks to give you fundamental control of DSP algorithms. Following is a list of the blocks available.

- AB In / CD Out Condition
- AB in/out Condition
- Absolute Value
- Clipper
- Delay
- Division
- DSP Readback
- Feedback
- Linear Gain
- Logic - And, Or, Nand, Nor
- Multiply
- Signal Add
- Signal Invert
- Tolerance Analyzer
- Value Hold
- Voltage Controlled Delay

The individual-block pages contain specific information. Also take a look at the Basic DSP Examples page to see a sample schematics using some of these basic blocks.

## AB In / CD Out Condition

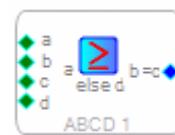
This algorithm works only for DSPs with a conditional instruction.

The AB In / CD Out Condition block lets you compare the sample-by-sample level of two incoming signals (AB) and output the sample of one of two new signals (CD), depending on the condition.

Click the blue icon in the block to select what condition you want to execute:

- greater than
- less than
- greater than or equal to
- less than or equal to.

When the condition is met, your output sample is C; otherwise it's D.



---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## AB in/out Condition

The AB In / CD Out Condition block lets you compare the sample-by-sample level of two incoming signals (AB) and output the sample of the signal meeting the condition.



This algorithm works only for DSPs with a conditional instruction.

Click the blue icon in the block to select what condition you want to execute:

- greater than
- less than
- greater than or equal to
- less than or equal to.

When the condition is met, your output sample is A; otherwise it's B.

Note that you will need to recompile your project every time you select a different condition.

For a sample design using this block, see the Basic DSP example.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Absolute Value

---

Use the Absolute Value block to convert all negative components of the input signal to positive values.



Note: Right-click the block to add multiple pins, by selecting  
**Add Algorithm > IC N > Absolute Value**

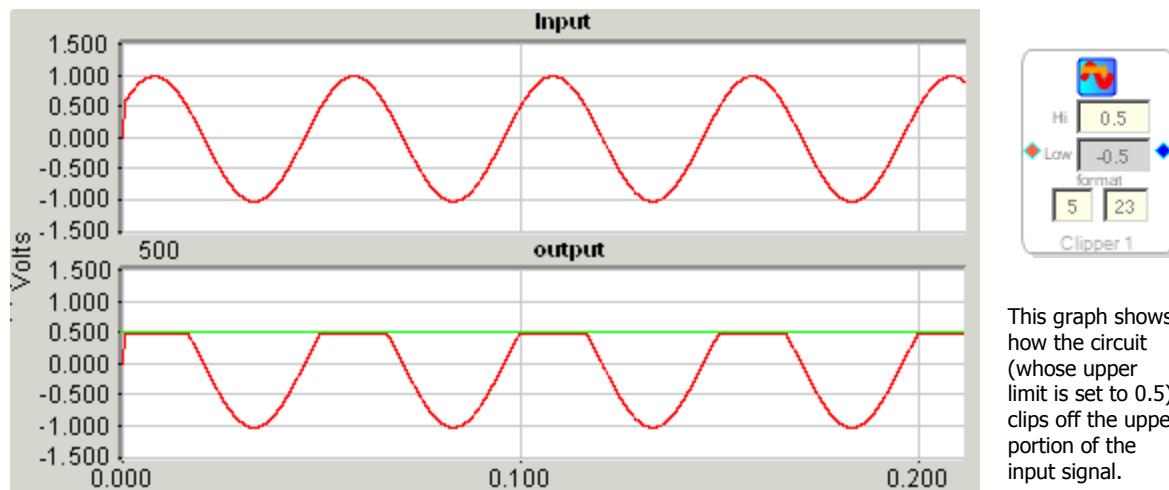
---

© 2006-2007 Analog Devices, Inc. All rights reserved.

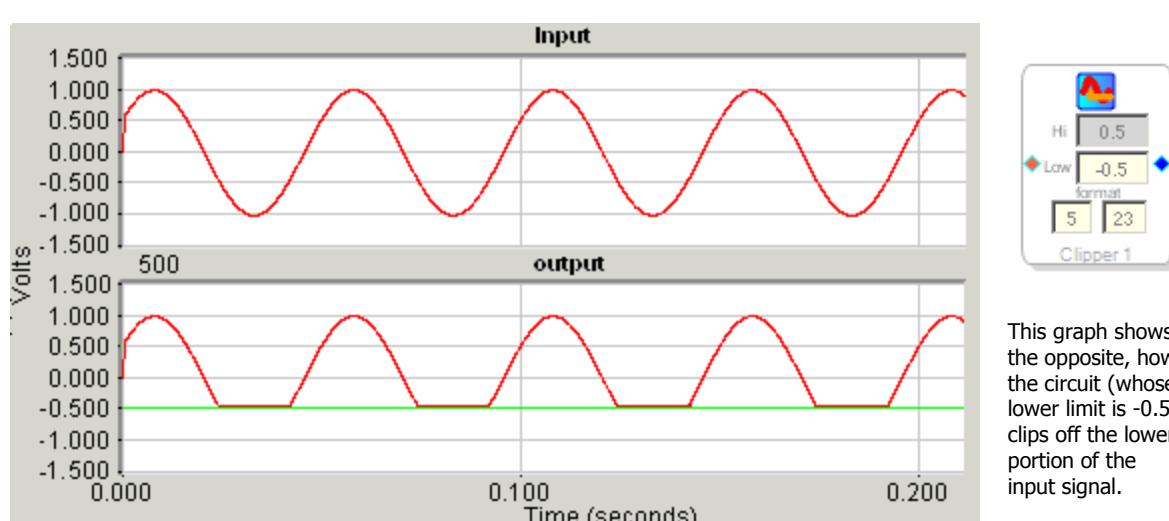
## Clipper

This block clips off portions of signal voltages above and/or below certain limits. In other words, the circuit limits the range of the output signal.

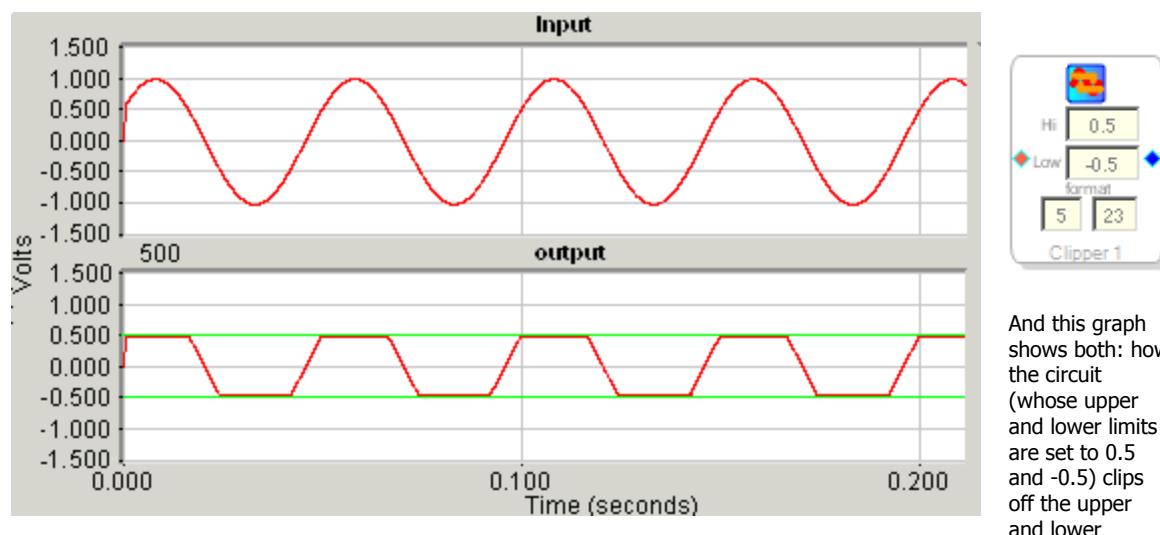
Click the blue icon to switch among the clipper options: Hi, Low, or both. Enter upper and lower limits in their text fields. Refer to the graphs below.



This graph shows how the circuit (whose upper limit is set to 0.5) clips off the upper portion of the input signal.



This graph shows the opposite, how the circuit (whose lower limit is -0.5) clips off the lower portion of the input signal.



And this graph shows both: how the circuit (whose upper and lower limits are set to 0.5 and -0.5) clips off the upper and lower portions of the input signal.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Delay

The Delay block (using the Z<sup>-a</sup> algorithm) adds a variable delay into the signal flow, between 1 and the maximum data memory available on the DSP.



The maximum delay available for a particular delay block depends on the total available system data RAM, which is specified in the DSP processor data sheet. Setting the **Max** control's value, allocates memory on the DSP, reserving that memory for use by this particular block only, and reducing the available memory for all other delay blocks in the design. This is a compiler directive and modifies the assembly code, so any time you change the Max setting you must recompile and download the program.

Note: this applies to any blocks that allocate memory (e.g. compressors). The maximum delay value range is limited to the remaining unallocated memory of the RAM.

## Division

---

The Division block allows you to divide two incoming signals. The division is performed using the Newton-Raphson iteration.



For a sample design using this block, see the Basic DSP example.

### To use this block:

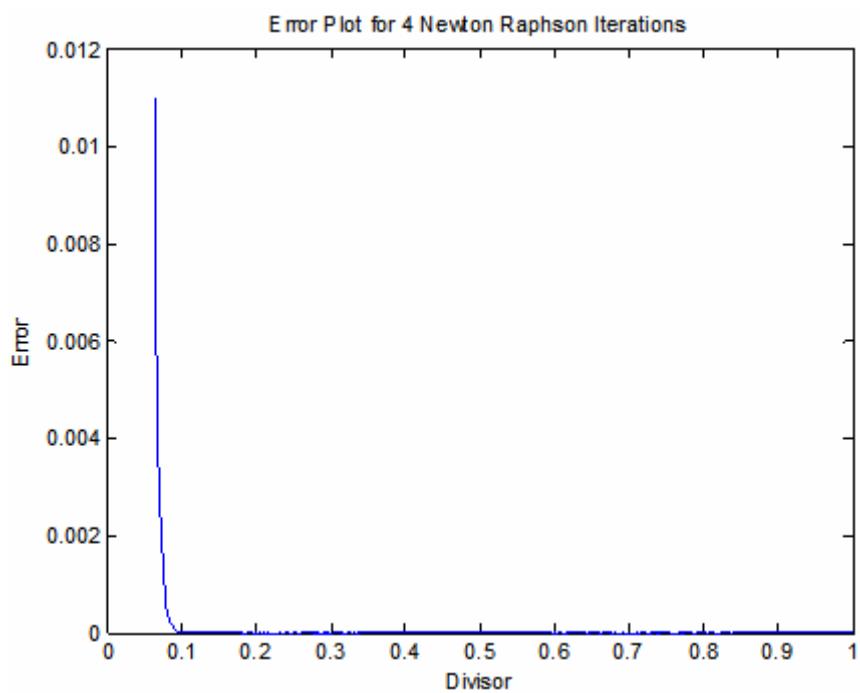
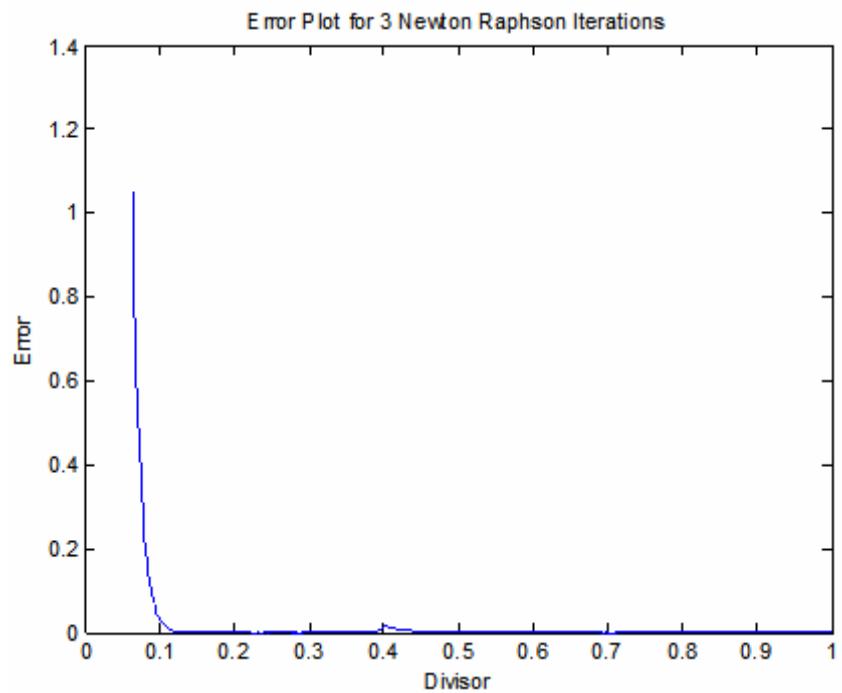
1. Drag and drop it into the workspace.
2. Right-click it and select **Add Algorithm > IC N >**
  - **Newton Raphson 3 Iterations**
  - **Newton Raphson 4 Iterations**
3. Connect your signals to the pins of the block so that you are performing the division  
pin1 / pin2

The Newton-Raphson iteration is performed according to the equation:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

This block lets you select the precision of the algorithm, whether to compute 3 or 4 iterations. There's a tradeoff between number of instructions and accuracy of computation for divisors (values of pin 2) less than 0.1. This means simply that fewer iterations are not as precise as more, but the more iterations the more instructions are entailed, with less room for programming the current DSP.

Below are error graphs for the 3- (below top) and 4-iteration (below bottom) algorithms, showing a difference of approximately two orders of magnitude:

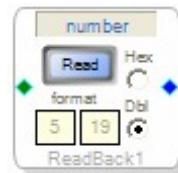


## DSP Readback

---

The DSP Readback block lets you read values back from the DSP at any point in your schematic design.

The number displayed onscreen is the data value sent back from the DSP considering all the blocks to the left of the Readback block. Every time you click Read, this value will be updated with the latest from the DSP. By displaying the output value from *any* block, in any format desired, Readback is used chiefly for debugging, and probably will prove very handy.



Values can be read back in either hex or decimal. For the latter, you must specify what format you want the number to be displayed in. The default is 5.19: 5 bits for the integer, 19 for the decimal. Any changes to this format will shift the decimal value of the number displayed.

For a sample design using this block, see the Basic DSP example.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Feedback

---

The feedback algorithm generates a delay in the signal path and reroutes signal to an input occurring earlier in the path. (Showing this reverse flow, It's the only module with the green Input on the right side and the blue output on the left side.) Note that this block *must* be used if feedback is required in your design.



For a sample design using this block, see the Basic DSP example.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Linear Gain

The Linear Gain block scales the signal by the value specified in the text field.

1. Drag it into the workspace.
2. Right-click the block and select **Add Algorithm > IC N >**
  - **Gain (slew)**
  - **Gain (no slew)**
- (Slew instructions mean better quality but at the expense of memory.)
3. Enter the desired scale factor into the text field.



For a sample design using this block, see the Basic DSP example.

The value you specify is in 5.23 format (5 bits for the integer, 23 for the decimal). The accepted values for this block are therefore in the range: -16 - 15.999.

You can choose a slew or no-slew algorithm. Using slew RAM gradually ramps the signal from original to target value, while using no-slew RAM jumps the signal immediately.

For more information on target/slew ram, see Target Slew RAM in the Basic Sigma DSP Architecture book section of this helpfile.

## Logic - And, Or, Nand, Nor, Xor

---

The Logic block applies one of five selectable logic operations to the two input signals and outputs the result on the output pin. *And*, *Or*, *Nand*, *Nor*, and *Xor* operators are available.



### To change the operator:

Left click on the icon button, each click will toggle the active operator which is shown in the icon image and label.



Note: the operator is a compile time setting you will have to compile and download the project each time you change the operator.

### Input Values:

This block should be used with DC input signals (positive 5.23 format). Also, this block is designed for use with boolean input values only, either 1.0 or 0.0. While not recommended, if necessary some operators can still function with arbitrary floating point values as input, but the output behavior may be ambiguous depending on the selected operator.

### And:

The logical conjunction of the two inputs:

Inputs:	Output:
In1 != In2	0.0
In1 == In2	In1



### Or:

The logical disjunction of the two inputs:

Inputs:	Output:
In1 == 0 In2 == 0	0.0
In1 == 0 In2 != 0	In2
In1 != 0 In2 == 0	In1
In1 != 0 In2 != 0	1.0**



\*\* If input values are non boolean, the output will be either the greater of In1 and In2 for integer inputs or the sum of In1 and In2 if either input is a non-integer value.

**Nor:**

The joint denial of the two inputs (opposite or Or):

Inputs:	Output:
In1 == 1 In2 == 1	0.0
In1 == 1 In2 == 0	0.0
In1 == 0 In2 == 1	0.0
In1 == 0 In2 == 0	1.0

**Nand:**

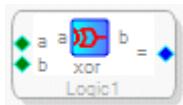
False (0.0) if both inputs are true (1.0):

Inputs:	Output:
In1 == 1 In2 == 1	0.0
In1 == 1 In2 == 0	1.0
In1 == 0 In2 == 1	1.0
In1 == 0 In2 == 0	1.0

**Xor:**

The logical exclusion of the two inputs:

Inputs:	Output:
In1 == 0 In2 == 0	0.0
In1 != 0 In2 == 0	In1
In1 == 0 In2 != 0	In2
In1 == 1 In2 == 1	0.0



© 2006-2007 Analog Devices, Inc. All rights reserved.

## Multiply

---

The Multiply block multiplies two signals together. This can be used for modulation or other design situations where a multiply operation is needed.



Right-click the block to add more pins by selecting **Add Algorithm > IC N > Multiplication**.

Care must be taken in using this block to avoid computational overflows, for example, putting in a value of 16 in 5.23 format, whose limit is 15.99999.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Signal Add

---

The Signal Adder block adds inputs together. No other modification of the signal is done.

Care must be taken to avoid clipping. If automatic gain reduction is needed to avoid clipping, use the Signal Merger block.



---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Signal Invert

The Signal Invert block takes the incoming signal, inverts its polarity, and outputs the inverted signal. This entails a 180-degree phase shift, making the positive components of the waveform negative and the negative components positive. Check the box to enable this block.



For a sample design using this block, see the Basic DSP example.

---

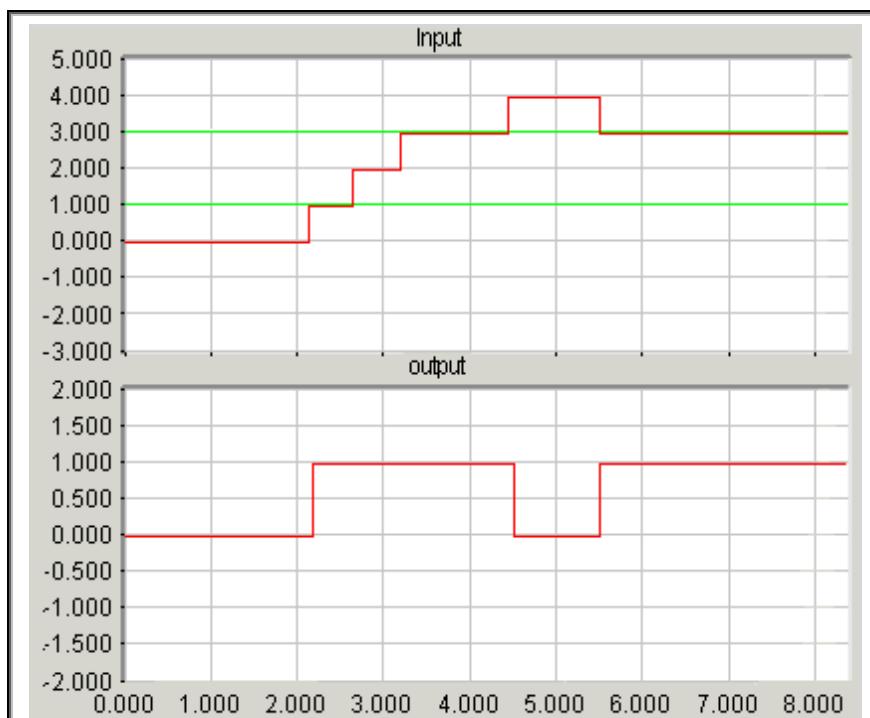
© 2006-2007 Analog Devices, Inc. All rights reserved.

## Tolerance Analyzer

This block is for any application where you need to verify a given value's tolerance limits. It is especially useful for testing environments where the Sigma DSP needs to perform system diagnostics. The Tolerance Analyzer outputs either one or zero based on input level: if the level falls within the limits specified, it outputs one at the output pin; otherwise it outputs zero.



The operation of the Tolerance Analyzer can be readily understood from this graph:



The graph shows the instantaneous values at input and output pins of the Tolerance analyzer whose tolerance limits are 1-3.

As long as the input remains within the limit (1 - 3, designated by green lines), the block outputs 1. When the signal is out of (above or below) the limit, the block outputs 0.

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Value Hold

---

The Value Hold block is used to retain an incoming signal (green pin). The signal is held based on the presence and level of a control signal (red pin).



This block holds the value of the signal on the green input pin, maintaining it on the blue output pin whenever the signal applied to the red pin is 0 (zero), but passing it for all values (positive or negative, 5.23 format) on the red pin.

This module can be used for such applications as applying and holding a value to be either analyzed in real time or read back later.

For a sample design using this block, see the Basic DSP example.

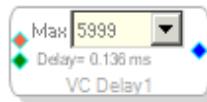
---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Voltage Controlled Delay

This block is used to add a very small variable delay to the signal flow (green to blue pins) based on the signal level ("voltage") applied to the red pin. (Typically an external dc signal is employed.)

Drag the block into your schematic. Right-click and add the algorithm desired: voltage-controlled delay with or without limit. The delay is set from the number of samples you choose in the Max dropdown list, with the corresponding delay value in a small fraction of a millisecond (ms) below it.



The two default algorithms with this block, delay without and with limit, function the same. The difference is that the latter keeps you from entering invalid data (less than 1 sample or more than 6k, all blocks totaled).

The maximum delay allowable for a particular DSP naturally depends on its available memory. Refer the appropriate data sheet to see the maximum RAM for a given DSP, and see the example for further details.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Counters

### Counters

---

Counters, timers and stopwatches help achieve time based conditional event triggering. In many applications it is important to be able to track time that has passed in order to trigger events, or turn flags on and off, or find out how long it takes to finish a task. It is also useful to trigger the start and stop of counting based on external signals.

Some typical applications include counting time during which a product is inactive in order to power-down the device after a certain time threshold of inactivity is met. Other more generic examples include sequencing of events depending on time that has passed. Thus counters can be used to pause the signal flow; a certain amount of time can be specified to wait for, and then the output will trigger the next event to occur.

In some applications it is useful to enter an actual value that sets the counter, whereas in other situations, the signal flow should determine when the counting begins, ends and is reset. SigmaStudio offers a wide range of timing algorithms that allow for flexible control of the counting method.

The toolbox's Counter library includes the following blocks:

- Counter
- Stopwatch
- Stopwatch w/ External Reset
- Timer w/ External Reset

## Counter

The Counter block can be used with a State Machine to process your input at a particular time interval, in ms. The counter starts from 0 (or any other user-defined value).

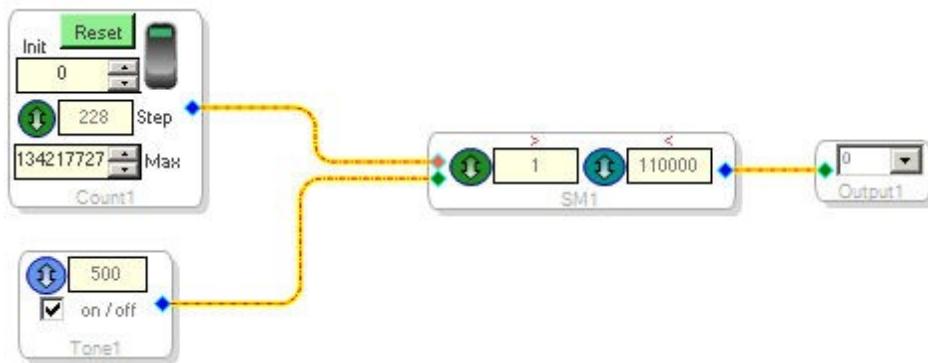


### Controls (parameters):

<b>Max</b>	The maximum value (limit of $2^{27}$ ms, or a day and a half) for the counter is given in this box, and you can also enter it in the text field and press the adjacent arrows.	<input type="text" value="134217727"/> Max
<b>Start</b>	Begins the count.	
<b>Init</b>	Enter the start count value in the Init field, or use the arrows.	<input type="text" value="4654"/> Init
<b>Reset</b>	This button resets the counter.	
<b>Step</b>	Control the count rate by entering the desired value in the Step field or using the arrows.	<input type="text" value="1"/> Step

### Counter example:

This schematic shows how the Counter (in conjunction with a Tone cell) can be used with the State Machine. The state machine block passes signal (green pin to blue pin) only when the Counter's control value to the red pin is between the given limits.



---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Stopwatch

The stopwatch block allows for a stop, start, and reset counting mechanism. When one needs to know how long something takes (i.e. how many samples have passed), you signal the start and stop of an event, and a stopwatch measures the time between the signaling of the start and stop. Stopwatches also allow for the count to be reset to zero to measure additional time intervals.



The stopwatch allows external signals to control the start and stop of the counter. The output of the stopwatch block is the total sample count, in 28.0 data format, for the active period. When active, the output value and count are incremented for every sample period.

**Inputs:**

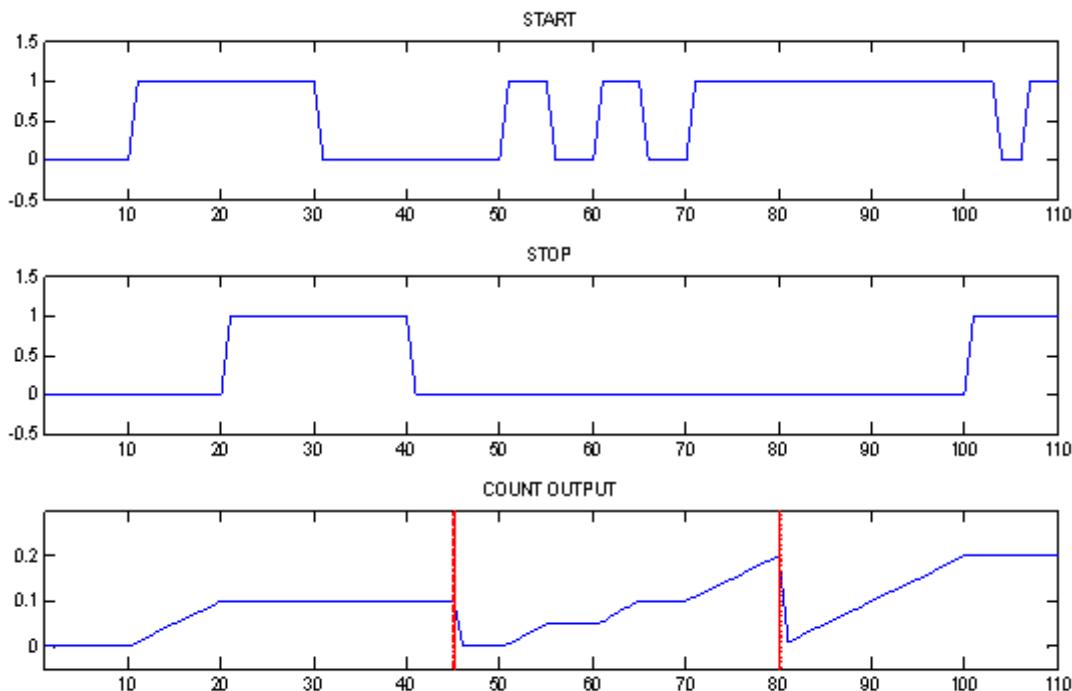


- The top input pin is the **START** pin. While the STOP pin signal is zero (see below), an input value greater than zero triggers the counter to start, a zero value will stop the counter.
- The bottom input pin is the **STOP** pin. An input value greater than zero will stop the counter (regardless of the value of the start pin).

**Reset button:**

The **Reset** button resets the value of the counter and output to zero. If the count is active (e.g. START = 1 and STOP = 0) the counter remains active and continues counting from zero. If the count is inactive (e.g. START = 0 or STOP = 1) the reset button clears the count and the output will remain at 0. The RESET button is controlled via a parameter coefficient.

Below are graphs showing values for START, STOP, and the resulting output values. The vertical red lines on the count output denote when the RESET button is pressed.



To better understand the counter output, imagine that for every sample that passes when the count is active, you are incrementing by  $2^{-23}$ . This is the smallest amount possible to

increment by (since our data is in 5.23 format the LSB is  $2^{-23}$ ). However the format of the count output is in 28.0 thus the maximum value that can be achieved by the count output is  $2^{28} - 1 = 268435455$ . If the count reaches this value it will no longer be incremented and will not wrap around, but just maintain this max value. This maximum value in samples can be converted to a meaningful time value in seconds by knowing the sampling frequency of the block.

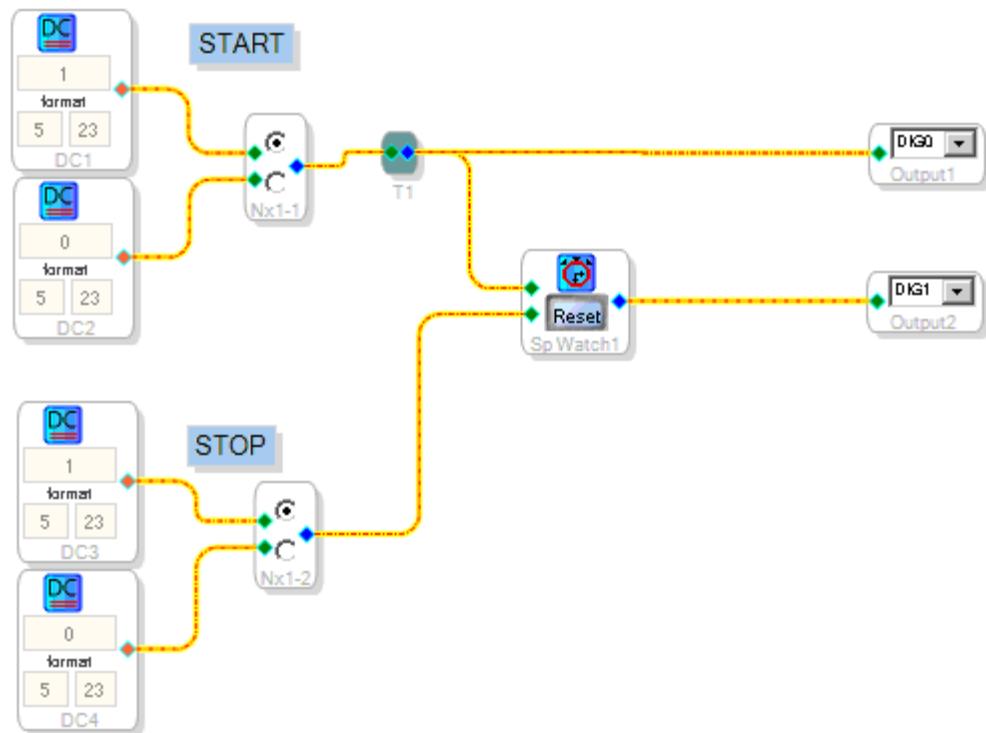
**For example @  $f_s = 48000$  Hz:**

$$268435455 \text{ samples} * 1 \text{ s} / 48000 \text{ samples} = \\ 5592.4053 \text{ s} = 93.02 \text{ minutes}$$

In this example, the stopwatch can count up to a maximum of 93.02 minutes.

**Example:**

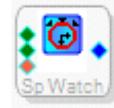
Below is a sample schematic showing the basic operation of the Stopwatch. The DC input blocks along with a mono switch, allow for the start and stop pins to be toggled from 0 to 1. This controls the beginning and end of the count. The reset button on the stopwatch controls a coefficient parameter that clears the value of the counter.



© 2006-2007 Analog Devices, Inc. All rights reserved.

## Stopwatch w/ External Reset

The stopwatch block allows for a stop, start, and reset counting mechanism. When one needs to know how long something takes (i.e. how many samples have passed), you signal the start and stop of an event, and a stopwatch measures the time between the signaling of the start and stop. Stopwatches also allow for the count to be reset to zero to measure additional time intervals.



**Note:** This block differs from the Stopwatch block because it can be reset by an external input signal. The regular Stopwatch can only be reset by writing to its reset parameter coefficient.

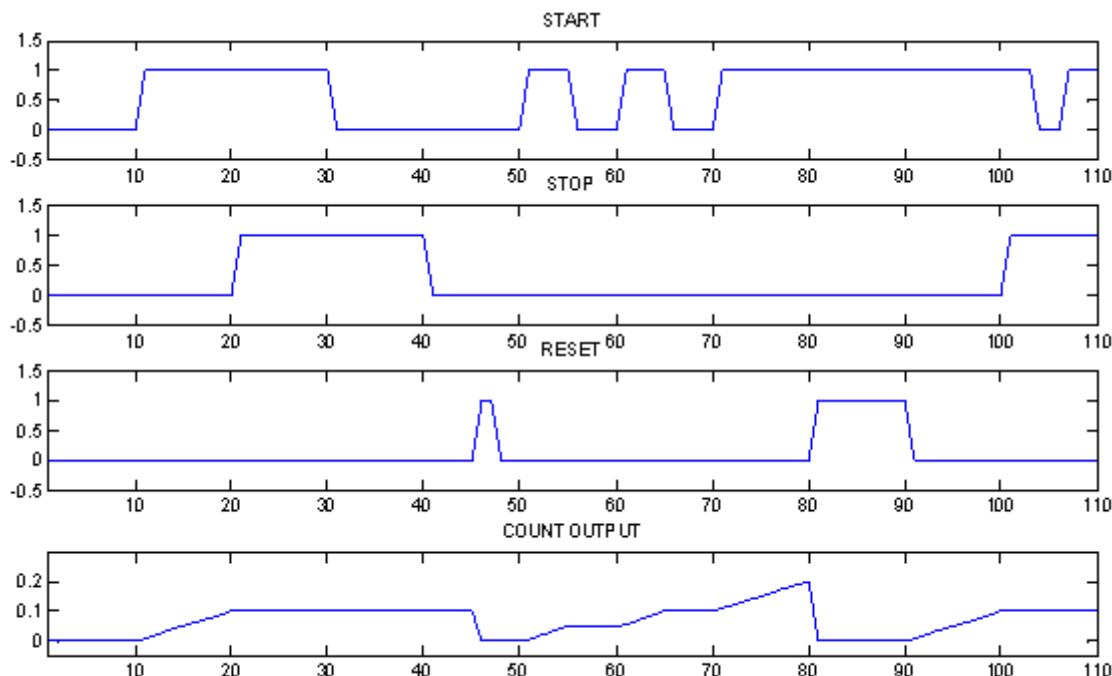
The stopwatch allows external signals to control the start and stop of the counter. The output of the stopwatch block is the total sample count, in 28.0 data format, for the active period. When active, the output value and count are incremented for every sample period.

**Inputs:**



- The top input pin is the **START** pin. While the **STOP** pin signal is zero (see below), an input value greater than zero triggers the counter to start, a zero value will stop the counter.
- The middle input pin is the **STOP** pin. An input value greater than zero will stop the counter (regardless of the value of the start pin).
- The bottom (orange) pin is the **RESET** control pin. When the reset signal is zero, it will reset the value of the counter and output to zero. If the count is active (e.g. START = 1 and STOP = 0) the count will reset to zero continue to increment from zero. If the count is inactive (e.g. START = 0 or STOP = 1) the reset signal clears the count and the output remains at 0.

Below are graphs showing values of the START, STOP, RESET, and the resulting output values.



To better understand the counter output, imagine that for every sample that passes when the count is active, you are incrementing by  $2^{-23}$ . This is the smallest amount possible to increment by (since our data is in 5.23 format the LSB is  $2^{-23}$ ). However the format of the count output is in 28.0 thus the maximum value that can be achieved by the count output is  $2^{28} - 1 = 268435455$ . If the count reaches this value it will no longer be incremented and will

not wrap around, but just maintain this max value. This maximum value in samples can be converted to a meaningful time value in seconds by knowing the sampling frequency of the block.

**For example @ fs = 48000 Hz:**

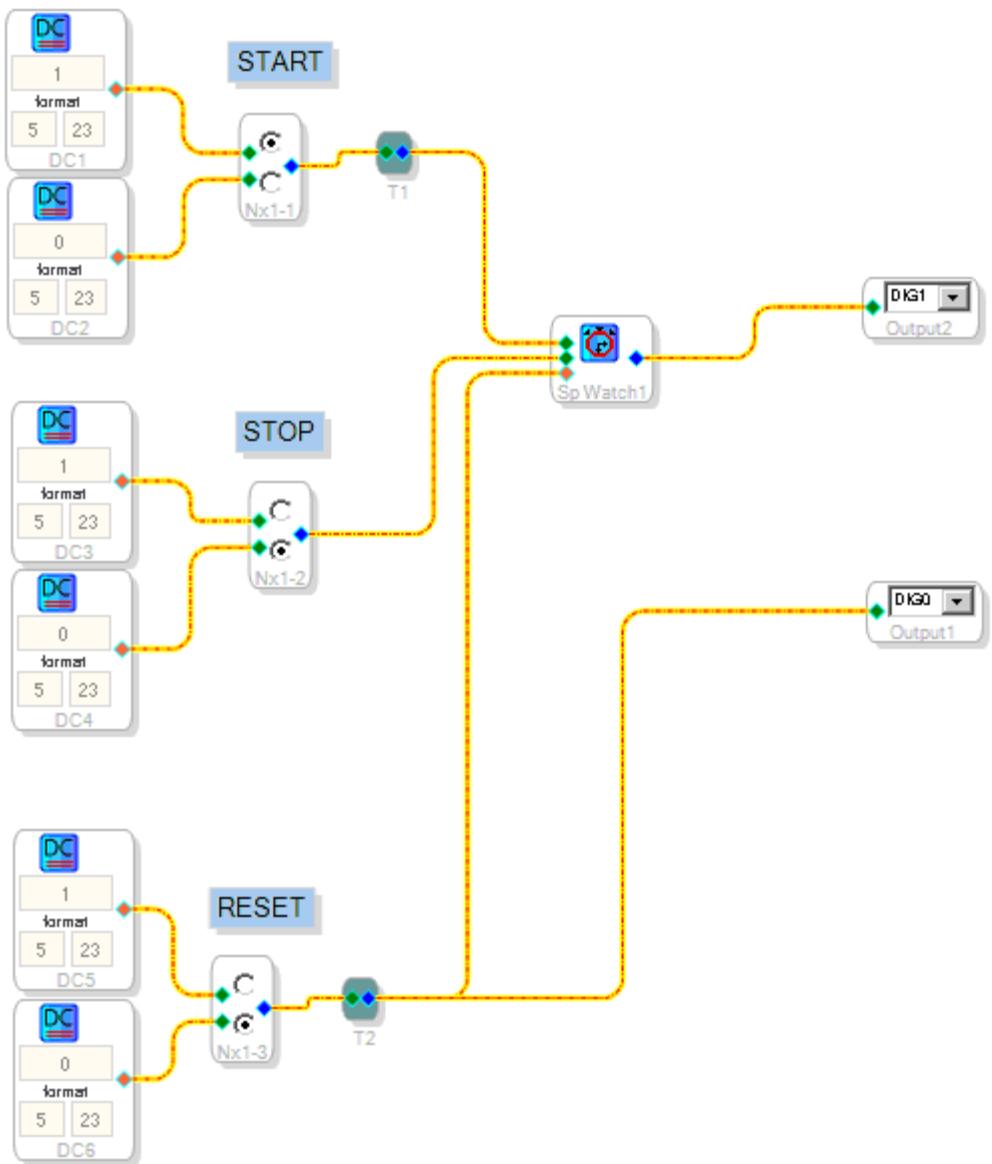
$$\begin{aligned} 268435455 \text{ samples} * 1 \text{ s} / 48000 \text{ samples} = \\ 5592.4053 \text{ s} = 93.02 \text{ minutes} \end{aligned}$$

In this example, the stopwatch can count up to a maximum of 93.02 minutes.

---

**Example:**

The stopwatch can be used in many different situations. Below is a sample schematic showing the basic operation of the Stopwatch w/ External Reset. The DC input blocks along with a mono switch, allow for the start, stop, and reset pins to be toggled from 0 to 1. This controls the beginning, end, and clearing of the count. The main difference between the Stopwatch and Stopwatch w/ external reset is how the clearing of the count output is handled. In this case an external data signal can be used to control the reset function of the count. With the stopwatch a coefficient parameter (controlled by a button in SigmaStudio) allows the count to be cleared.



© 2006-2007 Analog Devices, Inc. All rights reserved.

## Timer w/ External Reset

---

Time Timer counts for a specified amount of time and then sets a flag upon completion. When you want to prevent an event from occurring until a certain amount of time (samples) have passed, the timer can be used to trigger or delay schematic functionality.



The Timer w/ external reset allows an external signals to control the start and stop of the count. The output of the timer block is a 5.23 format value of zero or one, indicating whether the value of the timer has been reached. While the count value has not been reached the output will be zero, and once the value has been reached, the output will be one.

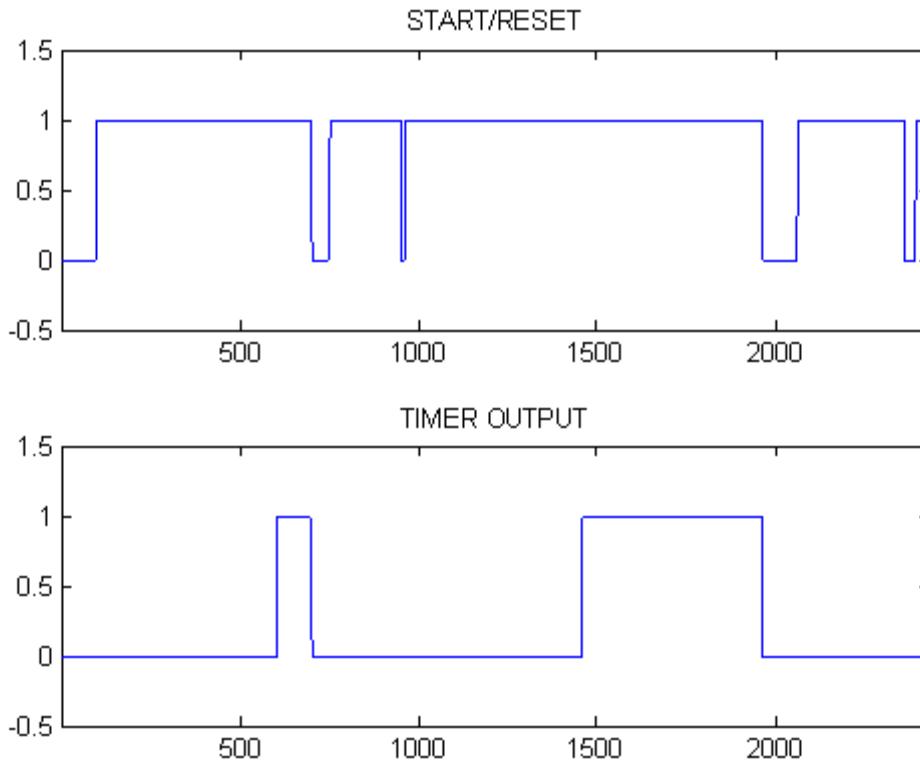
### Inputs:



- The top pin is the **START/RESET** pin. When the value does not equal zero, the timer will start. When the value is set to zero, the timer is reset.
- The bottom pin is the **TIMER** pin. This pin allows an external signal to set the timer's duration in samples, as a 28.0 format integer value.

---

The timer's counter begins incrementing upon the top Start/Reset pin being set greater than zero. The output will be zero until the timer value is reached, designated by the bottom Timer pin. If the START/RESET pin is set to zero before the count is reached, the output remains at zero and the count will not begin again, until the START/RESET pin is set back to one. Below are graphs showing the values of the START/RESET pins and the output according to a TIMER length of 500 samples.



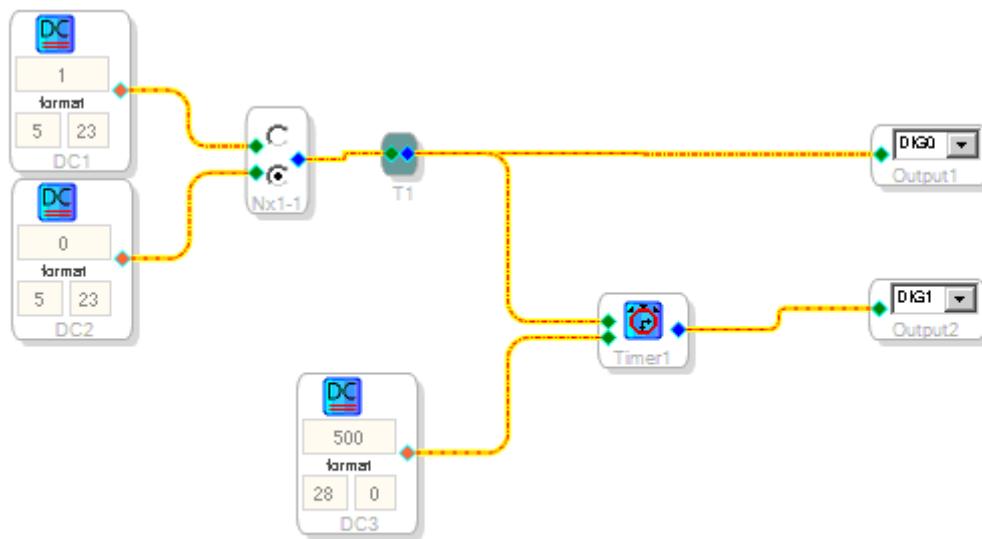

---

### Example:

The timer can be used in many situations in which it is important for the output to trigger an on/off or flagged event. Unlike the stopwatch algorithms, the output of this block is solely a 1

or 0 depending on whether the count value has been reached. (The stopwatch algorithms output the actual current count value).

In this example schematic, DC input blocks along with a mono switch, allow for the START/RESET pin to be set. This pin controls both when the count starts, and also when the value is cleared. A second DC block (NOTE: 28.0 for integer value) sets the length of the timer to 500 samples. The output of the timer in this case is going straight to the hardware output, but this value of 0 or 1 could also be used for logic conditioning.



© 2006-2007 Analog Devices, Inc. All rights reserved.

# Dynamics Processors

## Dynamics Processors

---

The Dynamics Processors library of the ToolBox provides a variety of blocks that can be programmed to control the dynamic range of signals. These blocks provide compression, limiting, and expansion — and allow you to graphically control the dynamic response by manipulating the response curves. For usage samples, take a look at the Dynamics Processor Examples page.

- **Envelope Detectors**
  - Peak
  - RMS
- Limiter
- Compressors
  - Hi-Resolution RMS
  - N-Channel
  - Peak (gain)
  - Peak (no gain)
  - RMS (gain)
  - RMS (no gain)
  - RMS (display)
  - RMS (no gain, hold, decay)

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Envelope

## Envelope

The Envelope blocks calculate and output the "envelope" of an input signal. This is sometimes referred to as an envelope follower. There are 2 envelope blocks available, one which calculates the signal's peak envelope and another that calculates the signal's RMS (Root mean square) envelope. The envelope is typically used as a control signal. See the block descriptions for more information.

- **Envelope Peak**
- **Envelope RMS**

---

### **Envelope Definition:**

The envelope of an audio signal (or any amplitude-modulated input) is that boundary within which the signal is contained, when viewed in the time domain. It is a quantitative measurement that describes the signal's amplitude variations over time. This "boundary" has an upper and lower part (positive and negative), but in practice, it is customary to consider only one of these boundaries as 'the envelope' (typically the upper or positive boundary).

Any AM signal can be written in the following form:

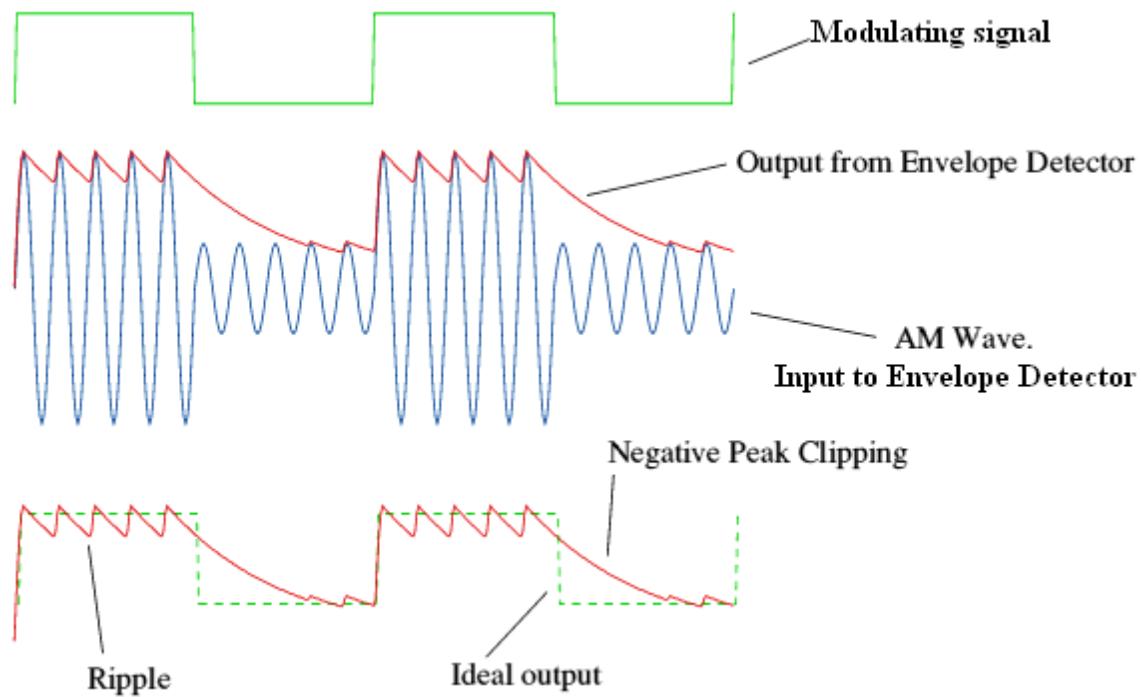
$$x(t) = R(t) \cos(\omega t + \varphi(t))$$

In the case of AM the phase component of the signal,  $\varphi(t)$ , is constant and can be ignored, so all the information in the signal is in  $R(t)$ , which is called its envelope. Hence an AM signal is given by the equation.

$$x(t) = (C + m(t)) \cos(\omega t)$$

with  $m(t)$  representing the original audio frequency message,  $C$  the carrier amplitude, and  $R(t)$  equal to  $C + m(t)$ . If the envelope of the AM signal can be extracted, then, the original message can be recovered.

Envelope block functionality can be understood as follows:



---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Envelope Peak

The Peak Envelope is a the envelope of the maximum or 'peak' level's of input signal over time unlike the RMS envelope, which is essentially an average on the input level.



### Controls:

<b>Hold (ms)</b>	0	Controls the time (in ms) the envelope maintains the output level before it starts decreasing/increasing as the input level decrease/increases.
<b>Decay (db/s)</b>	10	Controls the rate at which the envelope signal changes (increases or decreases) in response to input signal level changes, i.e. the smoothness of the envelope.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Envelope RMS

Mathematically speaking, RMS refers to the square root of the average of a set of values. For our purposes it easier to think of the RMS Envelope as the average level of the input signal whereas the Peak Envelope follows the maximum level of the input.

For a sample design using this block, see the Dynamics Processor Example.



### Controls:

<b>rms TC (db/s)</b>	<b>RMS TC (db/s)</b> 120	Controls the time constant (TC) in dB/second that is used for calculating the RMS value. The time constant determines how rapidly the envelope will adapt to increases in input signal level, e.g. the "Attack" time of envelope detector.
<b>Hold (ms)</b>	<b>Hold (ms)</b> 0	Controls the time (in ms) the envelope maintains the output level before it starts decreasing/increasing as the input level decrease/increases.
<b>Decay (db/s)</b>	<b>Decay (db/s)</b> 10	Controls the rate at which the envelope signal decreases in response to decrease in the input signal level, e.g. the "Release" rate of the envelope detector.

## Limiter

The Limiter block is an extreme compressor, completely preventing signals from exceeding the threshold. Whenever the level signal starts to go above it, the limiter immediately stops it and keeps it at threshold.

This block can control the detected rms value and attack-time constant (TC), as well as the processor's decay.

The following graph shows the input/output relationship for a 1kHz tone with 6dB-increments thresholds.



Signal output is computed according to the following formula:

$$G_n = \begin{cases} \text{threshold / RMS}_{\text{avg}} & \text{if } \text{RMS}_{\text{avg}} \geq \text{threshold} \\ 1 & \text{if } \text{RMS}_{\text{avg}} \leq \text{threshold} \end{cases}$$

output:  $y_n = G_n x_n$

Signals below threshold remain unaffected; those above it are attenuated by the firm ratio shown above.

In the block figure, top right, the blue pin outputs the dynamically limited input signal. The first red pin (middle of three pins) outputs ZERO (a flag) when the rms value of the input is below threshold. If the rms value exceeds the threshold, it will output ONE, giving you the option to read whether the limiter is active.

The second red pin outputs the instantaneous-limiting ratio, and you can use this value to derive the compressed signal by employing a multiplier block to measure the signal envelope and the input. Compressed-signal displays are widely used in professional audio equipment.

For a sample design using this block, see the Dynamics Processor Example.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Compressors

## Compressors

---

The compressor blocks share similar functionality. Compare the follow block descriptions to find the appropriate compressor for your particular application.

- **Hi-Resolution RMS**
  - **N-Channel**
  - **Peak (gain)**
  - **Peak (no gain)**
  - **RMS (gain)**
  - **RMS (no gain)**
  - **RMS (display)**
  - **RMS (no gain, hold, decay)**
- 

Compressors reduce the overall dynamic range of the program signal passing through. All of them make loud signals quieter and if desired, can make quiet sounds louder.

The amount of gain applied (positive, negative, or none) is determined by comparing the input level with a threshold, a level-match point where there is no dynamic gain processing.

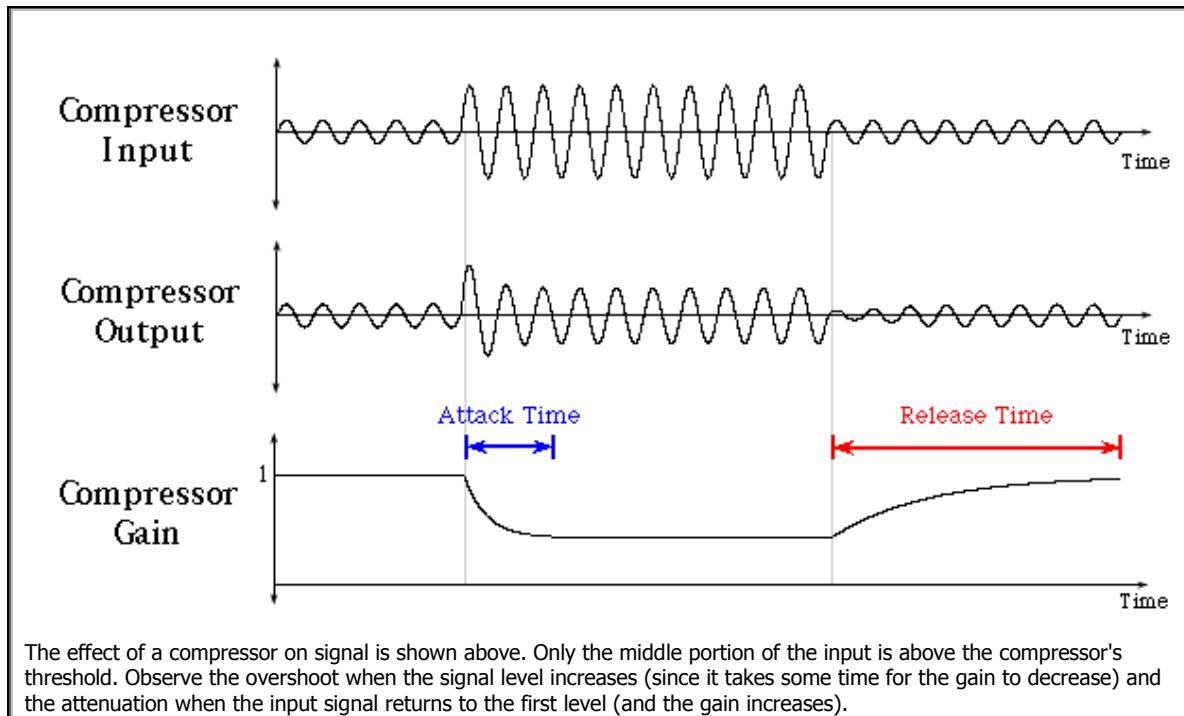
With typical recording compressors, incoming signals below threshold remain unaffected, but whenever the level exceeds the threshold, the output is turned down by a ratio you determine. For example, if you set the compressor ratio for 1.6:1, for every 1.6-dB increase above threshold, the output increases only 1 dB.

If you set a much higher ratio, the compressor will keep the output below a fixed limit, and regardless of how high the input level goes above threshold the maximum output will not exceed the limit. This processor is called a **Limiter**. Such limiters typically are used to protect broadcast transmitters and other situations where no signal can exceed a set point. Protection of loudspeakers from overdriving is another common use for compressors.

Below the threshold, the signal can pass unaltered, as mentioned above, or, if desired, its level can be boosted. Such compressors are commonly employed for broadcasting and/or playback, as compressing loud levels and boosting quiet levels can make for more comfortable listening as well as increasing intelligibility.

**Expanders** do the opposite of compressors: they expand the dynamic range of the signal passing through, operating below the threshold point and making those quiet sounds even quieter. For example, if you set the ratio at 2:1 and there's a decrease in input level of 10dB, the expander attenuates the output by 20dB. This functionality is chiefly used for noise reduction. Other applications might include broadcast ducking, where the music is reduced when the announcer starts speaking.

The individual block descriptions contain further detail about algorithm behavior. The figure below shows general above-threshold compressor behavior, plus attack- and release-time action.



© 2006-2007 Analog Devices, Inc. All rights reserved.

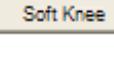
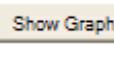
## Hi Resolution RMS

This block uses a high-resolution rms dynamics processor that lets you control the rms TC (time constant), Hold, Decay, and Soft Knee behavior, and displays the compression curve graph for your curve drawing.

Hi-Resolution RMS works on a longer average than peak processors, but, like them, it still won't allow fast loud transients to pass without compression.

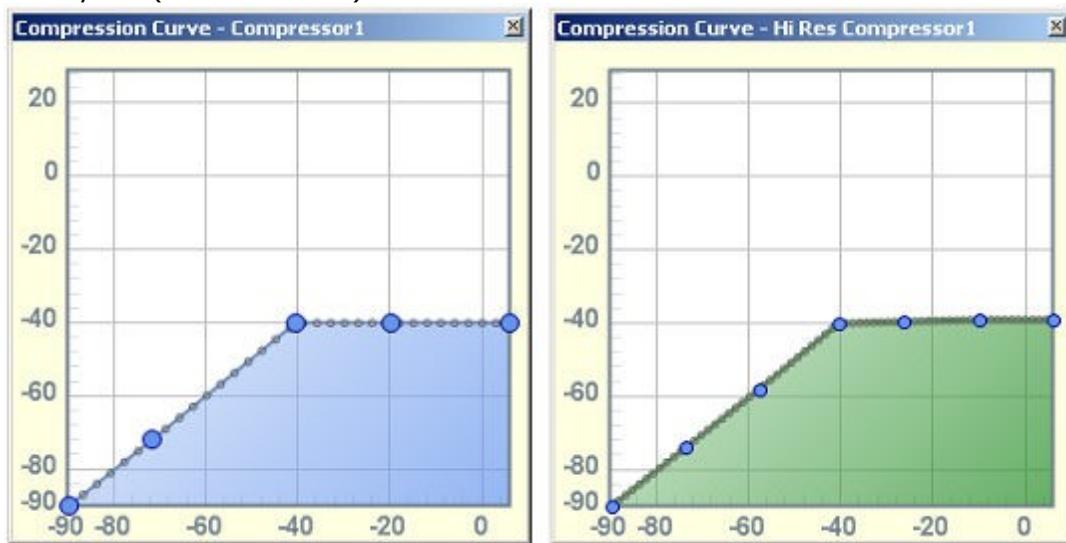
Push Show Graph to open the Compression Curve window and drag, add and remove (right-click) control points to achieve your desired processing curve.

### Controls:

<b>RMS TC (db/s)</b>	 120	Controls the time constant (TC) in dB/second that is used for calculating the RMS input value. The time constant determines how rapidly the compressor will respond to input signal level changes, e.g. the "Attack" time.
<b>Hold (ms)</b>	 0	Controls the time (in ms) the compressor maintains its current output gain setting before it starts decreasing as the input level decrease.
<b>Decay (db/s)</b>	 10	Controls the rate at which the compressor gain decreases in response to decrease in the input signal level, e.g. the "Release" time.
<b>Soft Knee</b>		<p>Soft-knee lets the compressor ease into action, making a less-abrupt change from unprocessed signal to compressed signal. Typically it sounds better. If it is not activated, the default is hard-knee behavior, meaning compression reduces signal level immediately when the threshold is passed.</p> <p><input type="checkbox"/> Click here to see a comparison of soft-knee and hard-knee responses</p>
<b>Show Graph</b>		Opens the Compression Curve editor window.

Shown below are the curves of the Hi-Resolution RMS dynamics processor (right) and the regular RMS processor (left). The number of data points in this block is double that in the

ordinary one (66 instead of 33).



---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## N-Channel

---

This block, for multichannel use, uses an rms dynamics processor that lets you control the rms TC (time constant), Hold, Decay, and Soft Knee behavior, and displays the compression curve graph for your curve drawing. An rms compressor works on a longer average than a peak compressor, allowing some fast loud transients to pass while longer loud moments are prevented from doing so.

The block compares among the  $n$  channels and selects the one with the highest energy. In a 5-channel situation, the sound will be controlled and processed based on the loudest channel.

Push Show Graph to open the Compression Curve window.

For this block to be active, there must be signal at the red pin. That input acts as a detect signal, and the block compresses based upon it.

For multichannel use you will need to grow your algorithm, of course. Right-click the block's border or title and select **Grow Algorithm > 1. N-Channel > 1.** The signal applied to the red pin will scale the signals applied to the green pins accordingly.

### Controls:

<b>RMS TC (db/s)</b>	 120	Controls the time constant (TC) in dB/second that is used for calculating the RMS input value. The time constant determines how rapidly the compressor will respond to input signal level changes, e.g. the "Attack" time.
<b>Hold (ms)</b>	 0	Controls the time (in ms) the compressor maintains its current output gain setting before it starts decreasing as the input level decrease.
<b>Decay (db/s)</b>	 10	Controls the rate at which the compressor gain decreases in response to decrease in the input signal level, e.g. the "Release" time.

<b>Soft Knee</b>	<a href="#">Soft Knee</a>	Soft-knee lets the compressor ease into action, making a less-abrupt change from unprocessed signal to compressed signal. Typically it sounds better. If it is not activated, the default is hard-knee behavior, meaning compression reduces signal level immediately when the threshold is passed.  <input type="checkbox"/> Click here to see a comparison of soft-knee and hard-knee responses
<b>Show Graph</b>	<a href="#">Show Graph</a>	Opens the Compression Curve editor window.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Peak (gain)

This block uses a peak-detecting dynamics processor that lets you control Hold, Decay, and Soft Knee behavior, and displays the compression curve graph for your curve drawing.

It also lets you control Post Gain, the signal level after processing.

Peak detection operates on any signal passing above the threshold, no matter how fast a transient it is. However, this compressor has no TC (time constant) setting.

Because it can stop the briefest and quickest loud signals, peak compression (limiting, effectively) is useful for preventing any overdriving of inputs — for example, a transmitter. However, it sometimes sounds less natural than rms compression.

1. Drag the block into the workspace.
2. Right-click it and select the algorithm for your application:
  - **Stereo 1 Peak w/ gain**
  - **Stereo Separate Detect w/ gain**
3. Set these parameters to fit your application and click Show Graph and drag, add and remove (right-click) control points to achieve your desired processing curve.

**Note:** For the figure above, Stereo Separate Detect w/ gain was selected. Unlike with the other Detect algorithms, for peak limiting you need two separate signals to activate stereo performance.

As an application example, this block could be used at a radio station to attenuate background music according to the voice level of the announcer ("ducking"). The voice signal would be applied to the red pin, the music signal to the green one.

### Controls:

<b>Post Gain</b>		Post-processing gain, for increasing the overall signal level following dynamics processing.
<b>Hold (ms)</b>		Controls the time (in ms) the compressor maintains its current output gain setting before it

		starts decreasing as the input level decrease.
<b>Decay (db/s)</b>		Controls the rate at which the compressor gain decreases in response to decrease in the input signal level, e.g. the "Release" time.
<b>Soft Knee</b>		Soft-knee lets the compressor ease into action, making a less-abrupt change from unprocessed signal to compressed signal. Typically it sounds better. If it is not activated, the default is hard-knee behavior, meaning compression reduces signal level immediately when the threshold is passed.  <input type="checkbox"/> Click here to see a comparison of soft-knee and hard-knee responses
<b>Show Graph</b>		Opens the Compression Curve editor window.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Peak (no gain)

This block functions exactly like the Peak (gain) block (see that topic for details) but does not include the Post Gain knob.

### Controls:

<b>Hold (ms)</b>	 0	Controls the time (in ms) the compressor maintains its current output gain setting before it starts decreasing as the input level decrease.
<b>Decay (db/s)</b>	 10	Controls the rate at which the compressor gain decreases in response to decrease in the input signal level, e.g. the "Release" time.
<b>Soft Knee</b>		<p>Soft-knee lets the compressor ease into action, making a less-abrupt change from unprocessed signal to compressed signal. Typically it sounds better. If it is not activated, the default is hard-knee behavior, meaning compression reduces signal level immediately when the threshold is passed.</p> <p><input type="checkbox"/> Click here to see a comparison of soft-knee and hard-knee responses</p>
<b>Show Graph</b>		Opens the Compression Curve editor window.

## RMS (gain)

This block uses an rms dynamics processor that lets you control the rms TC (time constant), Hold, Decay, and Soft Knee behavior, and opens the compression curve graph for your curve drawing.

RMS works on a longer average than peak processors, thus allowing some fast loud transients to pass without compression, but operating more on longer segments that exceed the threshold.

1. Drag the block into the workspace
2. Right-click it and select the algorithm for your application:
  - **Stereo 1 RMS**
  - **Stereo 1 RMS/Detect**
  - **Mono 1 RMS**
  - **Mono 1 RMS/Detect**
3. Set these parameters to fit your application and click Show Graph and drag, add and remove (right-click) control points to achieve your desired processing curve.

**Note:** For the picture above, the block was chosen to include the /Detect algorithm, shown by the red pin. For this compressor to be active, signal has to be connected to it.

### Controls:

<b>Post Gain</b>	 Post Gain	Post-processing gain, for increasing the overall signal level following dynamics processing.
<b>RMS TC (db/s)</b>	 RMS TC (db/s) 120	Controls the time constant (TC) in dB/second that is used for calculating the RMS input value. The time constant determines how rapidly the compressor will respond to input signal level changes, e.g. the "Attack" time.
<b>Hold (ms)</b>	 Hold (ms) 0	Controls the time (in ms) the compressor maintains its current output gain setting before it starts decreasing as the input level decrease.
<b>Decay (db/s)</b>	 Decay (db/s) 10	Controls the rate at which the compressor gain decreases in response to decrease in the input

		signal level, e.g. the "Release" time.
<b>Soft Knee</b>	<a href="#">Soft Knee</a>	Soft-knee lets the compressor ease into action, making a less-abrupt change from unprocessed signal to compressed signal. Typically it sounds better. If it is not activated, the default is hard-knee behavior, meaning compression reduces signal level immediately when the threshold is passed.  <input type="checkbox"/> Click here to see a comparison of soft-knee and hard-knee responses
<b>Show Graph</b>	<a href="#">Show Graph</a>	Opens the Compression Curve editor window.

---

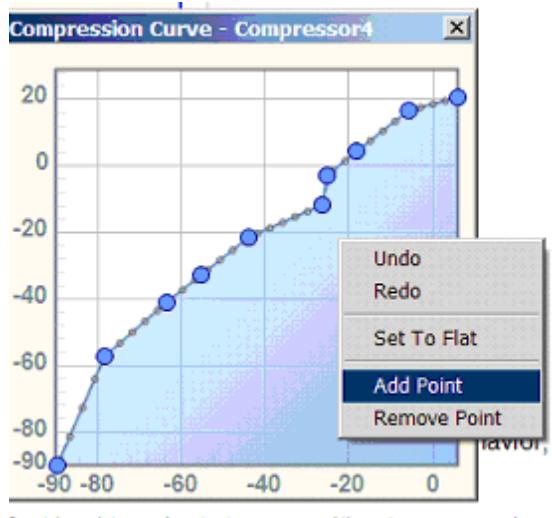
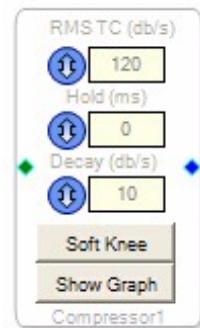
© 2006-2007 Analog Devices, Inc. All rights reserved.

## RMS (no gain)

This block functions exactly like RMS (gain) but does not include the knob for post gain.

For the block picture at right, the Mono Algorithm was chosen, meaning there is only one set of input/output pins.

Push Show Graph to open the Compression Curve window. Right-click along the curve to add or remove control points according to your needs.



This block uses an rms dynamics processor that can control the rms TC (time constant), Hold, Decay, and Soft Knee behavior, and also lets you open the compression curve graph for your curve creation.

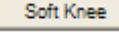
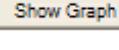
RMS works on a longer average than peak processors, thus allowing some fast loud transients to pass without compression, operating more on longer segments that exceed the threshold.

To use this block:

1. Drag and drop into the workspace
2. Right-click the block and select the algorithm for your application:
  - **Stereo 1 RMS**
  - **Stereo 1 RMS/Detect**
  - **Mono 1 RMS**
  - **Mono 1 RMS/Detect**

3. Set these parameters to fit your application and click Show Graph and drag, add and remove (right-click) graphical points to draw your desired processing curve.

**Controls:**

<b>RMS TC (db/s)</b>	 <b>RMS TC (db/s)</b> 120	Controls the time constant (TC) in dB/second that is used for calculating the RMS input value. The time constant determines how rapidly the compressor will respond to input signal level changes, e.g. the "Attack" time.
<b>Hold (ms)</b>	 <b>Hold (ms)</b> 0	Controls the time (in ms) the compressor maintains its current output gain setting before it starts decreasing as the input level decrease.
<b>Decay (db/s)</b>	 <b>Decay (db/s)</b> 10	Controls the rate at which the compressor gain decreases in response to decrease in the input signal level, e.g. the "Release" time.
<b>Soft Knee</b>		Soft-knee lets the compressor ease into action, making a less-abrupt change from unprocessed signal to compressed signal. Typically it sounds better. If it is not activated, the default is hard-knee behavior, meaning compression reduces signal level immediately when the threshold is passed.  <input type="checkbox"/> Click here to see a comparison of soft-knee and hard-knee responses
<b>Show Graph</b>		Opens the Compression Curve editor window.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## RMS (Display)

This block functions exactly like RMS (gain) with the Stereo 1 RMS algorithm, except that it comes with a signal level display.

The left bar shows the input level and the right bar the compression ratio. The **On** button toggles the level display on and off.

For a sample design using this block, see the Dynamics Processor Example.

### Controls:

<b>Post Gain</b>		Post-processing gain, for increasing the overall signal level following dynamics processing.
<b>RMS TC (db/s)</b>	 120	Controls the time constant (TC) in dB/second that is used for calculating the RMS input value. The time constant determines how rapidly the compressor will respond to input signal level changes, e.g. the "Attack" time.
<b>Hold (ms)</b>	 0	Controls the time (in ms) the compressor maintains its current output gain setting before it starts decreasing as the input level decrease.
<b>Decay (db/s)</b>	 10	Controls the rate at which the compressor gain decreases in response to decrease in the input signal level, e.g. the "Release" time.
<b>Soft Knee</b>		Soft-knee lets the compressor ease into action, making a less-abrupt change from unprocessed signal to compressed signal. Typically it sounds better. If it is not activated, the default is hard-knee behavior, meaning compression reduces signal level immediately when the threshold is

		<p>passed.</p> <p><input type="checkbox"/> Click here to see a comparison of soft-knee and hard-knee responses</p>
<b>Show Graph</b>		Opens the Compression Curve editor window.

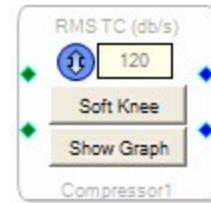
---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## RMS (no: gain, hold, decay)

This block uses an rms dynamics processor that lets you control the rms TC (time constant) and Soft Knee behavior, and opens the compression curve graph for your curve drawing.

RMS works on a longer average than peak processors, thus allowing some fast loud transients to pass without compression, but operating more on longer segments that exceed the threshold.



1. Drag the block into the workspace
2. Right-click it and select the algorithm for your application:
  - **Stereo 1 RMS (NO: gain, hold, decay)**
  - **Stereo 1 RMS/Detect (NO: gain, hold, decay)**
  - **Mono 1 RMS (NO: gain, hold, decay)**
  - **Mono 1 RMS/Detect (NO: gain, hold, decay)**
3. Click Show Graph and drag, add and remove (right-click) control points to achieve your desired processing curve.

For a sample design using this block, see the Dynamics Processor Example.

### Controls:

<b>RMS TC (db/s)</b>		Controls the time constant (TC) in dB/second that is used for calculating the RMS input value. The time constant determines how rapidly the compressor will respond to input signal level changes, e.g. the "Attack" time.
<b>Soft Knee</b>		Soft-knee lets the compressor ease into action, making a less-abrupt change from unprocessed signal to compressed signal. Typically it sounds better. If it is not activated, the default is hard-knee behavior, meaning compression reduces signal level immediately when the threshold is passed.  <input type="checkbox"/> Click here to see a comparison of soft-knee and hard-knee responses
<b>Show Graph</b>		Opens the Compression Curve editor window.

© 2006-2007 Analog Devices, Inc. All rights reserved.

# Filters

**Navigation:** Toolbox >  
Filters



## Filters

The Filters library of the Toolbox lets you access numerous filter / EQ blocks for shaping the frequency content of your signal. For information about the algorithms driving the blocks, check Algorithms. The following blocks are available:

- DC-Blocking
- De-Emphasis
- FIR
- General (1st-Order)
- General (1st-Order w/ Param / Lookup / Slew)
- General (2nd-Order)
- General (2nd-Order / Lookup)
- General (2nd-Order w/ Param / Lookup / Slew)
- General (2nd-Order / Index Selectable) Graphical
- Medium-Size EQ
- Pinking
- State-Variable
- State-Variable (Q input)
- Text-In (linked)
- Text-In (unlinked)
- Tracking

The topic pages in this book contain specific information about the individual blocks. Be sure to take a look at the Filters Example page to see sample schematics using some filter blocks.

**Note:** Most of these filters will let you select between double- and single-precision computation. If there available system resources, double-precision should normally be used: it uses 56 bits for each calculation and takes 10 instructions per filter. Single-precision uses 28 bits for calculations, takes 6 instructions per filter, but saves you 3 RAM spaces over the double-precision algorithm. Single-precision should not be used for signal content below 1/10 the sampling frequency or for high-Q filters.

To view the response of a filter's settings, drag a filter block into the workspace attach a Simulation Stimulus block to the input and a Simulation Probe to the filter's output. Now Click the Probe button Stimulus button. Take a look at the filter examples.

© 2006-2007 Analog Devices, Inc. All rights reserved.

**Navigation:** Toolbox > Filters >  
DC-Blocking



## DC-Blocking

This filter is pre-configured, that is, it has set parameters for blocking direct-current components. It is used to remove dc that may be present in your signal.



While the block is ready to use after dragging into the workspace, there is the option to Add to the algorithm.

The dc-blocking behavior is computed according to the following transfer function:

$$H(z) = \frac{1 - z^{-1}}{1 - Rz^{-1}}$$

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## De-Emphasis

---

The De-Emphasis filter is pre-configured, with set parameters, and is used to attenuate the high-frequency components boosted during recording with pre-emphasis.



When the De-Emphasis block is used on audio recorded with pre-emphasis, its signal-to-noise ratio is greatly improved.

This block is ready to use, with no need (or ability) to add or grow algorithms.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

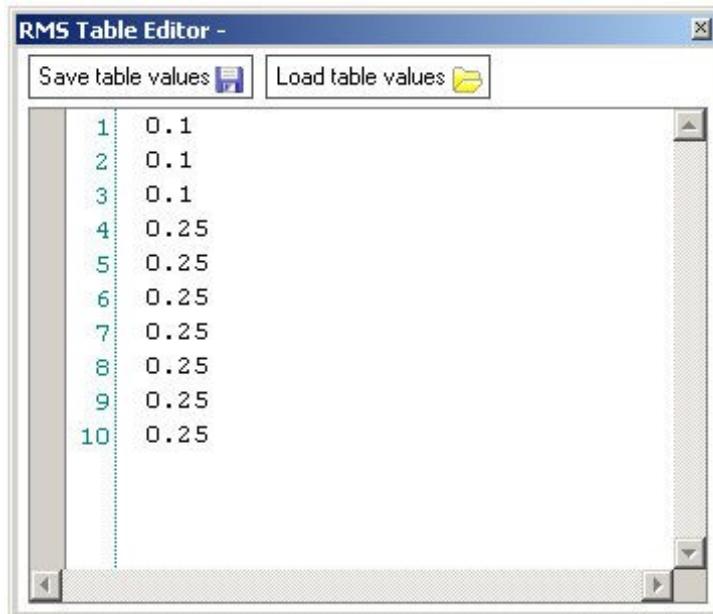
## FIR

The FIR (finite impulse response) block lets you design any FIR filter desired.



- 1) Drag the block into the workspace.
- 2) Click Table.
- 2) Enter the coefficients as calculated by your chosen software (see below). (Max is 800.)

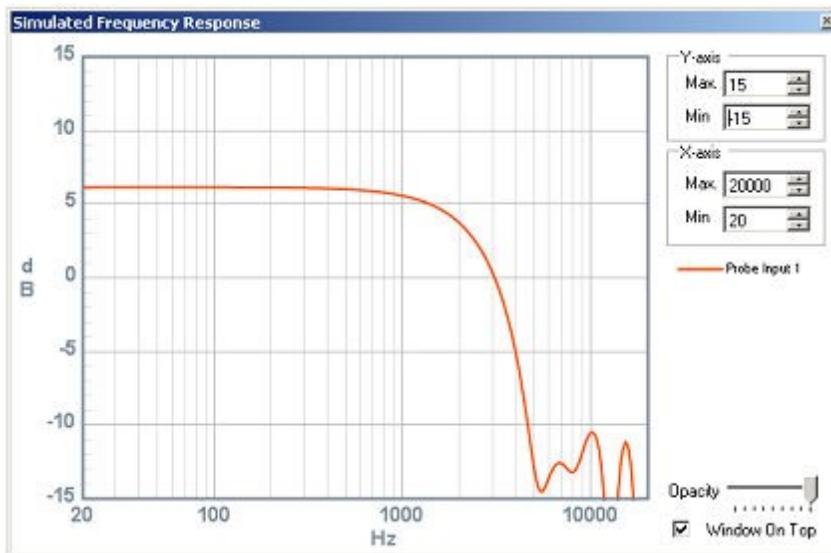
Frequency response can be shaped by specifying the appropriate filter coefficients, as shown here:



You can use this popup window to input as many more coefficients as you desire, resulting in your own custom FIR filter. (Remember, the more numbers the closer to an ideal IIR filter, but at the cost of memory.)

CAD programs are available which simplify the design of lowpass, highpass, bandpass, or bandstop FIR filters. A popular one was developed by Parks and McClellan and uses the Remez exchange algorithm. The design begins by specifying such parameters as passband ripple, stopband [attenuation] ripple, and the transition region. One such CAD program is QED1000 from Momentum Data Systems; a free version is downloadable from [www.mds.com](http://www.mds.com).

The frequency response of the filter coefficients in the RMS Table editor figure shown above is pictured below:



To add input / output sets, right-click and Add Algorithm, IC1, FIR. After the default algorithm is established, this block can have algorithms added to it. (If you're using more than one DSP board, you will need to add the initial default algorithm for the desired board.) Right-click the block and select Add Algorithm > IC N > Pink Noise Filter. This adds another set of input/output pins for connection.

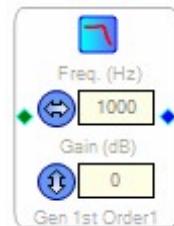
For background detail and theory, see FIR Filter Algorithm.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## General (1st-Order)

The General 1st-Order block allows you to design 1st-order lowpass and highpass filters.



Drag the block into the workspace and it's ready to use. As with other blocks, there's the option to Grow or Add to this algorithm. Observe, however, that with this block **growing** the algorithm will add another frequency band to the block, which is equivalent to having two filters in series. **Adding** an algorithm adds another input/output pair to the block, which is equivalent to adding a filter in parallel.

To switch among highpass, lowpass, and flat, click the blue frequency response icon. This can be done in real time, without needing to recompile the project. Enter your desired values in the text fields to set the cutoff frequency and overall gain (sometimes called scale gain) of the filter. Or click the arrows to increment values for these parameters. To increment them very quickly, click and hold, dragging a little.

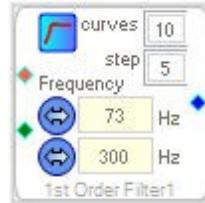
For details about the algorithms driving this block, see General 1st-Order Filters in Algorithm Information.

To review filter concepts and terms in general (e.g., Q), see the next topic page, covering General-Purpose (2nd-Order) filters.

To view your work, drag a filter block into the workspace, a Simulation Stimulus block as input, a Simulation Probe to come after the filter, set some test parameters, and click Probe, then Stimulus. To see some results, take a look at the filter examples.

## General (1st-Order w/ Param / Lookup / Slew)

The General (1st-Order / Lookup) block provides a selectable set of High-pass or Low-pass 1st-order filter responses with smooth (slew) transitions when selecting among responses.



The block allows you to define a set of filter responses (low or high pass) which be selected through the external control input in the end system. The number of selectable responses or **curves** is variable as is the slew rate or **step** which controls how quickly the filter response changes.

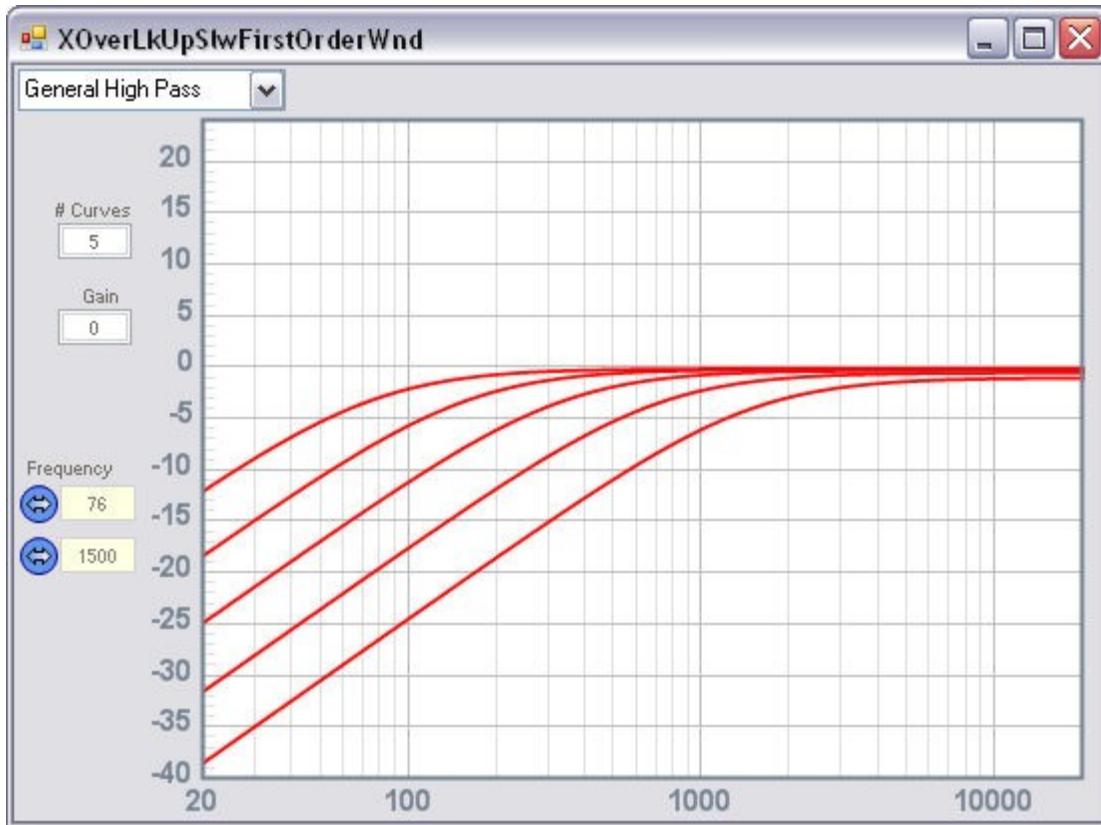
The filter's response is displayed in the Filter Control window (see below). Note that the response curves are linearly spaced between the low and high cutoff frequency values.

This block's algorithm stores a set of filter coefficients in a table on the DSP. To select a curve, use the Index Lookup Table, the Counter block, or the DC Input block and connect to the red pin. Using the GPIO blocks you could control the selected responses with a knob, rotary encoder or button.

### To open the Filter Control Window:

Click the icon button:

- Enter the number of curves desired in the # Curves field.
- Enter desired filter Gain (-/+ dB).
- Enter the desired frequency range for the curves (low and high frequency targets)



**Slew:**

The transition between two curves (selected by the red control input pin) is smooth, without any clicking sounds or instabilities because the hardware/software Slew is used to transition between one filter response and another. Essentially one filter is cross-faded into the next.

Note: Using Slew requires double the memory space of non-slew filters.

## General (2nd-Order)

The General (2nd-Order) block gives access to a wide variety of 2nd-order (biquad)filter algorithms.

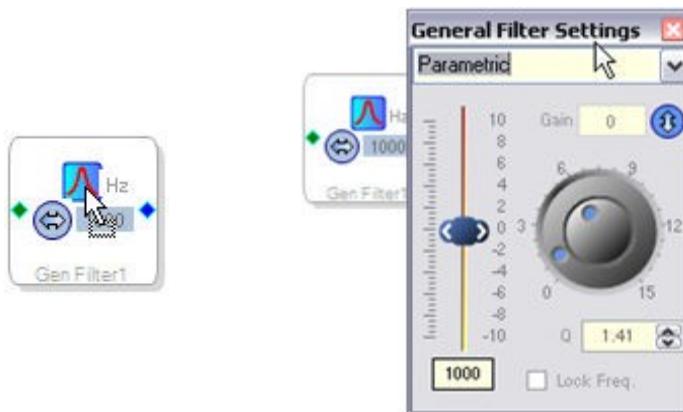
The available filter types are:

- Parametric
- Shelving
- General High-Pass
- General Low-Pass
- General Band-Pass
- General Band-Stop
- Butterworth Low-Pass / High-Pass
- Bessel Low-Pass / High-Pass
- Tone Control
- IIR Coefficient (direct coefficient entry)
- 1st-Order Low-Pass / High-Pass
- All-pass
- Peaking
- Notch
- Chebyshev Low-Pass / High-Pass



**To open the filter control window, click on the icon button:**

Select the desired filter type from the drop-down combo-box list. The parameter control's will change to reflect the currently selected filter type as will the image in the block's icon button.



---

This block's algorithms use biquad filter designs based on Robert Bristow-Johnson's work in this field.

$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}$$

Common variables:

- $\omega_0 = 2\pi f_0 / F_s$
- $gainLinear = 10^{(gain/20)}$

### Peaking

Transfer Function

$$H(s) = \frac{s^2 + \frac{A}{Q}s + 1}{s^2 + \frac{s}{A*Q} + 1}$$

Coefficients       $\alpha = \sin(\omega_0)/(2*Q)$

```

a0 = 1 + alpha/A
a1 = -2 * cos(omega0)
a2 = 1 - alpha/A
b0 = (1 + alpha*A) *
      gainLinear
b1 = -(2 * cos(omega0)) *
      gainLinear
b2 = (1 - alpha*A) *
      gainLinear
  
```

### Low-Shelf

Transfer Function

$$H(s) = A * \frac{s^2 + \frac{\sqrt{A}}{Q}s + A}{As^2 + \frac{\sqrt{A}}{Q}s + 1}$$

Coefficients       $\alpha = \sin(\omega_0)/2 * \sqrt{(A + 1/A)(1/S - 1) + 2}$

```

a0 = (A+1) + (A-1)*cos(omega0) + 2*sqrt(A)*alpha
a1 = -2*( (A-1) + (A+1)*cos(omega0) )
a2 = (A+1) + (A-1)*cos(omega0) - 2*sqrt(A)*alpha
b0 = A*( (A+1) - (A-1)*cos(omega0) + 2*sqrt(A)*alpha ) *
      gainLinear
b1 = 2*A*( (A-1) - (A+1)*cos(omega0) ) * gainLinear
b2 = A*( (A+1) - (A-1)*cos(omega0) - 2*sqrt(A)*alpha ) *
      gainLinear
  
```

### High-Shelf

Transfer Function

$$H(s) = A * \frac{As^2 + \frac{\sqrt{A}}{Q}s + 1}{s^2 + \frac{\sqrt{A}}{Q}s + A}$$

Coefficients

$$\text{alpha} = \sin(\omega_0)/2 * \text{sqrt}( (A + 1/A)*(1/S - 1) + 2 )$$

$$\begin{aligned} a0 &= (A+1) - (A-1)*\cos(\omega_0) + 2*\text{sqrt}(A)*\text{alpha} \\ a1 &= 2*( (A-1) - (A+1)*\cos(\omega_0) ) \\ a2 &= (A+1) - (A-1)*\cos(\omega_0) - 2*\text{sqrt}(A)*\text{alpha} \\ b0 &= A* ( (A+1) + (A-1)*\cos(\omega_0) + 2*\text{sqrt}(A)*\text{alpha} ) \\ &\quad * \text{gainLinear} \\ b1 &= -2*A* ( (A-1) + (A+1)*\cos(\omega_0) ) * \text{gainLinear} \\ b2 &= A* ( (A+1) + (A-1)*\cos(\omega_0) - 2*\text{sqrt}(A)*\text{alpha} ) \\ &\quad * \text{gainLinear} \end{aligned}$$

### Lowpass

Transfer Function

$$H(s) = \frac{1}{s^2 + \frac{s}{Q} + 1}$$

Coefficients

$$\text{alpha} = \sin(\omega_0)/(2*Q)$$

$$\begin{aligned} a0 &= 1 + \text{alpha} \\ a1 &= -2*\cos(\omega_0) \\ a2 &= 1 - \text{alpha} \\ b0 &= (1 - \cos(\omega_0)) * \text{gainLinear} / 2 \\ b1 &= 1 - \cos(\omega_0) * \text{gainLinear} \\ b2 &= (1 - \cos(\omega_0)) * \text{gainLinear} / 2 \end{aligned}$$

### Highpass

Transfer Function

$$H(s) = \frac{1}{s^2 + \frac{s}{Q} + 1}$$

Coefficients

$$\text{alpha} = \sin(\omega_0)/(2*Q)$$

$$\begin{aligned} a0 &= 1 + \text{alpha} \\ a1 &= -2*\cos(\omega_0) \\ a2 &= 1 - \text{alpha} \\ b0 &= (1 + \cos(\omega_0)) * \text{gainLinear} / 2 \\ b1 &= -(1 + \cos(\omega_0)) * \text{gainLinear} \\ b2 &= (1 + \cos(\omega_0)) * \text{gainLinear} / 2 \end{aligned}$$

### Bandpass

Transfer Function

$$H(s) = \frac{s}{s^2 + \frac{s}{Q} + 1}$$

Coefficient

$$\text{alpha} = \sin(\omega_0) * \text{sinh}(\ln(2)/2 * \text{bandwidth} *$$

$$\omega_0/\sin(\omega_0)$$

```
a0 = 1 + alpha
a1 = -2*cos(omega0)
a2 = 1 - alpha
b0 = alpha * gainLinear
b1 = 0
b2 = -alpha * gainLinear
```

### **Bandstop**

Transfer Function 
$$H(s) = \frac{s^2 + 1}{s^2 + \frac{s}{Q} + 1}$$

Coefficients  $\text{alpha} = \sin(\omega_0) * \sinh(\ln(2)/2 * \text{bandwidth} * \omega_0/\sin(\omega_0))$

```
a0 = 1 + alpha
a1 = -2*cos(omega0)
a2 = 1 - alpha
b0 = 1 * gainLinear
b1 = -2*cos(omega0) * gainLinear
b2 = 1 * gainLinear
```

### **Butterworth LP**

Transfer Function 
$$H(s) = \frac{1}{s^2 + \sqrt{2}s + 1}$$

Coefficients  $\text{alpha} = \sin(\omega_0) / 2.0 * 1/\sqrt{2}$

```
a0 = 1 + alpha
a1 = -2*cos(omega0)
a2 = 1 - alpha
b0 = (1 - cos(omega0)) * gainLinear / 2
b1 = 1 - cos(omega0) * gainLinear
b2 = (1 - cos(omega0)) * gainLinear / 2
```

### **Butterworth HP**

Transfer Function 
$$H(s) = \frac{s^2}{s^2 + \sqrt{2}s + 1}$$

Coefficients  $\text{alpha} = \sin(\omega_0) / 2.0 * 1/\sqrt{2}$

```
a0 = 1 + alpha
a1 = -2*cos(omega0)
a2 = 1 - alpha
b0 = -(1 + cos(omega0)) * gainLinear / 2
b1 = -(1 + cos(omega0)) * gainLinear
b2 = -(1 + cos(omega0)) * gainLinear / 2
```

### **Bessel LP**

Transfer Function	$H(s) = \frac{3}{s^2 + 3s + 3}$
Coefficients	alpha = sin( $\omega_0$ ) / 2.0 * 1/sqrt(3)
	a0 = 1 + alpha
	a1 = -2*cos( $\omega_0$ )
	a2 = 1 - alpha
	b0 = (1 - cos( $\omega_0$ )) * gainLinear / 2
	b1 = 1 - cos( $\omega_0$ ) * gainLinear
	b2 = (1 - cos( $\omega_0$ )) * gainLinear / 2

**Bessel HP**

Transfer Function	$H(s) = \frac{s^2}{s^2 + 3s + 3}$
Coefficients	alpha = sin( $\omega_0$ ) / 2.0 * 1/sqrt(3)
	a0 = 1 + alpha
	a1 = -2*cos( $\omega_0$ )
	a2 = 1 - alpha
	b0 = -(1 + cos( $\omega_0$ )) * gainLinear / 2
	b1 = -(1 + cos( $\omega_0$ )) * gainLinear
	b2 = -(1 + cos( $\omega_0$ )) * gainLinear / 2

For all of the above filters, the coefficients are divided by  $a0$ , normalizing them and making  $a0 = 1$  so that only 5 coefficients must be stored. In the actual implementation on the DSP, when the coefficients are stored in parameter RAM,  $a1$  and  $a2$  need to be inverted. This is done automatically, in software, before the parameters are written to memory.

© 2006-2007 Analog Devices, Inc. All rights reserved.

## General (2nd-Order / Lookup)

The General (2nd-Order / Lookup) block gives access to a wide variety of 2nd-order IIR (infinite impulse response) filter algorithms. See General 2nd-Order Filters (in Algorithm Information) for details about the algorithms driving these blocks.

The filters available are:

- Tone
- Peaking
- General LP/HP
- Butterworth LP/HP
- Bessel LP/HP
- Chebyshev LP/HP



This block is implemented by a biquad filter that has multiple sets of coefficients in tables on the DSP. To select curves (lookup), use an Index Lookup Table, a Counter block, or a DC Input block in your design and connect it to the red pin. Using the GPIO blocks you could control the selected responses with a knob, rotary encoder or button.

### To open the Filter Control Window:

Click the icon button:

The curve is defined using the Tone Control window (shown below).

Enter the number of curves desired in the # Curves field.

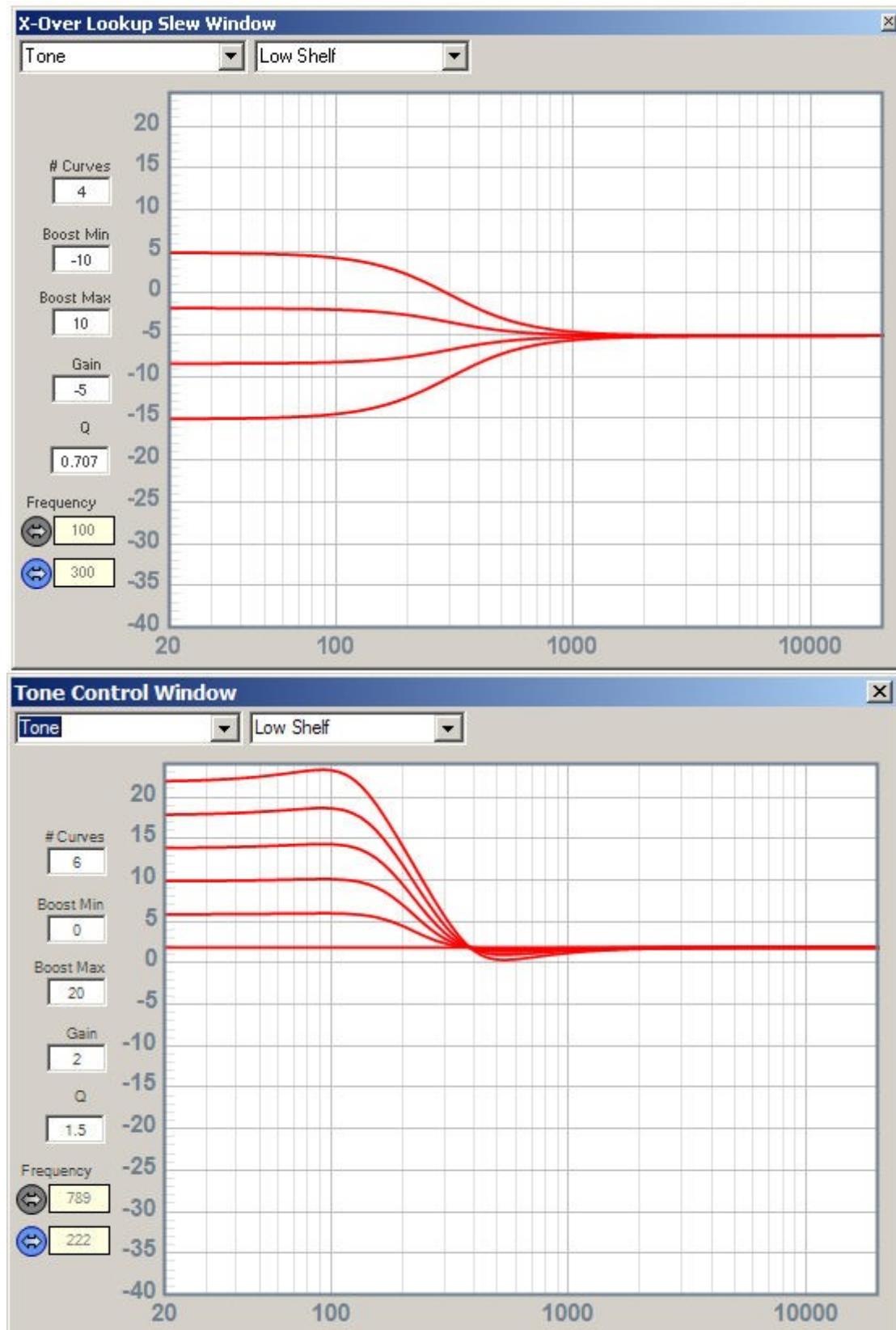
Enter Boosts, (overall) Gain, and Q in their fields.

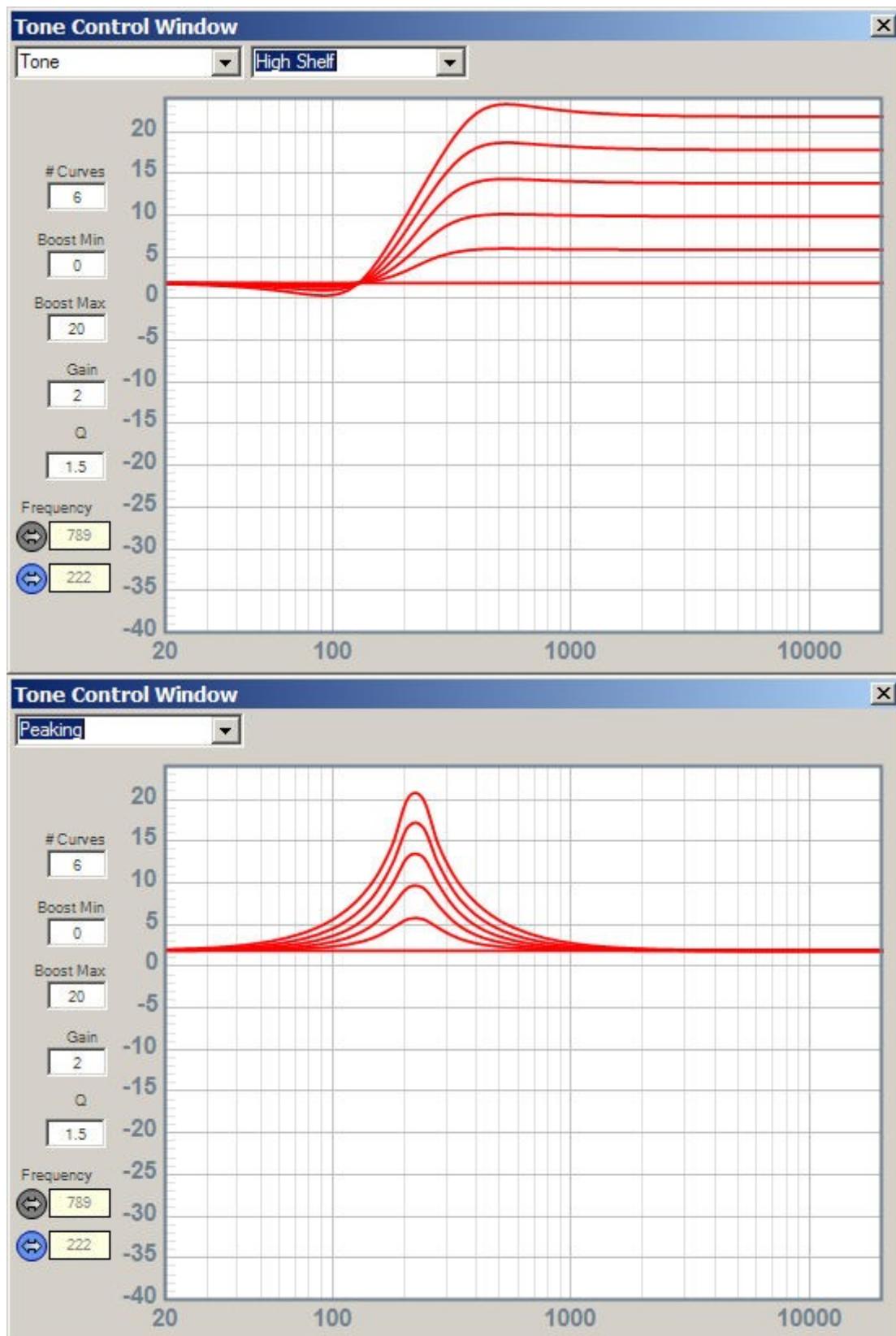
Enter the desired cutoff or center (peaking filters) frequency in the Frequency fields.

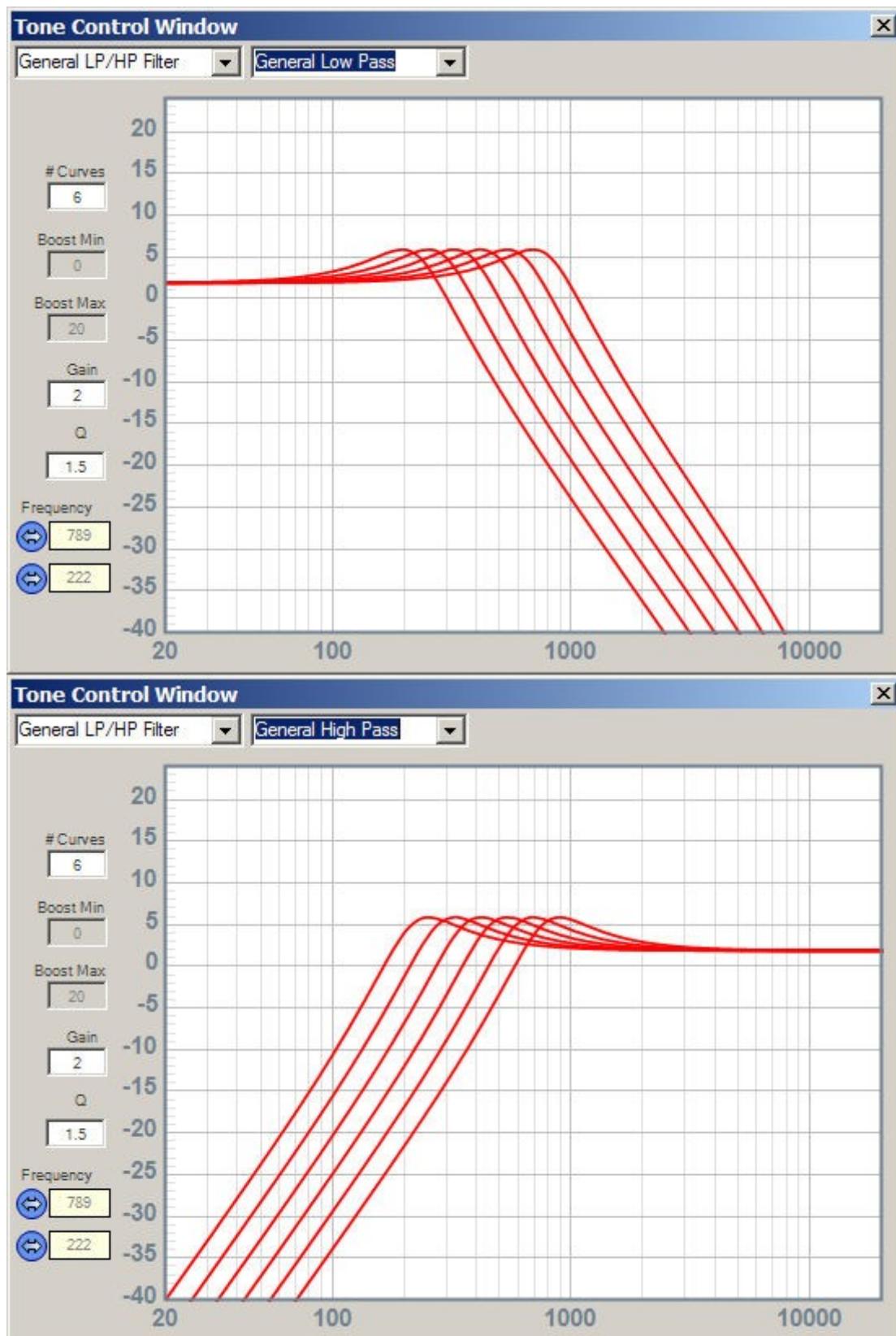
Other parameters to enter will vary with filter type.

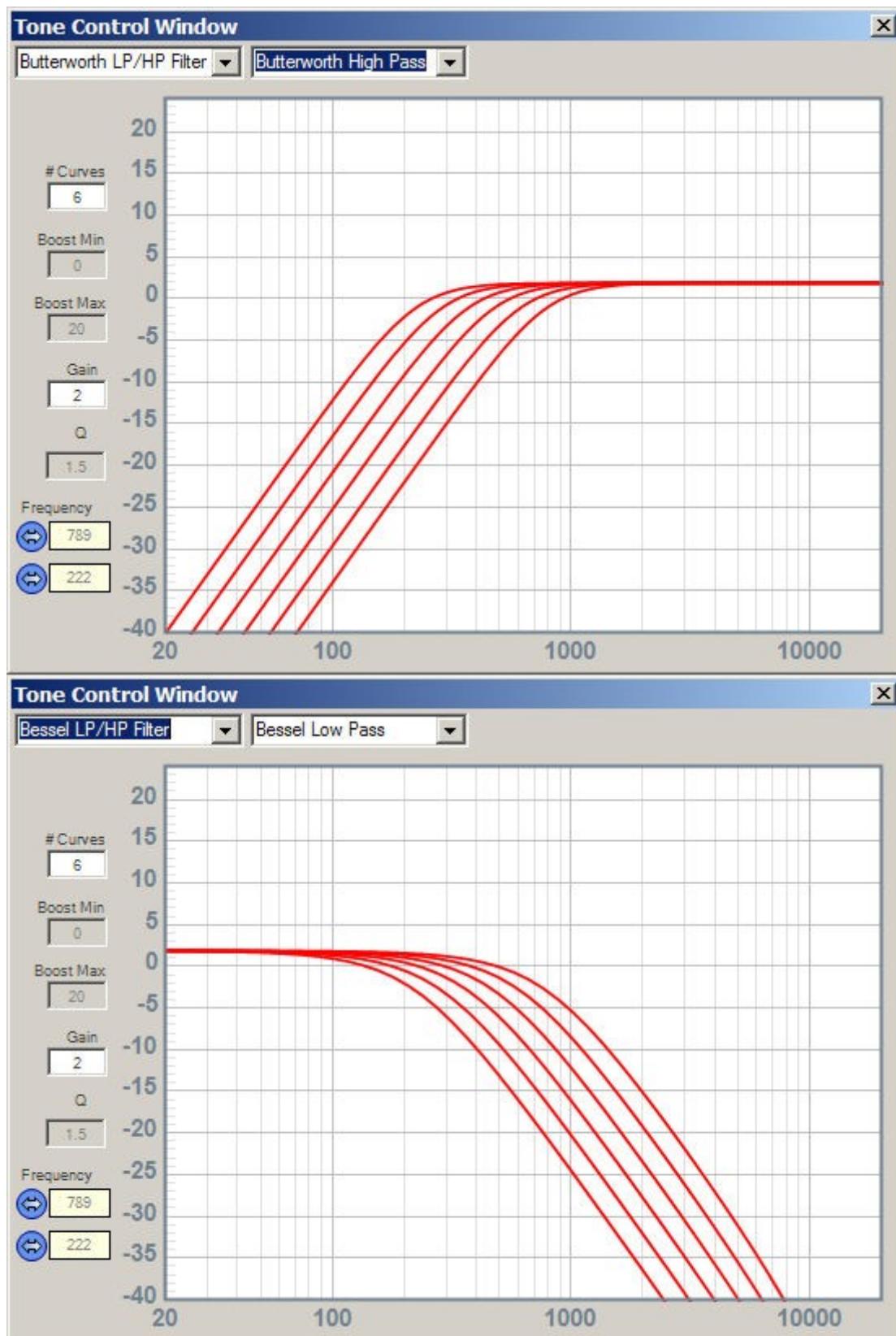
The variety and range of filters are remarkable, as can be seen from the following examples:

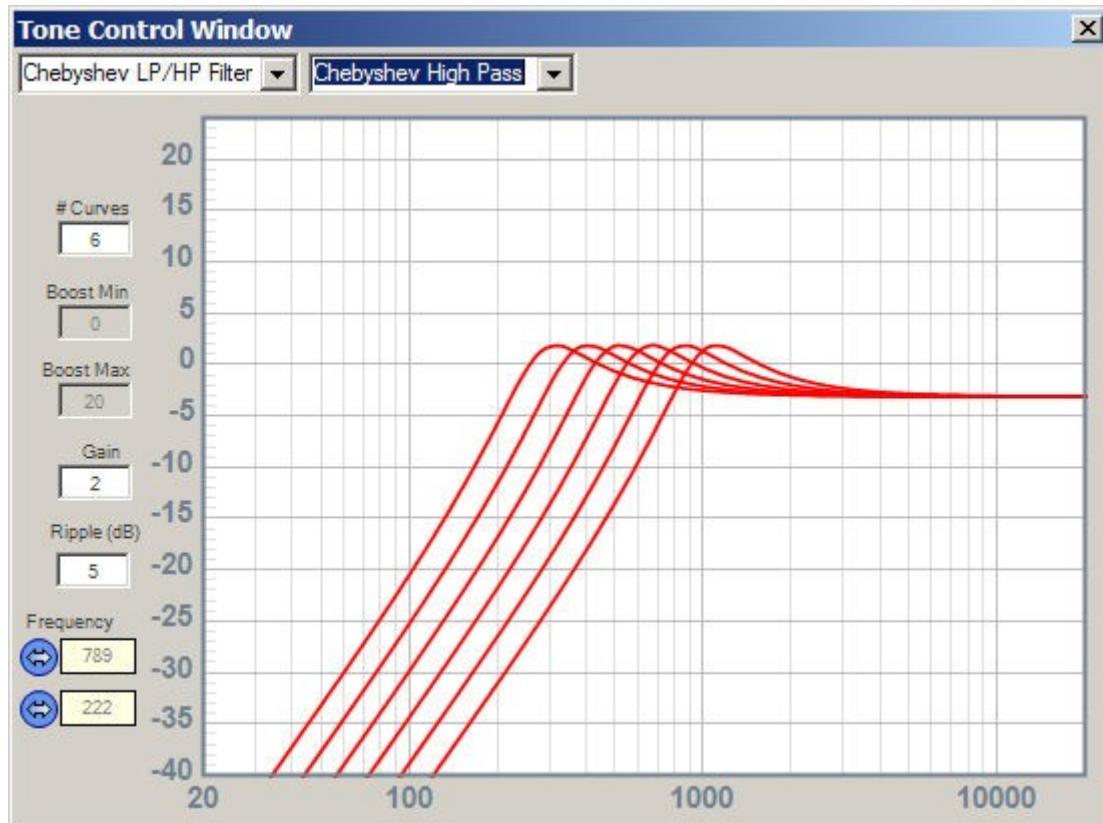
## SigmaStudioHelp











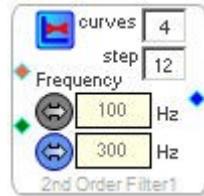
**Note:** For  $n$  curves, the selected index should not exceed  $n - 1$ , i.e., the index range is 0 to  $n - 1$ . If you select an  $n$ th curve, misbehavior or errors may result.

## General (2nd-Order w/ Param / Lookup / Slew)

The General (2nd-Order / Lookup) block gives access to a wide variety of 2nd-order IIR (infinite impulse response) filter algorithms. See General 2nd-Order Filters (in Algorithm Information) for details about the algorithms driving these blocks.

The filters available are:

- Tone
- Peaking
- General LP/HP
- Butterworth LP/HP
- Bessel LP/HP
- Chebyshev LP/HP



The block is simply a biquad filter that has stored a set of coefficients in tables in the DSP. To select curves (lookup), use an Index Lookup Table, a Counter block, or a DC Input block in your design and connect it to the red pin. Using the GPIO blocks you could control the selected responses with a knob, rotary encoder or button.

### To open the Filter Control Window:

Click the icon button:

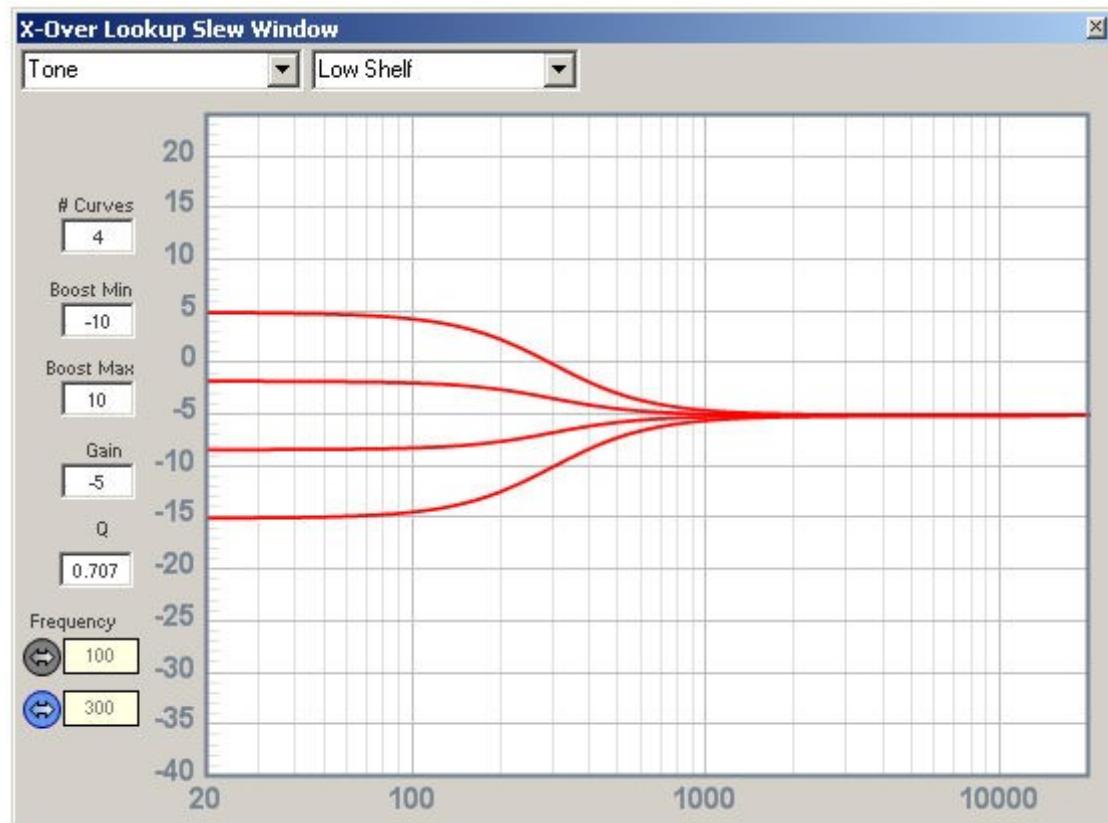
Enter the number of curves desired in the # Curves field.

Enter Boosts, (overall) Gain, and Q in their fields.

Enter the desired cutoff or center (peaking filters) frequency in the Frequency fields.

Other parameters to enter will vary with filter type.

The variety and range of filters are remarkable, as can be seen from the many examples in the General (2nd-Order / Lookup) topic page. Below is one:



For more details see the Example illustrating usage of this block.

The chief difference between the General (2nd-Order / Lookup) block and this Slew version may be observed when the index changes (red pin) to select a new curve. The transition from previous curve to next curve is smooth, without any occasional zipping or clicking sounds, the reason being that the Slew version takes more memory space. As the designer, you will want to match your hardware appropriately. Consider the General (1st-Order w/ Param / Lookup / Slew) if system resources are scarce.

**Note:** For  $n$  curves, the selected index shouldn't exceed  $n - 1$ , i.e., the index range is 0 to  $n - 1$ . If you select an  $n$ th curve, misbehavior or errors may result.

**Navigation:** Toolbox > Filters >  
**General (2nd-Order / Index Selectable) Graphical**



## General (2nd-Order / Index Selectable) Graphical

The General (2nd-Order / Index Selectable) block provides a wide variety of 2nd-order filter algorithms. This block extends the functionality of the General (2nd-Order / Lookup) filter with an enhanced graphical filter design tool. In addition, each of this filter's response curves can be designed independently, allowing you to use complex filter configurations to match your desired system response.



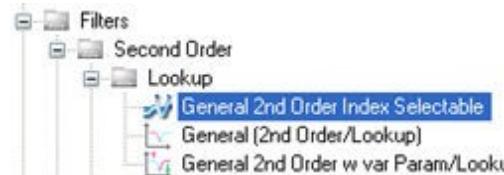
This filter provides:

- Selection from a set of filter responses via external control input pin.
- Graphical design of filter response curves.
- Independent filter type and settings for each response curve.

This block is implemented as a biquad filter that has multiple sets of coefficients stored in tables on the DSP. See General 2nd-Order Filters (in Algorithm Information) for details about the biquad filter design techniques used for this block.

### Tree ToolBox Path:

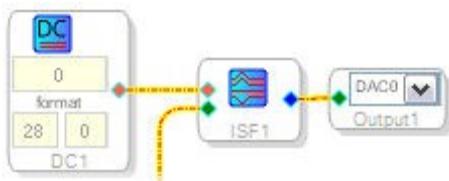
Filter -> Second Order -> Lookup -> General Second Order Index Selectable



### **Index Selection (Control Pin):**

The Multiple Index Selectable Filter block has an external input (orange pin) that is used to select any of the filters defined in the control window. To select a particular filter response curve, use an Index Lookup Table, a Counter block, or a DC Input block in your design and connect it to the control input pin.

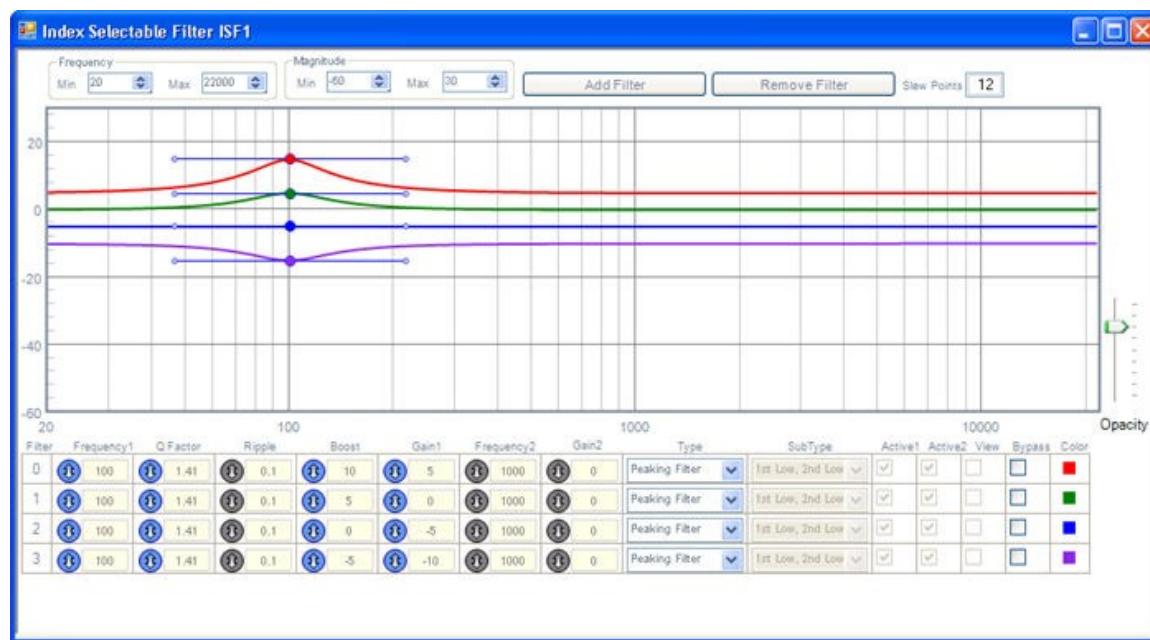
The selection input should be a 28.0 format integer value between 0 and the number of curves. For example if your filter has 6 curves, the selection input value should be a 28.0 integer value from 0 to 5.



### **Filter Control Window:**

The window will show initially have four different filter curves of the same type, as depicted in the figure below. If you wish to add additional filter curves, by default the Boost will decrease by a factor of 5 dB each time a new filter is added.

**To open the Filter Control Window, Click the icon button:**

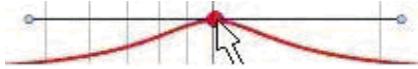


You can add additional filter curves by clicking the **Add Filter** button; the block supports a **maximum of 15** filter response curves. To remove a filter, select the filter in the graph or clicking its index in the Filter column and press the **Remove Filter** button.

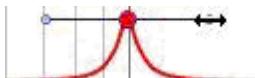
### **Controls:**

On the window we observe second order filters represented by different colors, and adjusted from independent controls shown at the bottom of the graph.

- To change a filter's characteristics, like "**Frequency**" or "**Boost**", just click on the colored circle on filter graph, and drag to the desirable position. If preferred, any of the parameters can be changed directly from the spin controls.



- To change the "**Q**" factor using the graph, click on the horizontal line control above the colored circle, and drag it left or right, adjusting the filter's width.

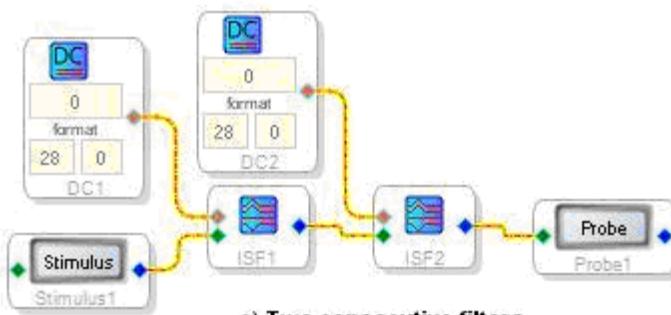


- In the "**Type**" column, choose between Low-Pass, High-Pass, Peaking, Tone, IIR, and First Order filters.



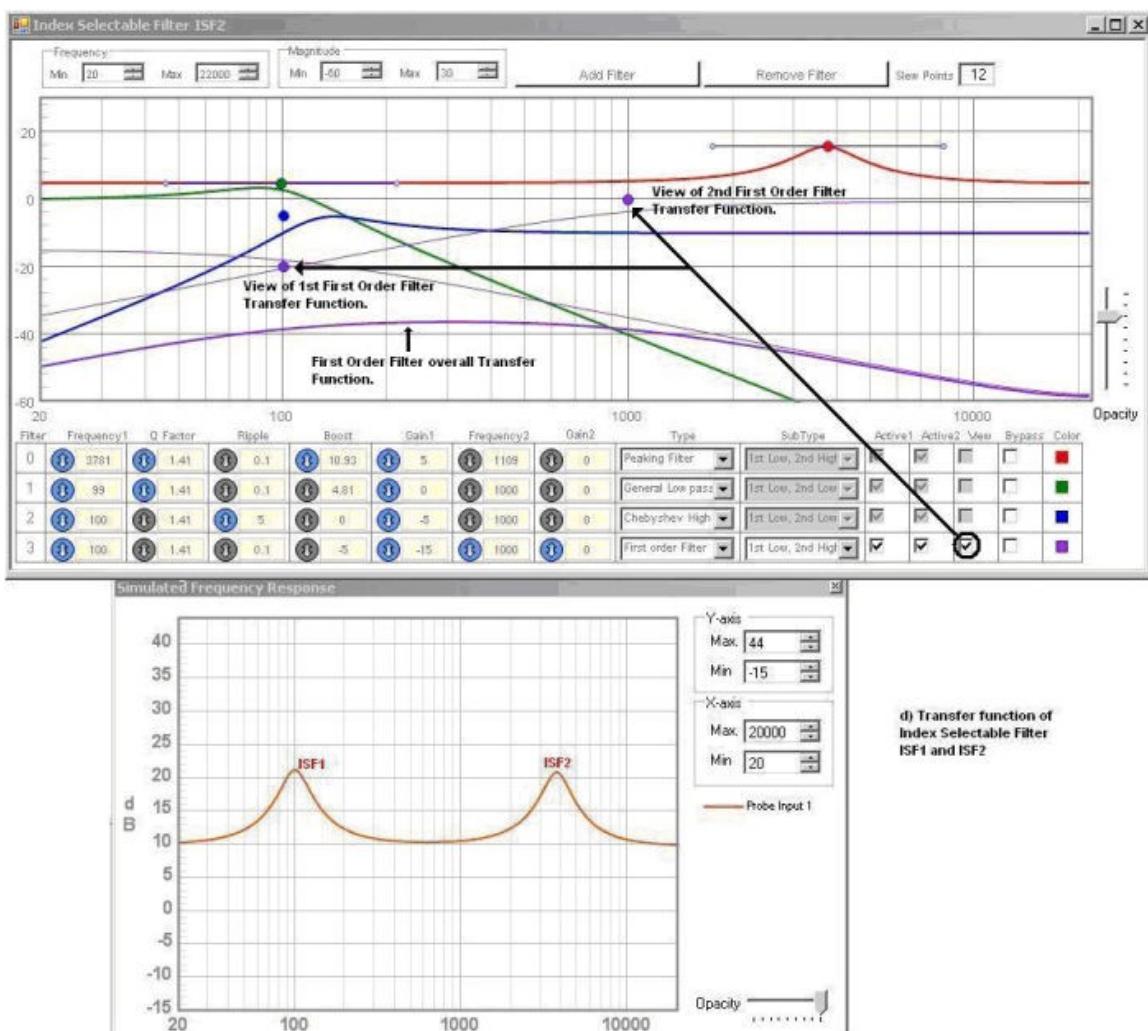
- The "**Sub Type**" section is only available for the First Order filter type. In first order mode, you have the flexibility of choosing between two cascaded first order filters in a configuration of Low - Low, Low - High, High - Low, or High - High combinations. The "**Active 1**" and "**Active 2**" check boxes also apply to First Order filters only, and allow you to independently enable or disable either of the two first order filters.
- The "**View**" section only applies for the First Order filter type, this tool allows you to manage each first order filter and control its settings graphically. Refer to the Example for more details.
- The "**Bypass**" check box disables the filter when checked.
- The plot view can be modified by changing the **Frequency** and **Magnitude** settings. These settings apply to the interface only and do not affect filter response.
- The "**Slew Points**" control sets how many transition points the algorithm uses to transition from one selected filter curve to another, increasing the number of points will provide smoother transitions.
- The "**Opacity**" control sets the transparency value of the control window.

#### **Example:**



c) Two consecutive filters

In this example, ISF1 and ISF2 are the names of the two consecutive blocks on Figure c. Both blocks contain four filters, and are set to the "zeroth" index filter, curve 0, selected by the DC input in "28.0" format. (Other formats cannot be used for selection). The ISF1 window is shown below. The cascaded Transfer Function is represented at the bottom. A different filter response curve can be selected by changing the value of the DC input block to corresponding filter index.



Using cascaded Index Selectable Filters (ISF) as in the above example, it is possible to simulate

complex responses like the Fletcher & Manson psychoacoustic contours of loudness.

© 2006-2007 Analog Devices, Inc. All rights reserved.

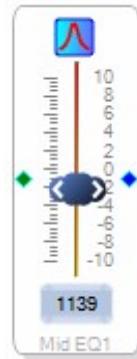
**Navigation:** Toolbox > Filters >  
Medium-Size EQ



## Medium-Size EQ

This block gives access to two general 2nd-order filters: Peaking and Shelving EQ. The algorithms driving this block are the same as for the other 2nd-order filters, but it simply offers an alternate layout and control of parameters that may prove more useful for your application.

As can be seen from the figure at right, the block controls frequency, gain, and filter type.



For information on this block's algorithms, see General 2nd-Order Algorithms.

To use this block:

1. Drag it into the workspace, right-click it and select **Add Algorithm** > **IC N.**
2. Choose which algorithm meets your needs. (1 - 6 channels governs the number of your block's ins and outs, and for information on precision, refer back to the note with Filters.)
  - 1-channel - Double-Precision
  - 1-channel - Single-Precision
  - 2-channel - Double-Precision
  - 2-channel - Single-Precision
  - 3-channel - Double-Precision
  - 3-channel - Single-Precision
  - 4-channel - Single-Precision
  - 4-channel - Double-Precision
  - 5-channel - Double-Precision
  - 5-channel - Single-Precision
  - 6-channel - Double-Precision
  - 6-channel - Single-Precision
3. Click the blue icon to choose which filter you want to implement: Peak or High- / Low-Shelf.
4. Double-click right in the middle of the block to bring up the EQ



parameter window:

5. Select from the dropdown menu which of the two types of filters you want and enter the parameters desired. Peaking is the default.

After choosing the filter type, right-click the block to either Grow or Add to it. Growing adds another frequency band to the block, equivalent to having two individual filters in series, while adding an algorithm adds another input/output pair, equivalent to adding a filter in parallel.

### **Peaking EQ**

Peaking boosts or cuts a designated center frequency.

Gain: This field sets the overall gain (scale gain) of the filter. Edit your desired value (+/-15) in the upper-right Gain field or click the arrows.

Boost / cut level: Control the level of the filtered portion of the response with the slider.

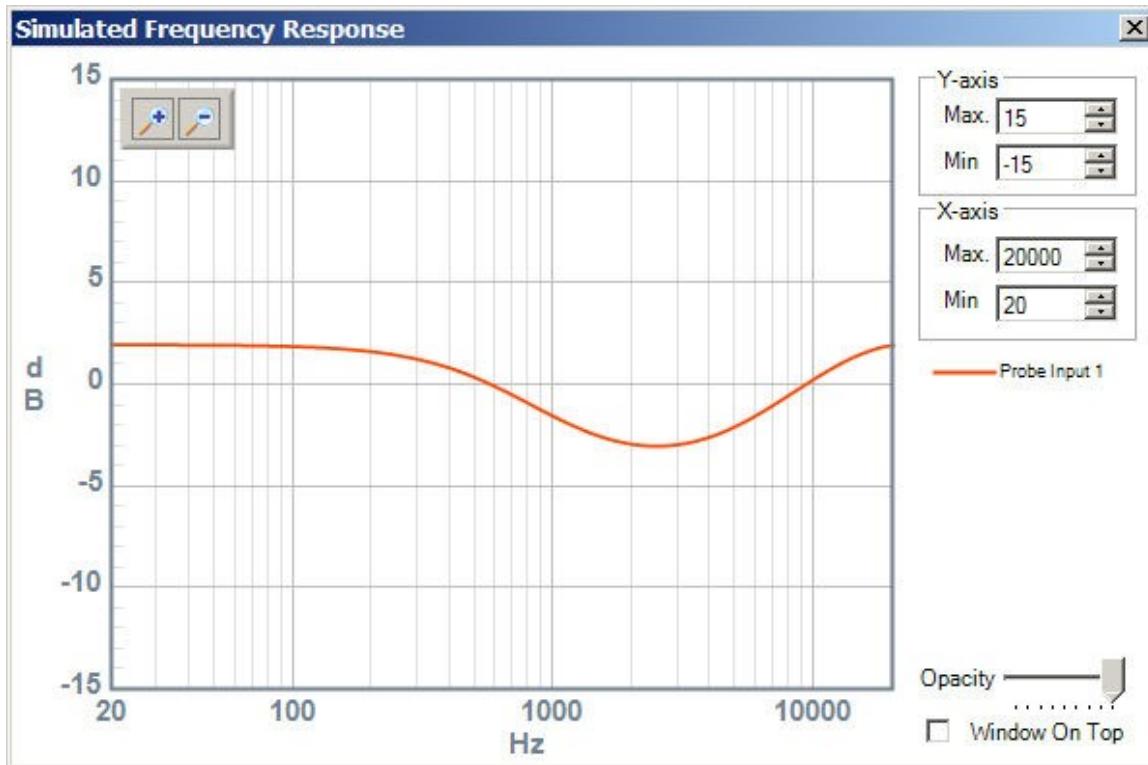
Frequency: Enter the desired peak or dip center frequency in the lower-left field, or click the slider's < > arrows. (Hold or try to drag them for faster change.)

Q: Set the Q you want (max 51) by entering a value in the field, clicking its arrows, or using the concentric knobs, where the outer one controls the integer value and the inner one the decimal value. Q governs the narrowness of the filter, being the ratio of the center frequency to the half-power points (-3dB) on either side. The higher the Q the faster the transition between passband and stopband.

This is a quick and versatile block, as can be seen by comparing the below curves with their settings. It is highly useful for developing an educated ear.



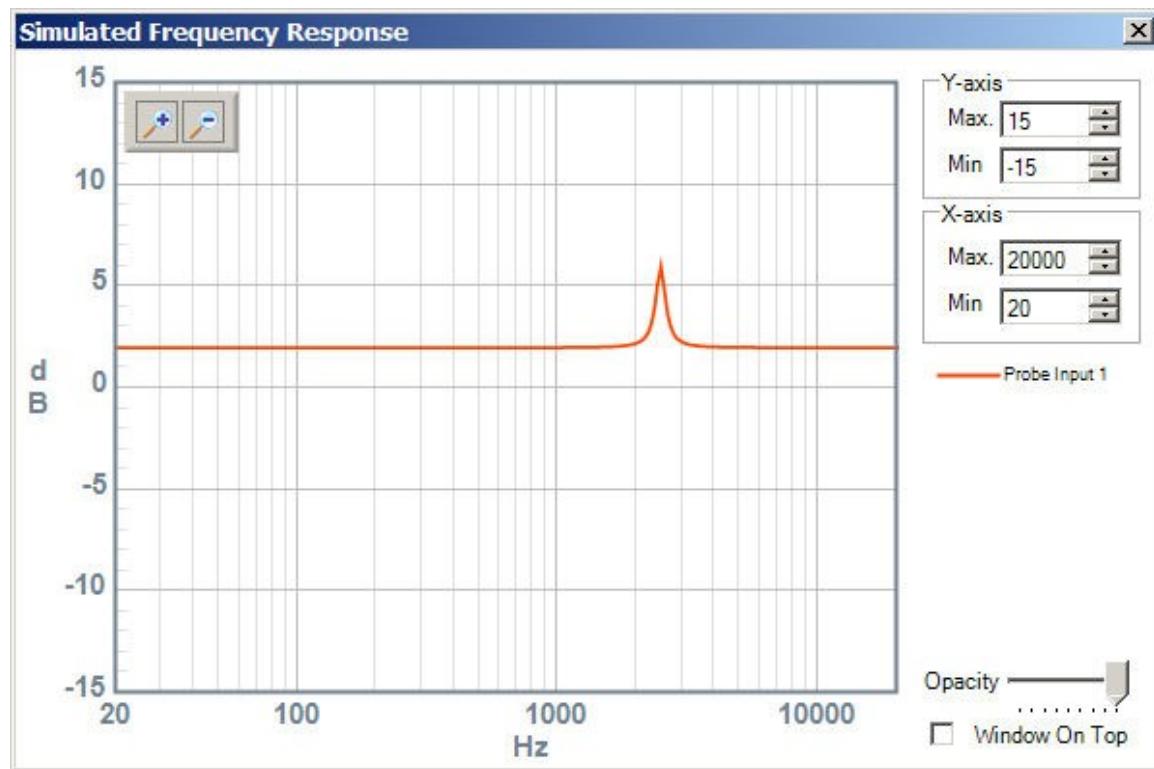
Negative gain and shallow Q (above) produce this:



while positive gain and narrower Q (same scale gain and same frequency; below),



quickly produce this:



### Shelving EQ: Low / High Shelf

Shelving evenly boosts or cuts all frequencies either above (= High Shelf, ) or below (= Low Shelf, ) the cutoff frequency.

The other parameters:

Cutoff frequency: Enter the cutoff frequency in the field below the slider, or click its arrows. (Hold and attempt to drag the < > for rapid incrementation.) This frequency is the cutoff point between the shelf boost/cut and the unaffected (flat) response.

Gain (filter): Control the filter boost or cut with the slider. Negative values give a cut for all frequencies above (High Shelf) or below (Low Shelf) the cutoff, while positive values give a boost for all frequencies above (High Shelf) or below (Low Shelf) the cutoff.

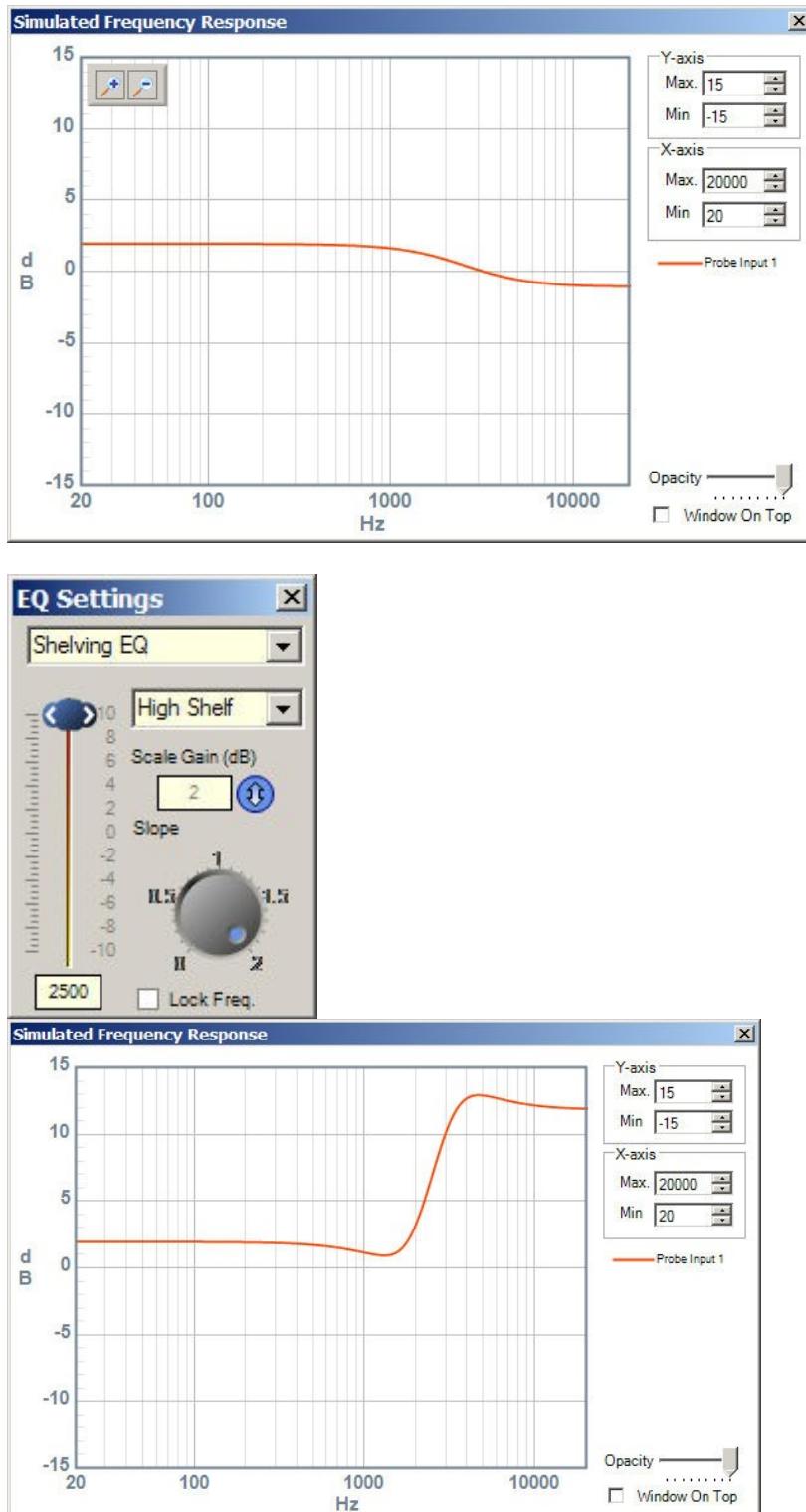
Slope (Q): Edit the slope of the filter with the control knob, 0 - 2. Right-click it to enter more-precise values.

Slope controls filter steepness and hence the transition between the boost/cut and the flat response.

Scale Gain (dB): This value controls the overall gain of the filter. Enter it in the field or click the arrows at right.

Handy for ear training, this is a quick and versatile block, as can be seen by comparing the below curves with their parameters.





As with Peaking, Shelving parameters may be changed handily and the results immediately checked for audible effect.

© 2006-2007 Analog Devices, Inc. All rights reserved.

**Navigation:** Toolbox > Filters >  
**Pinking**



## Pinking

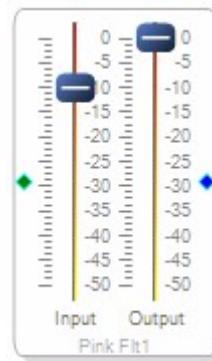
The classic use of this type of filter is to convert white noise, which is equal energy per hertz, to pink noise, which is equal energy per proportional or constant-percentage (e.g., logarithmic) band.

Such energy displays as flat on any log scale graph when bundled (integrated) appropriately.

The Pinking cell takes any input signal and outputs a signal with a 3dB drop per octave in linear terms.

See Pink Filter Algorithm for information on the algorithm, and also look at the Level Detectors Example to see the filter used in a schematic.

After you have established your default algorithm, this cell can have algorithms added to it. If you're using more than one DSP board, you will need to add the initial algorithm for the desired board. In either case, right-click the cell and select **Add Algorithm > IC N > Pink Noise Filter** to add another set of input/output pins.

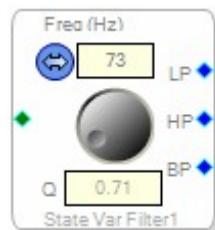


---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## State-Variable

The State-Variable block allows for simultaneous access to three different filter types: lowpass, highpass, and bandpass. See State Variable Algorithms for information about the parameters and calculations for this algorithm.



Set the center frequency in the Freq (Hz) field or by dragging the arrows. For examples and details, see Filters Example.

To have external control over Q from the block, refer to the State-Variable (Q input) Filter.

The three output pins let you choose among LP, HP, BP filters. The nature of this algorithm is to compute the coefficients for all filter types, giving you access to all of the filters simultaneously. If your application doesn't require use of all filter types, you can connect the output to a terminal block.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## State-Variable (Q input)

This block allows for simultaneous access to three different filter types: lowpass, highpass, bandpass. See State-Variable Algorithms for information about the parameters and calculations for this algorithm.



As can be seen from the figure (above right), this block has two inputs and three outputs. The green (standard input) pin is for the signal and the orange is for a value to control filter Q. It's common to control the Q of the filter either by sending a DC input value to this pin or by using it with the RMS table to generate Q parameters from the input signal instead.

To have control over Q from a block parameter, refer to the State-Variable Filter.

The three output pins allows to choose among LP, HP, BP filters. This algorithm computes the coefficients for all filter types, giving simultaneous access. If your application does not require the use of all filter types, then the output may be connected to the terminal block.

Set the center frequency in the Freq (Hz) field or by dragging the <--> arrows.

---

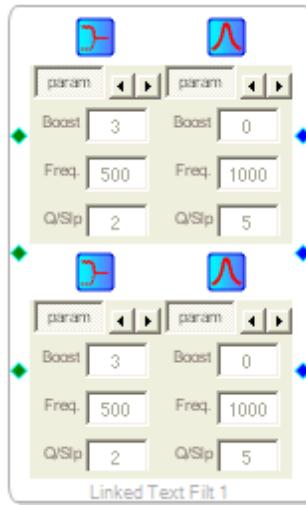
© 2006-2007 Analog Devices, Inc. All rights reserved.

## Text-In (linked)

An alternative to the General-Purpose filter, the Text-In block lets you set parameters using text fields, without having to open another window. The algorithms are the same, giving access to the wide variety of 2nd-order-filter behaviors, although not all the algorithms are offered with Text-In.

The Text-In (linked) filter differs from the unlinked version in that (1) it allows simultaneous control of added algorithms and (2) when you grow upon an added algorithm, a corresponding parallel filter will be added to match the previously added algorithm.

At right is an example of a default algorithm of 1ch - Double Precision grown by 1, with an added 2ch - Double Precision. The parameters are the same in the vertical settings because of the linked nature of this block; see below for more about that.



See General 2nd-Order Algorithms for information about the algorithms driving these blocks. The available filters are:

- Peaking EQ
- Shelving EQ: Low-/High-Shelf
- General: Low-/Highpass

To use this block:

1. Drag into the workspace, right-click the block and select **Add Algorithm > IC N.**
2. Select your desired algorithm:
  - 1-channel - Double-Precision
  - 1-channel - Single-Precision
  - 2-channel - Double-Precision
  - 2-channel - Single-Precision
  - 3-channel - Double-Precision
  - 3-channel - Single-Precision
  - 4-channel - Single-Precision
  - 4-channel - Double-Precision
  - 5-channel - Double-Precision
  - 5-channel - Single-Precision
  - 6-channel - Double-Precision
  - 6-channel - Single-Precision
4. Click the blue icon to choose your filter: Peak, Lowpass, Highpass, Low-Shelf, High-Shelf.

5. Use the < / > arrows to select gain, with the Gain text field, or param, clicking which displays the text fields Boost (Peak and Shelf EQ only), Freq, Q/Slope:

- Boost is the amount of boost or cut applied to the designated frequency range
- Freq is the center frequency
- Q/slope is the sharpness of the curve
- Gain is the overall gain of filter.

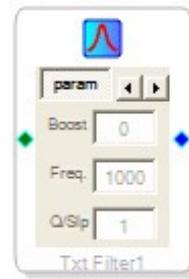
As with other blocks, there's the option to either Grow or Add to this algorithm. Observe that with this block, **growing** the algorithm adds another frequency band to the block, which is equivalent to having two filters in series. **Adding** an algorithm adds another input/output pair to the block, which is equivalent to adding a filter in parallel.

Since this is a linked block, changing the values in one parameter box after you add an algorithm automatically updates the other filter in parallel.

Better yet, after adding an algorithm, if you grow it the series filter automatically adds parallel filters to match the previous case. The same is true when you add onto a grown algorithm: the parallel filter will automatically add series filters to match.

## Text-In (unlinked)

An alternative to the General-Purpose filter, the Text-In block lets you set parameters using text fields, without having to open another window. The algorithms are the same, giving access to the wide variety of 2nd-order-filter behaviors, although not all the algorithms are offered with Text-In.



See the General 2nd Order Algorithms page for information about the algorithms driving these blocks. The available filters are:

- Peaking EQ
- Shelving EQ: Low-/High-Shelf
- General: Low-/Highpass Filter

To use this block:

1. Drag into the workspace, right-click the block and select **Add Algorithm > IC N.**
2. Select your desired algorithm:
  - 1-channel - Double-Precision
  - 1-channel - Single-Precision
  - 2-channel - Double-Precision
  - 2-channel - Single-Precision
  - 3-channel - Double-Precision
  - 3-channel - Single-Precision
  - 4-channel - Single-Precision
  - 4-channel - Double-Precision
  - 5-channel - Double-Precision
  - 5-channel - Single-Precision
  - 6-channel - Double-Precision
  - 6-channel - Single-Precision
4. Click the blue icon to choose your filter: Peak, Lowpass, Highpass, Low-Shelf, High-Shelf.
5. Enter values in the param tab for Boost (Peak and Shelf EQ), Freq, Q/Slope. Click the right arrow and the gain tab and enter the filter gain.
  - Boost is the amount of boost / cut applied to the designated frequency range
  - Freq is the center frequency
  - Q/slope is the sharpness of the curve
  - Gain is the overall gain of filter.

As with other blocks, there's the option to either Grow or Add to this algorithm. Observe that with this block, growing the algorithm adds another frequency band to the block, which is equivalent to having two filters in

series. Adding an algorithm adds another input/output pair to the block, which is equivalent to adding a filter in parallel.

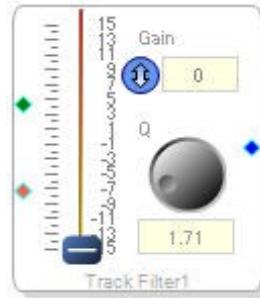
The behavior of this block in terms of growing and adding algorithms is like the other filter blocks. A special version of this filter, the Text-In (linked) (which see), forces added algorithms to share the same parameters, and grown algorithms must add a corresponding parallel algorithm. In other words, it allows simultaneous control of added algorithms, and when you grow upon an added algorithm, a corresponding parallel filter will be added to match any previously added algorithm.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Tracking

Tracking Filters are useful for dynamic shifting of filtering. This particular tracking Filter is a Peaking Filter algorithm. In many applications rather than having a fixed center frequency for a Peaking filter, it would be useful to have a moving center frequency. For example a LFO (low-frequency oscillator) could drive a Tracking Filter in order to produce a phasing audio effect. The tracking filter allows for the center frequency to be determined by an external input.



**Input Pins:**

There are two inputs and one output to the tracking filter block. The first input is the actual audio input that you want to filter. The second input pin is the value of the center frequency of the filter. You will notice that this pin is orange, denoting that it is a control pin and not an audio pin. Here you can connect LFOs or DC input blocks to designate the desired fixed or moving center frequency of the tracking filter.

**Algorithm Description:**

The Tracking Filter is a unique filter in the SigmaStudio Filter library. It is one of the only filters in which the coefficients are calculated dynamically by the DSP processor. The equations are embedded in the block's algorithm code, and depending on the input center frequency, the coefficients are then generated.

The input center frequency that the tracking filter algorithm is expecting, is a 5.23 data format value, computed by:  $\text{Hz}/(\text{fs}/2)$ . For example if you are operating at  $\text{fs} = 44100 \text{ Hz}$  and you have an  $\text{fc} = 500 \text{ Hz}$  the value being sent should be:  $0.0226757 = 500 / (44100/2)$ .

The Tracking Filter is based on the coefficient generation for an IIR Peaking Filter. The following equations show how the DSP attains the 5 biquad coefficients from the controls. The controls of the tracking filter are overall\_gain which is the dB value of the overall gain of the filter, boost which is the dB value of the cut or boost of the Peaking filter, and Q which determines the width of the frequency cut or boost. The center frequency fc of the Peaking Filter is given by the second input pin.

```

gainLinear = 10^(overall_gain/20)
w = 2*pi*fc/Fs
alpha = sin(w)/(2*Q)
A = 10^(boost/40)
a0 = 1 + alpha/A
a1 = -2 * cos(w)
a2 = 1 - alpha/A
b0 = (1 + alpha*A) * gainLinear
b1 = -(2 * cos(w)) * gainLinear
b2 = (1 - alpha*A) * gainLinear

```

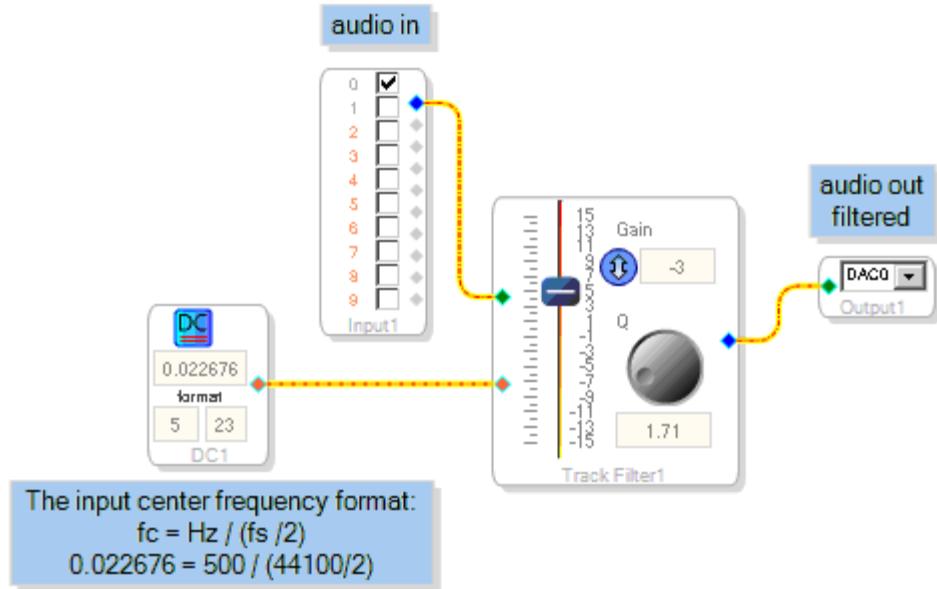
**Usage:**

The tracking filter can be used in many situations. It should be used when a changing center frequency to a peaking filter is desired. For fixed center frequency applications you should use a general second order filter. There is a lot of overhead in MIPS/Memory in order for the changing center frequency to be supported for this algorithm. However, the first example here using the tracking filter has a fixed center frequency just for ease in explaining the format of the center frequency.

---

**Example 1:**

Here a DC entry block is connected to the second pin of the tracking filter. The value inside the block corresponds to 500Hz. Thus the audio coming in on the first pin will be boosted 5dB at 500Hz with a Q of 1.71, and an overall gain adjustment of -3dB will be applied over the entire frequency range.

**Example 2:**

Here is a more complex example showing how the tracking filter can be used with a LFO to create a phasing effect. The first input pin is receiving a merged version of the input signal (essentially a mono version of the input signal). The blocks preceding the second input pin create a LFO that sweeps the center frequency at a rate of .1Hz. The range of the swept center frequency is 441 – 11025 Hz.

These are the equations explaining how the range is determined:

$$Fc \text{ control input} = (F2 - F1) * \sin(.1\text{Hz}) + F2 + F1$$

Since a normal sine tone ranges from  $\pm 1$ , this LFO ranges from:

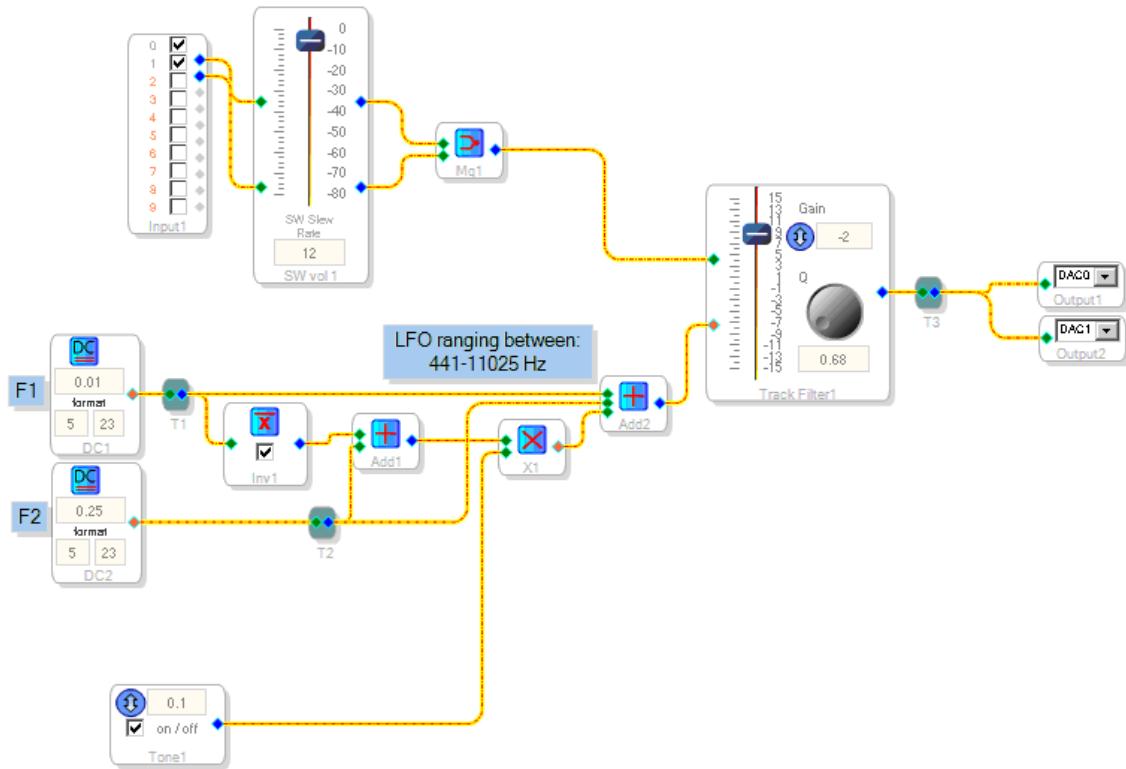
$$\begin{aligned} +(F2-F1) + F2 + F1 &= 2*F2 \\ -(F2-F1) + F2 + F1 &= 2* F1 \end{aligned}$$

Then to figure out what the frequencies are in Hz corresponding to .01 and .25 in the schematic use the following formula:

$$(44100/2) * Fc = Value \text{ in Hz}$$

Keep in mind though that F1 and F2 are the numbers in the DC block, but the range will go from  $2*F1$  to  $2*F2$ , so first multiply by 2 to get the correct range value, then use the above formula to find out the value in Hz:

$$\begin{aligned} (0.01*2)*44100/2 &= 441 \text{ Hz} \\ (0.25*2)*(44100/2) &= 11025 \text{ Hz} \end{aligned}$$



© 2006-2007 Analog Devices, Inc. All rights reserved.

## GPIO Conditioning

### GPIO Conditioning

---

GPIO stands for general-purpose input/output. The Toolbox's GPIO Conditioning library gives you access to the following blocks:

- Push and Hold
- Rotary Encoder
- Software Debounce
- Up/Down Control, Index Output

See the GPIO Conditioning Example for schematic samples utilizing these blocks.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Push and Hold

This block can be used for functions like a pushbutton, to condition a GPIO input to create pulses in response to the user pushing or holding the button.

A typical application would be a volume control comprising two buttons, one for up and the other for down.



1. Drag the block into the workspace.
2. Right-click it and select the algorithm:

- **push\_hold**
- **push/hold 2-in 2-out**
- **push/hold with two-button mute**

3. Set the parameters to fit your application:

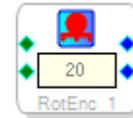
<b>Hold (ms)</b>	Determines how long the signal is held before the repeat pulses are generated.	
<b>Repeat (ms)</b>	Sets the interval between repeated pulses. Enter the time in milliseconds in the field.	

**Note:** For the picture above right, push\_hold was selected. Use push/hold 2-in 2-out to condition two GPIO inputs, for example one up and one down. Push/hold with mute works similarly but with the extra feature that if both buttons are pressed, a mute pulse is generated (bottom output pin). To un-mute, any of the buttons can be pressed.

## Rotary Encoder

---

The Rotary Encoder block processes the inputs from a rotary encoder and outputs an "up" and a "down" signal. The algorithm also incorporates a software "de-bouncer" for each of the inputs.



- 1) Drag the block into the schematic.
- 2) Connect the inputs to two GPIOs. These would correspond to the 2 out-of-phase output pins of a rotary encoder. In hardware configuration, set the GPIOs to "no debounce."

The value in the field sets the time constant (in samples) for the debouncer; the default is 20. Adjust this value by trial and error using the rotary encoder in your end system.

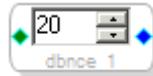
The Example illustrates use of the block.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Software Debounce

The contacts of mechanical switches and encoders can "bounce" when changing positions; meaning the voltage may fluctuate between states several times when a button is pushed. When the transition is not clean erroneous states can be set in your system. This block debounces (removes the transition ripple) from a signal, by waiting a specified amount of time between sampling periods. This provides a clean transition signal to the output.



Typically, this block is used to debounce a GPIO input signal.

To use this block, drag it into the schematic and connect the input to a GPIO signal.

The debounce time control sets the time constant for the debouncer, in samples; the default is 20. For best results adjust the value by trial and error for whatever hardware is connected to the GPIO input.

---

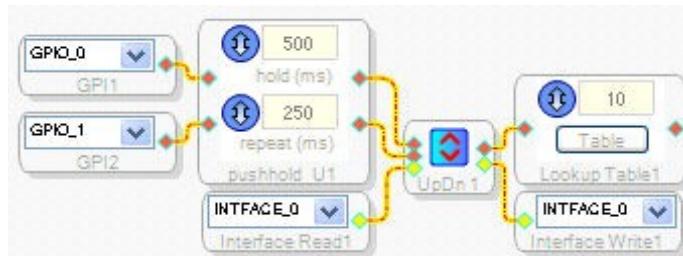
© 2006-2007 Analog Devices, Inc. All rights reserved.

## Up/Down Control, Index Output

This block takes in two inputs, one up and one down, and uses them to generate an index for a lookup table. The starting index is pre-loaded into an interface register and the up/down inputs increment or decrement the value.



1. Drag the block into your schematic.
2. Connect the red control inputs to GPIOs that have been conditioned by Push and Hold, or to the outputs of the Rotary Encoder.
3. Connect the yellow input pin to an Interface Read block.
4. Connect the yellow output to an Interface Write block, with the same register selected for both the interface read and write blocks.
5. The output will be an index value, so it should be connected to a block such as a lookup table or tone control.



The Example illustrates use of this block.

### Algorithm

The algorithm first reads an index from a stored value in the interface register. This value is then adjusted by two control signals, up/down signals, generated either by buttons conditioned by Push and Hold or by the GPIOs connected to outputs of the Rotary Encoder conditioned by that block. The control signals increment or decrement the index value, which is then sent to the output pin. The algorithm also writes the value back to the interface register to be stored.

# IO

## Inputs and Outputs

The toolbox's I/O library gives you access to the input/output blocks essential for bringing the signal from the hardware IC's physical input connections, into the schematic design, and back out to the hardware's output connections. The available input and output blocks will depend on the particular processor used in your design.

- ASRC Input/Output
- Aux ADC Input
- General-Purpose Input
- General-Purpose Output
- Input
- Interface Read
- Interface Write
- Output
- SPDIF Input/Output

**Note:** The Input and output blocks represent physical and limited hardware resources. If you attempt to insert more input or output blocks than there are available resources you will receive the "Not enough DSP Resources for this Algorithm" error. For example, only one input block per processor can be used in a schematic. In addition, inputs and outputs can only be assigned to a single block at a time. For example, you cannot create two output blocks that represent the same output channel.

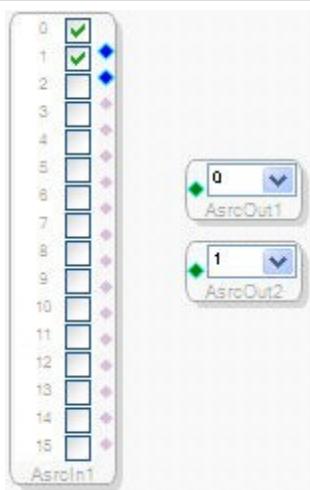
See theGPIO Example topic for sample schematics.

## ASRC Input/Output

The ASRC Input and Output blocks route signals between the schematic design and the hardware ASRCs (Asynchronous Sample Rate Converters).

Use the input block's check boxes to enable or disable particular inputs. Use the output block's drop-down list control to select from the available ASRCs.

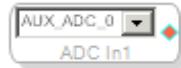
- Every enabled input must be connected to an output, else there will be errors on compilation.
- Observe that as you drag more ASRC output blocks into the schematic, the number of available outputs in the drop-down decreases because they can only be represented by one block at a time.
- To change the ASRC input's Sampling Rate , Right-click the block name and select Set Sampling Rate, which brings up the Sampling Rate window (default is 44.1 kHz).



**Note:** These blocks are only available for use with DSPs that have integrated ASRCs.

## Aux ADC Input

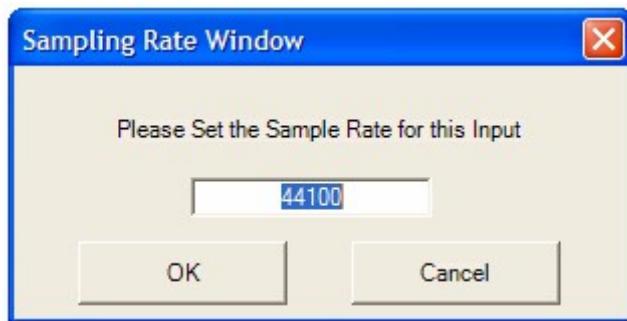
The Auxiliary ADC Input algorithm takes the digital signal from the auxiliary A/D (analog-to-digital converter) and makes it available in the design.



**Note:** This block is only available for use with DSPs that have auxiliary ADC (e.g. ADAU1701).

Use the block's drop-down list control to select from the available ADC inputs.

- Every enabled input must be connected to an output, else there will be errors on compilation.
- Observe that as you drag more ADC input blocks into the schematic, the number of auxiliary ADCs available in the drop-down decreases because they can only be represented by one block at a time.
- To change the Sampling Rate for an Aux ADC Input, Right-click the block name and select Set Sampling Rate, which brings up the Sampling Rate window (default is 44.1 kHz):



---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## General-Purpose Input

The General Purpose Input block allows signals from the hardware's GPIO input pins to be used in a schematic design. The orange output pin should typically be connected to a GPIO Conditioning block.

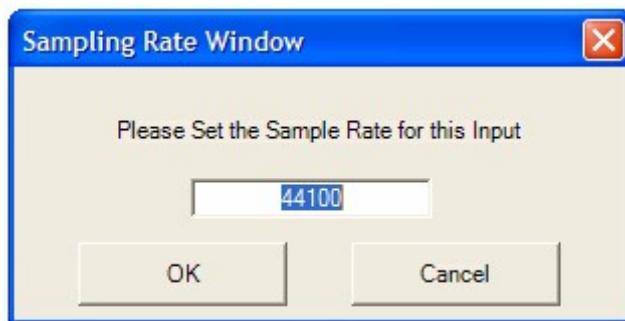


There are 12 selectable inputs to this GPIO block.

**Note:** The block is only available for DSPs that include GPIO. To configure the hardware's GPIO input and outputs settings, see the processor's Register Control Window.

Select the channel desired for sending your signal using the General Purpose Input drop-down control.

- Every enabled input must be connected to an output, else there will be errors on compilation.
- Observe that as you drag more output blocks to your schematic, your number of input channels available in the drop-down decreases.
- To change the Sampling Rate for the Input, Right-click the block name and select Set Sampling Rate, which brings up the Sampling Rate window (default is 44.1 kHz):



## General-Purpose Output

The General Purpose Output block routes a signal to the hardware's GPIO output. Each block is linked to one output channel.



**Note:** The block is only available for DSPs that include GPIO. To configure the hardware's GPIO input and outputs settings, see the processors Register Control Window.

Use the drop-down to select the hardware GPIO output that is assigned to the block.

Observe that as you drag more blocks to your schematic, your number of output channels available in the drop-down list decreases.

---

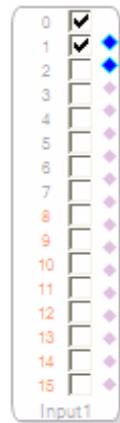
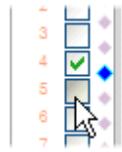
© 2006-2007 Analog Devices, Inc. All rights reserved.

## Input

The Input block receives input from the hardware's input pins and makes it available in the schematic design.

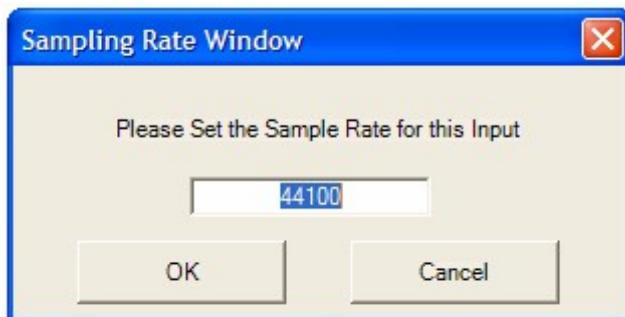
### To enable/disable an input:

Check the block for the input channel(s) you wish to enable. Un-check the box to disable an input channel. The pin will turn blue when an input is enabled and grey when disabled. The default block has two pins enabled, for stereo connection.



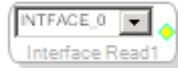
If you have multiple DSP processors in your design, specify which DSP is associated with the input block by right-clicking and selecting Add Algorithm > IC # > *DSP Type*.

- Every enabled input must be connected to an output, else there will be errors on compilation.
- Only a single input block can be associated with a processor. You will receive an error if you attempt to add multiple inputs to a schematic.
- To change the Input's Sampling Rate, Right-click the block name and select Set Sampling Rate, which will open the Sampling Rate window (default is 44.1 kHz):



## Interface Read

The Interface Read block takes value from one of eight interface registers and makes it available in the schematic design. The yellow pin can connect to GPIO Conditioning block's yellow input pins. It is especially useful for parts that self-boot and use external interface registers.



Select a particular interface from the drop-down.

- Every input must be connected to an output, else there will be errors on compilation.
- Observe that as you drag more output blocks to your schematic, your number of interfaces available decreases.
- To change the block's Sampling Rate, Right-click the block name and select Set Sampling Rate, which will open the Sampling Rate window (default is 44.1 kHz):

**Note:** This block is only available for DSPs with GPIOs or auxiliary ADC.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Interface Write

The Interface Write block sends signals from the schematic to an interface output register.



Each block is associated with only one interface register; select the appropriate one from the drop-down menu.

Observe that as you drag more blocks to your schematic, your number of interfaces available decreases.

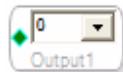
**Note:** This block is only available for DSPs with GPIOs or auxiliary ADC.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Output

The Output block routes signals to the hardware's physical outputs. Each block is linked to a single output channel.



Using the drop-down list, select the output channel to associate with a particular block.

Observe that as you drag more output blocks to your schematic, the number of output channels available in the drop-down list decreases because each output can only be associated with a single output block at a time.

If you have multiple DSP processors in a design, specify which processor to associate with the output block by right-clicking the block and selecting Add Algorithm > IC # > *DSP Type* from the menu.

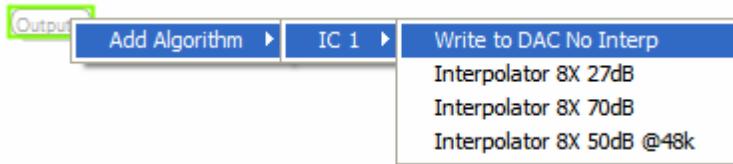
The hardware outputs for a particular processor are limited. While designing you can see the number of outputs that are still available from the HWOutputs item of the Resources window.

Resources	
IC 1	
AvailableDataM	2000
HWInputs	0
HWOutputs	11
MaxDataMemory	2000
MaxParameterMemory	1024
MaxProgramMemory	1024
Type	DSPSigma100

### AD1953 Output

The AD1953 has the DACs built into the DSP core, so you must choose an interpolation filter for the output. To use Output blocks with the AD1953 you will need to add an algorithm. As shown below, right-click the block to select which interpolating

filter you would like to implement in the DSP core.



Next, designate the channel on the block to be left, right, or sub. Only one block can write to the same output channel.

Following are descriptions of the interpolation filters:

#### **Write to DAC No Interp**

Writes to the DAC registers with no interpolation. Useful for subwoofer outputs, where it does not matter that distortion above 2kHz rises (output slewing) and HF response suffers (sinc(x) droop).

#### **Interpolator 8x 27dB**

27dB DAC interpolation filter. Response flat to 20kHz with a 48kHz sample rate. Stopband starts at 40kHz ??TKTK, so images of frequencies above 8 kHz will not be attenuated strongly. Lowest-quality, but very low MIPS: uses 37 instructions.

#### **Interpolator 8x 70dB**

Highest-quality interpolation filter: flat to 20kHz with a 44.1kHz sample rate. Uses 80 instructions.

#### **Interpolator 8x 50dB @48kHz**

50dB interpolation filter. Flat to 20kHz with a 48kHz sample rate; uses 53 instructions.

## SPDIF Input/Output

The SPDIF Input and Output blocks route signals between the schematic design and the hardware's SPDIF pins.



Use the input block's check boxes to enable or disable an input. Use the output block's drop-down list control to select from the available SPDIF outputs.

- Every enabled input must be connected to an output, else there will be errors on compilation.
- To change the SPDIF Input block's Sampling Rate , Right-click the block name and select Set Sampling Rate, which brings up the Sampling Rate window (default is 44.1 kHz).

**Note:** These blocks are only available for use with DSPs that have SPDIF I/O.

## Level Detectors / Lookup Tables

### Level Detectors / Lookup Tables

---

The Level Detectors / Lookup Tables library of the ToolBox gives you access to blocks which provide graphical representations of signal level.

The following are available:

- Index Lookup Table
- Level Detector Designer
- RMS Table
- Seven-Band Level Detector
- Single-Level Detector

The topic pages contain detailed information, but also look at the Example for sample schematics.

---

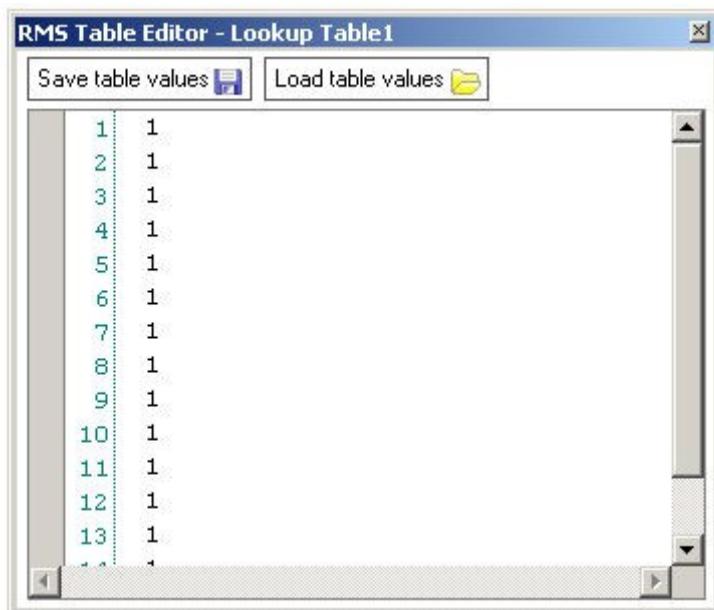
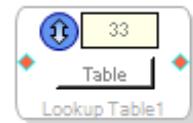
© 2006-2007 Analog Devices, Inc. All rights reserved.

## Index Lookup Table

The Index Lookup Table block lets you access values stored in a lookup table (LUT). They're accessible through a table index.

To use this block:

1. Drag it into the workspace and click Table to open the RMS Table Editor window:



There are two ways to enter values:

- Enter them directly and use Save table value.
- Use Load table value to load data from a file.

The first option lets you enter maximum value(s).

### Description

For this algorithm, the input is the index value into the LUT. It should be a number in 28.0 format, beginning at the LSB of a 28-bit word. Zero (0) will index the first value in the table. This algorithm checks if the index at the input is greater than the maximum index and, if so, indexes the last value in the table.

Similarly, if the index is below zero, it will select the first value in the table.

The Example further illustrates use of this block.

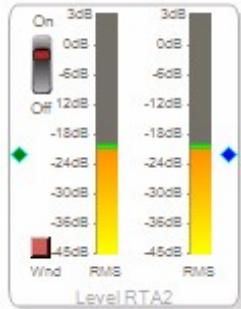
---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Level Detector Designer

The Level Detector blocks calculate the input signal level, reading directly from the hardware in real time, and display the level graphically in meter displays. The Level Detector Designer block lets you define your own frequency bands and time constants for the display.

The level detector performs analysis only and does not modify the input signal. The signal at the output pin is identical to the input.



Use the **On / Off** button to enable or disable the display. The level detector will not function until the schematic design has been compiled and downloaded to the hardware and a USB communication channel is properly configured.

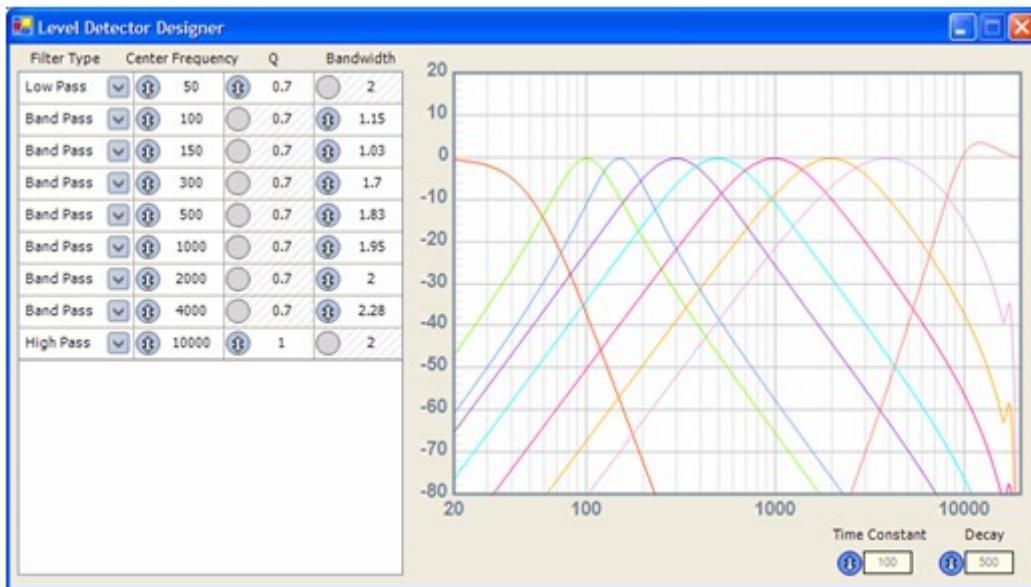
Note: The refresh rate of the display is approximately 10 Hz, while the green cross-bars track the maximum rms value with a slight delay. The display's performance is limited by your PC system and USB communication resources. Using multiple level detectors may degrade the responsiveness.

To use this block:

1. Drag it into the workspace.
2. Right-click it and select Add Algorithm > IC #.
3. From the list, select the algorithm that meets your needs:
  - Passthrough
  - RTA to output
  - Passthrough / minimal reading, single-precision
  - Passthrough / minimal reading, double-precision
4. Click the red **Wnd** button to bring up the window (below) for entering your design parameters.

After selecting your default algorithm, there's an option to Grow it. The default block contains only one color bar, but when you grow your algorithm you can account for multiple frequency bands. The figure above right shows the Passthrough algorithm grown by 1.

To make the most of your level detector, it's important to understand the parameters in the designer window. Shown below is that window for a passthrough algorithm grown by 8. There are a 9 bands total, each with the option of choosing Filter type, Center Frequency, and Q or Bandwidth depending on filter type. (There also is an overall Time Constant and Decay value that can be set in dB/s to control the refresh time of the color display.)



#### Parameters:

**Filter Type:** Lowpass, highpass, and bandpass filters can be set for the individual bands of the color display.

**Center Frequency:** Specify the center / cutoff frequency of the filter.

**Q:** Available with LP and HP filters, Q determines the steepness of the filter skirts and its -3dB points.

**Bandwidth:** Available with the BP filter, Bandwidth determines the range of frequencies your design will affect.

**Time Constant:** This value, in dB/s, designates the averaging time of the detector: how rapidly it assesses and responds to signal level changes.

**Decay:** Another type of time constant, Decay designates the rate at which signal returns to a lower detected level. Decay is responsible for releasing the signal at the given rate.

#### Algorithms:

**Passthrough:** This algorithm receives a 5.19 number (5 bytes to represent the integer portion of the value, 19 bytes for the decimal portion) during readback and converts it to a decimal value represented in decibels. This is what is seen on the display bar. Because this is a passthrough system, the output is the same signal being sent to the block.

**RTA to output:** The RTA-to-output algorithm outputs the rms level value through the output pin. This pin is red because no actual audio is being sent out, just the level values for each frequency band. Observe that when you grow this algorithm, each pin corresponds to a frequency band.

**Passthrough / minimal reading, single- and double-precision:** This functions in a manner similar to Passthrough but saves on communication bandwidth at the expense of losing precision. It uses only 1 byte for the readback value, then converts the value to a decimal which is represented in decibels.

Normally you should use double-precision: 56 bits for each calculation and 10 instructions per filter. Single-precision uses 28 bits for calculations and 6 instructions per filter, saving 3

data ram spaces over the double-precision algorithm. Note that single-precision should not be employed either for frequencies below 1/10 the sampling rate or for high-Q filters.

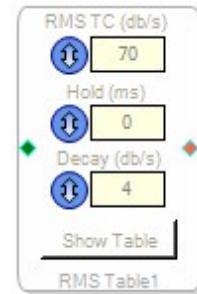
---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## RMS Table

The RMS Table takes an input signal and outputs the interpolated mapped value of the signal, relative to the table, depending on the calculated rms input. The block uses rms average values and maps them to the user-selectable table values, employing linear interpolation in between table values.

Three parameters are available through the edit boxes / control arrows:



**RMS TC  
(dB/s)**



Controls the time constant (TC) used for calculating the rms value. This determines how rapidly the gain will adapt to changes in the input level; this is also called the attack time.

**Hold (ms)**



Controls the time the rms average holds its current output setting before it detects a lower value and starts ramping down.

**Decay  
(dB/s)**



Controls the rate at which the output signal returns to a lower detected level. Decay is responsible for releasing the signal at a given rate. This is also referred to as the release time.

Press Show Table to view and edit the table values. The calculated signal RMS values are mapped to an index in the table. The maximum table resolution is approximately 96 dB with indices spaced 3 dB apart (as shown in the example below).

Table Index	Detected Value (dB)	Mapped Values (user-specified)
1	-90	0.01
2	-87	0.01
3	-84	0.01
4	-81	0.02
5	-78	0.02
6	-75	0.02
7	-72	0.02
8	-69	0.03
9	-66	0.03
10	-63	0.04
11	-60	0.04
12	-57	0.05
13	-54	0.06
14	-51	0.07
15	-48	0.08
16	-45	0.09
17	-42	0.10
18	-39	0.12
19	-36	0.13
20	-33	0.15

21	-30	0.18
22	-27	0.21
23	-24	0.24
24	-21	0.28
25	-18	0.32
26	-15	0.37
27	-12	0.42
28	-9	0.49
29	-6	0.57
30	-3	0.65
31	0	0.75
32	3	0.87
33	6	1.01

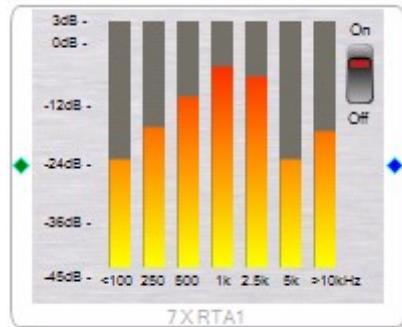
---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Seven Band Level Detector

The Seven-Band Level Detector block calculates the input signal level, reading directly from the hardware in real time, and displays the level graphically in seven frequency band meter displays:

- <100Hz
- 250Hz
- 500Hz
- 1kHz
- 2.5kHz
- 5kHz
- >10kHz



The level detector performs analysis only and does not modify the input signal. The signal at the output pin is identical to the input.

Use the **On / Off** button to enable or disable the display. The level detector will not function until the schematic design has been compiled and downloaded to the hardware and a USB communication channel is properly configured.

Note: The refresh rate of the display is approximately 10 Hz. The display's performance is limited by your PC system and USB bandwidth. Using multiple level detectors may degrade the responsiveness.

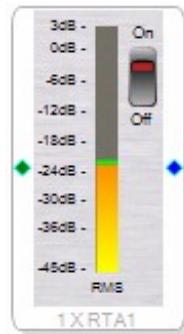
## Single Level Detector

The Level Detector blocks calculate the input signal level, reading directly from the hardware in real time, and display the level graphically in meter displays. The Single-Level Detector calculates and displays the rms level of the signal, shown in dB.

The level detector performs analysis only and does not modify the input signal. The signal at the output pin is identical to the input.

Use the **On / Off** button to enable or disable the display. The level detector will not function until the schematic design has been compiled and downloaded to the hardware and a USB communication channel is properly configured.

The refresh rate of the display is approximately 10 Hz, while the green cross-bar tracks the maximum rms value with a slight delay. Note that the display's performance is limited by your PC system and USB communication resources. Using multiple level detectors may degrade the responsiveness.



---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Licensed Algorithms

### Licensed Algorithms

---

The Toolbox's Licensed Algorithms library lets you access proprietary licensed blocks (including some developed by Analog Devices) for use in your designs.

The following Algorithms are available:

#### Analog Devices

- ADI Surround
- ADI Virtual
- External LED Array Driver
- Subharmonic Synth (general)
- Subharmonic Synth (LF)

#### BBE Sound - [www.bbesound.com](http://www.bbesound.com)

- BBE
- BBE MP
- BBE ViVA

#### Dolby Laboratories - [www.dolby.com](http://www.dolby.com)

- Audistry
- Dolby Automotive Entertainment Program (DAEP)
- Dolby Headphone
- Dolby Pro Logic II
- Dolby Virtual Speaker

#### SRS Labs - [www.srslabs.com](http://www.srslabs.com)

- Circle Surround Automotive Decoder (CS Auto)
- Circle Surround II Decoder
- TruBass
- TruSurround XT
- WOW
- WOW HD

**Note:** None of these algorithm blocks are included with the SigmaStudio application installation. Please contact Analog Devices for algorithm evaluation and licensing information, [sigmadsp@analog.com](mailto:sigmadsp@analog.com).

## Analog Devices

### ADI Surround

**Note:** This block is not included in the SigmaStudio installation. Please contact Analog Devices for evaluation and licensing information, [sigmadsp@analog.com](mailto:sigmadsp@analog.com).

ADI Surround uses matrix-surround decoding techniques: deriving five channels of output from a two-channel input. This process entails using a sum channel (the combination of the left and right channels, L+R), which contains frontal information, and a difference channel (the difference between the left and right channels, L-R / R-L), which contains ambience information.

Depending on the blending and distribution of the difference channel with the sum channel, we can derive the left, right, center, and side or rear surround channels. The L and R front speakers carry music, frontal sound effects, and directional dialog; the center speaker carries the rest (meaning most) of the dialog, and the surround speakers (ideally placed to the side of and slightly above the listeners) provide ambience and surround effects.

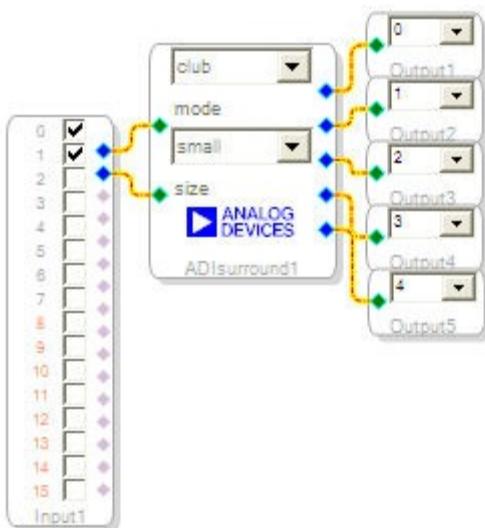
ADI Surround has presets for the following modes:

- Club
- Music
- Stadium
- Cinema
- Hall
- Rock
- Jazz
- User1
- User2
- User3

It also features two room sizes:

- Size1 = small
- Size2 = large

The schematic shown below features an input block (two-channel), ADI Surround (configuration = Club, size = small) and output blocks.



---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## ADI Virtual

**Note:** This block is not included in the SigmaStudio installation. Please contact Analog Devices for evaluation and licensing information, [sigmadsp@analog.com](mailto:sigmadsp@analog.com).

ADI Virtual enhances the stereo soundfield, creating a more spacious listening environment in several styles. It uses sophisticated signal processing to synthesize virtual loudspeakers and give the sensation of being inside the stereo tracks and/or making the music surround you. Built-in spectral shaping filters fine-tune frequency and phase responses. 5.1-channel surround audio (left, right, center, subwoofer, left and right surround channels) can be converted to two-channel format, binaural for headphones and stereo specially treated for loudspeaker playback.

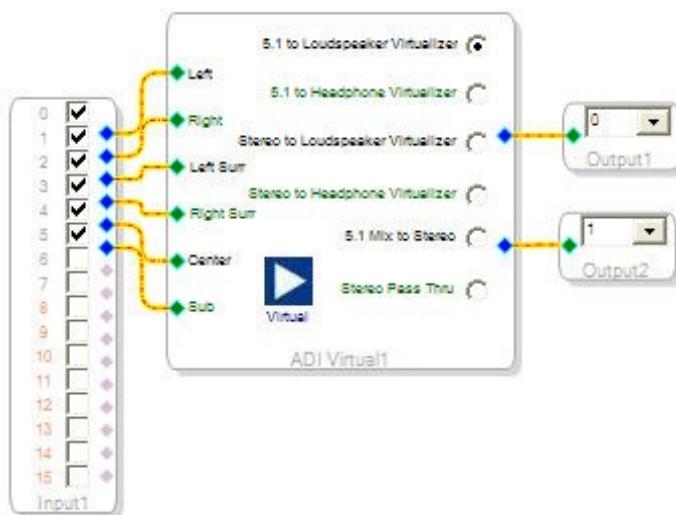
ADI Virtual is intended for lightweight implementation on embedded DSP.

Following are the processing configurations available:

- 5.1 to loudspeakers virtualizer
- 5.1 to headphones virtualizer
- Stereo to loudspeaker virtualizer
- Stereo to headphones virtualizer
- 5.1 mix to stereo
- Stereo passthrough TTKT

Click the appropriate choice to select the input configuration desired.

The schematic shown below features 6-channel input block, ADI Virtual block (5.1 to Loudspeakers Virtualizer selected), and two output blocks.



© 2006-2007 Analog Devices, Inc. All rights reserved.

## External LED Array Driver

### External LED Array Driver

**Note:** This block is not included in the SigmaStudio installation. Please contact Analog Devices, [sigmadsp@analog.com](mailto:sigmadsp@analog.com), for more information.

The ADAU1701 block drives an external LED level meter using three of its multipurpose pins. The external circuit requires only an LED driver IC and the proper number of LEDs for the level meter.

ADI's A6276EA serial-input LED driver can drive two 8-LED meters. The driver IC requires three control signals (clock, latch, data) to drive the LED meters.

These three signals are obtained from the External LED Array Driver block, which takes the stereo signal as input and outputs the clock, latch, and data signals in its first, second, and third blue output pins. Clicking Table brings up the LEDTable window (right), where the linear values that turn on each LED can be set. (To the right of these linear values are their decibel equivalents.)

For more details, see the Example.

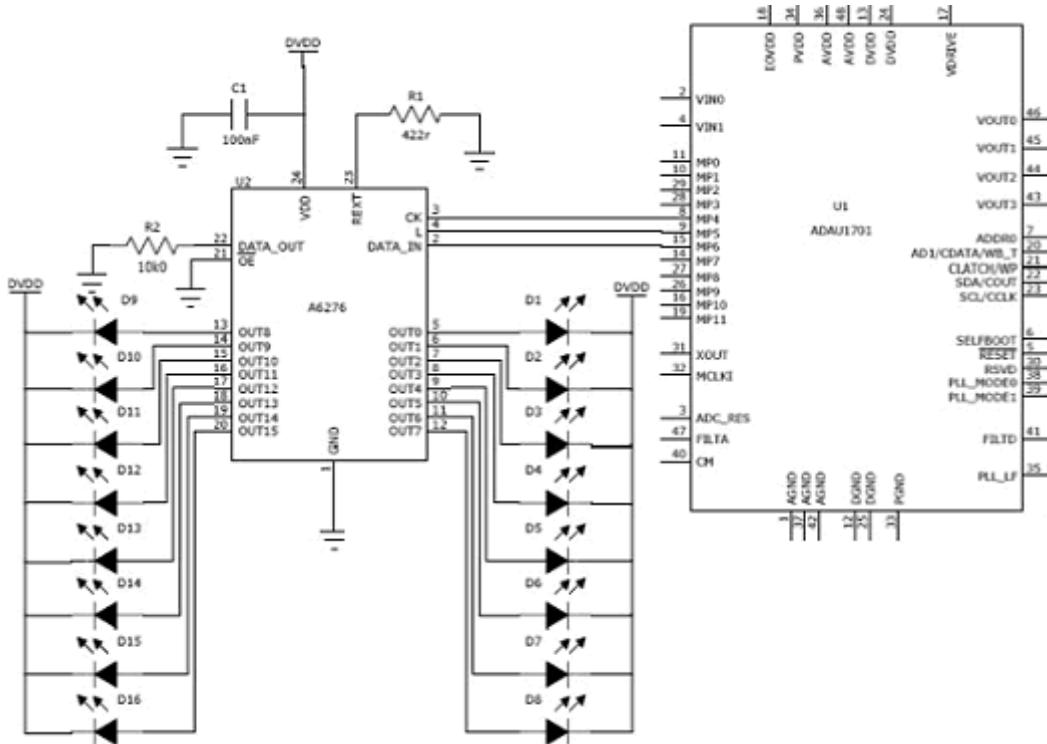
**Note:** ADI's LED driver add-on board can be easily connected to the breadboard section of the ADAU1701 GPIO interface board; click [here](#) for details.



LEDTable		
LED 0	0.75	-2.5 dB
LED 1	0.8	-1.94 dB
LED 2	0.825	-1.67 dB
LED 3	0.85	-1.41 dB
LED 4	0.875	-1.16 dB
LED 5	0.9	-0.92 dB
LED 6	0.925	-0.68 dB
LED 7	0.95	-0.46 dB

# Using Allegro MicroSystems™ A6276 with ADAU1701

The circuit below shows connection of the Allegro MicroSystems A6276 to the ADAU1701 to drive an LED-display. The ADAU1701's multipurpose pins are set to emulate the SPI-type data interface that the A6276 is expecting. The serial data stream is decoded in the A6276 and output on the connected LEDs. Resistor R1 is in place to set the part's output load current per bit. A  $422\text{-}\Omega$  resistor provides approximately 40 mA/bit, which is certainly enough to drive an LED. In this design, D8 and D9 are the highest-level output indicators, while D1 and D16 indicate the lowest signal levels. DVDD for the A6276 is 5V.



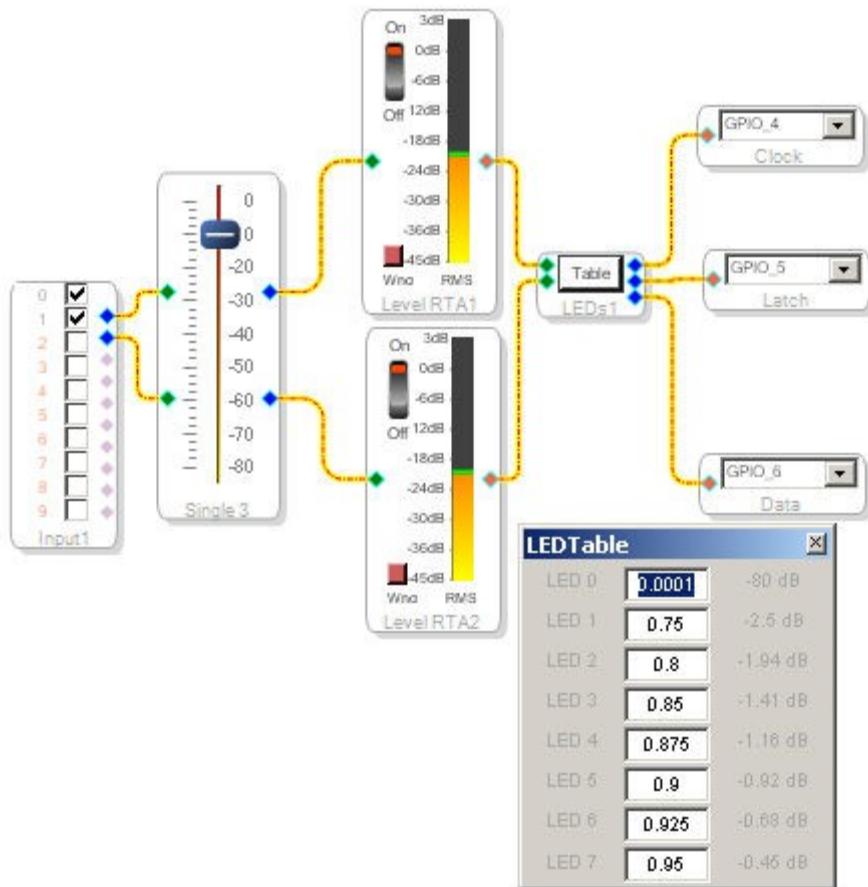
© 2006-2007 Analog Devices, Inc. All rights reserved.

## External LED Array Driver Example

This schematic uses Single Volume Control, Level Detector Designer, External LED Array Driver, Input, and General Purpose output blocks.

Here, after the volume control, the audio in the project is fed into a level detector block, which outputs the rms values of the incoming signal levels. This control signal (rms signal) is sent to the LED Array Driver block, which outputs three signals: clock, latch, and data. These three signals are implemented with the GPIO block, and in this case GPIO 4, 5, and 6 are selected (as shown).

Clicking Table brings up the LEDTable window, where the linear values that turn on each LED can be set. (To the right are the equivalent decibel values.)



## Subharmonic Synth (general)

**Note:** This block is not included in the SigmaStudio installation. Please contact Analog Devices for evaluation and licensing information, [sigmadsp@analog.com](mailto:sigmadsp@analog.com).

The Subharmonic Synth (general) cell intelligently generates new bass sounds at half the frequency below the fundamentals. For emphasis, it also partly suppresses the fundamental component (using appropriate filtration).



---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Subharmonic Synth (LF)

**Note:** This block is not included in the SigmaStudio installation. Please contact Analog Devices for evaluation and licensing information, [sigmadsp@analog.com](mailto:sigmadsp@analog.com).

The Subharmonic Synth (low freq) cell is a synthesizer processor which produces and adds deep, typically near-infrasonic bass to the original audio signal.



Unlike an equalizer, which indiscriminately boosts bass that may be loud enough already, this cell automatically and intelligently creates new harmonics at half the frequency of the fundamentals present in the signal. The deep bass generated by this cell is mixed with the regular audio to produce clear but significantly bass-enhanced sound.

**Max Harmonic.** Enter the maximum desired frequency of the subharmonic in the text field (or click and hold the up/down arrow), 40-120 Hz.

## BBE

### BBE

**Note:** This block is not included in the SigmaStudio installation. Please contact Analog Devices for evaluation and licensing information, [sigmadsp@analog.com](mailto:sigmadsp@analog.com).

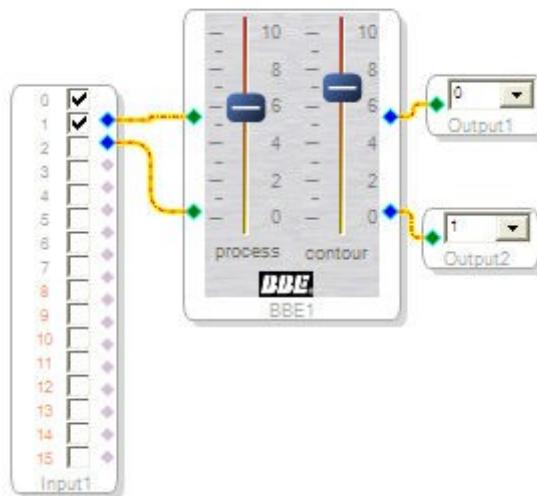
BBE improves overall audio clarity, giving more brightness with apparent increased depth, detail and definition. BBE has two sliders, for LF and HF enhancement, called Process and Contour respectively.

*"BBE High Definition Sound is the core sound enhancement technology licensed by BBE Sound and featured in the BBE Sonic Maximizer range of professional audio signal processors."*

For detailed information about this algorithm, please visit [www.bbesound.com](http://www.bbesound.com).

**Example:**

The simple example schematic below shows input block, BBE block, and stereo output blocks.



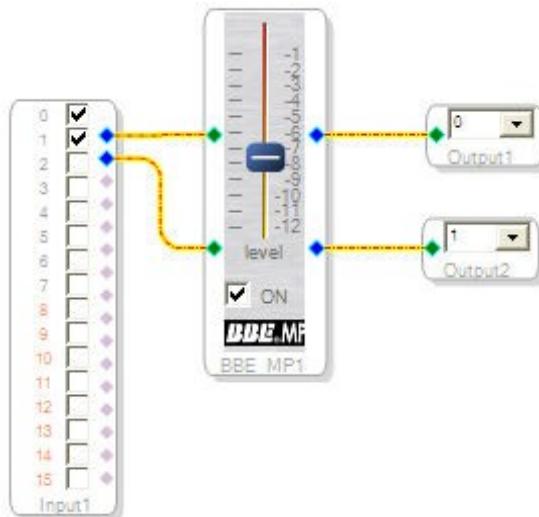
## BBE MP

**Note:** This block is not included in the SigmaStudio installation. Please contact Analog Devices for evaluation and licensing information, sigmadsp@analog.com.

BBE's proprietary MP process (Minimized Polynomial Nonlinear Saturation) improves the sound of data-reduced audio, for example MP3 files. It does this by restoring and enhancing high frequency harmonics lost in the data-reduction process.

For details about this algorithm, please visit [www.bbesound.com](http://www.bbesound.com).

A simple schematic, with input block, BBE MP block and output blocks, is shown below.



---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## BBE ViVA

**Note:** This block is not included in the SigmaStudio installation. Please contact Analog Devices for evaluation and licensing information, [sigmadsp@analog.com](mailto:sigmadsp@analog.com).

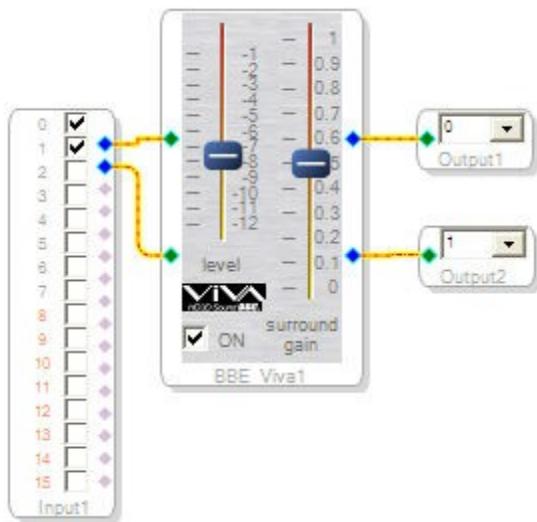
BBE ViVA creates a 3D enhancement effect for stereo speakers.

*"ViVA creates an authentic and exciting 3D sound effect from stereo speakers while preserving the clarity, definition, presence and texture for which BBE High Definition Sound is noted."*

For detailed information on this algorithm, please visit [www.bbesound.com](http://www.bbesound.com) .

**Example:**

A simple schematic, with input, BBE ViVA block, and output blocks, is shown below.



---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Dolby

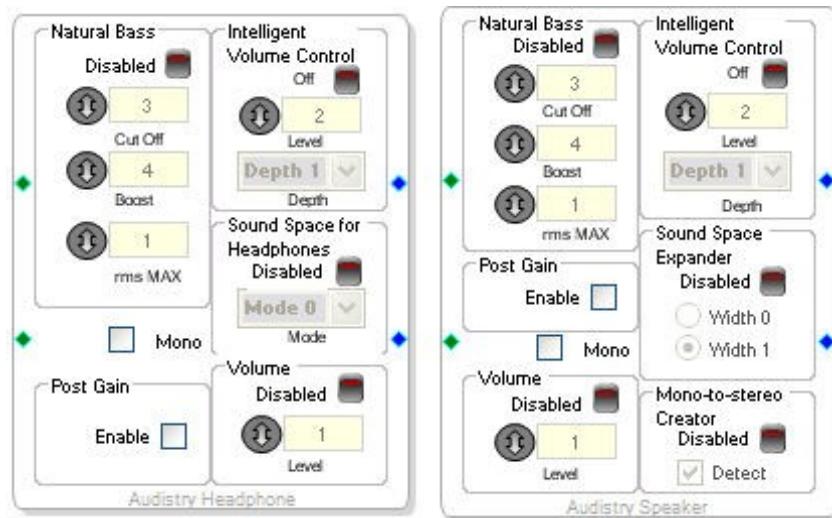
### Audistry™

**Note:** This block is not included in the SigmaStudio installation. Please contact Analog Devices for evaluation and licensing information, [sigmadsp@analog.com](mailto:sigmadsp@analog.com).

Audistry™ by Dolby provides a set of tools to enhance the listening experience. There are two Audistry block's available: one designed for headphone processing and the other optimized for loudspeakers. It includes the following 5 tools which are "designed to work together in any combination to deliver an exceptional listening experience".

- Sound Space Expander
- Sound Space for Headphones
- Intelligent Volume Control
- Natural Bass
- Mono-to-stereo Creator

For detailed information about this algorithm, please visit [www.audistry.com](http://www.audistry.com).

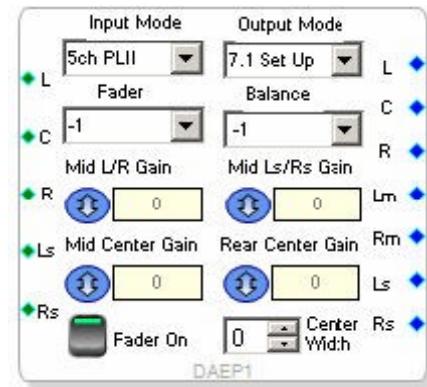


## Dolby Automotive Entertainment Program (DAEP)

**Note:** This block is not included in the SigmaStudio installation. Please contact Analog Devices for evaluation and licensing information, [sigmadsp@analog.com](mailto:sigmadsp@analog.com).

The Dolby Automotive Entertainment Program (DAEP) bundles selected core Dolby technologies such as Dolby Prologic II with newly developed post-processing technologies, to provide an immersive surround sound entertainment experience for the automotive environment.

For more information on this algorithm, please contact Dolby Laboratories, [www.dolby.com](http://www.dolby.com).



---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Dolby Headphone

**Note:** This block is not included in the SigmaStudio installation. Please contact Analog Devices for evaluation and licensing information, sigmadsp@analog.com.

Dolby Headphone's signal processing permits headphone listeners to get a 7.1-channel surround-sound experience from a stereo source. It lets users wear any set of headphones and listen to music, watch movies, or play video games with the surround effects of a 7.1-channel soundtrack.

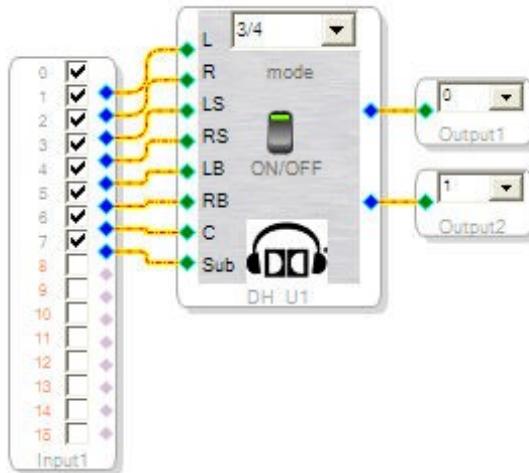
Dolby Headphone has settings for the following configurations:

- Center only (C only)
- Stereo (L & R)
- LCR
- Mono surround (L,C,R, LB = RB)
- 5-channel (L,C,R, LB, RB)
- 6-channel (L,C,R, LS, RS, LB = RB)
- 7-channel (L,C,R, LS, RS, LB, RB)

For detailed information about this algorithm, please visit [www.dolby.com](http://www.dolby.com).

### Example:

The schematic, with input block, DH block, and the output blocks is shown below.



## Dolby Pro Logic II

**Note:** This block is not included in the SigmaStudio installation. Please contact Analog Devices for evaluation and licensing information, [sigmadsp@analog.com](mailto:sigmadsp@analog.com).

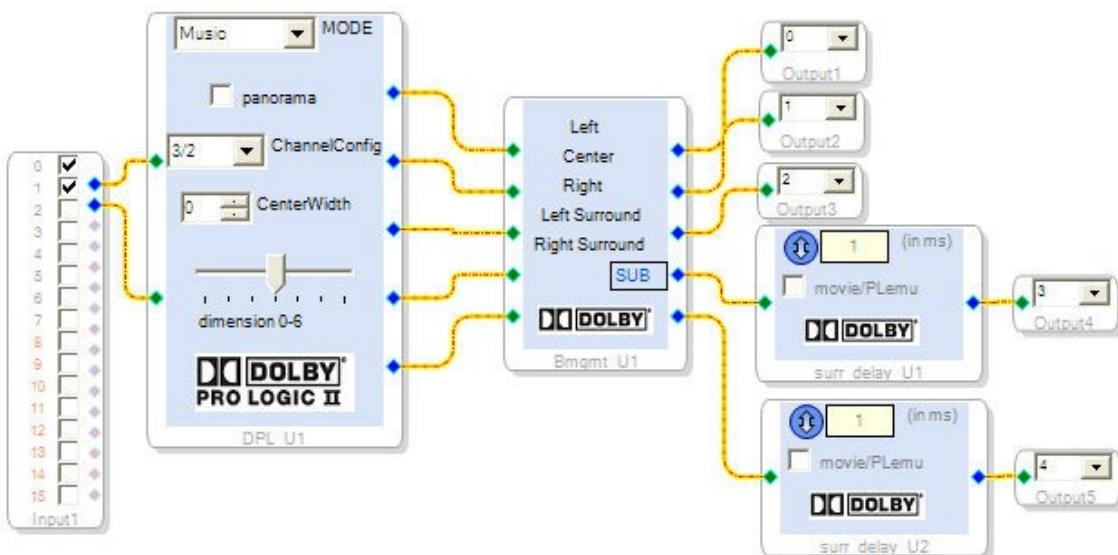
Dolby Pro Logic II processes two-channel stereo movie and music audio and produces five surround sound output channels.

It detects the directional cues that occur naturally in the stereo content, and uses them to create a five-channel surround sound playback experience.

For detailed information on this algorithm, please visit [www.dolby.com](http://www.dolby.com).

### Example:

A schematic, with input block, DPL II block, and the output blocks, is shown below.



© 2006-2007 Analog Devices, Inc. All rights reserved.

## Dolby Virtual Speaker

**Note:** This block is not included in the SigmaStudio installation. Please contact Analog Devices for evaluation and licensing information, sigmadsp@analog.com.

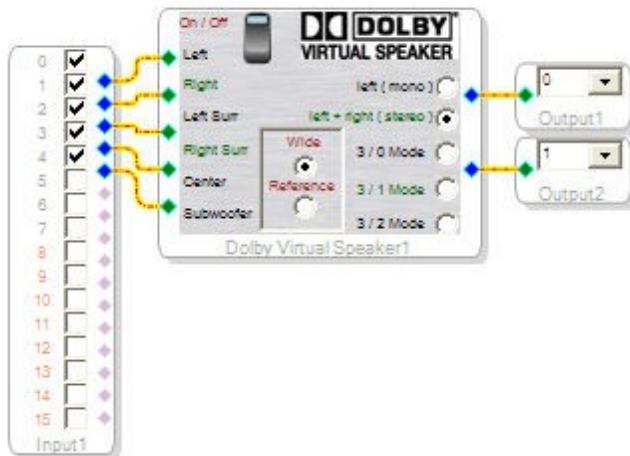
Dolby Virtual Speaker processing simulates a 5.1-speaker surround sound listening environment from a two-speaker configuration. It is an ideal technology for mini stereos, PCs, and similar audio situations. DVS employs crosstalk cancellation processing to keep the surround cues for each ear from being canceled by cues from the other speaker.

For details, please visit [www.dolby.com](http://www.dolby.com).

The Dolby Virtual Speaker features presets ??tktk for the following output modes:

- Left (mono)
- Left + Right (L, R stereo)
- 3/0
- 3/1
- 3/2

The schematic with input cell, DVS cell and the output cells is shown below.



## SRS

### Circle Surround Automotive Decoder (CS Auto)

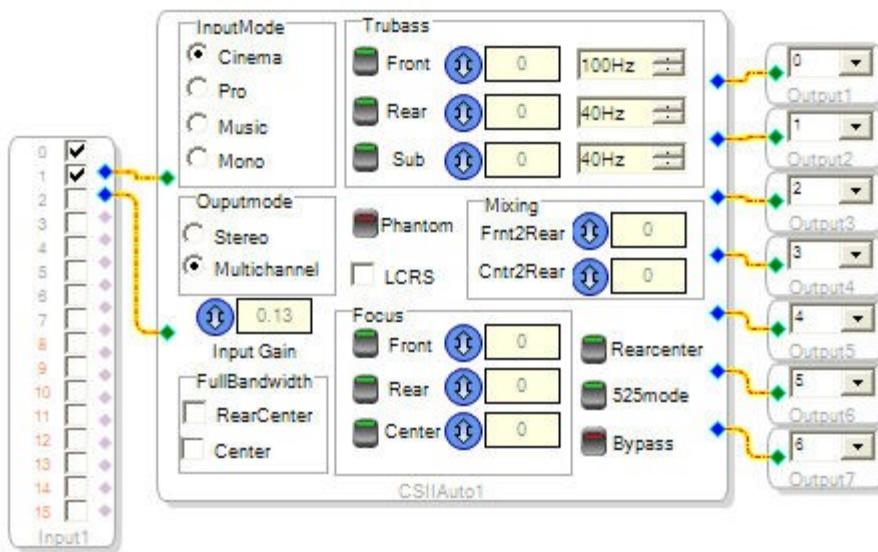
**Note:** This block is not included in the SigmaStudio installation. Please contact Analog Devices for evaluation and licensing information, [sigmadsp@analog.com](mailto:sigmadsp@analog.com).

Circle Surround Automotive (CS Auto) is a 6.1-channel surround-sound decoding system. The decoder features a FOCUS module, which repositions the sonic image from non-optimally located speakers, and a TruBass module, which enhances the bass of systems without subwoofer(s).

For detailed information about this block, please visit [www.srslabs.com](http://www.srslabs.com).

**Example:**

A schematic with input block, CS Auto block, and output blocks is shown here:



© 2006-2007 Analog Devices, Inc. All rights reserved.

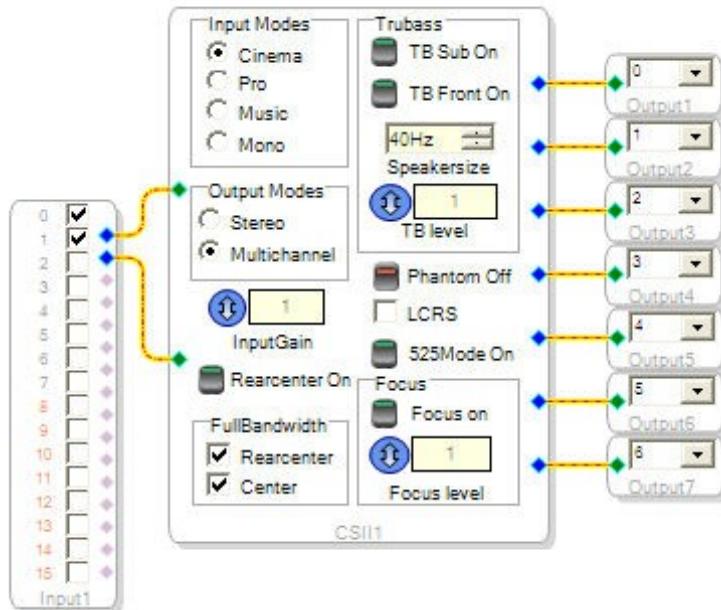
## Circle Surround II Decoder

**Note:** This block is not included in the SigmaStudio installation. Please contact Analog Devices for evaluation and licensing information, sigmadsp@analog.com.

The CSII decoder splits a stereo signal into 6 channels of surround audio plus an LFE / subwoofer signal. The CS II includes a FOCUS module, which can electronically reposition a sound image propagating from a non-optimally located speaker, and a TruBass module, which is used to enhance the bass for small systems without a subwoofer or other extra LF components.

For detailed information about this algorithm, please visit the website [www.srslabs.com](http://www.srslabs.com).

A schematic, with input cell, CSII cell, and the output cells is shown below:

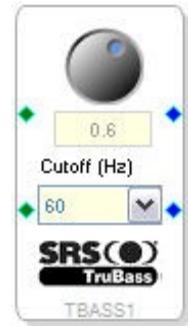


## TruBass

**Note:** This block is not included in the SigmaStudio installation. Please contact Analog Devices for evaluation and licensing information, [sigmadsp@analog.com](mailto:sigmadsp@analog.com).

SRS TruBass provides psychoacoustic bass (Low Frequency) enhancement for headphones and systems with small loudspeakers. "It uses psychoacoustics and speaker driver physics to increase the perception of bass while decreasing demands on the driver."

For detailed information about this algorithm, please visit the website [www.srslabs.com](http://www.srslabs.com).



---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## TruSurround XT

**Note:** This block is not included in the SigmaStudio installation. Please contact Analog Devices for evaluation and licensing information, [sigmadsp@analog.com](mailto:sigmadsp@analog.com).

SRS TruSurround XT provides virtual surround sound for stereo reproduction systems. "It delivers a premium virtual surround sound experience from any multichannel source over just two speakers or headphones".

For detailed information about this algorithm, please visit the website [www.srslabs.com](http://www.srslabs.com).



---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## WOW®

**Note:** This block is not included in the SigmaStudio installation. Please contact Analog Devices for evaluation and licensing information, [sigmadsp@analog.com](mailto:sigmadsp@analog.com).

SRS WOW® improves the dynamic audio performance of compressed audio, expands the size of the audio image and improves the perception of low frequency sound.

For detailed information about this algorithm, please visit the website [www.srslabs.com](http://www.srslabs.com).



---

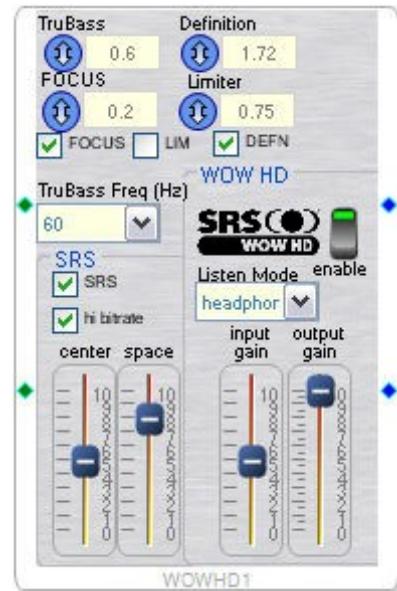
© 2006-2007 Analog Devices, Inc. All rights reserved.

## WOW HD

**Note:** This block is not included in the SigmaStudio installation. Please contact Analog Devices for evaluation and licensing information, [sigmadsp@analog.com](mailto:sigmadsp@analog.com).

SRS WOW HD™ is an advanced version of its SRS WOW technology which provides enhanced width, improved bass and definition controls.

For detailed information about this algorithm, please visit the website [www.srslabs.com](http://www.srslabs.com).



---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Mixers / Splitters

### Mixers/Splitters

---

The Mixers/Splitters library of the ToolBox gives access to blocks that let you combine or split signals in various ways. The following block's are available:

- Cross-Mixer (2 inputs)
- Cross-Mixer (3 inputs)
- Cross-Mixer (8 inputs)
- Multiple-Control Mixer
- Multiple-Control Splitter
- Signal Merger
- Single-Control Mixer
- Single-Control Splitter
- Stereo Mixer

The pages within this topic book contain specific information about these blocks. Also take a look at Mixers/Splitters Example to see a sample schematic.

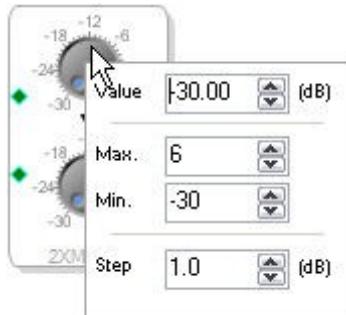
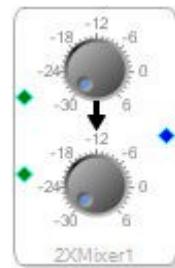
---

© 2006-2007 Analog Devices, Inc. All rights reserved.

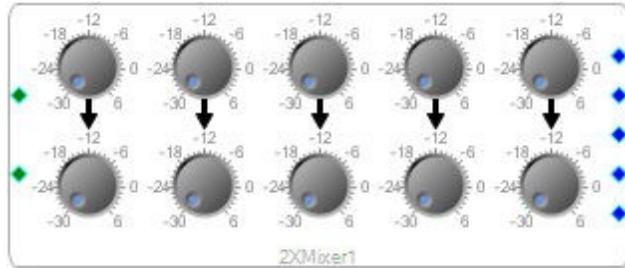
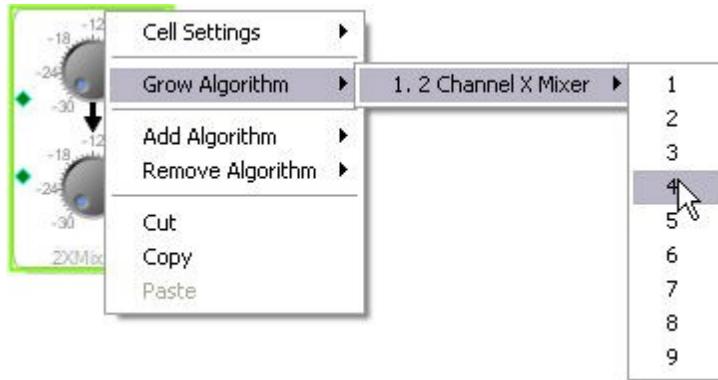
## Cross-Mixer (2 inputs)

The Cross-Mixer (2 inputs) allows two input signals to be mixed, with variable gain, down to one output signal.

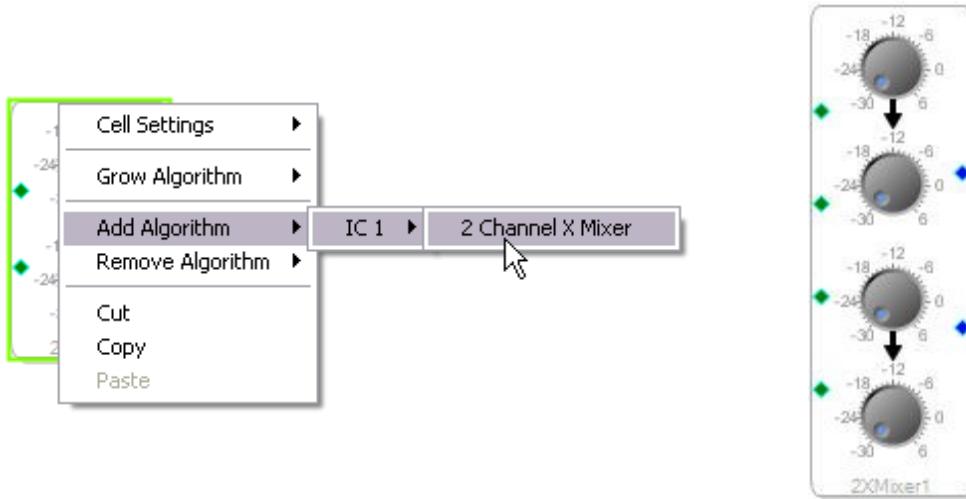
There are two control knobs for setting the gain, one knob for each input pin. To set the gain, click the knob with the left mouse button and drag while holding the button down. Each knob's range, value, and step size can be customized by **right-clicking** on the knob control.



To add additional mixer outputs, you can perform Grow Algorithm on this block (**right-click** and select **Grow Algorithm** from the menu), increasing the number of output pins, as shown below. This example features growth by 4, resulting in five total outputs. (The maximum is +9.) This allows you to create different mixes from the same inputs.

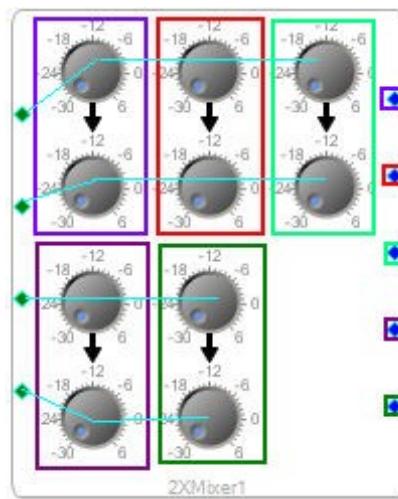


You can also add additional mixer algorithms to the block by **right-clicking** and selecting **Add Algorithm** from the menu. Adding an algorithm creates another set of input/output pins and control knobs. Each block algorithm is independent. See the Add Algorithm topic for more information.



This second mixer control is entirely separate even though it appears on the same block. You can add additional algorithms; the input to output ratio will always be 2:1.

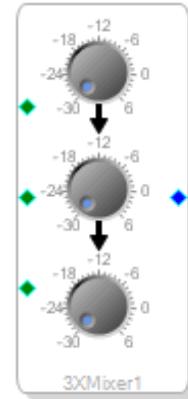
It is also possible to perform Grow and Add Algorithm on the same block. In this situation it is important to keep track of input and output pin relationships. Below is a design with a mixer block containing two algorithms, the first grown by 2, the second grown by 1. The colored boxes indicate the mixer configuration and the corresponding input/output pins.



© 2006-2007 Analog Devices, Inc. All rights reserved.

## Cross-Mixer (3 inputs)

The Cross Mixer (3 inputs) block mixes three input signals to a single output, allowing each input's gain to be independently controlled. You also can Grow and/or Add Algorithms to the this block; refer to the Cross-Mixer (2 inputs) for more information.



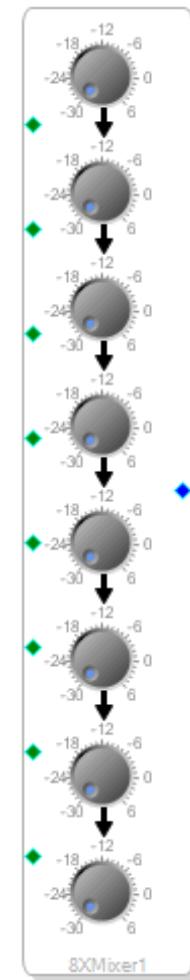
---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Cross-Mixer (8 inputs)

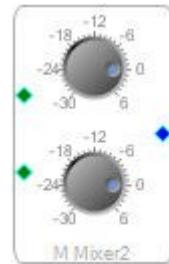
This block mixes 8 input signals to a single output. Each input has an independent and customizable gain control knob.

You also can Grow and/or Add Algorithms to the this block; refer to the Cross-Mixer (2 inputs) for more information.



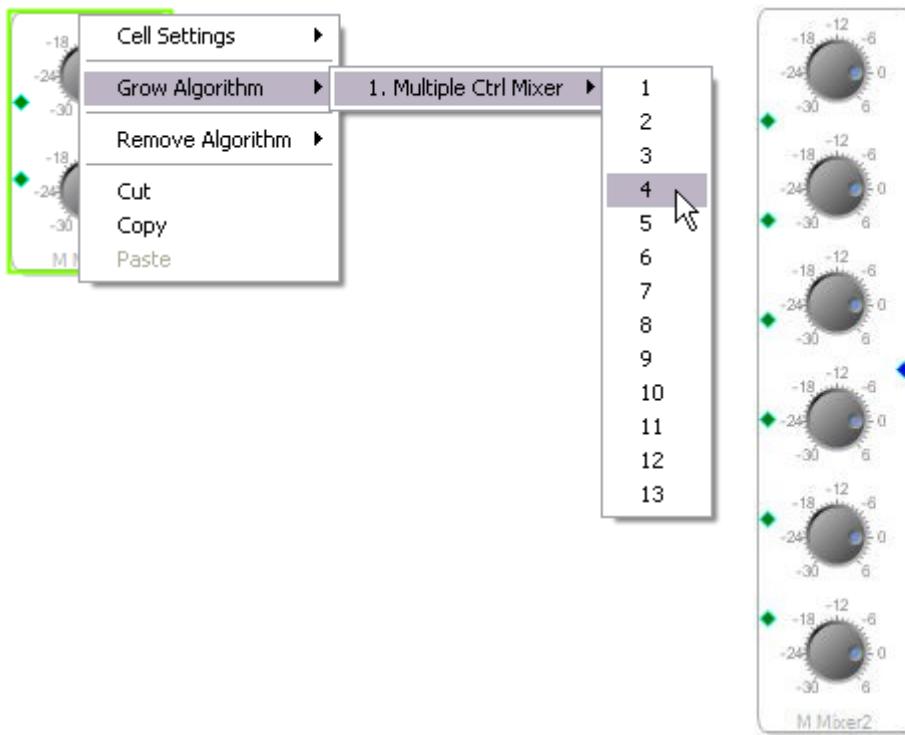
## Multiple-Control Mixer

The Multiple Control Mixer allows a set of input signals to be mixed, with variable gain, into a single output signal. The Multiple-Control Mixer is similar to the Cross-Mixer blocks, however this block has a different Grow behavior. Growing this block will increase the number of inputs while growing a cross mixer block increases the number of outputs.



### To add additional inputs:

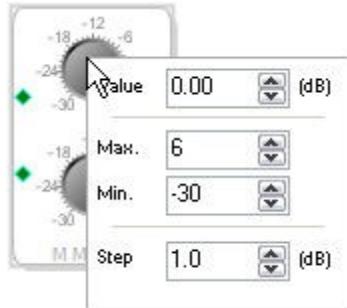
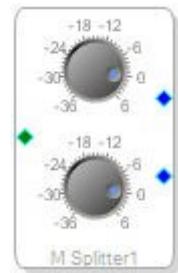
Right-click, select **Grow Algorithm** from the menu, and choose the number of input pins to add. By default this block will have two input pins.



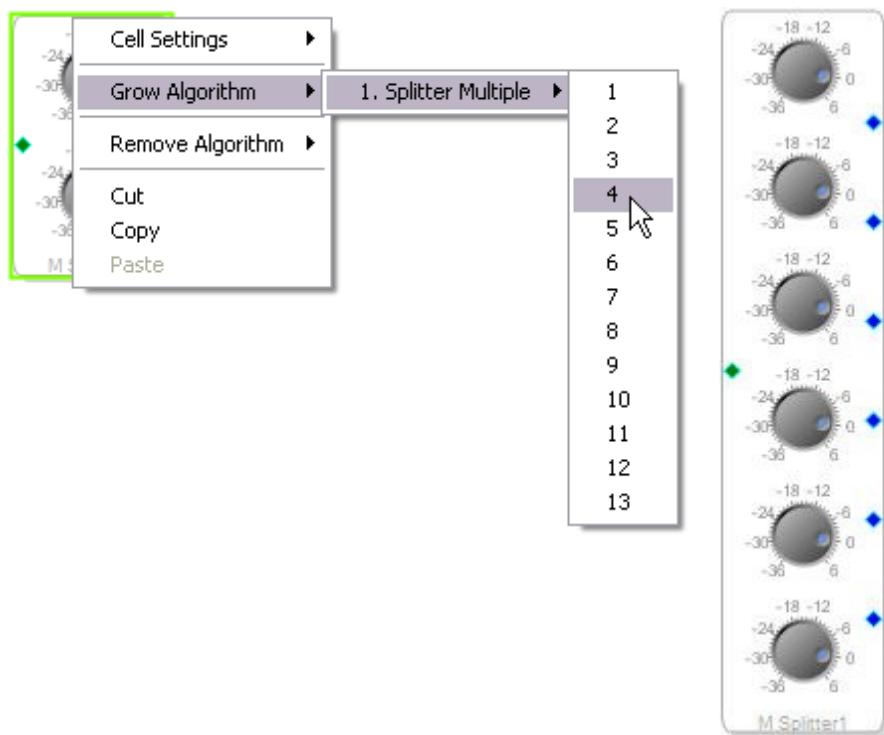
## Multiple-Control Splitter

The Multiple Control Splitter allows an input signal to be split into two or more outputs with independent control of each output's gain settings

There is a control knob for setting the gain of each output. To set the gain, click the knob with the left mouse button and drag while holding down the button. Each knob's range, value, and step size can be customized in the control pop-up window. To change the knob settings, right-click on the knob control.



The block can be grown by right-clicking and selecting Grow Algorithm from the menu. Growing creates additional output pins and control knob. Up to 15 outputs maximum can be created (13 more than the initial 2 outputs).



---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Signal Merger

The Signal Merger mixes a group of input signals and automatically decreases the signal levels in proportion to the number of inputs. This block's algorithm can be grown up to 15 inputs.



This block helps to avoid level overages (clipping) without the need to manually adjust mix levels.

---

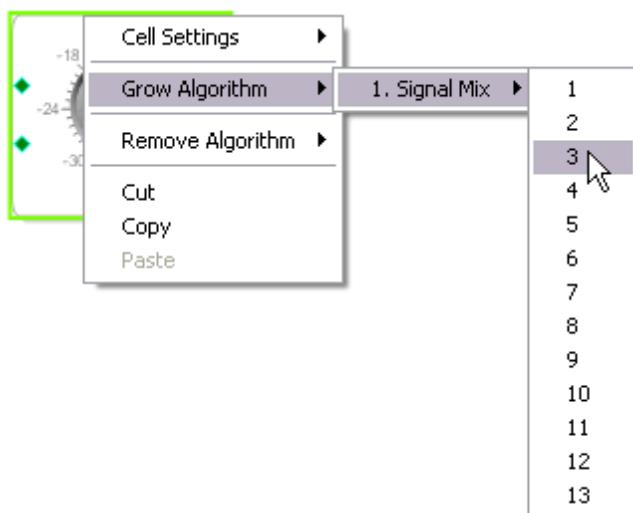
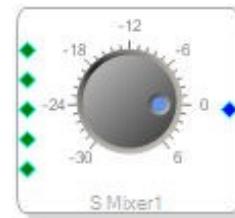
© 2006-2007 Analog Devices, Inc. All rights reserved.

## Single-Control Mixer

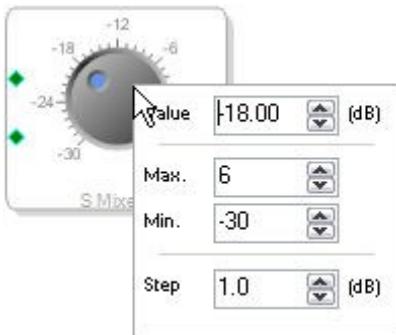
The Single Control Mixer lets you mix multiple inputs down to one output, with a single knob to set the gain of all inputs.

The default block has two inputs; right-click and select Grow Algorithm from the menu to add additional inputs.

The example at right shows the result of Grow Algorithm > 1. Single Control Mixer > 3, creating 5 total inputs.



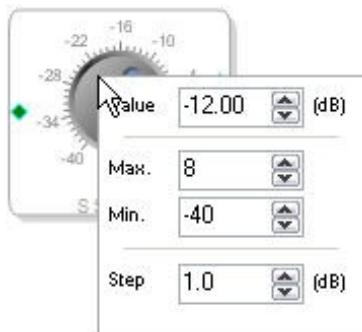
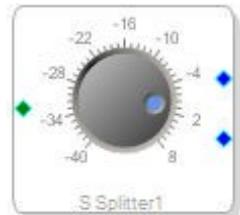
Right-click the knob control to change its settings:



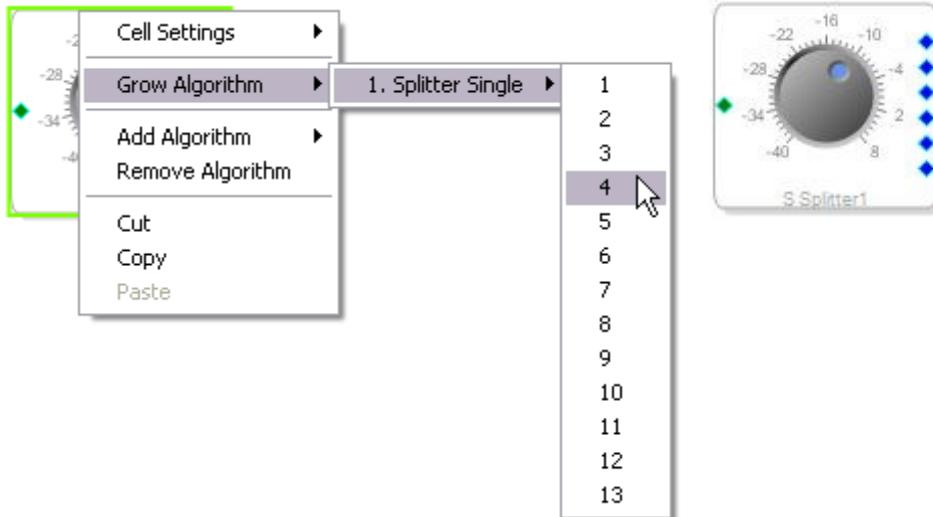
## Single-Control Splitter

This Single Control Splitter block splits one input into multiple outputs, with a single knob control for all the output gains. Two outputs is the default (see right), but right-click the block edge to grow it up to any number of outputs, max 15.

To set the output gain, click the knob with the left mouse button and drag while holding down the button. The knob's range, value, and step size can be customized in the control pop-up window. To change the knob's settings, right-click on the knob control.



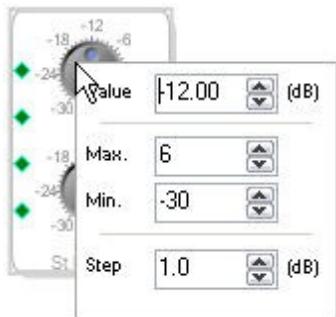
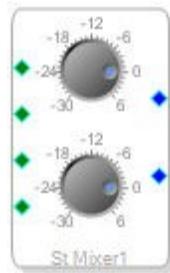
Additional outputs can be created by growing the algorithm. Right-click and select Grow Algorithm from the menu. Up to 15 outputs maximum can be created.



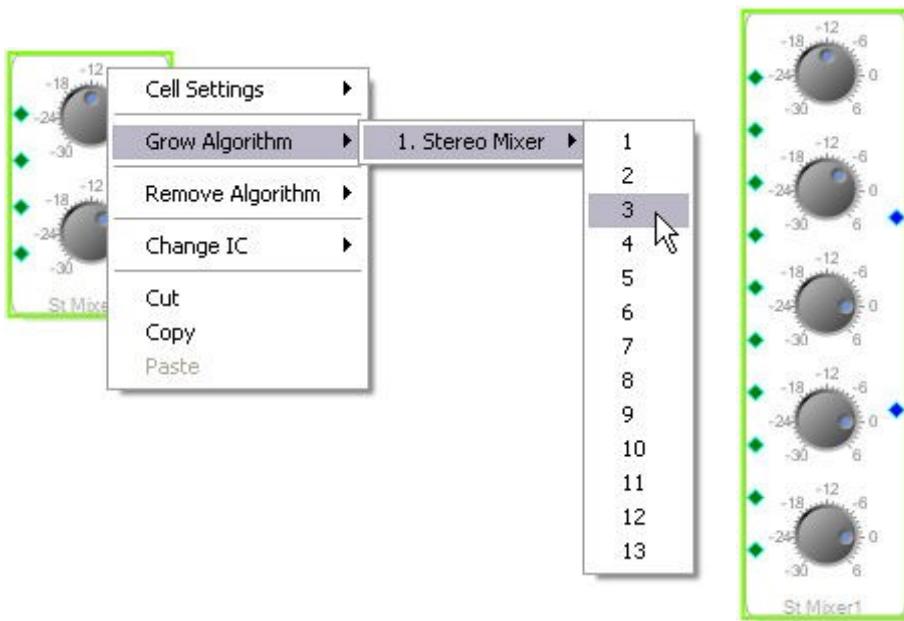
## Stereo Mixer

The Stereo Mixer mixes two or more stereo input channels to a single stereo output. Each stereo input's mix level can be independently adjusted.

There are two control knobs for setting the gain, one knob for each pair of input pins. To set the gain, click the knob with the left mouse button and drag while holding the button down. Each knob's range, value, and step size can be customized by **right-clicking** on the knob control.



To add additional stereo inputs, perform a Grow Algorithm on this block, **right-click** and select **Grow Algorithm** from the menu, as shown below. A maximum of 15 stereo inputs is supported.



---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Multiplexers / Demultiplexers

### Multiplexers/Demultiplexers

---

The Multiplexer and Demultiplexer library of the ToolBox contains blocks for multiplexing or demultiplexing signals in various ways.

The following blocks are available:

- Crossfade(data controlled)
- Index-Selectable DeMultiplexer
- Index-Selectable Multiplexer
- Index-Selectable Slewing Mux
- Mono Switch 1xN
- Mono Switch Nx1
- Multichannel Switch Nx4
- Multichannel Switch Nx6
- Multichannel Switch Nx8
- State Machine
- Stereo Switch 2xN
- Stereo Swith Nx2

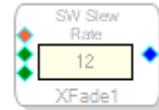
Take a look at the Multiplexer/Demultiplexer Examples see sample schematics utilizing some of the blocks.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Crossfade (Data-Controlled)

This block creates a smooth transition between two input signals: the volume of one signal is decreased while the other input signal level increases, creating a gradual switch (cross-fade) between the inputs.

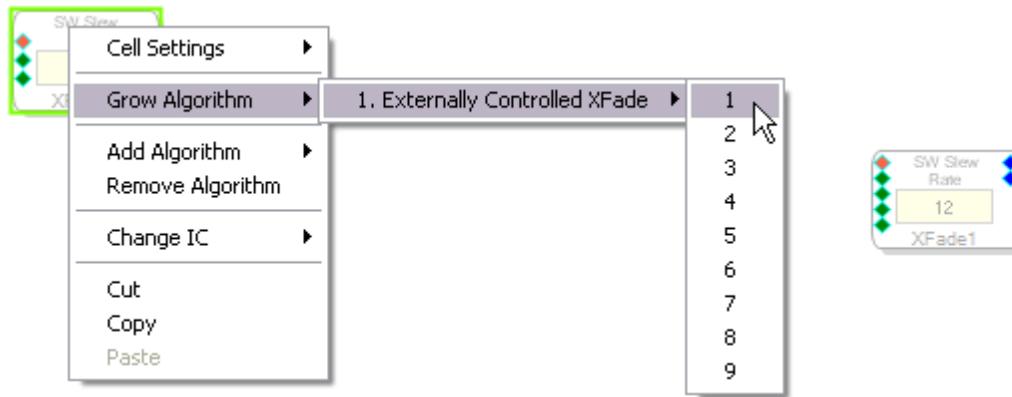


The output signal is selected by setting a value on the input control pin (top orange pin) to the desired input's index of either 0.0 or 1.0. The control signal should be a 28.0 format integer value, typically from a DC input block or RMS Table.

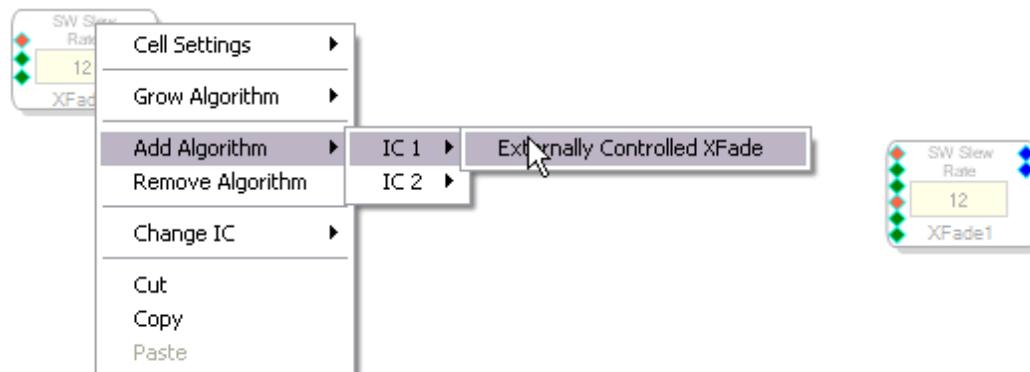
When the control input value changes, a cross fade is initiated between the current output and the newly selected input signal. The cross-fade transition rate (slew rate) can be adjusted using the numerical control on the block, (the maximum SW Slew Rate value is 23).

This block's algorithm can be grown or added:

- Select Grow Algorithm from the right-click menu to create an additional output and input pair (sharing the signal select control pin and SW slew ram parameter).



- Select Add Algorithm from the right-click menu to create an additional algorithm with its own independent input select control pin.



See the Multiplexer/Demultiplexer Example of the Crossfade (Data-Controlled) block.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Index Selectable DeMultiplexer

This block lets you route an input to one of two selectable output pins. The output is selected based on a control signal value (orange pin), in 28.0 integer format from an Index Lookup Table, RMS Table, or DC Input block.



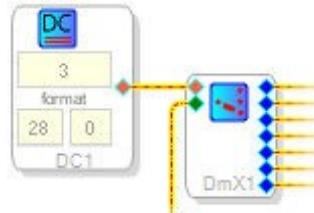
See the Multiplexer/Demultiplexer Example utilizing this block.

### Algorithms:

The default algorithm for this block is the *Mono Demux(unselected outputs 0)*. Selecting the *Stereo Demux(unselected outputs 0)* allows you to use a single control signal to control 2 input to output groups. Typically, this is used for routing a stereo signal to one of two pairs of stereo output pins.



This block can be grown by Right-clicking and selecting Grow Algorithm from the menu. Growing will create additional output pins: single pins with the Mono algorithm, stereo output pin pairs with the Stereo algorithm. To select an output signal, set the control signal to the output pin's integer index starting from 0.



## Index Selectable Multiplexer

This block routes the signal from either of its two input pins, based on the control signal input index value from the Index Lookup Table, RMS Table or DC Input.

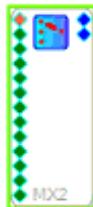


See the Multiplexer/Demultiplexer Example utilizing this block.

Drag the block into the workspace and right-click to select Add Algorithm > IC # > Mono/Stereo. Mono is shown above right. Stereo, which can be used for routing a stereo signal from one input pair, is shown here:



Like its companion blocks, this block can be grown. Right-click and select Grow Algorithms from the context menu. Growing will create additional input pins, single pins with the Mono algorithm, stereo input pin pairs with the Stereo algorithm. To select a particular input signal, set the control signal to an input pin's integer index starting from 0, using a 28.0 format integer value.

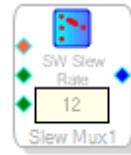


---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Index Selectable Slew Mux

This block routes any one of its inputs, based on the index value from the Index Lookup Table, RMS Table or DC input, letting you ramp the source level up and down without any clicking noises when switching inputs.



Incrementing the value in the SW Slew Rate field slows the slewing speed. For the AD1940, the Index-Selectable Slew Mux uses one hardware volume control in the target/slew RAM per input pair.

Like its companion blocks, this block can be grown. Right-click and select Grow Algorithms from the context menu. Growing will create an additional input pin or pins. To select a particular input signal, set the control signal to an input pin's integer index starting from 0, using a 28.0 format integer value.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Mono Switch 1xN

The Mono Switch block routes an input signal to one of many possible outputs, the strip of radio buttons is used to select an output (each radio button corresponds to a coefficient RAM parameter).



To begin, drag the block into the workspace, right-click it and select Add Algorithm > IC N Mono/Slew or Mono. (The result appears the same.) The Mono/Slew algorithm lets you rapidly ramp the source level up and down without any clicking noises when switching outputs. It uses  $N$  hardware volume controls in the target/slew RAM to do its job.

See the Multiplexer/Demultiplexer Example utilizing this block.

The default block is ready to use as a 1x2 connection, and like the others can be grown by right-clicking and selecting the desired number of outputs. The example below has been grown by 3:



---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Mono Switch Nx1

---

The Mono Switch block routes one of many possible input signals to the output, the strip of radio buttons is used to select the input signal (each radio button corresponds to a coefficient RAM parameter).

To begin, drag the block into the workspace, right-click it and select Add Algorithm > IC # Mono/Slew or Mono. The blocks look the same, but the Mono/Slew algorithm performs a rapid fade-out fade-in of the source signals to prevent clicking noises when switching among the inputs. For the AD1940 it uses one hardware volume in the target/slew RAM per input to perform the slew.

The default block is ready to use as a 2x1 connection, and like the others can be grown by right-clicking to select your desired number of inputs. Shown above is the default block grown by 3 (= 5 inputs total).

See the Multiplexer/Demultiplexer Example utilizing this block.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

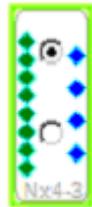
## Multi-channel Switch Nx4

---

Use the radio buttons in this multichannel block to select among sets of 4 inputs and send one set to one of the (4-channel) outputs.

To begin, drag the block into the workspace, right-click it and select Add Algorithm > IC # > 4 Channel/Slew or 4 Channel. (The result appears the same either way.) The 4 Channel/Slew algorithm lets you rapidly ramp the source level up and down without any clicking noises when switching among outputs. It uses  $N$  hardware volume controls in the target/slew RAM to do the job.

The default block (above right) is ready to use as a 2x1 connection, and like the others can be grown by right-clicking to select your desired number of input sets. The example to the right shows the default block grown by 3 (= 5 input sets total).



---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Multi-channel Switch Nx6

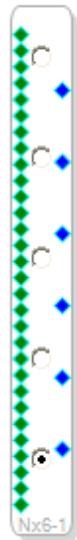
Use the radio buttons in this multi-channel block to select among sets of 6 inputs and send one set to one of the (6-channel) outputs.

To begin, drag the block into the workspace, right-click it and select Add Algorithm > IC # > 6 Channel/Slew or 6 Channel. (The result, shown at right top, appears the same either way.)



The 6 Channel/Slew algorithm lets you rapidly ramp the source level up and down without any clicking noises when switching among outputs. It uses  $N$  hardware volume controls in the target/slew RAM to do the job.

The default block (shown at right) is ready to use as a 2x1 connection, and like the others can be grown by right-clicking to select your desired number of input sets. Shown at lower right is the default grown by 3 (= 5 input sets total).



---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Multi-channel Switch Nx8

Use the radio buttons in this multi-channel block to select among sets of 8 inputs and send one set to one of the (8-channel) outputs.

To begin, drag the block into the workspace, right-click it and select Add Algorithm > IC # > 8 Channel/Slew or 8 Channel. (The result, shown at right top, appears the same either way.)

The 8 Channel/Slew algorithm lets you rapidly ramp the source level up and down without any clicking noises when switching among outputs. It uses  $N$  hardware volume controls in the target/slew RAM to do the job.

The default block (above right) is ready to use as a 2x1 connection, and like the others can be grown by right-clicking to select your desired number of input sets. At lower right is the default grown by 5 (= 7 input sets total).



---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## State Machine

This block outputs the input signal from the green pin, if and only if the control signal level (orange pin) falls within the range specified in the numerical spin controls (>) and (<).



If the control signal is out of range, the input is disabled and a zero value is output.

The control pin can be sourced by a DC Input Entry, Counter, or an Index Lookup Table block. The control value should be a 28.0 format integer.

See the Multiplexer/Demultiplexer Example utilizing this block.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Stereo Switch 2xN

---

Route a stereo input to one of several possible stereo outputs with this block, using the strip of radio button pairs to select which output.

To begin, drag the block into the workspace, right-click it and select Add Algorithm > IC N Stereo/Slew or Stereo. (The result appears the same; this default is shown at top right.) The Stereo/Slew algorithm lets you rapidly ramp the stereo source level up and down without any clicking noises when switching among the stereo outputs. It uses  $N$  hardware volume controls in the target/slew RAM to do the job.

The default block shown above right is ready to use as a 2x4 connection, and like the others can be grown by right-clicking to select your desired number of stereo outputs. Below the default at right is a block grown by 3 (3 pairs, equaling 6, totaling 10).



---

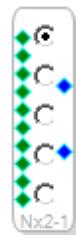
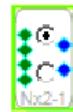
© 2006-2007 Analog Devices, Inc. All rights reserved.

## Stereo Switch Nx2

Use the radio buttons in this multichannel block to select between or among sets of stereo inputs and send one set to one of the (2-channel) outputs.

To begin, drag the block into the workspace, right-click it and select Add Algorithm > IC N Stereo/Slew or Stereo. (The result appears the same either way.) The 2 Channel/Slew algorithm lets you rapidly ramp the source level up and down without any clicking noises when switching among outputs. It uses  $N$  hardware volume controls in the target/slew RAM to do the job.

The default block (above right) is ready to use as a 2x1 connection, and like the others can be grown by right-clicking to select your desired number of input sets.



See the Multiplexer/Demultiplexer Example utilizing this block.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Sources

### Sources

---

The Sources library of the ToolBox provides blocks that generate signals. The following blocks are available:

- Beep Sources
- DC Input Entry
- On/Off Switch
- Sawtooth Wave
- Square Wave
- Sweep (Log)
- Sweep (Lookup/Sine)
- Tone (Lookup/Sine)
- Triangle Wave
- VCO
- White Noise Source

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Beep

This block uses an internal oscillator to generate tones. Set the frequency in the text field or using the arrows. Click click Beep button to enable the tone generation. This source is only active while the beep button is pressed.



You can add algorithms to the Beep blocks after the default has been established (if you're using more than one DSP processor, you'll need to add the initial default algorithm for the desired IC). Right-click the block to select Add Algorithm > IC # > Beep variable gain and add another output pin.

To change the source's Sampling Rate, Right-click in the block and select Set Sampling Rate, which will open the Sampling Rate window (default is 44.1 kHz).

### Usage

For most applications we recommend using Tone (Lookup/Sine) instead of Beep. The Beep block output level is inconsistent, the oscillator tends toward instability above 6-7 kHz and, most importantly, the output is never limited. This means that in certain circumstances downstream audio circuits can be overdriven (e.g., 12kHz beep combined with input audio to a mixer module). If you do use the Beep algorithm, be sure to add a limiter (see the Dynamics Processors topics).

Beep is most valuable for schematic testing as it produces less THD than the Tone (lookup/sine) block.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## DC Input Entry

The DC Input block allows you to generate a DC, or direct current, signal (a constant numerical value). This block can be used to generate control signals for Multiplexers and Lookup Tables.



The block's controls allow the DC value and the format to be set. There are 28 available bits which can be used to represent decimal values. The default format is 5.23: 5 bits for the integer, 23 for the decimal. The valid values for 5.23 format are -16 through 15.999.

**Note:** Many control signal input pins require 28.0 format integer values, which has a valid range from 0 to 134217727. See Numeric Formats for more information.

---

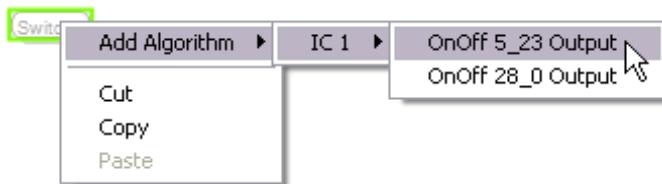
© 2006-2007 Analog Devices, Inc. All rights reserved.

## On/Off Switch

The On/Off Switch outputs a constant value of either 0.0 (OFF) or 1.0 (ON). Click the switch control with the mouse to toggle it on or off.



There are two available algorithms: *OnOff 5\_23 Output* and *OnOff 28\_0 Output*. The outputs numerical format will either be 5.23 or 28.0 depending on which algorithm is selected.

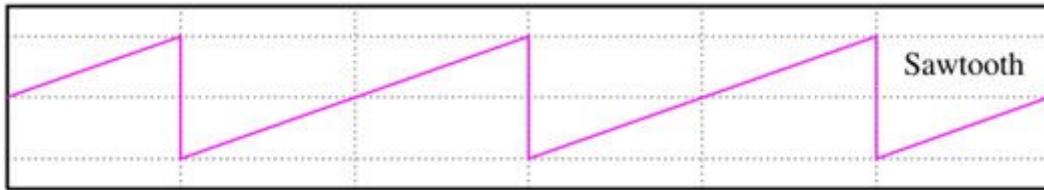


---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Sawtooth Wave

The Sawtooth Wave block generates a sawtooth wave (or saw wave) at a constant level and frequency. The output frequency is adjustable. Use the edit control or arrows to set the desired frequency; the checkbox control turns the signal on and off.



To change the source's Sampling Rate, Right-click in the block and select Set Sampling Rate, which will open the Sampling Rate window (default is 44.1 kHz).

**Note:** The sawtooth wave is digitally generated and is **non-bandlimited**. This waveform is suitable for use as an internal control signal, but will produce aliasing distortion if used directly for audio output. If this configuration is desired, it is recommended that you apply a low-pass filter to the block's output before routing the signal to hardware outputs.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Square Wave

The Square Wave block generates a square wave at a constant level and frequency. The output frequency is adjustable. Use the edit control or arrows to set the desired frequency; the checkbox control turns the signal on and off.



To change the source's Sampling Rate, Right-click in the block and select Set Sampling Rate, which will open the Sampling Rate window (default is 44.1 kHz).

**Note:** The square wave is digitally generated and is **non-bandlimited**. This waveform is suitable for use as an internal control signal, but will produce aliasing distortion if used directly for audio output. If this configuration is desired, it is recommended that you apply a low-pass filter to the block's output before routing the signal to hardware outputs.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Sweep (Log)

The Sweep (Log) block generates a sinewave that sweeps from a start frequency to an end frequency. The sweep rate is determined by the number of sweep steps and cycles per step.

Enter the desired start and stop frequency and step size using the edit controls:

**Start Freq**

The frequency at which to begin the sweep.

**Stop Freq**

The frequency at which to end the sweep.

**# Steps**

Determines the number of steps in the sweep.

**# Cycles/Step**

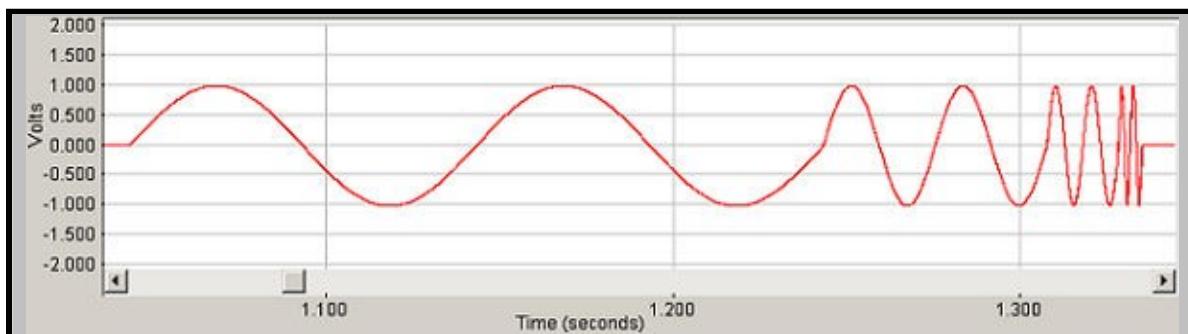
Determines the number of the cycles (sine wave cycles) in each step.

**On/Off switch**

Starts the sweep



The graph below shows a sweep from 10 Hz to 300 Hz achieved in four steps with two cycles in each one.



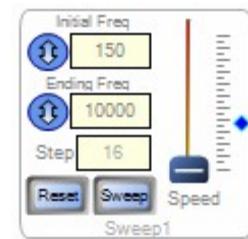

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Sweep (Lookup/Sine)

The Sweep (Lookup/Sine) block generates a sinewave that sweeps from a start to an end frequency. The sweep speed is determined by the number of steps and cycles per step.

Enter the desired start and stop frequency and step size using the edit controls:



**Initial Freq**

The frequency at which to begin the sweep.

**Ending Freq**

The frequency at which to end the sweep.

**Step**

Determines the number of steps in the sweep.

**Reset**

Resets the current sweep frequency to the "Initial Freq".

**Sweep**

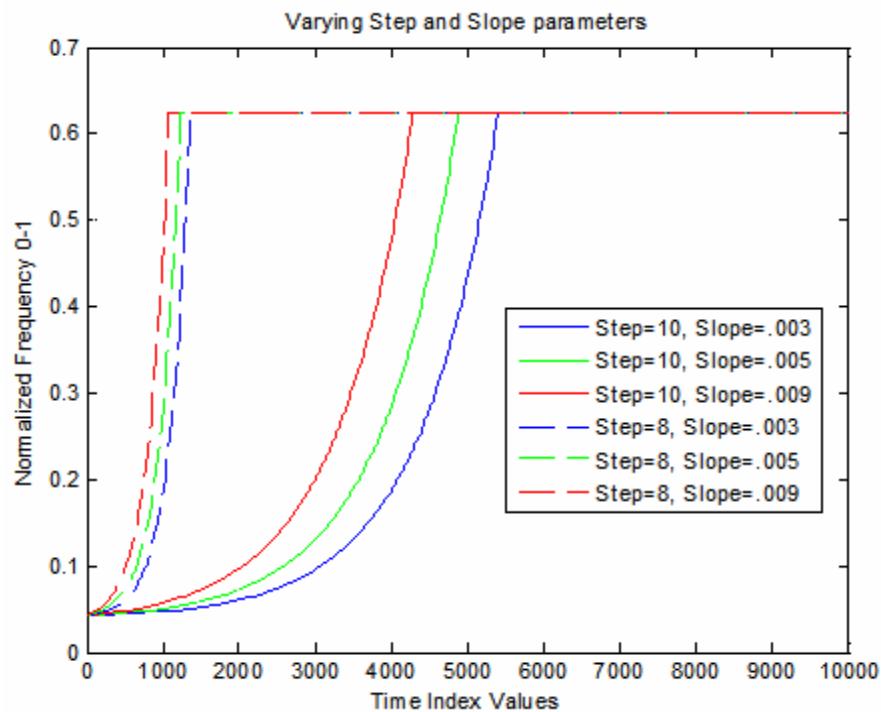
Starts the frequency sweep.

**Speed**

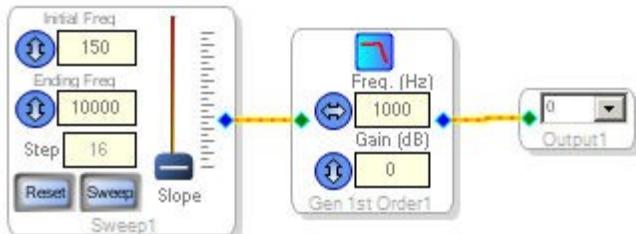
Sets the sweep speed, the relative duration of each sweep step.

A valid slope range will be computed depending on the step size entered.

The ending tone will be held until reset is clicked, which returns to the initial frequency. The plot below shows the correlation for sweep plots with varying step sizes and slopes.

**Example:**

This sample schematic uses the Sweep (Lookup/Sine), General (1st-Order) and an Output block to show how the Sweep block can be utilized with a filter.



This setup can be used in conjunction with a spectrum analyzer to determine the response of a particular filter or algorithm.

## Tone (Lookup/Sine)

The Tone (Lookup/Sine) block generates a tone from a lookup table, keeping the level constant regardless of frequency. Set the tone frequency in text field or use the arrows; the checkbox turns the tone on and off.



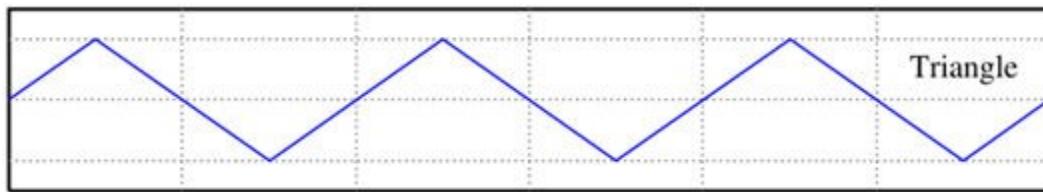
To change the source's Sampling Rate, Right-click in the block and select Set Sampling Rate, which will open the Sampling Rate window (default is 44.1 kHz).

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Triangle Wave

The Triangle Wave block generates a triangle wave at a constant level and frequency. The output frequency is adjustable. Use the edit control or arrows to set the desired frequency; the checkbox control turns the signal on and off.



To change the source's Sampling Rate, Right-click in the block and select Set Sampling Rate, which will open the Sampling Rate window (default is 44.1 kHz).

**Note:** The triangle wave is digitally generated and is **non-bandlimited**. This waveform is suitable for use as an internal control signal, but will produce aliasing distortion if used directly for audio output. If this configuration is desired, it is recommended that you apply a low-pass filter to the block's output before routing the signal to hardware outputs.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## VCO

The voltage-controlled oscillator block, typically used for modulation applications, takes a control input signal (orange pin) and outputs a frequency on the blue pin.



The input value may be from zero to one (5.23 format), and the VCO linearly interpolates its derived value into an output frequency from 0 Hz (0.0 input) to  $f_s/2$  (1.0 input).  $f_s$  is the sampling frequency.

If the input applied is greater than 1 V, the derived output value will fold back to frequencies below  $f_s/2$ .

To change the source's Sampling Rate, Right-click in the block and select Set Sampling Rate, which will open the Sampling Rate window (default is 44.1 kHz).

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## White Noise Source

The White Noise cell generates a signal that contains equal energy per frequency division (hertz or similar increment).



This random time-varying signal can be useful for testing equipment, although for audio situations white noise is commonly run through a pinking filter, to better simulate the behavior and response of the human ear.

A simple on/off toggle (right) controls the cell output.

You can add algorithms to the White Noise cell after the default has been established (if you're using more than one DSP board, you'll need to add the initial default algorithm for the desired board). Right-click the cell to select Add Algorithm > IC N > White Noise and add another output pin.

To change the source's Sampling Rate, Right-click in the block and select Set Sampling Rate, which will open the Sampling Rate window (default is 44.1 kHz).

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

# Volume Controls

## Volume Controls

---

The Volume Controls category of the ToolBox gives you access to blocks that control signal levels.

Following is a list of the available blocks:

- Adjustable Volume Control
- Mute
- Multiple Volume Control
- Single slew ext vol
- Single SW slew vol
- Single Vol (shared)
- Single Volume Control
- Surround Sound Volume Control
- Up/Down Control w/ Lookup Table

See Algorithm Information for more details about the different Volume Control algorithms.

## Adjustable Volume Control

This block lets you control which slew curve, and which time constant for the slew algorithm, to use.

Choose slew type using the top drop-down control, Linear, Const[ant] dB, and RC-type:

**Linear:** Slews to target using a fixed step size. The ramp ranges from 6.75 to 213.4 ms.

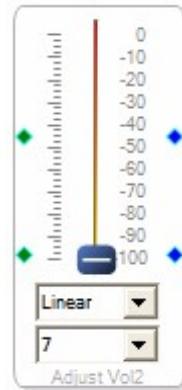
**Constant dB:** Slews to target using the current value to calculate the step size. The resulting curve has a constant rise and decay when measured in dB. The ramp ranges from 6.1 ms to 1.27 s.

**RC-type:** Slews to target using the difference between the target and current values to calculate the step size. This produces a simple RC-type curve for both rising and falling. The ramp ranges from 6.1 ms to 1.27 s.

With the bottom drop-down control, choose a time-constant value between 0 (fastest volume change) and -15 (slowest volume change).

The default block comes with the Gain slew algorithm; right-click the block border or title, per usual, to add input/output pins. The slider controls the input(s) / output(s) the same, and the time constant applies to all inputs.

The example at above right shows the default block with one algorithm added.



---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Mute

When enabled, the Mute block "mutes" the input signal so that there is no output (a 0.0 value DC signal). This is useful when powering up or down and when switching program flow.



### To Enable/Disable the mute:

Click the check box, the signal is muted when the box is checked.

The mute block includes Slew and Non-Slew type algorithms. Slew algorithms will smoothly transition the gain to zero, eliminating any click or pops, but require more system resources.

### Algorithms:

*No Slew (Standard)* - The default mute algorithm is a "no slew" type algorithm which mutes the signal immediately when enabled, there is no gain ramping, this algorithm requires less resources than the slew algorithms, but it can result in discontinuities (clicks and pops) when toggled in real-time.

*HW (RCtype Slew)* - The AD1940 / 1941 include support for mute controls that use a target/slew RAM hardware to slew (smoothly transition) from 0dB gain to muted.

*SW (RCtype Slew)* - Slew type algorithm that smoothly transitions from 0dB gain to muted and muted to 0dB gain. This slew algorithm is implemented in software, "SW", and thus requires more instructions than the No Slew algorithm.

This block supports both Grow Algorithm (adds additional inputs and outputs) and Add Algorithm (adds an independent mute algorithm plus associated inputs and outputs).

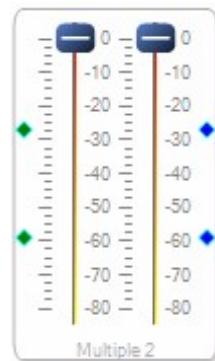
## Multiple Volume Control

The Multiple Volume Control permits gain adjustments to be made to each of the inputs individually. Every input pin has its own volume control.

### Algorithms:

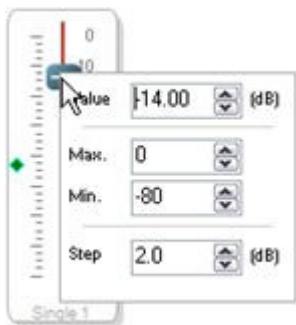
*Gain (no slew)* - The "no slew" algorithm performs gain changes immediately with no transition (ramping), this algorithm requires much lower program and memory resources than the slew algorithms, but it can result in discontinuities (clicks and pops) when making volume changes in real time.

*Gain (slew)* - The AD1940 / 1941 includes support for volume controls that use a target/slew RAM hardware to slew (smoothly transition) from one volume level value to another. For the Multiple Volume Control block, the slew curve is a linear-ramp with fixed time constant of 8.



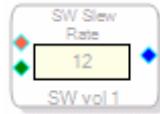
By default the block's algorithm has one input and one output. To add additional channels (input, output, & slider) Right-click the block and select Add Algorithm from the menu.

The Slider control's min/max value and step size can be customized. To modify the slider's settings, **right-click** on the control which will display the control pop-up window (shown below).



## Single slew ext vol

The Single Slew External Volume block permits external control of the level. Such capability would be appropriate for an application like external control communications to an 8-10-bit ADC.



Connect an input audio signal to the green pin and a control signal to the orange, external control pin. The text field allows you to specify the slew-rate time constant between 1 and 23 (Note this is a larger range than available for the Adjustable Volume Control block).

You can both Grow Algorithm (add additional inputs and outputs) and Add Algorithms (Add additional control input pins and associated inputs and outputs) to this block. Refer to Single Vol (Shared) for detailed discussion or adding and growing.

---

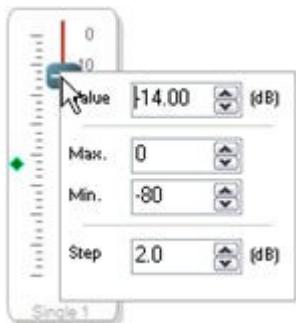
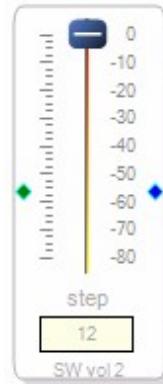
© 2006-2007 Analog Devices, Inc. All rights reserved.

## Single SW slew vol

The Single SW Slew Vol uses the same algorithm for volume control as the Single Slew Ext Vol, but the volume level is set using the slider (or by setting the associated coefficient parameter) rather than by external input. Connect an input signal and control its level with the slider. Optionally the slew rate can be adjusted in the range 1 to 23.

You can both Grow Algorithm (add additional inputs and outputs) and Add Algorithms (Add additional control input pins and associated inputs and outputs) to this block. Refer to Single Vol (Shared) for detailed discussion or adding and growing.

The slider control's min/max value and step size can be customized. To modify the slider's settings, right-click on the control which will display the control pop-up window (shown below).



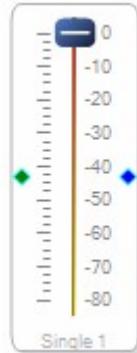
---

© 2006-2007 Analog Devices, Inc. All rights reserved.

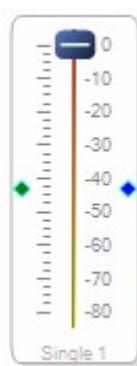
## Single Vol (shared)

The Single Vol (Shared) block is a volume slider programmed with the slew algorithm. If your DSP model does not support the slew algorithm, use the Single Volume Control (it's direct-write and has the option for the algorithm with no slew). If your DSP does support the slew algorithm, use this block, because it's more powerful. The ability to Add and Grow algorithms is important, especially in applications where multiple DSP boards will be used; this volume control lets you grow a particular algorithm already selected for a DSP.

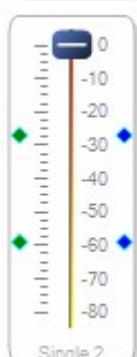
The following steps clarify the difference between an added and grown algorithm with two DSPs. Observe that there is no visible difference between a grown and added algorithm on the block itself, and the only indication of different DSP boards is wire color of block connections.

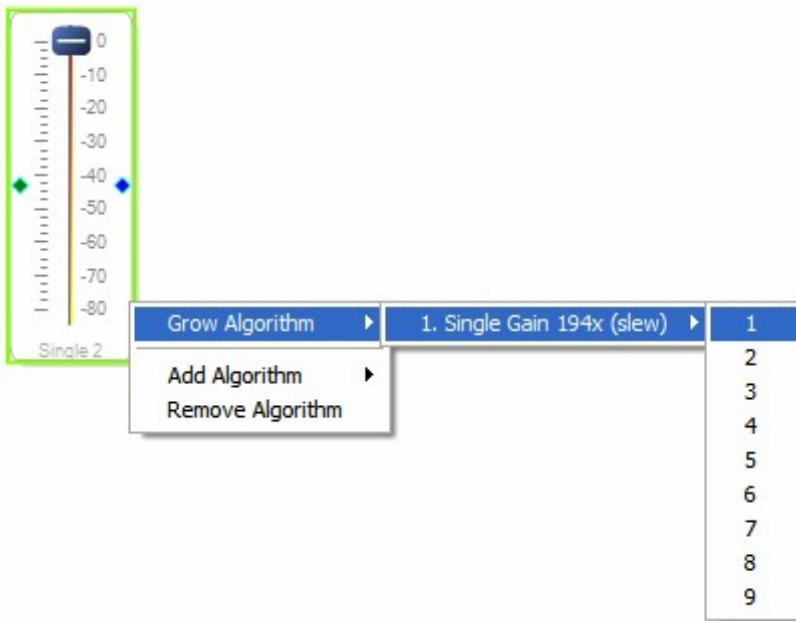


1. Drag the block into the workspace. If you have multiple DSP boards connected, It will be an empty block, as shown here.
2. Right-click it and select Add Algorithm > IC 1 > Single Gain xxxx (slew). (If you have only one DSP connected, you don't have to do this.)

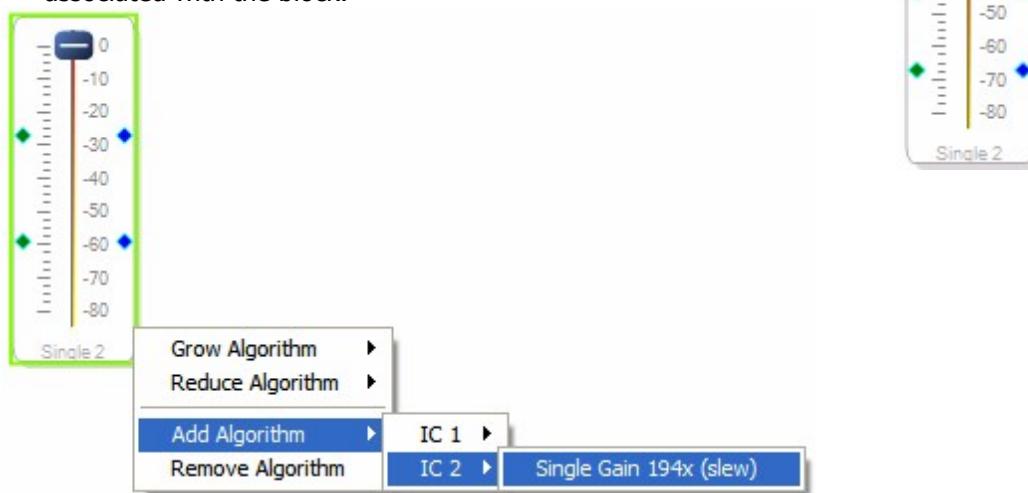


3. Right-click the block border or title and select Grow Algorithm > 1. Single Gain xxxx (slew) > 1. This adds another set of input/output pins for the same DSP board and algorithm as in step 2.

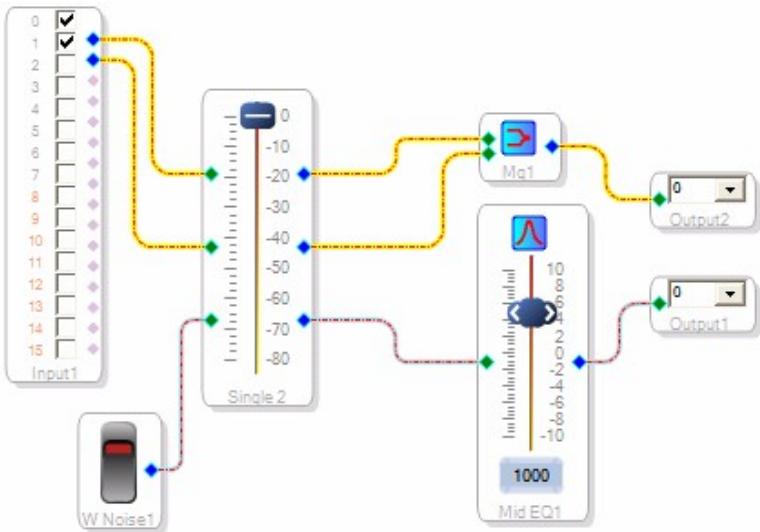




4. Right-click the block border or title and select Add Algorithm > IC 2 > Single Gain xxxx (slew) > 1. This adds another set of input/output pins, but corresponding to a *different* DSP board. Other blocks that you wish to connect to these pins must be independent (not associated with any DSP), or they must have the same algorithm associated with the block.



Now the top two pins are associated with the DSP under IC 1, while the bottom pin is associated with the DSP under IC 2. To wire this block to other blocks, you maintain the same DSP association; the software will not let you connect pins from one DSP to another. Observe the different-color traces for the different DSP boards below:



---

© 2006-2007 Analog Devices, Inc. All rights reserved.

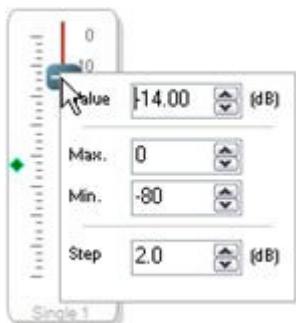
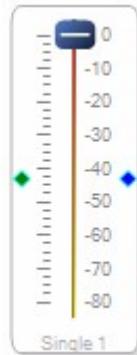
## Single Volume Control

The Single Volume Control block is a volume slider with access to both the slew and no-slew algorithms.

Right-click the block border or title to Add algorithms, which adds pins controlled by the slider. Each pin can be associated with a different DSP and/or algorithm (slew or no slew) under control of this slider. Pay attention when selecting multiple algorithms, because there is no visible indication.

Note: This block does not have the ability to grow (use the Single Vol (Shared) block).

The slider control's min/max value and step size can be customized. To modify the slider's settings, right-click on the control which will display the control pop-up window (shown below).



---

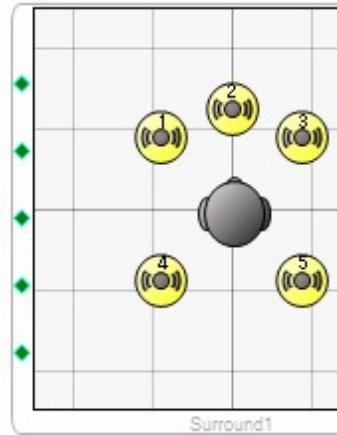
© 2006-2007 Analog Devices, Inc. All rights reserved.

## Surround Sound Volume Control

The Surround Sound Volume Control block enables positioning of source elements relative to a listener head. The default block has one source centered in front of the listener. Right-click the block border or title to grow the algorithm up to 6 sources. The figure at right is the default algorithm grown by 4, to achieve a typical 5-channel surround speaker setup.

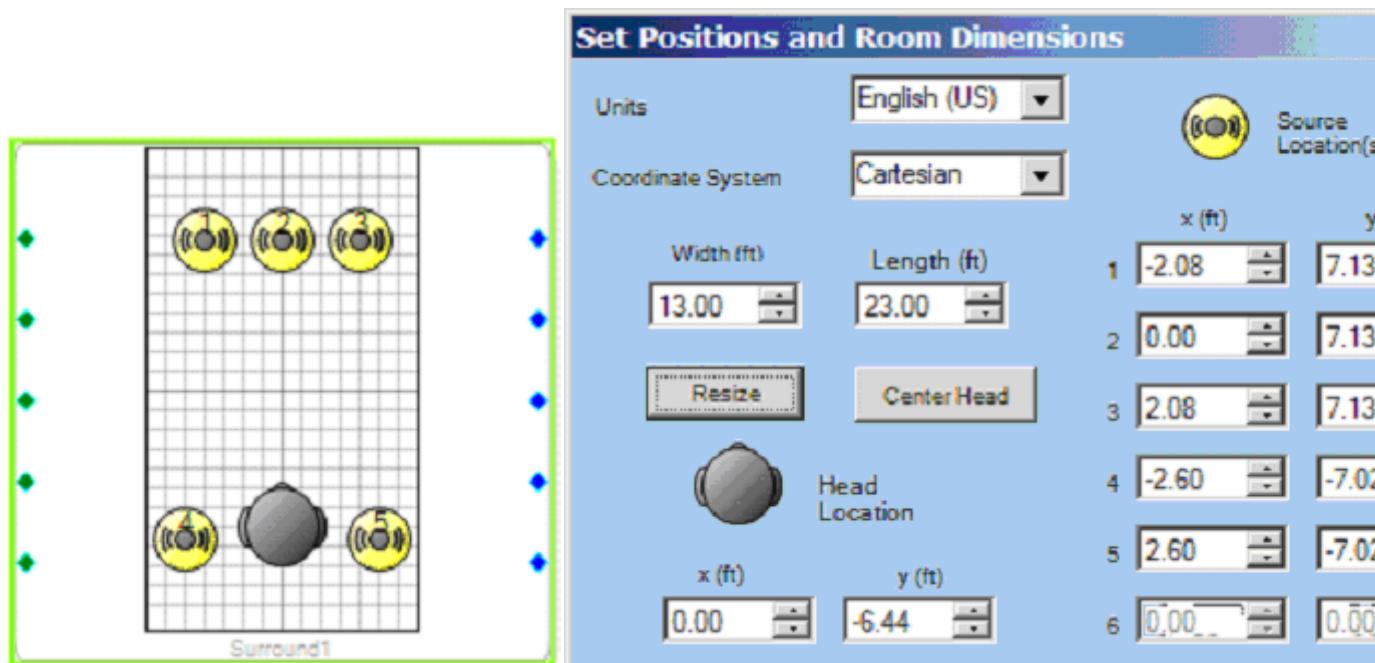
Sources and head can be dragged to any position in the room, or set more precisely via the Set Positions and Room Dimensions form (right-click the center of the block). Its parameters include:

- Units -- English (US) or Metric
- Coordinate System -- Cartesian or Polar
- Width and length of the room
- Head location
- Source locations



Press Resize to accept changes. Grayed-out source locations mean the source is not currently added.

A typical living room / HT setup is shown below:



© 2006-2007 Analog Devices, Inc. All rights reserved.

## Up/Down Control w/ Lookup Table

This block takes in two inputs, one “up” and one “down,” and uses them to generate an index into a lookup table. The starting index is one stored in an interface register, and the up/down inputs increment or decrement the value. The output is the value retrieved from the table.

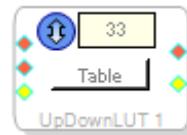
This block is available only for DSPs that support GPIO. Before using it, it may be useful to review the GPIO Conditioning examples.

### Description

The algorithm first reads an index from a stored value in the interface register. The value is then adjusted by two or three control signals, depending on the choice of algorithm.

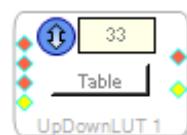
### Up/Down + INTF

This algorithm uses two control signals, up/down signals, generated either by up/down buttons that have been conditioned by Push and Hold or by the GPIOs connected to outputs of the rotary encoder that have been conditioned by the Rotary Encoder block. These control signals either increment or decrement the index value. This index is then used to retrieve a value stored in a lookup table, which is sent to the output pin. The algorithm also writes the index value back to the interface register for storage.



### Up/Down/Simultaneous + INTF

This algorithm uses three control signals. Two are up/down signals, generated by up/down buttons been conditioned by the Push and Hold block, and the third is the mute output from it. The up/down control signals either increment or decrement the index value. This index is then used to retrieve a value stored in a lookup table, which is sent to the output pin. If a mute signal is present, the algorithm will output the FIRST value in the table (index = 0), so the value must be selected with caution. Once the control signal is unmuted, the index resumes to the value it was at before muting. The algorithm also writes the current index value back to the interface register for storage.



## Examples

### Examples

---

The following schematics provide usage examples for some of the SigmaStudio ToolBox blocks.

- Basic DSP
- Dynamics Processor
- Filter
- GPIO Conditioning
- Level Detectors
- Mixers/Splitters
- Multiplexer/Demultiplexer
- System
- Volume Control

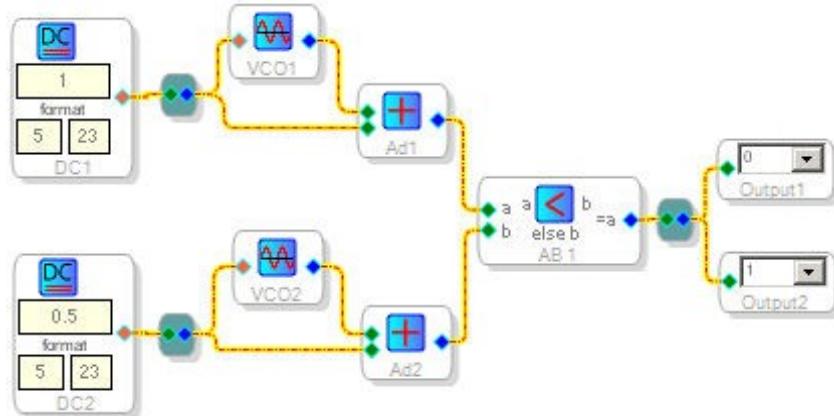
---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Basic DSP Examples

### Example 1:

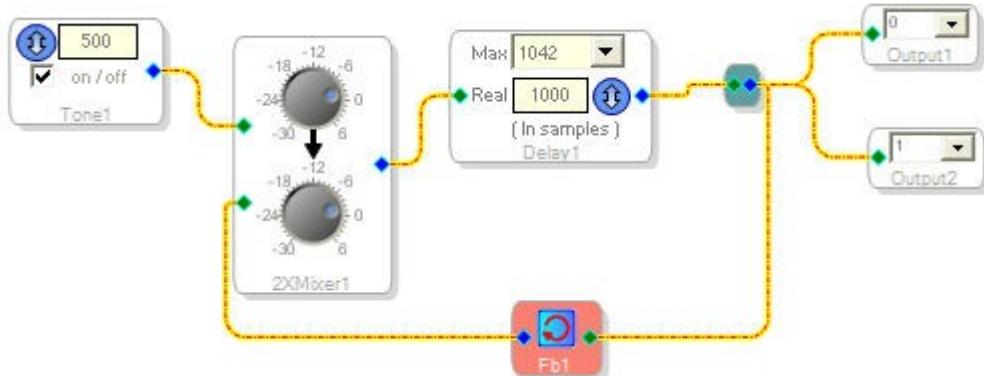
This schematic uses the DC Input block, Adders, and an AB In/Out Condition block to compare the level of two input VCO sources that have a dc value added to them. The output of the Condition block is split with a T Connection and sent to Output.



For the current configuration, the condition will evaluate false, and signal B will be sent to the output. If you click the Condition icon in the AB In/Out Condition block so that it shows greater than ( $>$ ), then recompile, the condition will evaluate true and the A signal will output.

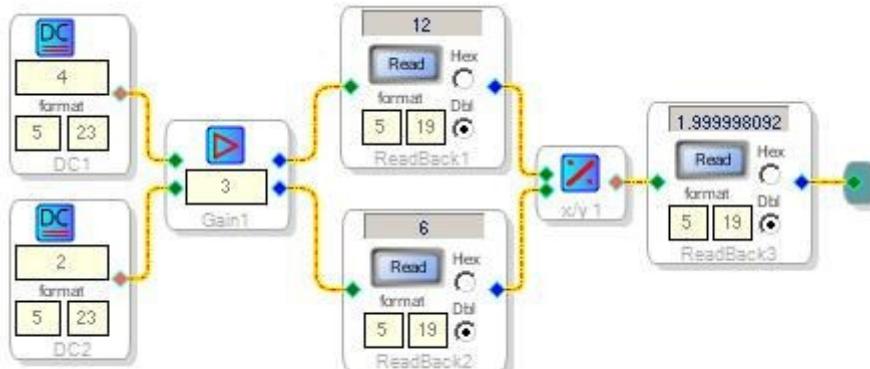
### Example 2:

This schematic shows how the output from the Delay block is split at a T Connection, to be sent to Output and also fed into the Feedback block and then back into the 2nd input of the Multiple Control Mixer.

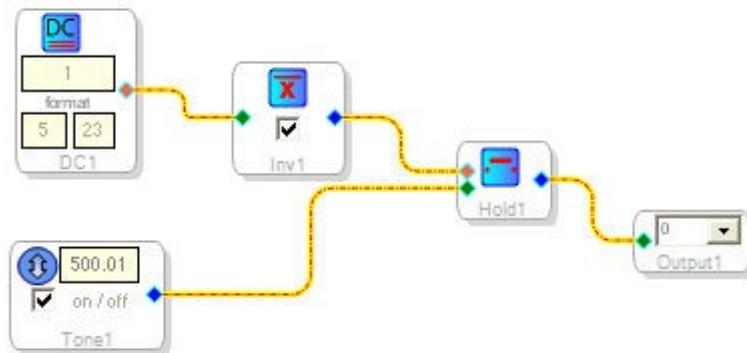


### Example 3:

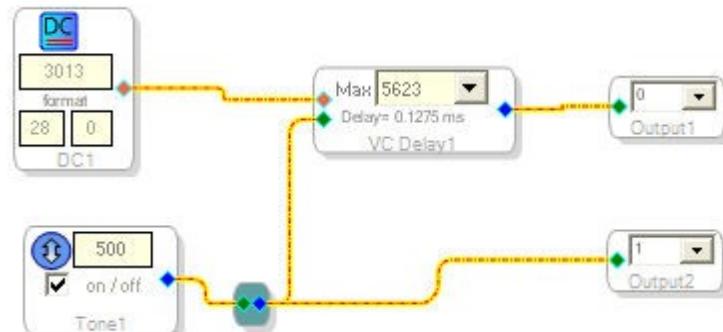
This schematic shows how the Linear Gain block can be used to apply plain boost, how the Divide block can be used to divide one value with another, and how the DSP Readback block reads back the value from the DSP.

**Example 4:**

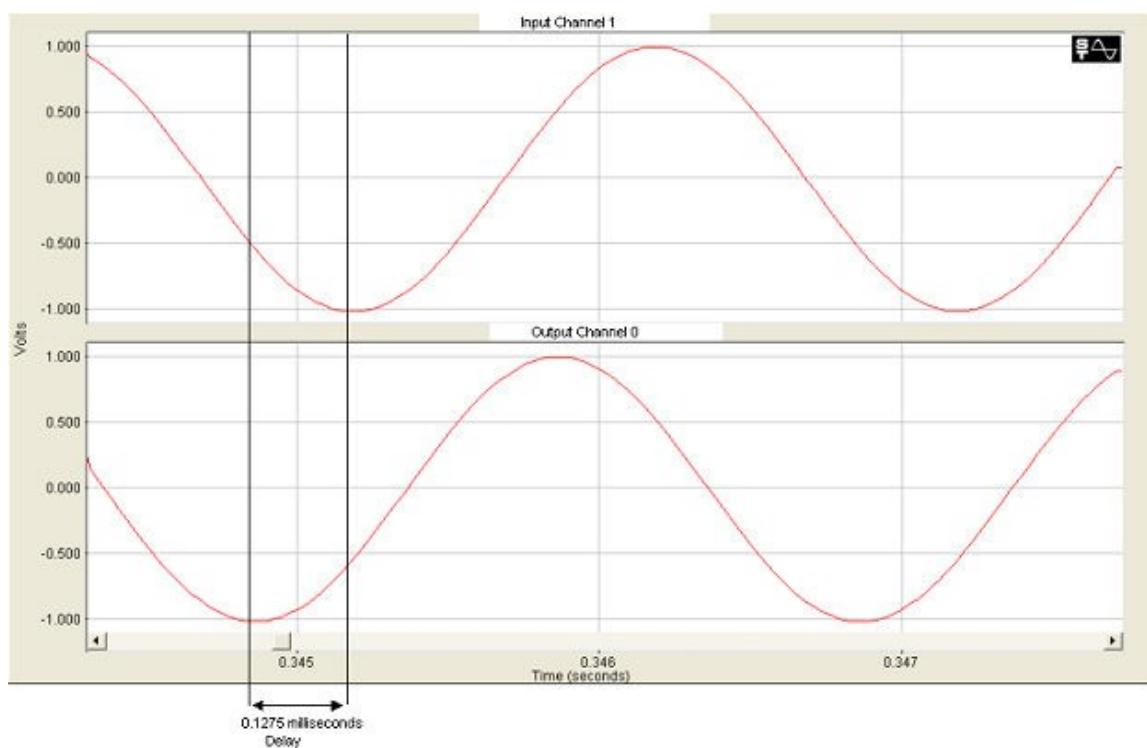
This schematic uses DC Input Entry, Tone (lookup/sine), Signal Invert, Value Hold and Output blocks. Here the Value Hold retains the incoming signal at the green pin as the Control input by the dc-input block at the red pin is inverted.

**Example 5:**

This schematic uses a DC Input block, Tone (lookup/sine), T Connection, Voltage-Controlled Delay, and Output blocks, and explains how it all works with the voltage-controlled delay algorithm. The dc input gives the samples of delay to the input signal when the delay is within the range of the maximum allowed and the output is undefined if it exceeds the maximum allowed or is set to zero. Here the voltage-controlled delay is set for a maximum delay of 5623 samples, which corresponds to 0.1275 milliseconds. The dc input is set for the maximum value of 3013.



The graph below shows the delay as offset between the lowest, -1 point on the input and the same on the output.



© 2006-2007 Analog Devices, Inc. All rights reserved.

## Dynamics Processor Examples

---

**Example 1:**

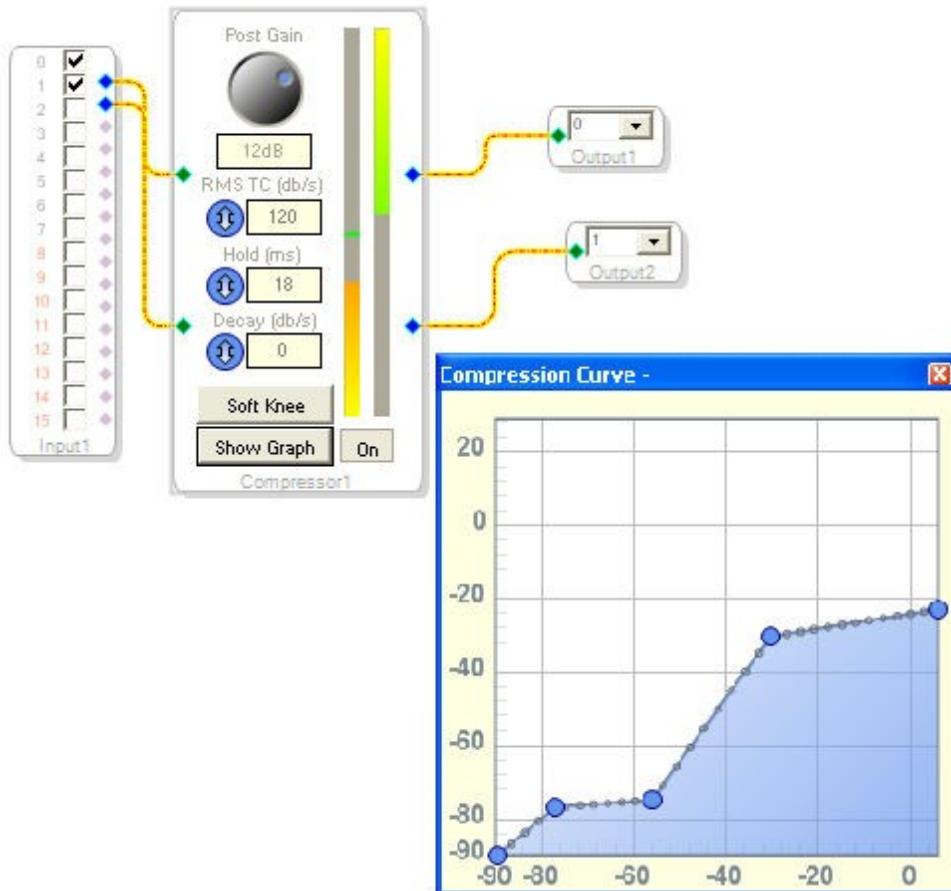
This design uses the RMS Detect (display) block, along with input/output blocks for audio flow. What is important to note about the example shown below are the Compression Curve window; the relative input/output levels on the display bars; and the Post Gain knob.

The threshold (level match point, where no dynamic processing takes place) is set at approximately 35 dB on the horizontal input level axis, as well as the same on the vertical output axis. The compression ratio above threshold has been drawn to approximate 2.7:1.

For all input signals below 35 dB, then, the output level will be unaffected. When the input exceeds this threshold, however, for every 2.7 dB it increases, the output will increase only 1 dB.

The display bars indicate the levels of the input and the compression respectively (greener = more compressed). The input green horizontal bar shows the level of the last value before the current level.

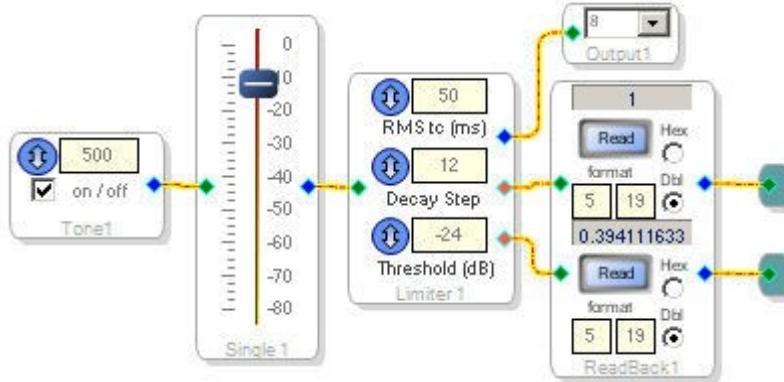
In this case the compression significantly lowered the output level, so to make it up, it was necessary to turn the output up an added 12 dB (Post Gain).



**Note:** While the interface lets you easily create complex dynamics processing curves (as shown above by the behavior below -55 dB), this example deals only with a linear ratio constructed using only one active graph point, the one at -35.

**Example 2:**

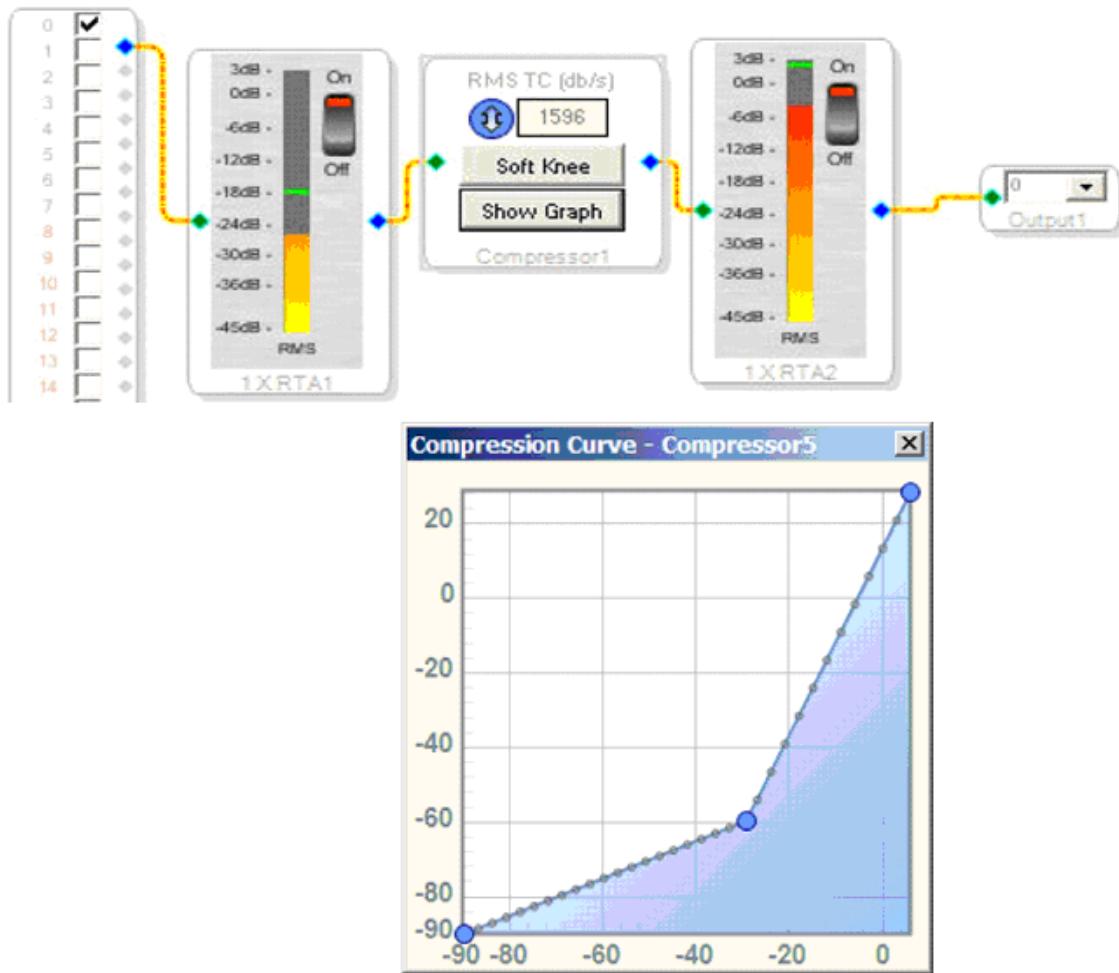
This design uses Tone (lookup/sine), Single Vol (shared), Limiter, DSP Readback, Terminal and output blocks. Here the threshold point for the input signal is set as -24 dB, and when the rms value of the input exceeds it, the gain is reduced per the ratio given on the Limiter help page.



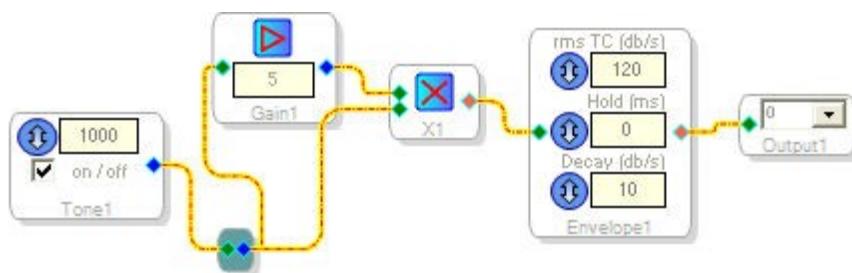
Looking at the Limiter block, the blue pin outputs the dynamically limited signal. The first red pin (middle of three pins) outputs 1 (one), which can be read from the numeric box in the Readback block, indicating that the limiter is currently clamping the input per the instantaneous limiting ratio ( $G_n$ ).

**Example 3:**

The following design uses RMS Detect (no: gain, hold, decay) and Single Level Detector, along with input/output blocks. This schematic shows the increase in the level of the output signal after processing. The threshold is set at -30 dB on the input axis. The output level is unaffected for inputs below -30 dB. However, when the input signal exceeds the threshold, the output increases steeply according to the expansion ratio of the curve shown (approximately 2.5:1 above threshold, which is strong expansion).

**Example 4:**

This schematic uses Tone (lookup/sine), Linear Gain, T Connection, Multiply, Envelope and Output blocks. Here the Multiply block multiplies the output of Linear Gain, which has scaled the 1 kHz tone by 5, and Envelope detects the energy envelope of the Multiply output.



© 2006-2007 Analog Devices, Inc. All rights reserved.

## Filter Examples

### Example 1:

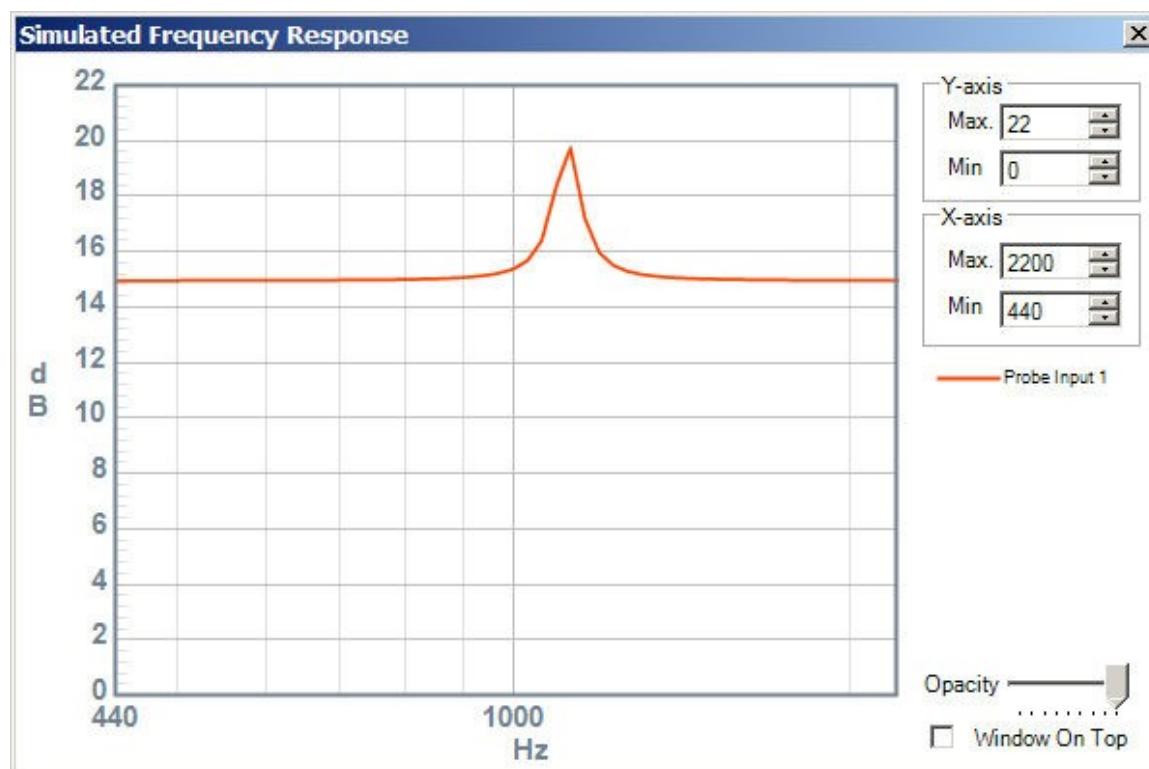
A simple filter circuit:



with these parameters:

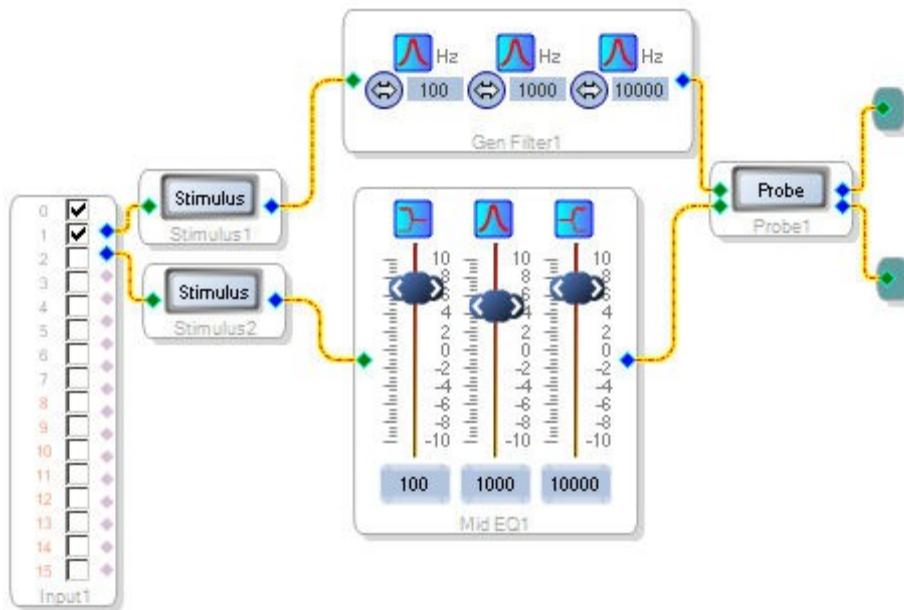


produces this filter frequency response:

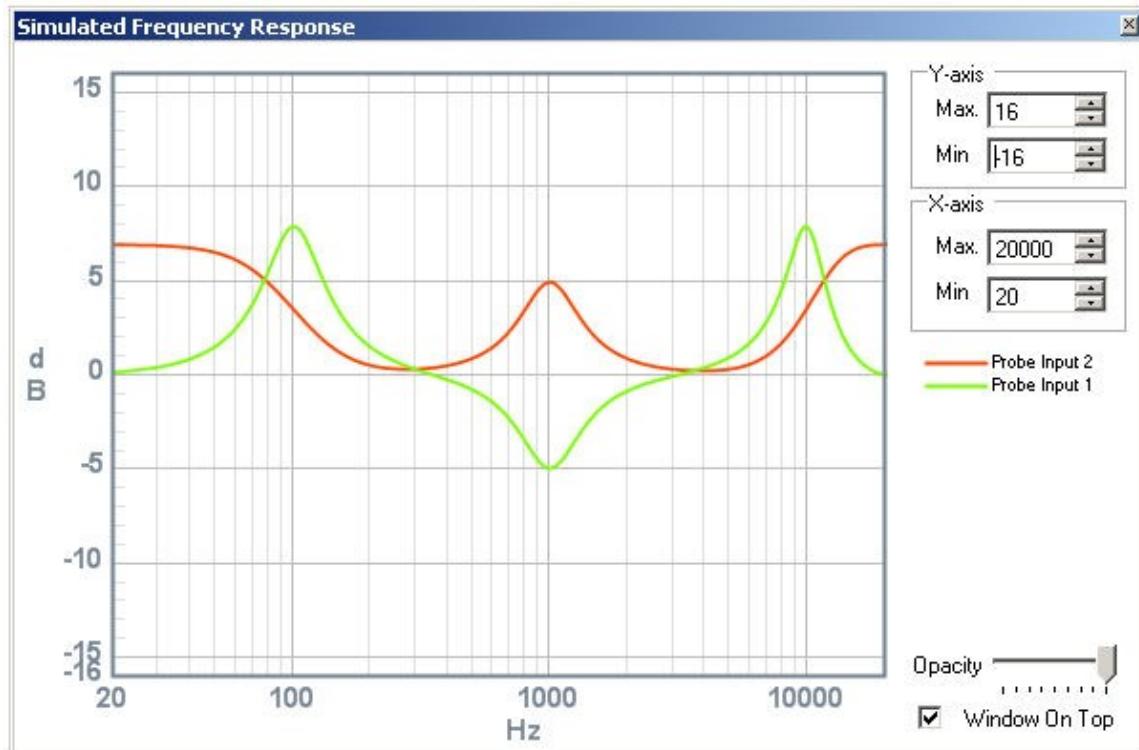


**Example 2:**

This schematic shows a General Purpose (2nd-Order) as well as a Medium-Size EQ, each with a 1-Channel - Double-Precision algorithm grown by 2. The schematic includes an input block and terminals for completion of signal flow.

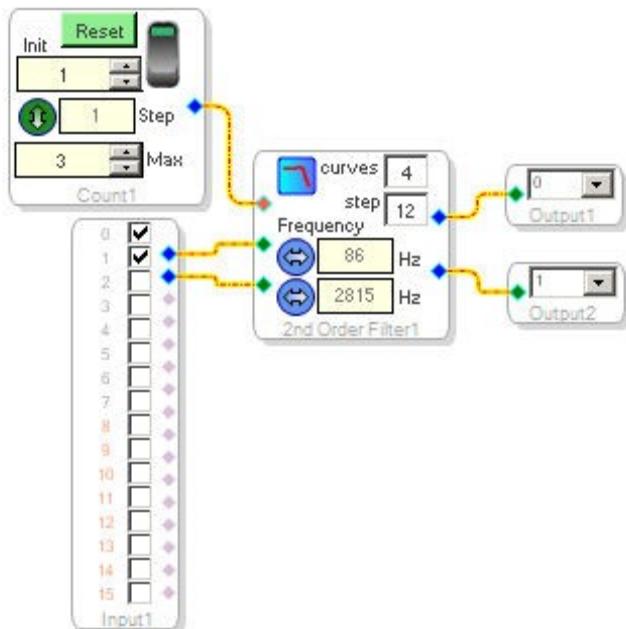


The configuration produces the following frequency responses, with the red line the General Purpose filter and the green line the Medium-Size EQ:

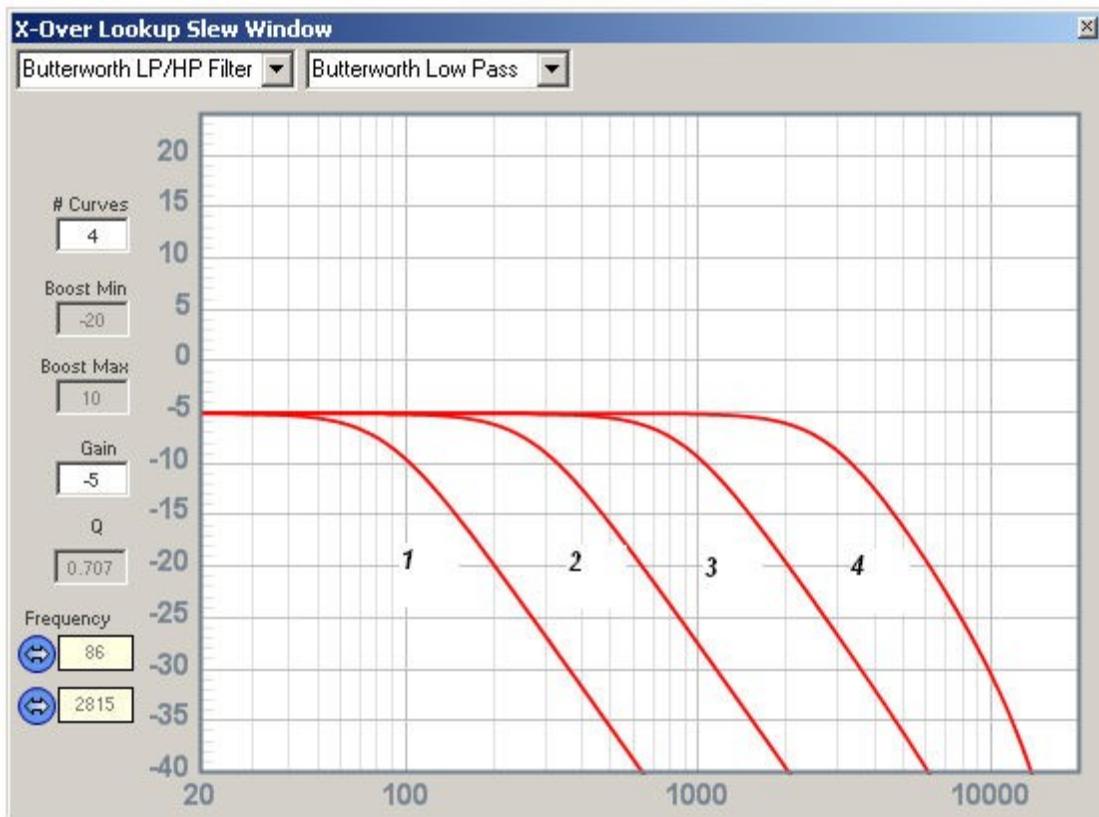


**Example 3:**

This schematic uses the General (2nd-Order / Lookup / Slew) block, a counter, an input and two output blocks.

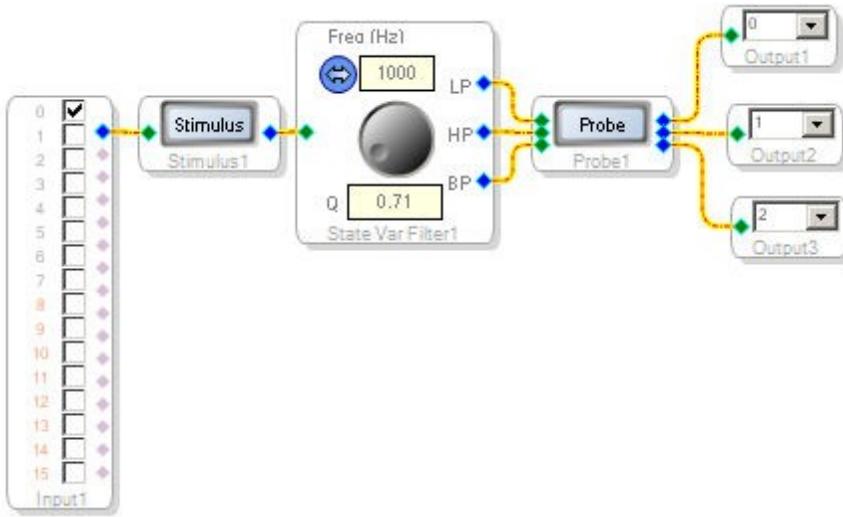


A Butterworth lowpass filter with 4 curves is selected. For them the index range possible is 0 - 3, and curve 2 uses index 1.

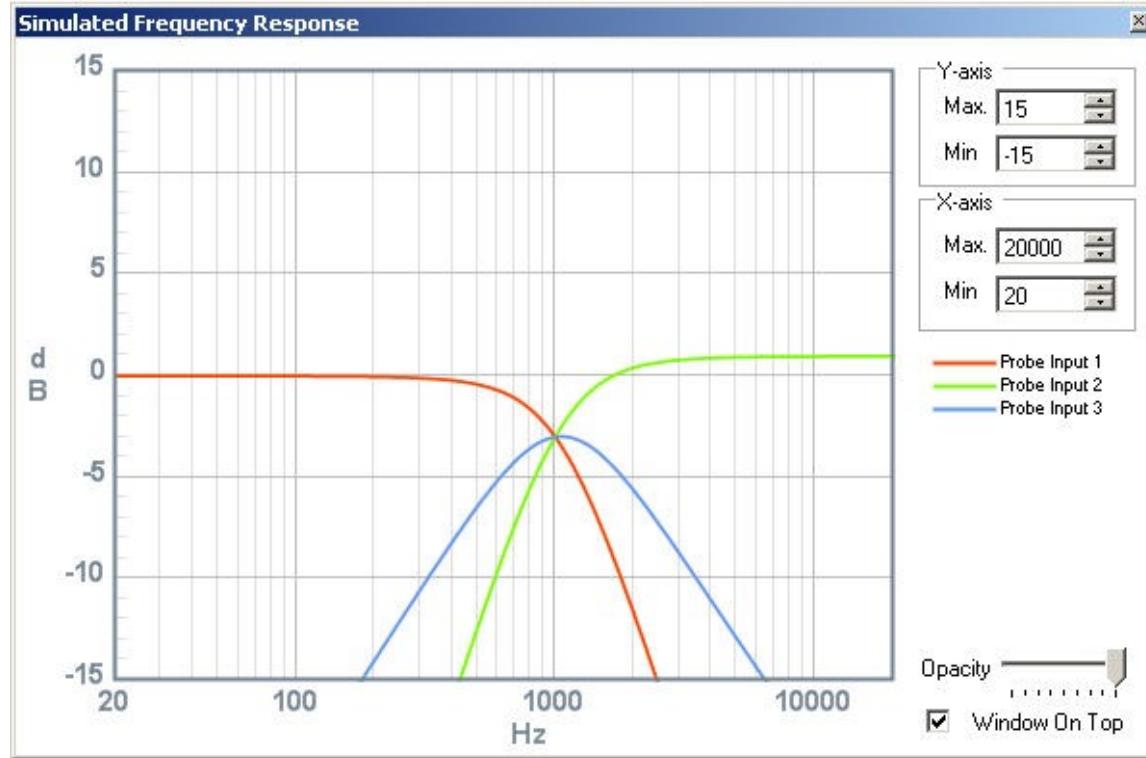


**Example 4:**

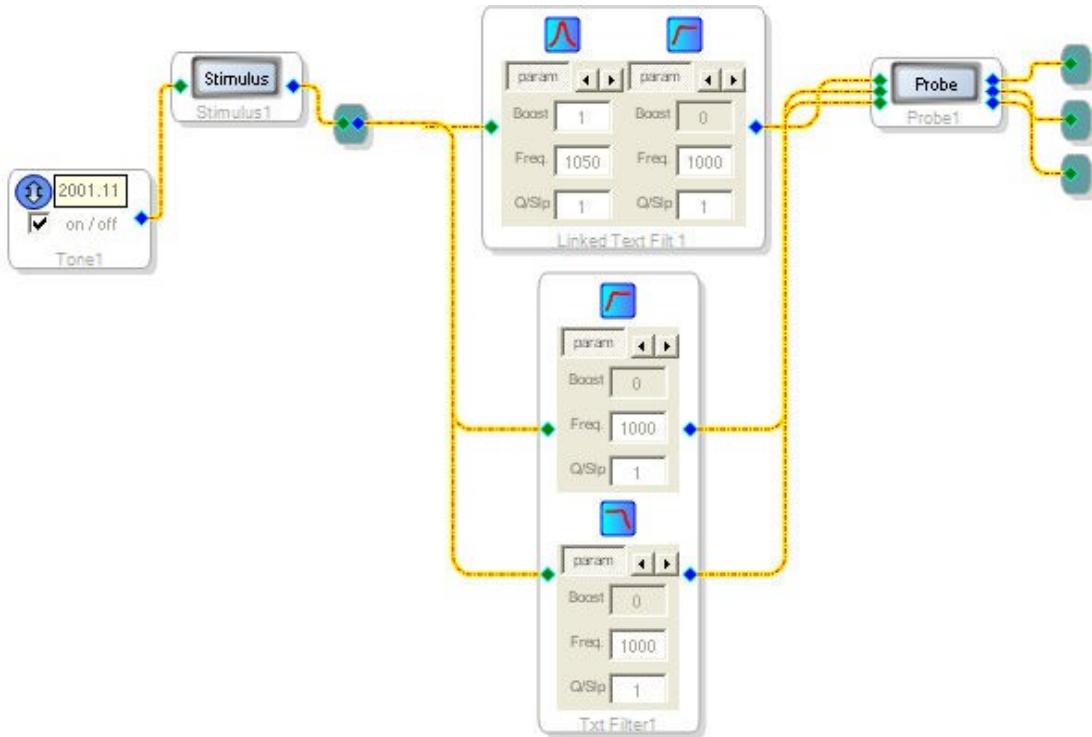
This schematic uses the State-Variable block, with a Stimulus, Probe, input and three output blocks.



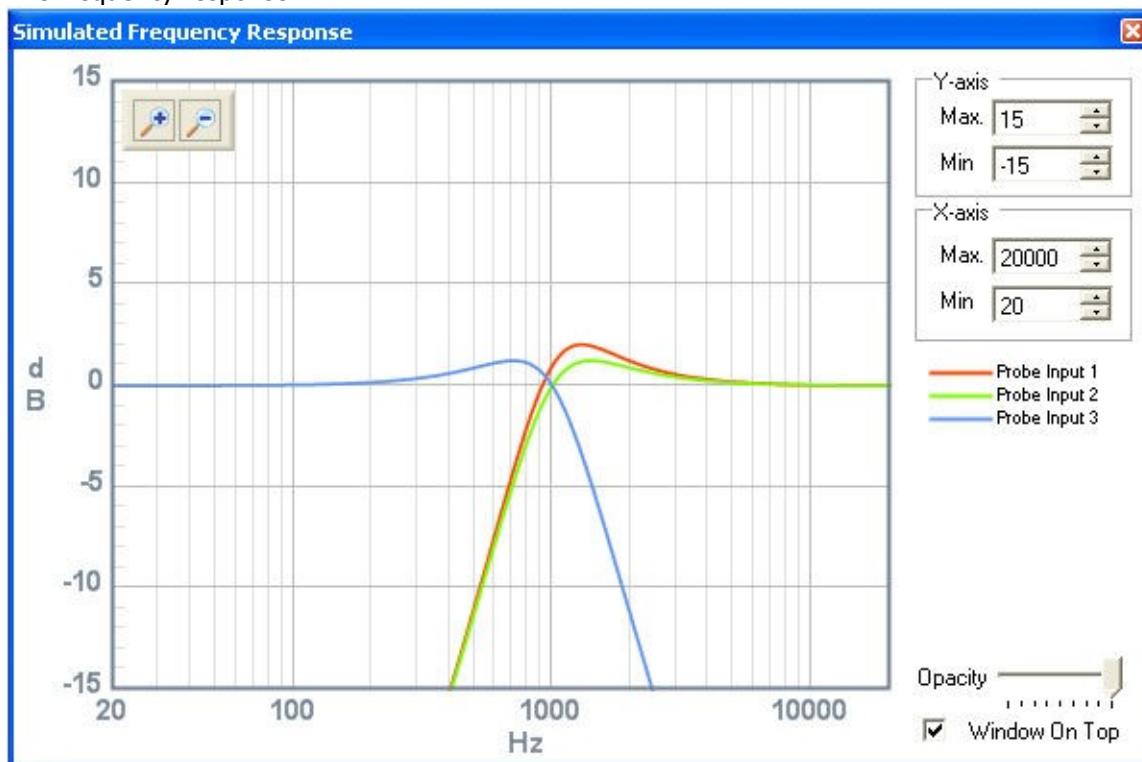
The frequency response produced, showing the three LP, HP, and BP filters, is:

**Example 5:**

This schematic uses the Text-In Filter (linked) and Text-In Filter (Unlinked) with probe and stimuli blocks. Linked has 1-ch - double-precision grown by 2, which behaves as a series filter, while Unlinked has 1-ch - double-precision added by 2, which behaves as a parallel filter. Tone (lookup/sine) is the input; T connection and terminals complete the signal flow.



The frequency response:



© 2006-2007 Analog Devices, Inc. All rights reserved.

## GPIO Conditioning

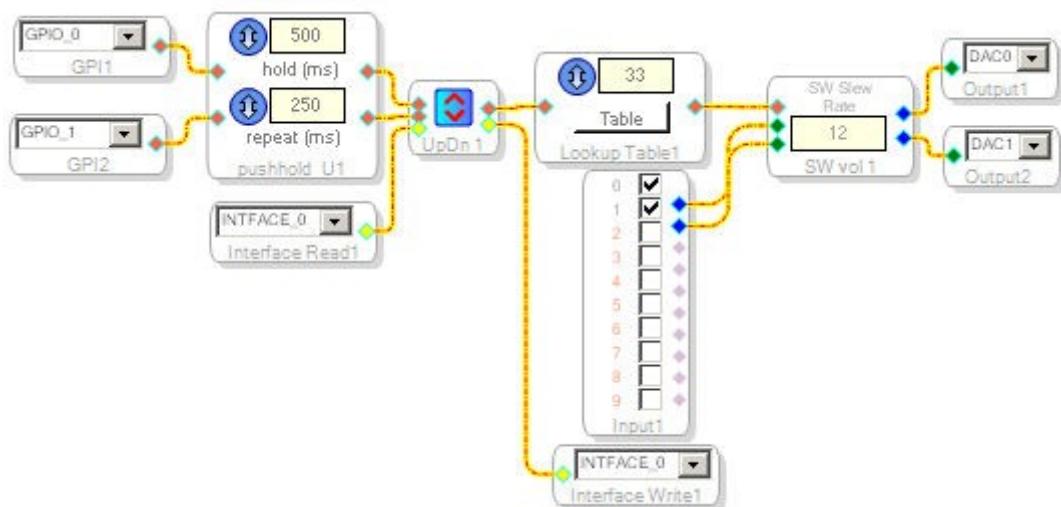
---

These examples all use the 1701 DSP evaluation board, which supports two analog and seven digital inputs.

**Example 1:**

Blocks: Up/Down Control, Index Lookup Table, Push and Hold

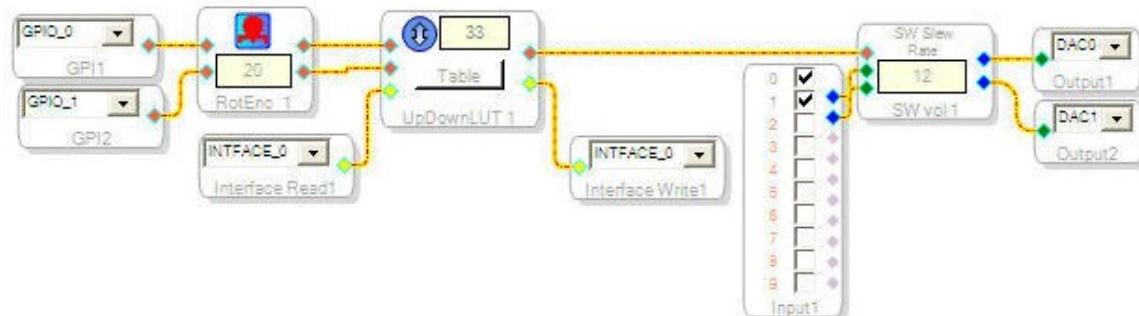
Two General Purpose input blocks drive the push and hold block. Its outputs are used to select the index for the Index Lookup Table. The Up/Down block increments or decrements the index, based level. The Interface Read block connects to the light-green pin of Up/Down Control block, which drives both the index lookup table and the Interface Write blocks. The output of the Lookup Table drives the red pin (used for external connection) of Single Slew Ext(ernal) Vol(ume). Then the signals are given to two DACs — which are nothing but the analog outputs of the board.



**Example 2:**

Blocks: Rotary Encoder and Up/Down Control w/ Lookup Table (Up/Down + INTF)

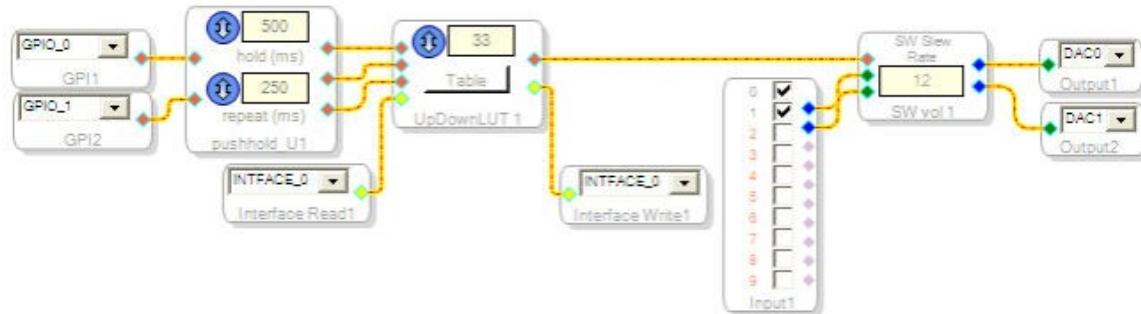
Two General Purpose input blocks drive the Rotary Encoder block. Its outputs are used to select the index for the Up/Down Control w/ Lookup Table. The Interface Read block connects to the light-green pin of Up/Down Control w/ Lookup Table block, which drives both the Single Slew Ext Vol and Interface Write blocks. Then the signals are given to two DACs, which are the analog outputs of the board.



**Example 3:**

### Blocks: Push and Hold, Up/Down Control w/ Lookup Table (Up/Down + INTF)

Two General Purpose input blocks drive the Push and Hold block. Its three outputs are used to select the index for the Up/Down Control w/ Lookup Table. The Interface Read block connects to the light-green pin of Up/Down Control w/ Lookup Table block, which drives both the Single Slew Ext Vol and Interface Write blocks. Then the signals are given to two DACs, which are the analog outputs of the board.



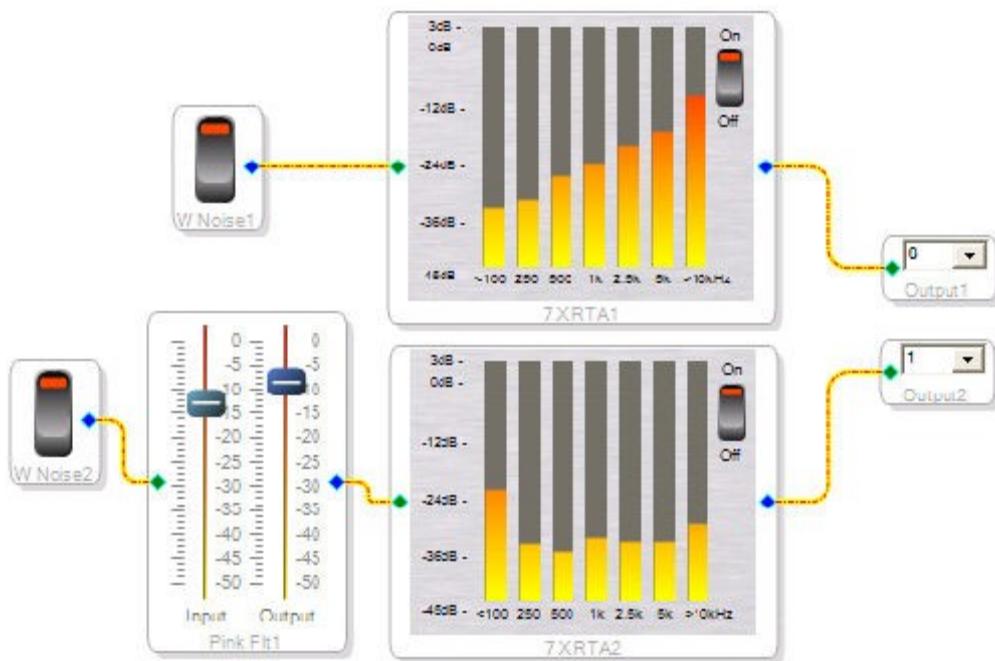
© 2006-2007 Analog Devices, Inc. All rights reserved.

## Level Detectors Example

### Example 1:

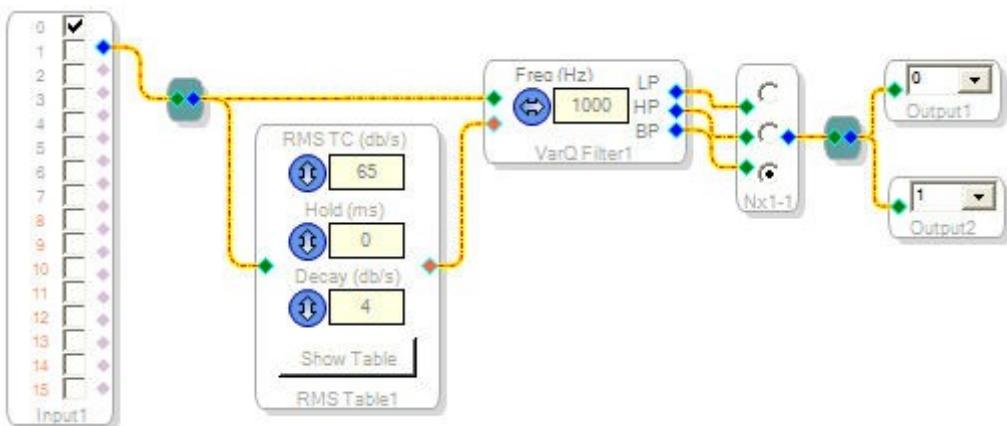
Two Seven-Band Level Detectors show the difference between white and pink noise. They were created using [White Noise Source](#) and [Pinking Filter](#) blocks. For the difference between these two noise sources, refer to the [Pink Filter page](#).

There are two outputs to complete the signal flow. Notice in the level detector displays that the pink noise levels tend to be constant while the white noise level fluctuate.



### Example 2:

This example design shows the State-Variable Filter (Q Input), RMS Table, Mono Nx1 Switch, T Connection, and Input/Output blocks. The Q of the state-variable filter is determined by the RMS table and the Mono Nx1 switch selects the output response: lowpass, highpass, or bandpass.



The RMS Table Editor contains the interpolated values, as the input level increases the filter Q increases:

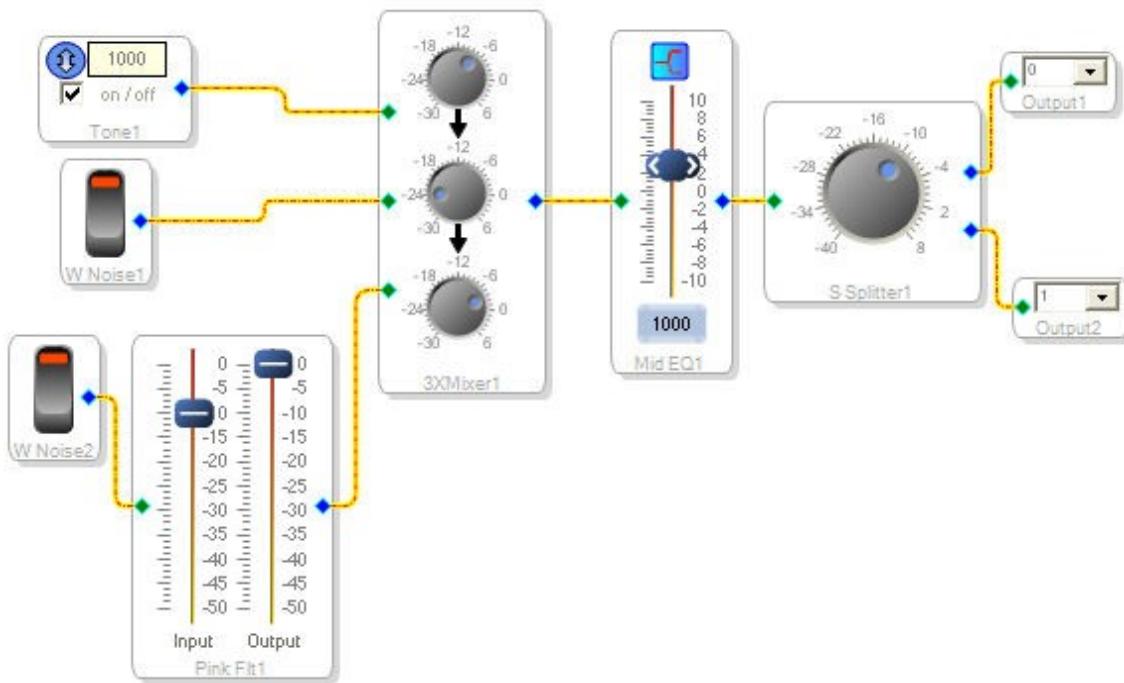
	RMS Table Editor - RMS Table1
1	0.01
2	0.02
3	0.03
4	0.04
5	0.05
6	0.06
7	0.07
8	0.08
9	0.09
10	0.1
11	0.11
12	0.12
13	.13
14	.14
15	.15
16	.16
17	.17
18	.18
19	.19
20	.20
21	.22
22	.23
23	.24
24	.25
25	.26
26	.27
27	.28
28	.29
29	.30
30	.31
31	.32
32	.33
33	

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Mixers/Splitters Example

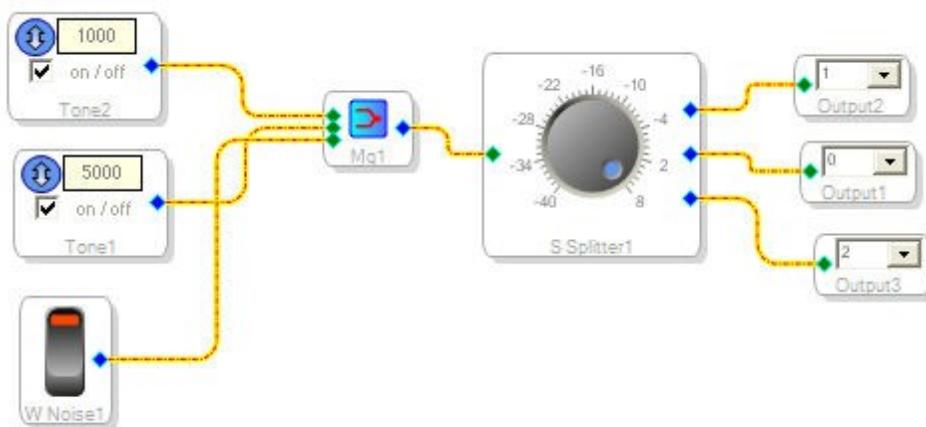
### Example 1:

This schematic shows a straightforward way of using a Three-Input Cross-Mixer to combine three sources — two White Noise (one passed through a Pinking Filter) and a tone source — and run their mixed output through a Mid EQ and finally through a Single-Control Splitter to produce two (same-gain) Outputs.



### Example 2:

This schematic shows a straightforward way of using a Signal Merger grown by 1 to combine two Tone and one White Noise sources and feed a Single-Control Splitter grown by 1, producing three Output blocks, their gain set by the splitter knob setting plus parameter values.



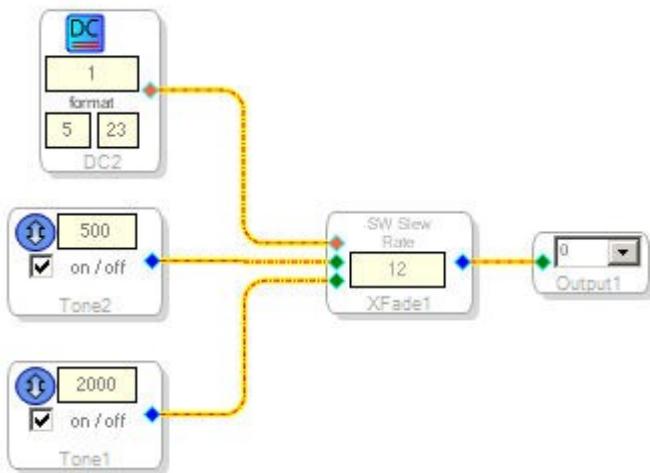
© 2006-2007 Analog Devices, Inc. All rights reserved.

## **Multiplexer/Demultiplexer Examples**

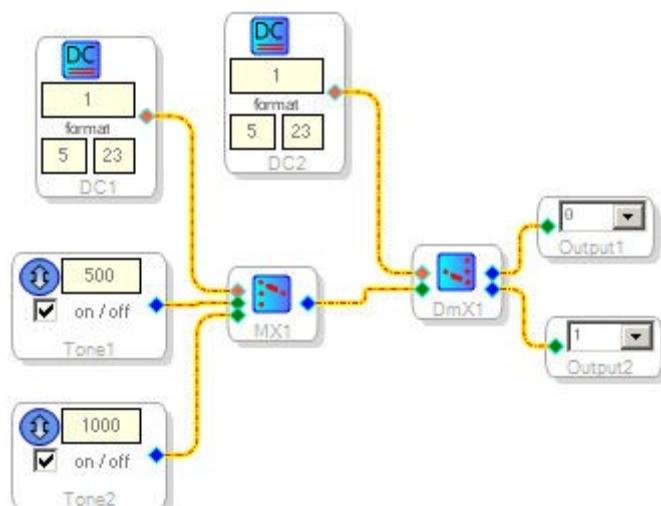
---

**Example 1:****Crossfade (Data-Controlled)**

This schematic shows a simple way of smoothly switching (crossfading) from one signal to the other (here 500 Hz and 2kHz). It uses these blocks: DC Input control entry, two Tone (Lookup/Sine), Crossfade (Data-Controlled), and Output.

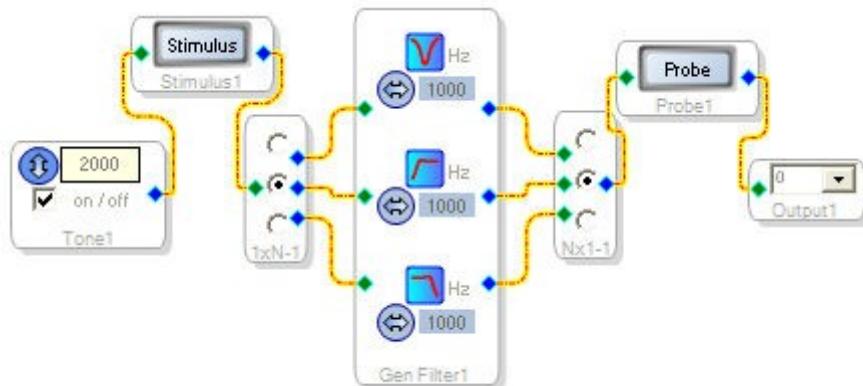
**Example 2:****Index-Selectable Mux/DeMux**

This schematic shows a simple way of multiplexing two sources and then demultiplexing the signals to two outputs. Two Tone (Lookup/Sine) blocks feed an Index-Selectable Multiplexer, with the switching between them controlled by a DC Input. The output goes to an Index-Selectable Demultiplexer, whose behavior is controlled by a second DC Input Entry to feed two Outputs.

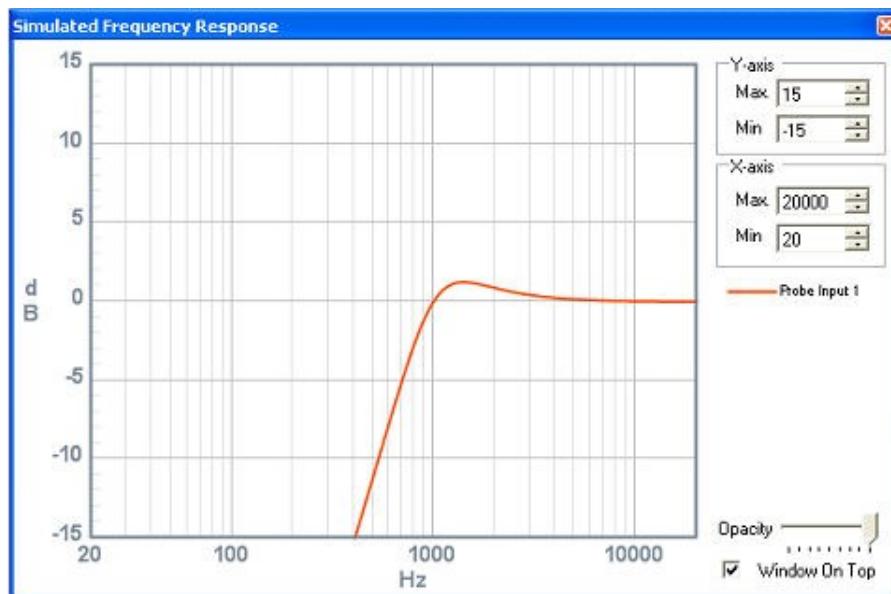
**Example 3:****Mono Switch**

This schematic uses Mono Switch 1xN and Mono Switch Nx1 flanking a General (2nd-Order) Filter grown by 2. The filter frequency response choice is highpass and the highpassed

signal is routed to the output. Tone (Lookup/Sine), Probe, Stimuli and Output blocks complete the signal flow.

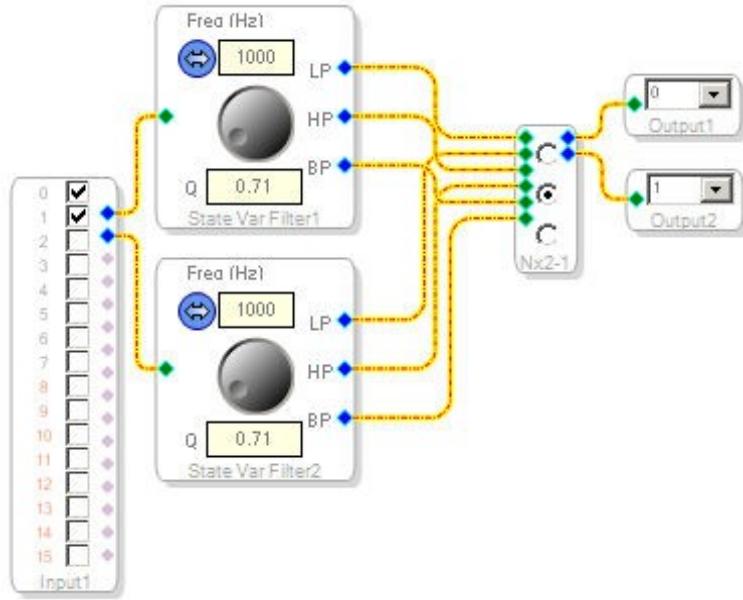


Below is the frequency response of the HP filter.



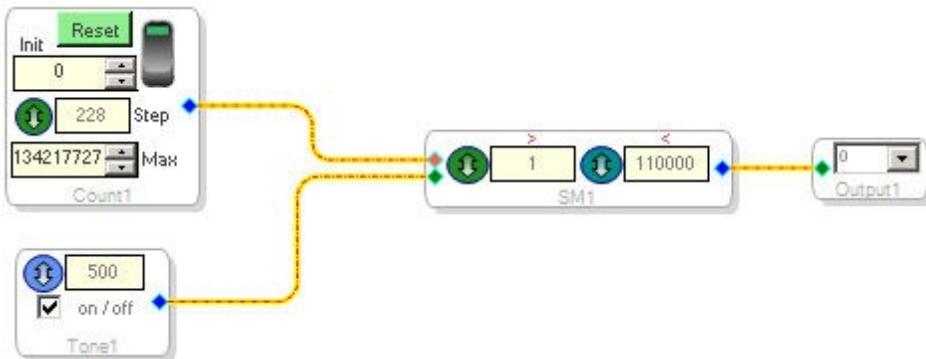
#### Example 4: Stereo Switch

This schematic shows a simple way of multiplexing two sources using two State-Variable filters, Stereo Switch Nx2, Input and Output blocks. The filters' high-pass outputs are selected and routed to the output blocks.



### Example 5: State Machine

This schematic uses a Counter as the control, a Tone (Lookup/Sine) and a State Machine block, in order to show how the tone passed through to the output gets blocked once the count goes beyond the specified range.

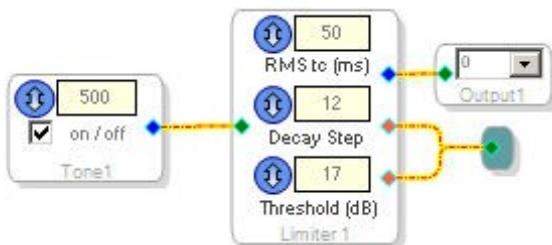


© 2006-2007 Analog Devices, Inc. All rights reserved.

## System Example

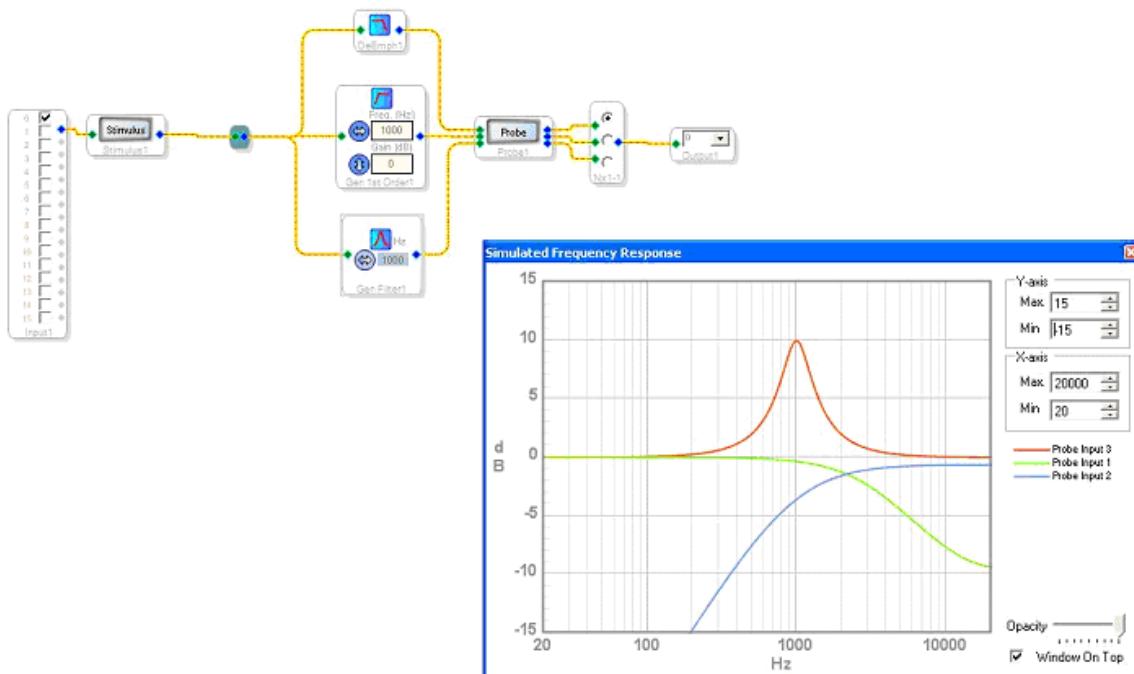
### Example 1:

The following schematic uses Tone (lookup/sine), Limiter, Output, and Schematic Terminal blocks. The red pins, which are unnecessary to the application, are sinked using the terminal block.



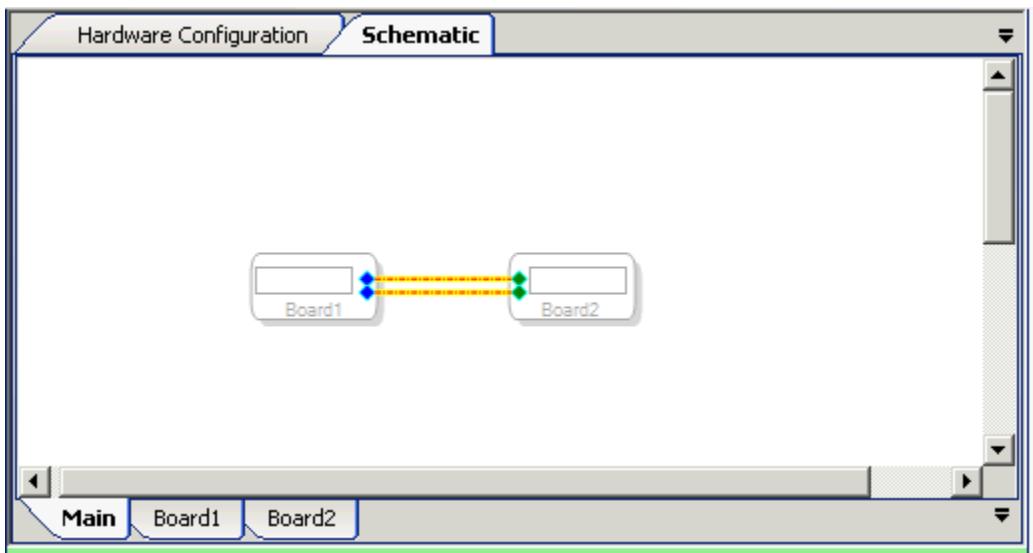
### Example 2:

The following example uses Simulation Stimuli, Simulation Probe, T Connection and Schematic Terminal to produce the frequency response of General (1st-Order), General (2nd-Order), and Deemphasis filters set at their default values. (Note that you have to press the **Stimuli** button to see the graphs.)

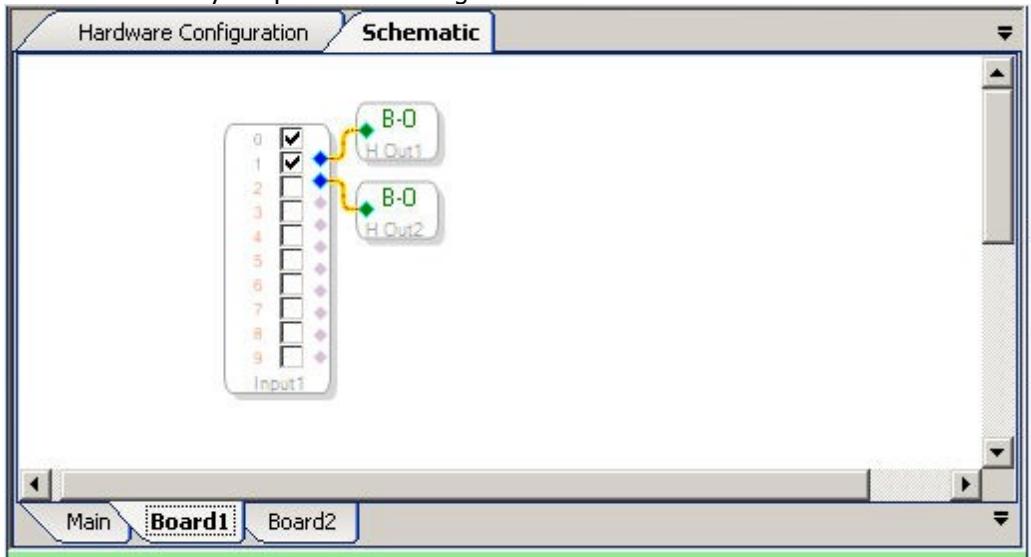


### Example 3:

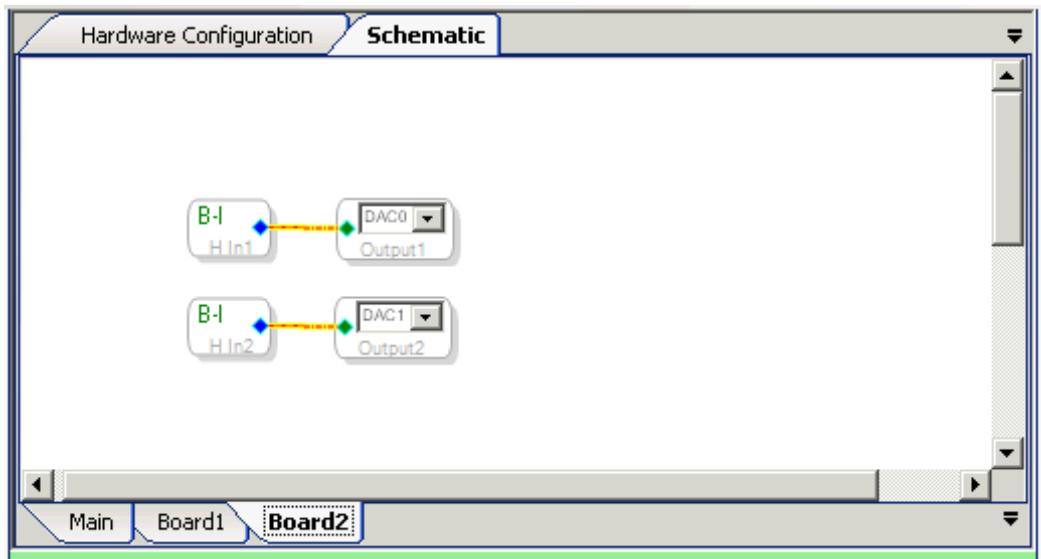
This simple example shows how the Hierarchy Board creates a new workspace page you can use for organizing your schematic and program flow.



How the Hierarchy Output flows the signal out of Board 1:



How the Hierarchy Input accepts the signal into Board 2 from 1:



---

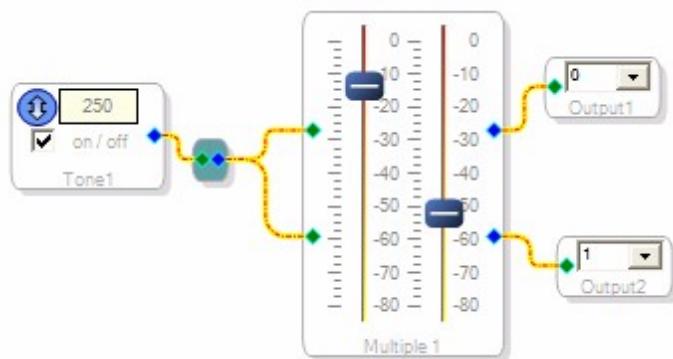
© 2006-2007 Analog Devices, Inc. All rights reserved.

## Volume Control Example

---

This example shows a Tone source feeding a TConnection and a Multiple Volume Control (one algorithm added), and finally two Outputs.

You can pan using level differences between the two outputs; the schematic shows slider positions that shift the tone from the center toward the left.



---

© 2006-2007 Analog Devices, Inc. All rights reserved.

# Algorithm Information

## Algorithm Information

These topic pages contain technical details for SigmaStudio algorithms. See the ToolBox for more information about the schematic blocks.

- AD1940/AD1941 Volume Controls
- AD1953/AD1954 Volume Controls
- Algorithm Information
- Allpass Filters
- Deemphasis Filter
- Dynamics Processors
- EQ
- FIR filter
- General 1st-Order Filters
- General 2nd-Order Filters
- Level-Detector Algorithm
- Pink Filter
- State-Variable Filters

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

**Navigation:** Algorithm Information >  
**AD1940/AD1941 Volume Controls**



## AD1940/AD1941 Volume Controls

---

The AD1940 / AD1941 volume controls use target/slew-RAM memory space to auto-ramp from one value to a desired final value. With the Adjustable Volume Control, you control ramping type and speed; with the other blocks the slew is constant dB and the time constant is 8.

### Time Constant

The time constant is a 4-bit value, ranging from 0 to 15 (fastest to slowest), and controls the ramping speed.

The three ramping curves are:

- **Linear** slews to target using a fixed step size.
- **Constant dB** slews to target using the current value to calculate the step size. The resulting curve has a constant rise and decay when measured in dB.
- **RC-type** slews to target using the difference between the target and current values to calculate the step size. This produces a simple RC-type curve for both rising and falling.

### Linear Update:

```
Step = 213 / 10 [ 2 * (tconst - 5) ] / 20
```

The result of this equation is normalized to the 5.23 data format. This gives a TC range from 6.75 ms to 213.4 ms (-60 dB relative to 0 dBFS).

### Constant dB and RC-type (Exponential):

Exponential slew is accomplished by shifts and adds with a range from 6.1 ms to 1.27 s (-60dB relative to full scale). When the ramp type is set to Constant dB, each step size is set to the current value in the slew data. When the ramp type is set to RC-type, the step sizes are equal to the difference between the values in the target RAM and slew RAM.

### Constant Time:

Constant time slew is accomplished by adding a step value calculated after each new target gain is loaded. The equation for this step size is:

```
Step = (target data - slew data) / number of steps  
Number of steps = 2tconst + 6
```

Number of steps ranges from 64 (tconst = 0) to 8196 (tconst = 7)

## AD1953/AD1954 Volume Controls

---

The AD1953 and AD1954 processors include hardware volume control.

- AD1953: 8 available volume controls
- AD1954: 3 available volume controls

The AD1953 and AD1954 include eight SPI registers for controlling gain, which include automatic-rampup circuitry for clickless adjustment. Gain values range from +2.0 to -2.0; the default is 1.0.

It takes 1024 frames to adjust the level from 2.0 down to 0 (zero), and in the normal case of max volume set to 1.0, it will take 512 audio frames to reach zero.

Mute is the same as setting the volume to zero, except that when the chip is un-muted the volume jumps back to its original value.

Volume ramp times assume that the AD1953 is set for the fast ramp speed. If you select the slow setting, it takes 9192 audio frames to reach zero from a setting of 2.0 and 4096 frames from the normal setting of 1.0. Note that the Single and Multiple Volume Controls are set permanently to the fast ramp speed.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Allpass Filters

---

The General (1st-Order) and General (2nd-Order) All-pass filter equations are shown here.

### Common variables:

```
omega0 = 2*pi*f0/Fs
alpha = sin(omega0)/(2 * Q)
gainLinear = 10^(gain/20)
```

### Allpass First-Order

Transfer Function  
(Laplace) 
$$H(s) = \frac{s - 1}{s + 1}$$

Transfer Function (z-  
domain) 
$$H(z) = \frac{b_0 + b_1 z^{-1}}{1 + a_1 z^{-1}}$$

Coefficients   
`a1 = 2.7^-omega0  
b0 = -gainLinear * a1  
b1 = gainLinear`

### Allpass Second-Order

Transfer Function  
(Laplace) 
$$H(s) = \frac{\frac{s^2 - \frac{s}{Q} + 1}{Q}}{\frac{s^2 + \frac{s}{Q} + 1}{Q}}$$

Transfer Function (z-  
domain) 
$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}$$

Coefficients   
`a0 = 1 + alpha  
a1 = -2 * cos(omega0) / a0  
a2 = (1 - alpha) / a0  
b0 = (1 - alpha) / a0 *  
gainLinear  
b1 = (-2 * cos(omega0)) / a0 *  
gainLinear  
b2 = (1 + alpha) / a0 *  
gainLinear`

## Deemphasis Filter

The Deemphasis Filter exactly attenuates the high frequencies boosted by pre-emphasis, as is sometimes used in audio recording. This process improves the signal-to-noise ratio.

### Common variables:

sampling frequency:  $F_s$   
de-emphasis factor:  $a = \exp(-2 * \pi * f / F_s)$

The algorithm is then computed recursively by:

```
Y1 = x1
Yi = xi + a * Yi-1
```

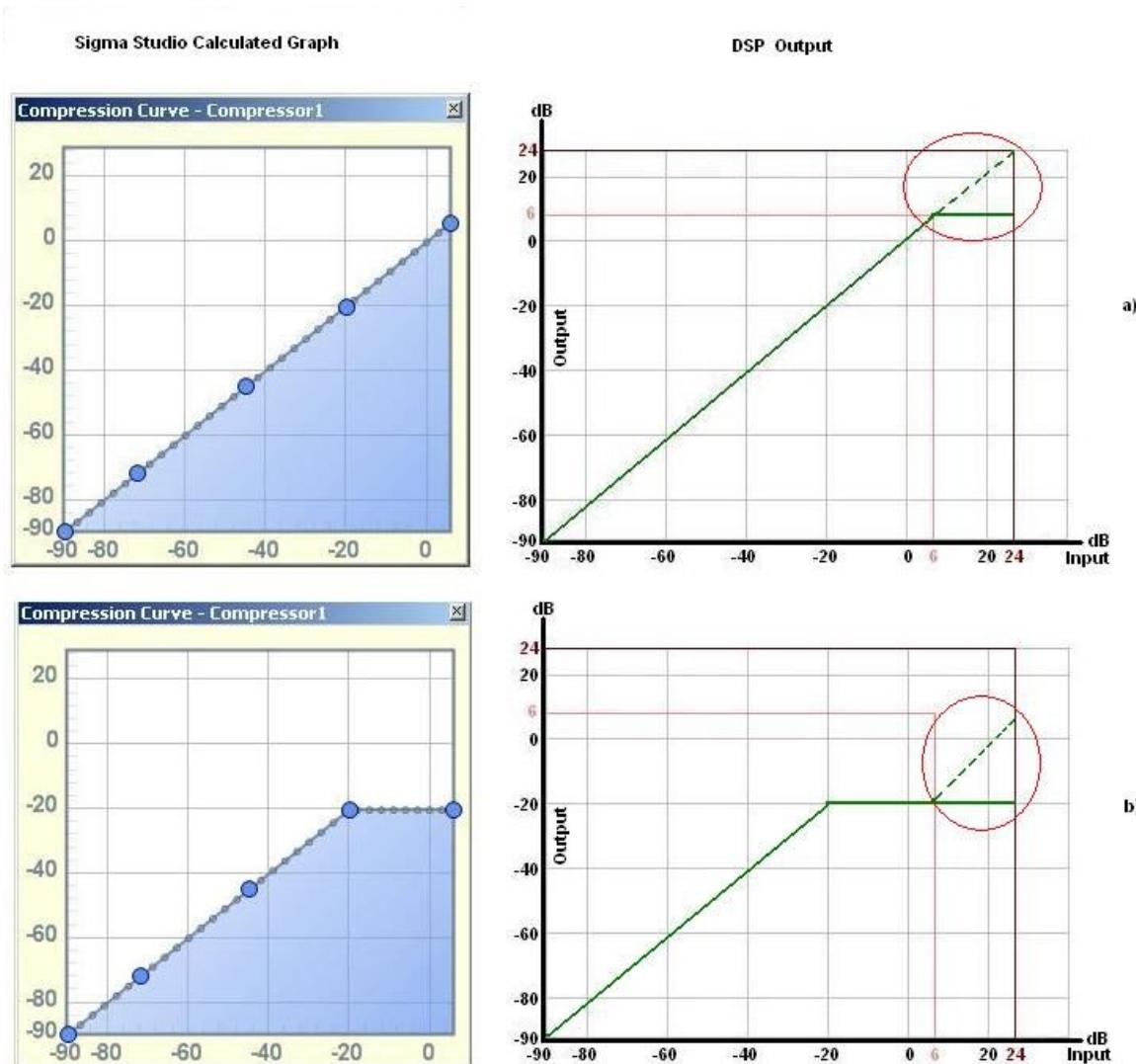
© 2006-2007 Analog Devices, Inc. All rights reserved.

## Dynamics Processors

The standard compressor algorithms supports input signal up to +6dB in level, for any signal exceeding 6dB the compressor will revert a linear compression ratio. If you have a compression curve mapped out up to 6dB, it will be ignored if the input signal exceeds the 6dB input mark and will return to a 1:1 ratio. Note this is not relevant to the Limiter block which supports internal gains higher than 6dB and attenuates the signal as expected.

### Full Range Dynamic Processor (Stereo / Mono):

The RMS (no gain) compressor includes "Full Range" algorithms that do not have this +6dB limitation. The Full Range Dynamic Processor will maintain a maximum output level of +6 dB if the input value exceeds +6 dB, giving a fixed output response between +6 dB and +24 dB equal to the last compression curve data point at the +6 dB value.



In figure (a), the standard dynamic processor algorithm, the maximum valid input value is +6 dB, the output value will be clipped at +6 dB for any inputs above +6dB in the range of +6 dB to +24

dB, regardless of the compression settings.

For the Full Range algorithm, figure (b), input signals above +6dB in the range +6 dB to +24 dB are properly compressed with a compression ratio equal to that specified in the graph control for +6dB input (-20dB in this example).

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## EQ Algorithm

---

The 2nd Order and medium-size EQ blocks use multiple biquad filters, based on Robert Bristow-Johnson's work as shown here:

### Common Variables

```
A = 10^(boost/40)
ω₀ = 2*pi*f₀/Fs
alpha = sin(ω₀)/(2*Q) (type: Peaking)
        = sin(ω₀)/2 * sqrt((A + 1/A)*(1/S - 1) + 2) (type: Shelving)
gainLinear = 10^(gain/20)
```

### Peaking

Transfer function

$$H(s) = \frac{s^2 + \frac{A}{Q}s + 1}{s^2 + \frac{s}{A*Q} + 1}$$

Coefficients

```
a₀ = 1 + alpha/A
a₁ = -2 * cos(ω₀)
a₂ = 1 - alpha/A
b₀ = (1 + alpha*A) *
      gainLinear
b₁ = -(2 * cos(ω₀)) *
      gainLinear
b₂ = (1 - alpha*A) *
      gainLinear
```

### Low-Shelf

Transfer function

$$H(s) = A * \frac{s^2 + \frac{\sqrt{A}}{Q}s + A}{As^2 + \frac{\sqrt{A}}{Q}s + 1}$$

Coefficients

```
a₀ = (A+1) + (A-1)*cos(ω₀) + 2*sqrt(A)*alpha
a₁ = -2*( (A-1) + (A+1)*cos(ω₀) )
a₂ = (A+1) + (A-1)*cos(ω₀) - 2*sqrt(A)*alpha
b₀ = A*( (A+1) - (A-1)*cos(ω₀) +
          2*sqrt(A)*alpha ) * gainLinear
b₁ = 2*A*( (A-1) - (A+1)*cos(ω₀) ) * gainLinear
b₂ = A*( (A+1) - (A-1)*cos(ω₀) -
          2*sqrt(A)*alpha ) * gainLinear
```

### High-Shelf

Transfer  
function

$$H(s) = A * \frac{As^2 + \sqrt{A}s + 1}{s^2 + \frac{\sqrt{A}}{Q}s + A}$$

Coefficients

```
a0 = (A+1) - (A-1)*cos(omega0) + 2*sqrt(A)*alpha
a1 = 2*( (A-1) - (A+1)*cos(omega0) )
a2 = (A+1) - (A-1)*cos(omega0) - 2*sqrt(A)*alpha
b0 = A*( (A+1) + (A-1)*cos(omega0) + 2*sqrt(A)*alpha ) *
      gainLinear
b1 = -2*A*( (A-1) + (A+1)*cos(omega0) ) * gainLinear
b2 = A*( (A+1) + (A-1)*cos(omega0) - 2*sqrt(A)*alpha ) *
      gainLinear
```

For each of these three filters, all the coefficients are divided by a0, normalizing them and making a0 = 1, so that only 5 coefficients must be stored.

In the implementation on the DSP, when the coefficients are stored in parameter RAM, a1 and a2 need to be inverted. This is done in software before the parameters are written to memory.

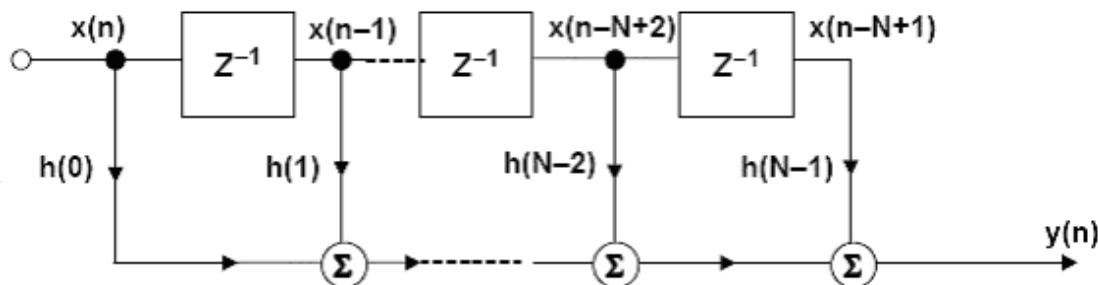
## FIR filter

An FIR filter must perform the following convolution equation:

$$y(n) = h(k)*x(n) = \sum_{k=0}^{N-1} h(k)x(n-k)$$

where  $h(k)$  is the filter coefficient array and  $x(n-k)$  is the input data array to the filter.  $N$  is the number of taps of the filter and relates to filter performance; the generalized form of an  $N$ -tap FIR filter is shown below:

### N-TAP FINITE IMPULSE RESPONSE (FIR) FILTER



■  $y(n) = h(n) * x(n) = \sum_{k=0}^{N-1} h(k) x(n-k)$

■  $*$  = Symbol for Convolution

■ Requires  $N$  multiply-accumulates for each output

The key theorem of FIR filter design is that the coefficients  $h(n)$  are simply the quantized values of the impulse response of the frequency transfer function  $H(f)$ .

The essence of FIR filter design is selection of the filter coefficients and number of taps to realize the desired transfer function  $H(f)$ . Various algorithms are available to translate the response  $H(f)$  into a set of FIR coefficients, and many commercial PC software implementations are available. Rolloff can be sharpened by adding more taps (stages) and stopband attenuation improved by properly selecting filter coefficients.

## General 1st-Order Filters

---

$$H(z) = \frac{b_0 + b_1 z^{-1}}{1 + a_1 z^{-1}}$$

The General 1st-Order block has two algorithms options.

### Common variables:

```
ω₀ = 2*pi*f₀/Fs
gainLinear = 10^(gain/20)
```

### Lowpass

Transfer Function

$$H(s) = \frac{1}{s^2 + \frac{s}{Q} + 1}$$

Coefficients

```
a₁ = 2.7^⁻ω₀
b₀ = gainLinear * (1.0 - a₁)
b₁ = 0
```

### Highpass

Transfer Function

$$H(s) = \frac{1}{s^2 + \frac{s}{Q} + 1}$$

Coefficients

```
a₁ = 2.7^⁻ω₀
b₀ = gainLinear * a₁
b₁ = -a₁ * gainLinear
```

In the actual implementation on the DSP, when the coefficients are stored in parameter RAM, *a1* needs to be inverted. This is done in software, automatically, before the parameters are written to memory.

## General 2nd-Order Filters

---

The General Purpose and Text Entry blocks have multiple options for biquad filters based on Robert Bristow-Johnson's work in this field.

$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}$$

### Common variables:

```
ω₀ = 2*pi*f₀/Fs
gainLinear = 10^(gain/20)
```

### Peaking

Transfer Function

$$H(s) = \frac{s^2 + \frac{A}{Q}s + 1}{s^2 + \frac{s}{A*Q} + 1}$$

Coefficients

```
alpha = sin(ω₀)/(2*Q)
```

```
a₀ = 1 + alpha/A
a₁ = -2 * cos(ω₀)
a₂ = 1 - alpha/A
b₀ = (1 + alpha*A) * gainLinear
b₁ = -(2 * cos(ω₀)) * gainLinear
b₂ = (1 - alpha*A) * gainLinear
```

### Low-Shelf

Transfer Function

$$H(s) = A * \frac{s^2 + \frac{\sqrt{A}}{Q}s + A}{As^2 + \frac{\sqrt{A}}{Q}s + 1}$$

Coefficients

```
alpha = sin(ω₀)/2 * sqrt((A + 1/A)*(1/S - 1) + 2)
```

```
a₀ = (A+1) + (A-1)*cos(ω₀) + 2*sqrt(A)*alpha
a₁ = -2*( (A-1) + (A+1)*cos(ω₀) )
a₂ = (A+1) + (A-1)*cos(ω₀) - 2*sqrt(A)*alpha
b₀ = A*( (A+1) - (A-1)*cos(ω₀) ) + 2*sqrt(A)*alpha * gainLinear
b₁ = 2*A*( (A-1) - (A+1)*cos(ω₀) ) * gainLinear
b₂ = A*( (A+1) - (A-1)*cos(ω₀) ) - 2*sqrt(A)*alpha * gainLinear
```

**High-Shelf**

Transfer Function

$$H(s) = A * \frac{As^2 + \frac{\sqrt{A}}{Q}s + 1}{s^2 + \frac{\sqrt{A}}{Q}s + A}$$

Coefficients

```

alpha = sin(omega0)/2 * sqrt( (A + 1/A)*(1/S - 1) + 2 )
)

a0 = (A+1) - (A-1)*cos(omega0) + 2*sqrt(A)*alpha
a1 = 2*( (A-1) - (A+1)*cos(omega0) )
a2 = (A+1) - (A-1)*cos(omega0) - 2*sqrt(A)*alpha
b0 = A*( (A+1) + (A-1)*cos(omega0) + 2*sqrt(A)*alpha
      ) * gainLinear
b1 = -2*A*( (A-1) + (A+1)*cos(omega0) ) * gainLinear
b2 = A*( (A+1) + (A-1)*cos(omega0) - 2*sqrt(A)*alpha
      ) * gainLinear
  
```

**Lowpass**

Transfer Function

$$H(s) = \frac{1}{s^2 + \frac{s}{Q} + 1}$$

Coefficients

```

alpha = sin(omega0)/(2*Q)

a0 = 1 + alpha
a1 = -2*cos(omega0)
a2 = 1 - alpha
b0 = (1 - cos(omega0)) * gainLinear / 2
b1 = 1 - cos(omega0) * gainLinear
b2 = (1 - cos(omega0)) * gainLinear / 2
  
```

**Highpass**

Transfer Function

$$H(s) = \frac{1}{s^2 + \frac{s}{Q} + 1}$$

Coefficients

```

alpha = sin(omega0)/(2*Q)

a0 = 1 + alpha
a1 = -2*cos(omega0)
a2 = 1 - alpha
b0 = (1 + cos(omega0)) * gainLinear / 2
b1 = -(1 + cos(omega0)) * gainLinear
b2 = (1 + cos(omega0)) * gainLinear / 2
  
```

**Bandpass**

Transfer Function

$$H(s) = \frac{s}{s^2 + \frac{s}{Q} + 1}$$

Coefficient	<pre> alpha = sin(omega0) * sinh( ln(2)/2 * bandwidth * omega0/sin(omega0)  a0 = 1 + alpha a1 = -2*cos(omega0) a2 = 1 - alpha b0 = alpha * gainLinear b1 = 0 b2 = -alpha * gainLinear </pre>
-------------	--

### **Bandstop**

Transfer Function

$$H(s) = \frac{s^2 + 1}{s^2 + \frac{s}{Q} + 1}$$

Coefficients

<pre> alpha = sin(omega0) * sinh( ln(2)/2 * bandwidth * omega0/sin(omega0)  a0 = 1 + alpha a1 = -2*cos(omega0) a2 = 1 - alpha b0 = 1 * gainLinear b1 = -2*cos(omega0) * gainLinear b2 = 1 * gainLinear </pre>
---

### **Butterworth LP**

Transfer Function

$$H(s) = \frac{1}{s^2 + \sqrt{2}s + 1}$$

Coefficients

<pre> alpha = sin(omega0) / 2.0 * 1/sqrt(2)  a0 = 1 + alpha a1 = -2*cos(omega0) a2 = 1 - alpha b0 = (1 - cos(omega0)) * gainLinear / 2 b1 = 1 - cos(omega0) * gainLinear b2 = (1 - cos(omega0)) * gainLinear / 2 </pre>
---

### **Butterworth HP**

Transfer Function

$$H(s) = \frac{s^2}{s^2 + \sqrt{2}s + 1}$$

Coefficients

<pre> alpha = sin(omega0) / 2.0 * 1/sqrt(2)  a0 = 1 + alpha a1 = -2*cos(omega0) a2 = 1 - alpha b0 = -(1 + cos(omega0)) * gainLinear / 2 b1 = -(1 + cos(omega0)) * gainLinear b2 = -(1 + cos(omega0)) * gainLinear / 2 </pre>
--

### **Bessel LP**

Transfer Function

$$H(s) = \frac{3}{s^2 + 3s + 3}$$

Coefficients

```

alpha = sin(omega0) / 2.0 * 1/sqrt(3)

a0 = 1 + alpha
a1 = -2*cos(omega0)
a2 = 1 - alpha
b0 = (1 - cos(omega0)) * gainLinear / 2
b1 = 1 - cos(omega0) * gainLinear
b2 = (1 - cos(omega0)) * gainLinear / 2

```

**Bessel HP**

Transfer Function

$$H(s) = \frac{s^2}{s^2 + 3s + 3}$$

Coefficients

```

alpha = sin(omega0) / 2.0 * 1/sqrt(3)

a0 = 1 + alpha
a1 = -2*cos(omega0)
a2 = 1 - alpha
b0 = -(1 + cos(omega0)) * gainLinear / 2
b1 = -(1 + cos(omega0)) * gainLinear
b2 = -(1 + cos(omega0)) * gainLinear / 2

```

For all of the above filters, the coefficients are divided by  $a0$ , normalizing them and making  $a0 = 1$  so that only 5 coefficients must be stored. In the actual implementation on the DSP, when the coefficients are stored in parameter RAM,  $a1$  and  $a2$  need to be inverted. This is done automatically, in software, before the parameters are written to memory.

## Level-Detector Algorithm

---

Level Detector blocks are driven by an algorithm for detecting the rms level of the signal, represented in dB with a color bar. The time constant, hold time, and decay time are derived from the following formulas:

```
Timeconstant = 1.0 - (dbperseconds / (10.0 * fs));
Hold = fs * milliseconds / 1000;
Decay = dbperseconds / (96.0 * fs);
```

The detector output is a 5.19-format number (5 bytes are used to represent the integer portion of the value, 19 bytes for the decimal part). Depending on the algorithm you selected, one of two formulas determines the output dB value from the 5.19 value. The default formula for all level detectors is:

```
dB_value = 96.32959861 * (readback_value / 219 - 1)
```

The minimal-reading algorithm uses the following formula:

```
dB_value = 96.32959861 * (readback_value / 256 - 1)
```

Note that the Level Detector Designer block is the only block that allows you to choose one of these algorithms, the single and seven band level detector blocks are Passthru algorithms only:

### Passthru

This is called a passthrough (passthru) system because the output is the same signal as sent into the block. The algorithm receives a 5.19 number during readback and converts it to a decimal value represented in dB. This is what is seen on the display bar.

### RTA to Output

The RTA-to-output algorithm outputs the rms-level value through the red pin. The pin is red because audio is not being sent out, just the level values for each frequency band. Notice that when you grow the algorithm, each pin corresponds to a frequency band.

### Passthru / Minimal Reading, Single- and Double-Precision

The Passthru / Minimal Reading works similarly to Passthru but saves on communication bandwidth at a loss of precision. Passthru / Minimal Reading uses only 1 byte for the readback value, then converts it to a decimal, represented in decibels.

Double-precision should normally be used: 56 bits for each calculation, 10 instructions per filter.

Single-precision uses 28 bits for calculations, takes 6 instructions per filter, and saves 3 data RAM spaces over double-precision. Single-precision should not be used for frequencies below 1/10 the sampling rate or for high-Q filters.

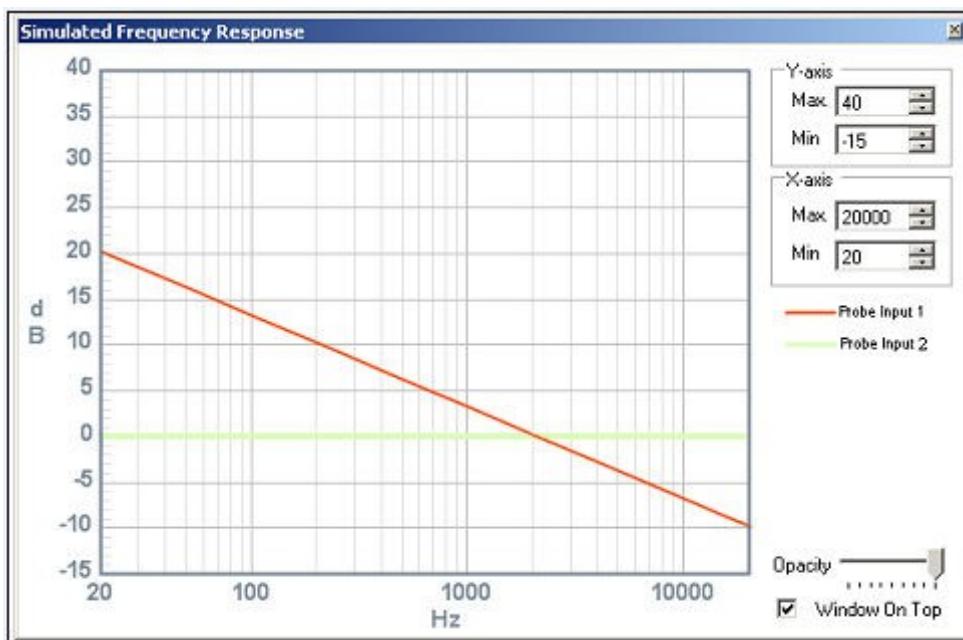
© 2006-2007 Analog Devices, Inc. All rights reserved.

## Pink Filter

The Pink Filter takes any broadband input and outputs a signal with a 3dB drop per octave. The classic use of this filter is to convert white noise (equal energy per hertz) to pink noise (equal energy per constant percentage, e.g., log bundling, as with an octave or subdivision).

To the human ear, which has an approximately logarithmic frequency response, white noise is distinctly trebly, while pink noise sounds broad and smooth, more like a waterfall.

The graph below (generated using simulation stimulus and probe) shows the two responses of white (green line) and pink (red line) noise. But realize that while these graphs are displayed on the familiar audio log scale, the data have **not** been subjected to, that is, not integrated in, constant-percentage bundling, but remain in constant-frequency depiction. This is in contrast to the RTA-style graphs in example 8.1, Level Detectors / Lookup Tables, where both the data and the graph scale are logarithmic.



---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## State-Variable Filters

---

The State-Variable filter offers lowpass, highpass, bandpass, and band-reject all within one block.

An added advantage over biquad section filters is that only one coefficient is needed, rather than their five coefficients. For the algorithm including Q, there's an additional coefficient for its control.

### Common variables:

- frequency-control coefficient:  $f$
- q coefficient:  $q$

### Bandpass

$$H(z)_{BP} = \frac{f(1-z^{-1})}{1+z^{-1}(f^2 - 2 + qf) + z^{-2}(1-qf)}$$

### Lowpass

$$H(z)_{LP} = \frac{f(H(z)_{BP})}{(1-z^{-1})}$$

### Highpass

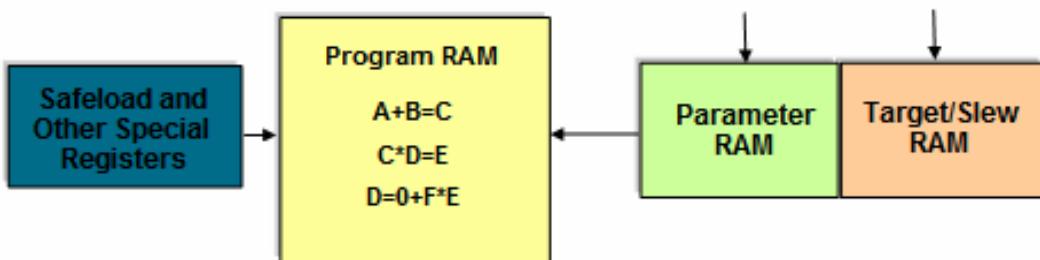
$$H(z)_{HP} = \frac{(1-z^{-1})}{f}$$

# SigmaDSP Architecture

## SigmaDSP Architecture (AD1940)

The basic AD1940 Sigma DSP architecture comprises these components:

- Program RAM stores functionality and execution information, meaning it dictates the signal flow and chain of operations (+, -,  $\times$ ,  $/$ , etc.) that are entered for execution.
- Parameter RAM and Data Memory. Parameter RAM stores parameter information (either from you or the result of a core operation). The data memory stores serial information, usually audio samples or the result of core operations.
- Safeload registers help load parameters smoothly.
- Target / Slew RAM are registers that help update parameters smoothly.
- Depending on the processor there are also additional registers: data capture; DSP core control; and serial input and output control.



This section of the helpfile contains a general summary of AD1940 features and microcontroller considerations. For details concerning these subjects beyond what is in the individual help topics, refer to the ADI website, navigating to your part's datasheet at [www.analog.com/sigmadsp](http://www.analog.com/sigmadsp).

## Program RAM Execution (AD1940)

It is important to understand how the program RAM is executed. The AD1940 SigmaDSP core generally works like a fixed program executing each one of its instructions (up to 1536 of them for the 1940) every new audio sample. This execution period is also known as a frame.

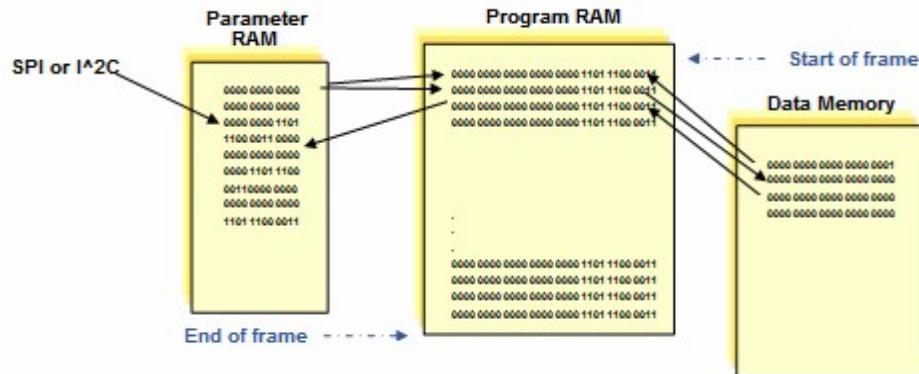


---

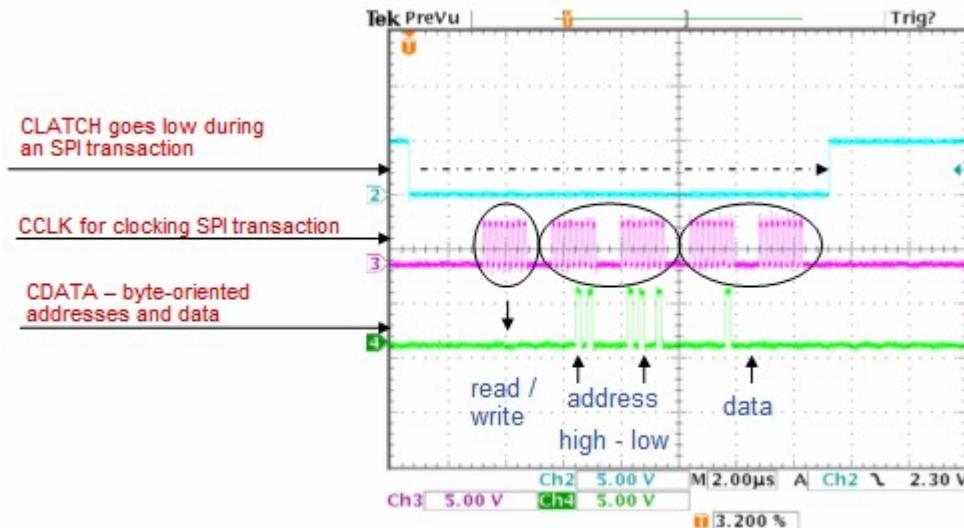
© 2006-2007 Analog Devices, Inc. All rights reserved.

# Parameter Ram and Data Memory (AD1940)

- Program RAM processes the information in the parameter RAM and data memory to produce results.
  - Parameter Memory is updated through the SPI or I<sup>2</sup>C port.
  - Data memory is usually updated every frame.

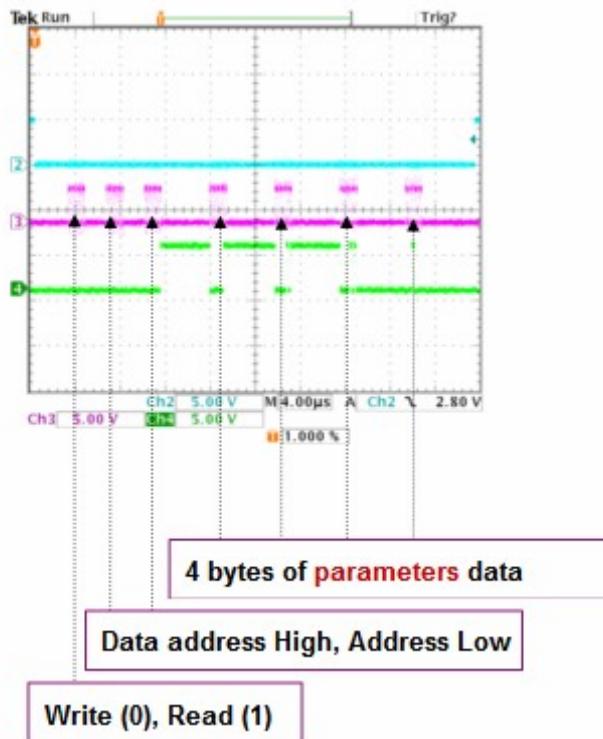


Among the most time-consuming tasks when developing an application is getting the communications to be set. The following illustration should be used as an example for starting basic SPI communication. It shows a simple operation on the AD1940 that will mute the DSP core. Observe that the word shown consists of 1 byte for writing, 2 bytes of the address, and 2 bytes for the data.

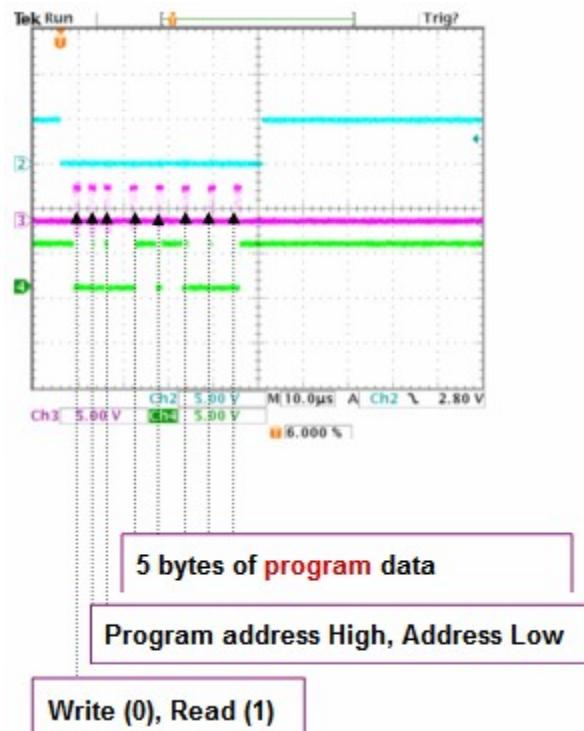


To write directly to each of them:

## Parameter Data



### Program Data

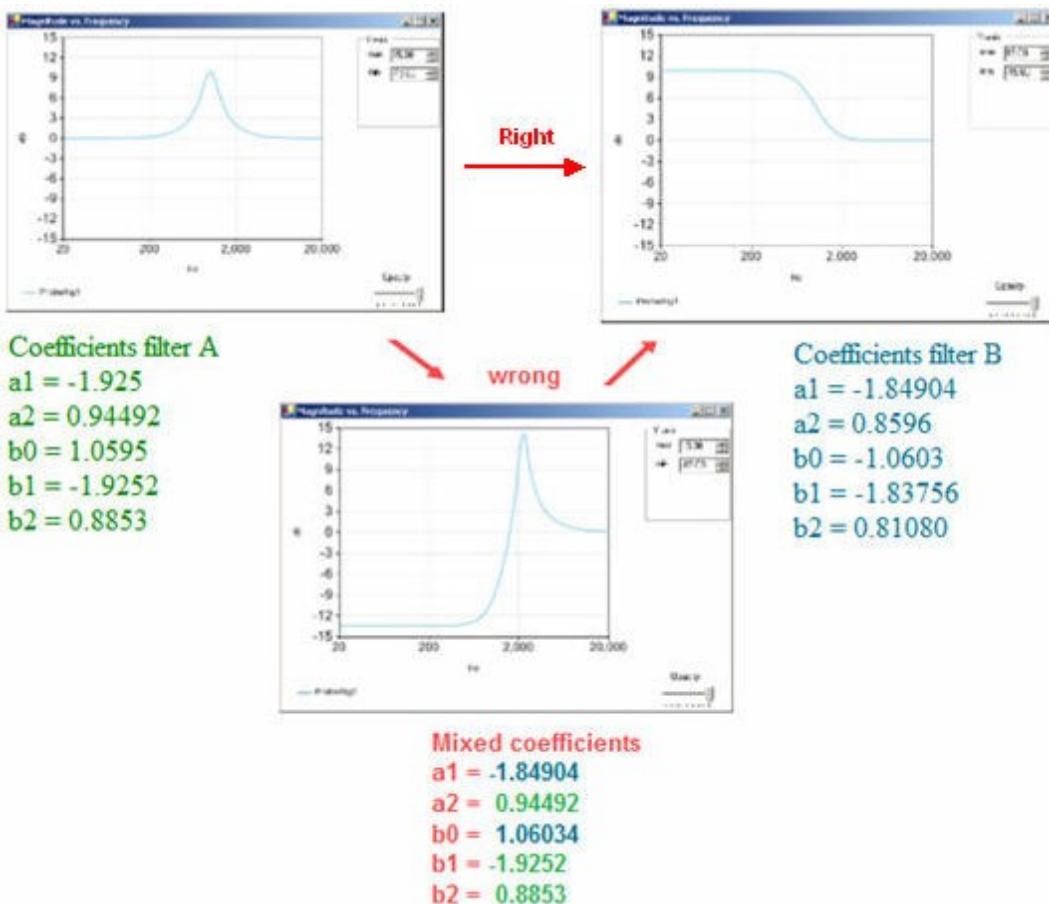


© 2006-2007 Analog Devices, Inc. All rights reserved.

## Safeload Registers (AD1940)

Filters are one of the most important elements in signal processing, and many are implemented using coefficients. Safeload registers are crucial in filter implementation because they are able to update coefficients safely. Whenever coefficients are changed, the transition has to be performed at the same time. Otherwise an undesirable response (audible pops etc.) can result, the filter may become unstable, and the output signal may harm the system because of this unpredictability.

Below are pictures showing filter coefficients being updated in the right and wrong ways, with mixed coefficients producing an undesirable frequency response:

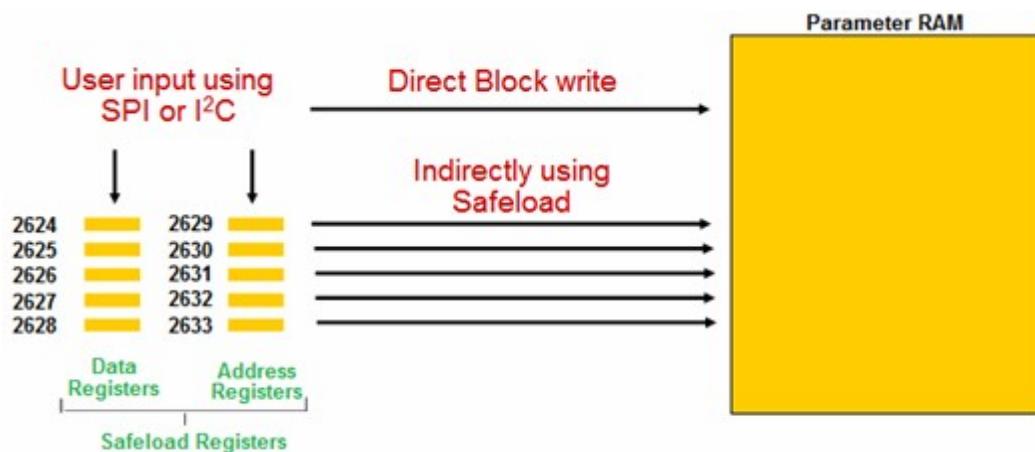


There are two approaches to solving this update problem:

- Mute the DSP, write directly into the filter's coefficient space in parameter memory, and then unmute the DSP to prevent unstable results. But this causes problems:
  - It is microcontroller-intensive – more control-port writes to the SigmaDSP.
  - It causes potential audio glitches every time we change the filter.

- The audio output will not be continuous.
- Use the safeload registers, as they are specifically designed to update coefficients at an appropriate time. Updates take place at the end of the program, before the start of a new frame, using five instructions from the program RAM. See below (the addresses shown are specific to the AD1940).

The following figures help describe what and where the safeload registers are:



Note that even though the length for parameter RAM is only 4 bytes, we need to reserve space for 5 bytes of data in the registers, as is required by Target / Slew RAM. Thus you need to keep in mind the format of your parameter data within the 5-byte space in the safeload registers:

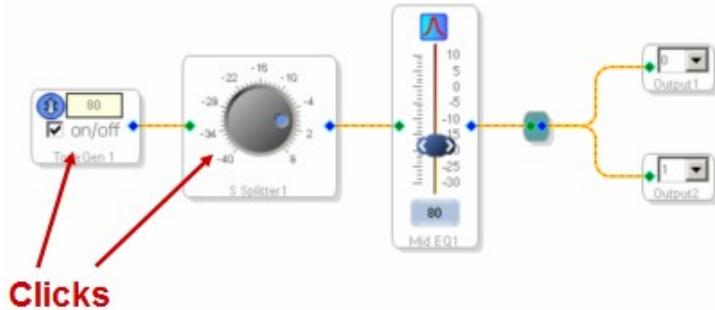
---

© 2006-2007 Analog Devices, Inc. All rights reserved.

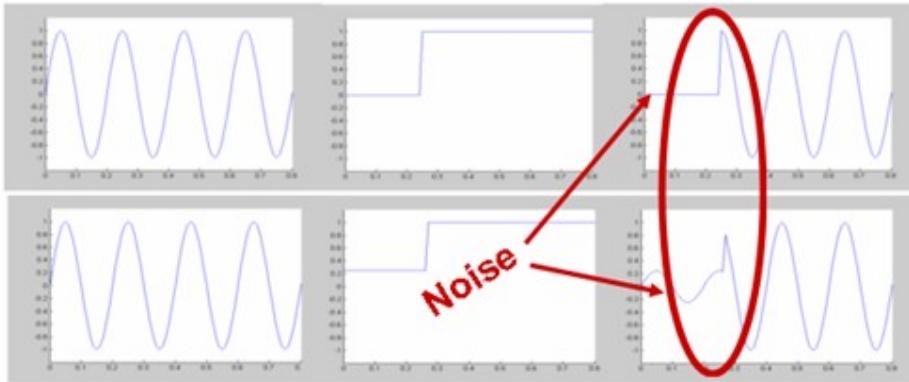
## Target Slew RAM (AD1940)

Target/Slew RAM is a hardware-optimized function that allows volume or other parameter level changes to ramp to subsequent levels without audible clicks/pops.

In audio systems, abrupt changes in volume position or other parameters produce unpredictable noises. See the sample schematic below showing possible real-time changes that can produce clicks/pops:

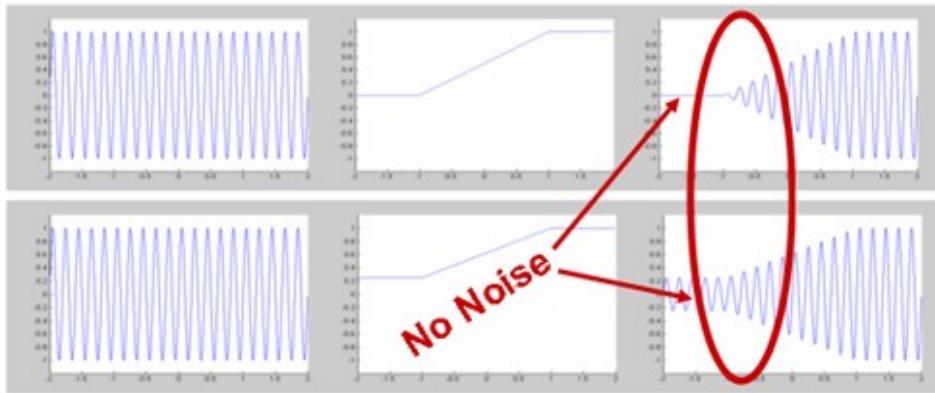


This noise arises from an audio signal getting scaled by a step function, which causes an unwanted response. The step function contains an impulse (Dirac delta function  $d(k)$ ). It can be shown that the impulse  $d(k)$  is a frequency-limited white noise. This noise is unwanted in all audio applications. See below for a graphic representation of this signal noise caused by the step function:



There are a couple of possible solutions to consider for this noise problem:

- We can continuously write an incremented value to the DSP until we reach the desired value. Problems:
  - Microcontroller-intensive
  - Difficult to write because of timing and slope-shape considerations.
- We can use target/slew RAM to automatically do the required ramping with one simple command without the need to mute the DSP. Below is the result:

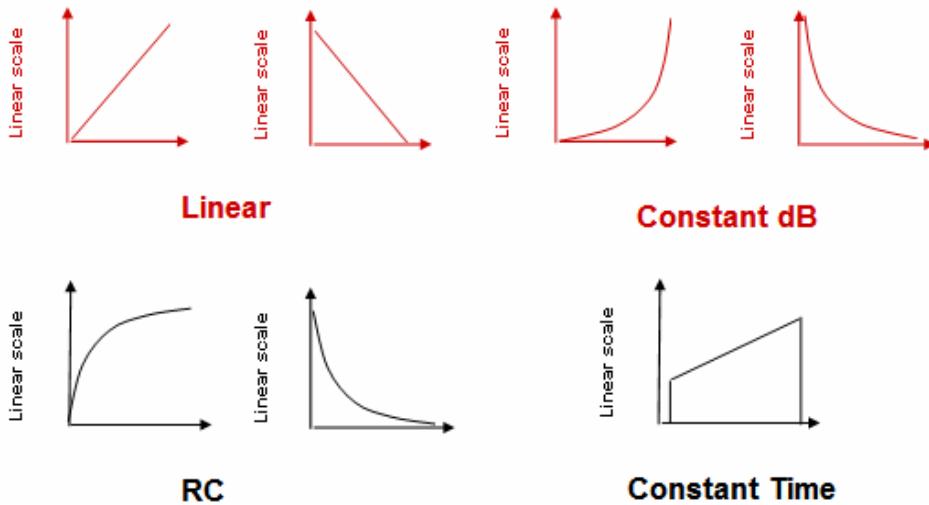


SigmaStudio's volume control and mux blocks both use the slew RAM to cleanly ramp volume from one level to another. Thus for all the volume control cells, you have the option of using target/slew RAM as your algorithm to prevent the noise inherent in switching levels.

There are 4 types of slope that can be used with the target/slew RAM:

- Linear -- fixed step size.
- Constant dB -- uses instantaneous slew value to calculate the next step size. In dB, it is constant up and down.
- RC-type -- Uses the difference between target and current values to calculate step size.
- Constant -- values change linearly, in a fixed number of steps.

For details, see the AD1940/1941 Volume Control Algorithms page.



Although target/slew RAM is useful for eliminating unwanted noise, there are some drawbacks:

- We are forced to write the values using the safeload registers.
- We cannot read back from the target/slew RAM to verify the written value.
- We can perform this functionality in software, although that will take possibly scarce program RAM (MIPS) and wouldn't be optimized.
- Limited number: the 1940 currently has 64 target/slew RAM locations, which should be used only when a parameter or set of parameters is going to get modified at

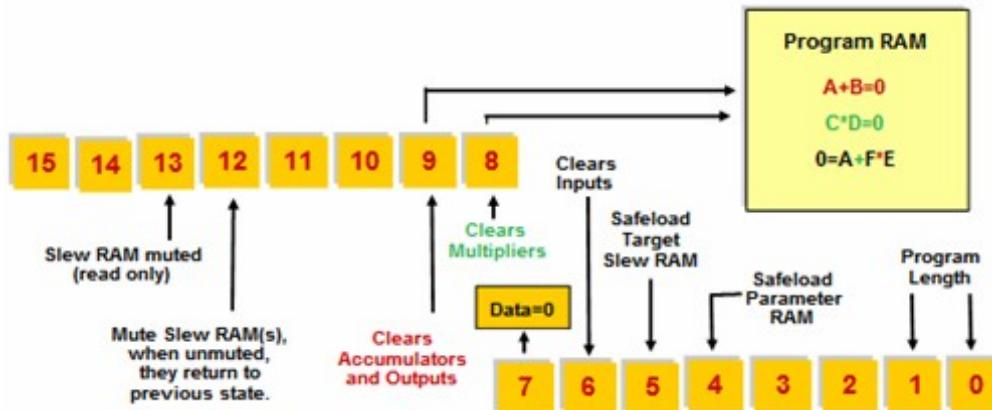
runtime in the final application.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Core Control Register (AD1940)

This is a 2-byte register that affects the DSP core. It is necessary for proper running of the SigmaDSP, by allowing you to easily set important core functions.



For more details concerning this register, go to the ADI website and refer to your part's datasheet: [www.analog.com/sigmadsp](http://www.analog.com/sigmadsp).

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

# Tutorials

## Tutorials

---

In this section are two simple examples to help you get started.

- The first tutorial shows how to set up input/output connections, a volume control, and 5-band EQ.
- The second tutorial shows you how to add probe and stimulus blocks and view the response of selected algorithms.

Refer to the ToolBox Examples for more design samples.

---

© 2006-2007 Analog Devices, Inc. All rights reserved.

## Stereo Audio with Volume Control and 5-Band EQ

This tutorial shows how to set up input and output connections, add a volume control, a 5-band Equalizer, and connect the schematic blocks and compile the program. This example uses the AD1940 with a slew RAM volume control, but the design is applicable to other SigmaDSP processors.

First, ensure that you have read through the Building Schematics section. Drag a AD1940 processor into the schematic, a USB communication channel, and connect them. The evaluation hardware should be configured for use with SigmaStudio and the hardware communication established. See Hardware Configuration Tab and the Quick Start topics for more information.

### Setting up I/O

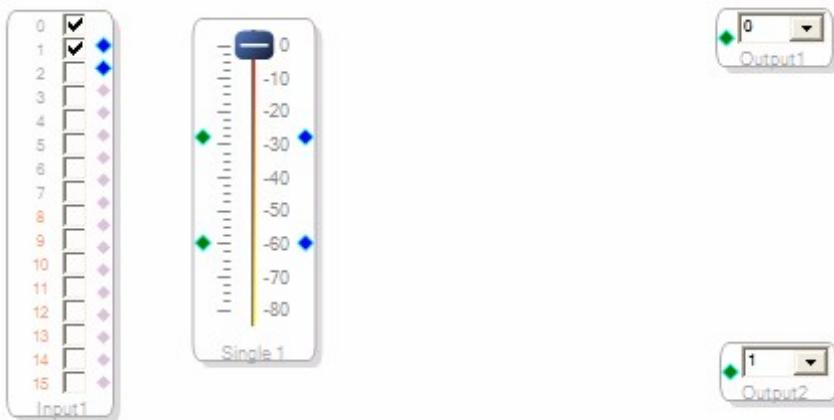
1. Click the **IO** tab of the ToolBox and drag the **Input** block into the workspace.
2. Two channels are already activated (boxes checked, channels 0 and 1).
3. Click and drag on an **Output** block. Since the example is stereo, repeat this step, to have two output channels. Your workspace should look something like this:



You are not ready to hear any output because nothing has been connected or compiled. You've just set the input/output layout.

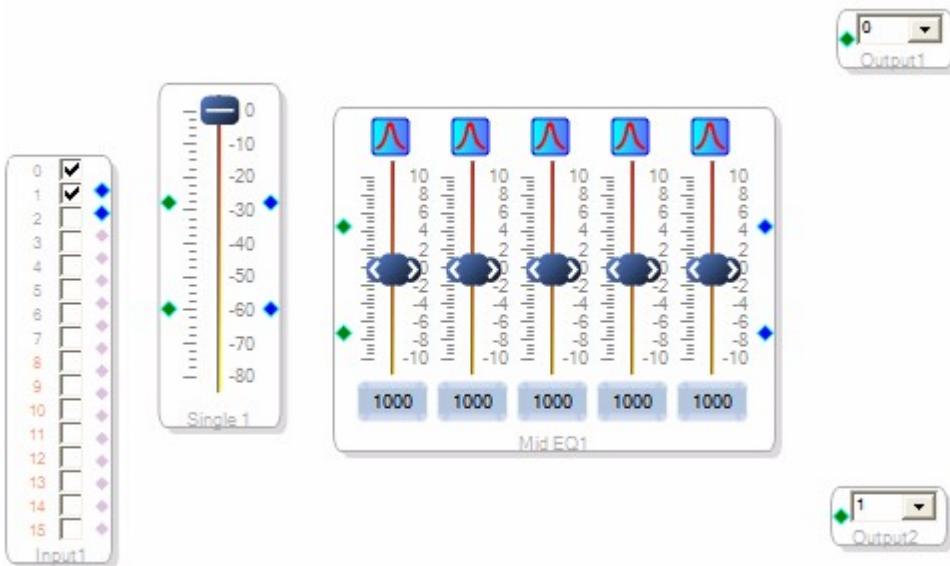
### Adding Volume Control

1. Click the **Volume Controls** tab.
2. Drag the **Single Volume Control** into the workspace
3. Right-click this block, **Single 1**, and select **Add Algorithm > IC 1 > Gain (slew)**. You should now see the slider block in the workspace. (For information on volume controllers, see Volume Controls.)
4. To make this application stereo, right-click the slider again and repeat step 3. You now will have two sets of connection nodes on your slider, as shown below. Again you are not yet ready to hear output.



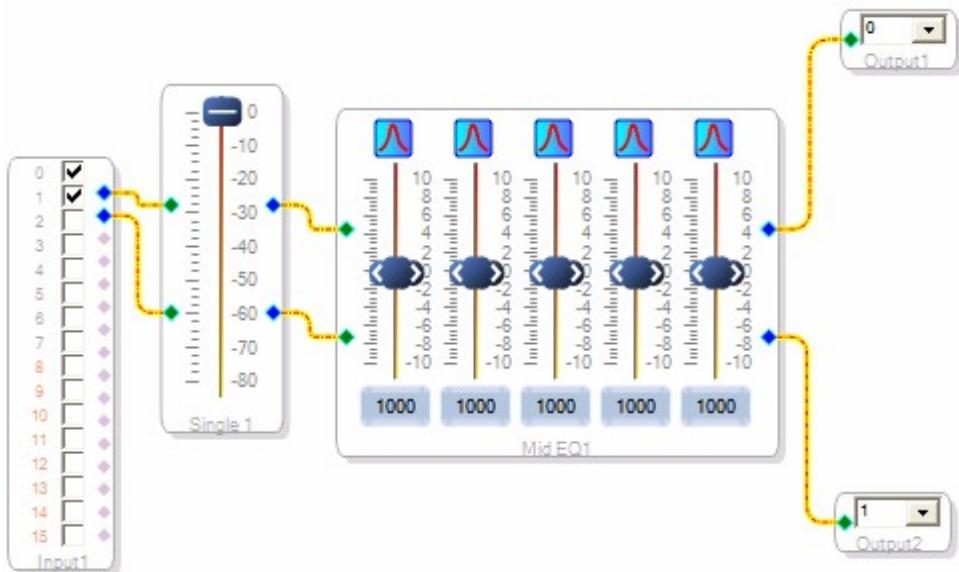
### Adding 5-Band EQ

1. Click the **Filters** tab of the ToolBox and drag a **Medium-Size EQ** into the workspace.
2. Currently all you see is a simple block that says **Mid EQ1**. Right click it and select **Add Algorithm > IC 1 > Double - Double Precision**.
3. By selecting **Double**, you are already in stereo, but currently have only one band. To increase bands, right-click the block and select **Grow Algorithm > 1. Double - Double Precision > 4**. Your circuit should look like the figure below. See the Filters topic book for detailed information on filters and the algorithm for this block.

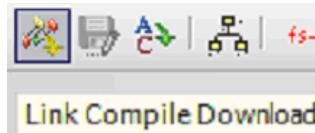


### Wire System and Compile

1. You now are ready to complete this project. Starting from the left, use your mouse (left button) and connect the nodes from blue to green, the output of one block to the input of the next.



2. Compile the project: Either click **Link Compile Download** on the toolbar or select **Action, Link Compile Download** from the main menu.



Once you have compiled the project, the blue bar at the bottom of the screen will turn green and the description will read **100% Ready - Downloaded**. At that point you are no longer in design mode, and have real-time control of the sliders and EQ controls and can hear the changes you make onscreen.

---

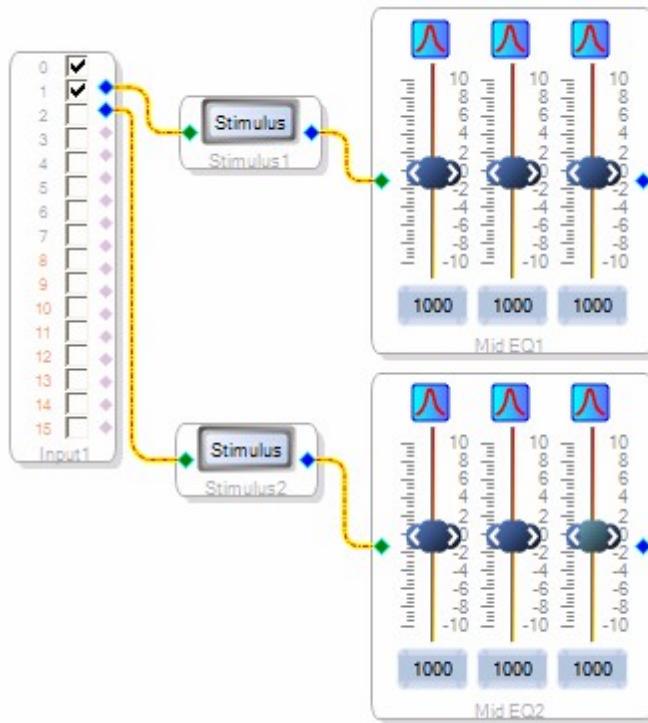
© 2006-2007 Analog Devices, Inc. All rights reserved.

## Probe and Stimulus Blocks

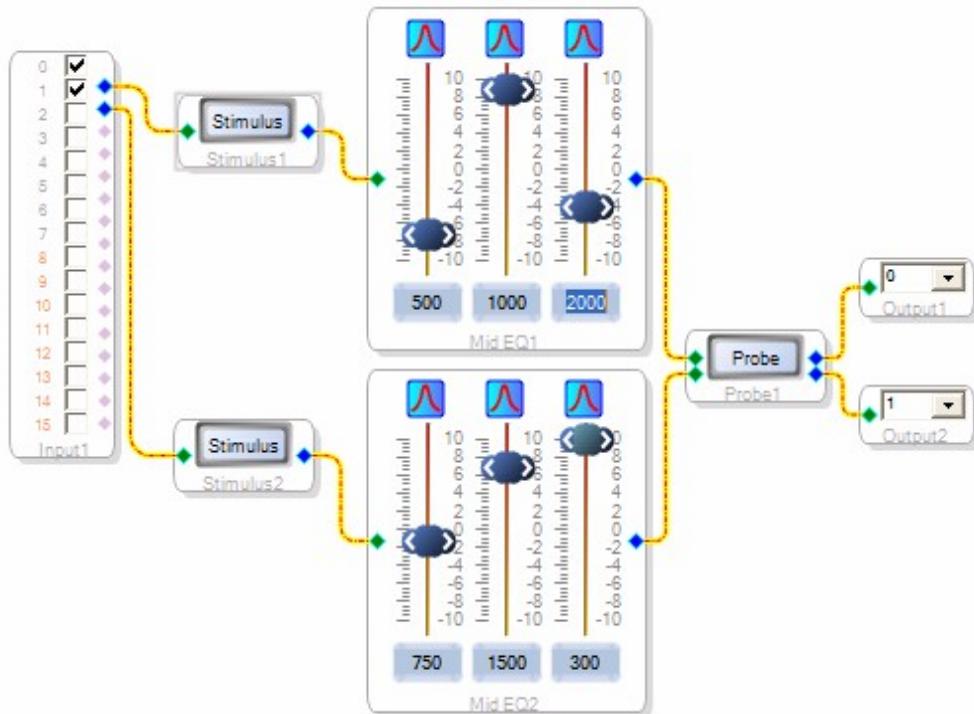
Complete the first tutorial in order for an explanation of the input, output, and EQ blocks that are used in this tutorial.

Its purpose is to show you how to use probe and stimulus blocks to monitor the frequency response of the filters you set.

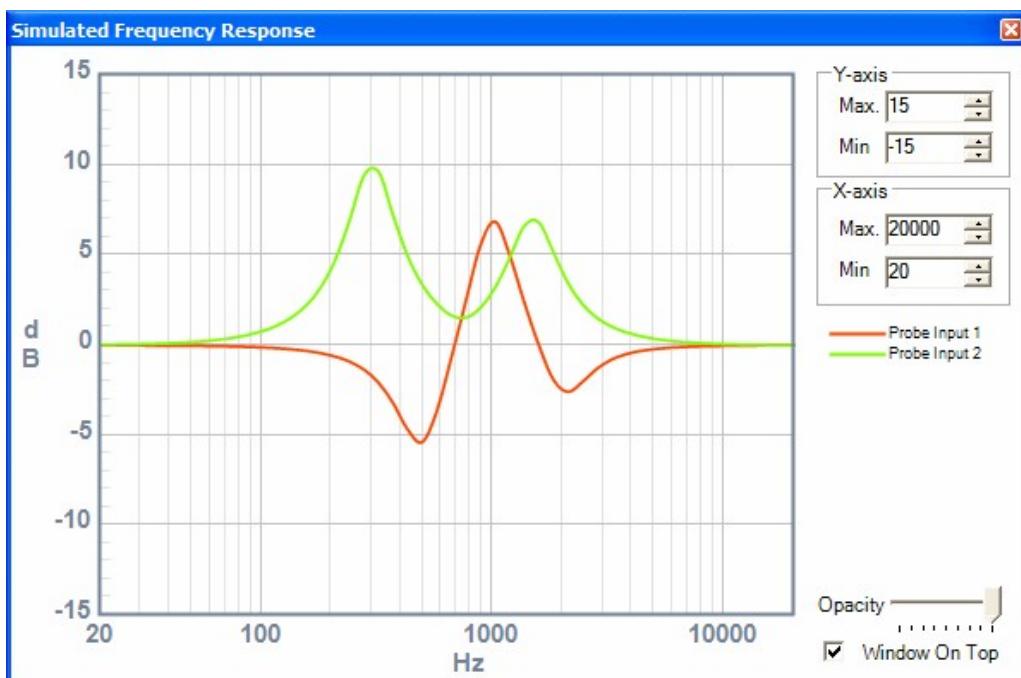
1. Insert an **Input** from the **IO** tab of the ToolBox.
  2. Click the **System** tab and drag two **Stimuliblock** blocks to the workspace.
  3. Connect wires from the input channels to the Stimuliblock block.
  4. Insert two **Medium Size Eq** blocks from the **Filters** tab.
  5. For each EQ block, right-click and select **Add Algorithm > IC 1 > Single - Double Precision**.
  6. Right-click each again and select **Grow Algorithm > 1. Single - Double Precision > 2**.
  7. Connect wires from the output of the Stimuliblock block to the input of the EQ block.
- Your workspace should look like this:



8. Click the **System** tab and drag one **Probeblock** block into your workspace.
9. Right-click the block and select **Add Pins**.
10. Connect wires from the output of the EQ to the input of the Probeblock.
11. Click the **IO** tab and drag two **Output** blocks into the workspace.
12. Connect wires from the output of the probes to the input of the Output block. Your workspace should look like this:



13. Compile the project: Click **Link Compile Download** on the toolbar or select **Action, Link Compile Download**.
14. Click the **Probe** block to bring up the Simulated Frequency Response window. There will be nothing displayed.
15. Click the **Stimulus** block buttons to bring up the Frequency Response window. You can now view in real time any changes you make to the EQ. The response for the above parameters is shown here:



© 2006-2007 Analog Devices, Inc. All rights reserved.

# Index

*	
*.bin.....	129
*.h .....	105
*.hex.....	105
*.params .....	105
_REG.h .....	105
<b>1</b>	
1940.....	5, 17, 22, 24
<b>2</b>	
28.0.....	118
<b>5</b>	
5.23.....	118
<b>A</b>	
AB in CD out Condition .....	171
AB in/out Condition .....	172
About Dialog .....	57
Absolute Value.....	173
AD1940.....	5, 17, 22, 24
Add .....	82
Add Filter .....	258
AddIns.....	58
Add-Ins Browser .....	58
AddIns.xml .....	58
ADI Algorithms.....	148
ADI Surround .....	310
ADI Virtual.....	312
Adjustable Slew Control.....	375
Algorithm.....	87
Align to Destination Pin .....	92
Align to Source Pin .....	92
Allow Realtime AB Testing.....	123
Allpass Filters.....	424
Architecture.....	441
ASRC Input.....	289
ASRC output.....	289
Audistry .....	322
automation .....	61
Auxiliary ADC Input.....	290
<b>B</b>	
Basic DSP.....	170
BBE.....	319
BBE MP.....	320
BBE ViVA.....	321
Beep Sources.....	362
Block Write .....	101
blocks .....	47
Board.....	45, 120, 137, 139, 140
Building Schematics .....	78
<b>C</b>	
Capture Output Data .....	101
Capture Window	
Capture.....	101
Capture Window.....	101
Change IC .....	126
Circle Surround II .....	328
Clipper.....	174
Compressors.....	216
Constant dB .....	375
Control.....	82
Copy.....	124
Core Control Register.....	452
Counter .....	194
Counters.....	193
Cross Mixer (2 inputs) .....	334
Cross Mixer (3 inputs) .....	337
Cross Mixer (8 inputs) .....	338
Crossfade (data controlled) .....	348
Cut.....	124
<b>D</b>	
DAEP .....	323
Data Capture.....	68
Data Memory.....	443
DC Blocking .....	237
DC Input Entry.....	363
De-Emphasis.....	238
Deemphasis Filter .....	425
Delay .....	176
Demultiplexer .....	347, 412
Development Environment .....	36
Divide .....	177
Division.....	177
Dolby Headphone .....	324
Dolby Pro Logic II .....	325

Dolby Virtual Speaker .....	326
DSP Readback .....	179
Dynamic Bass Boost.....	149
Dynamic Enhancement.....	151
Dynamics Processors .....	426
<b>E</b>	
E2Prom.....	73
E2Prom Read/Write Window .....	73
Enable Auto Layout.....	92
Envelope.....	210
Envelope Peak.....	212
Envelope RMS.....	213
Error - No Inputs are Defined for IC .....	78
EVAL-ADUSB1 .....	26, 30
EVAL-ADUSB2 .....	26, 34
Evaluation Board.....	17, 22, 24
Examples .....	387
Export System Files .....	105
External LED Array Driver.....	314
<b>F</b>	
FAQ .....	16
Feedback .....	180
filter coefficients .....	59
Filter Table Generator.....	59
Filters .....	235
FIR Filter .....	239
First Order w var	
Param/Lookup/Slew .....	242
Fixed-point numbers .....	118
Flanger.....	153
Flash Downloader .....	72
Freeze.....	120
Freeze Schematic.....	128
Full Range .....	426
<b>G</b>	
Gain (no slew).....	377
Gain (slew).....	377
General (1st order).....	241
General (2nd Order).....	244
General (2nd Order/Lookup).....	250
General 2nd Order Index Selectable .....	258
General 2nd Order w var	
Param/Lookup/Slew .....	256
General Purpose Input.....	291
General Purpose Output .....	292
GPIO Conditioning .....	283, 403
Grow.....	82
<b>H</b>	
hardware configuration..	5, 17, 22, 24
Hardware Configuration Tab .....	40
Hardware Windows .....	63
header file .....	105
Help.....	16
Hi Resolution RMS .....	220
Hide.....	120
Hierarchy Board .....	137
Hierarchy Board File.....	120
Hierarchy Boards .....	120
Hierarchy Input.....	139
Hierarchy Output .....	140
HW (RCtype Slew) .....	376
<b>I</b>	
Imploader .....	76
Index Lookup Table.....	300
Index Selectable DeMultiplexer...	350
Index Selectable Multiplexer .....	351
Index Selectable Slewing Mux ....	352
Input .....	293
Inputs and Outputs .....	288
Insert Point .....	92
Installation .....	3, 5
Interface Read.....	294
Interface Write .....	295
IO .....	288
<b>K</b>	
Knob.....	82
<b>L</b>	
Layout .....	82
Level Detector Designer.....	302
Level Detectors .....	299, 407
License Agreement .....	12
Licensed Algorithms .....	309
Limiter .....	214
Linear .....	375, 376
Linear Gain.....	181
Link/Compile/Download.....	96
Load Files.....	68
Logic - And,Or,Nand,Nor.....	182
Lookup .....	250
Lookup Tables.....	299

Loudness (Low & Hi).....	155
Loudness (Low & Hi) External Ctrl .....	156
Loudness (Lower End).....	159
<b>M</b>	
Max.....	82
Medium Size Eq.....	264
microcontroller .....	105
Midnight Mode .....	161
Min .....	82
Minimal Reading .....	437
Mixers .....	333, 410
Mono Switch 1xN .....	353
Mono Switch Nx1 .....	354
Multi CH Switch Nx4 .....	355
Multi CH Switch Nx6 .....	356
Multi CH Switch Nx8 .....	357
Multiple Control Mixer .....	339
Multiple Control Splitter.....	340
Multiple Volume Control.....	377
Multiplexer .....	347, 412
Multiply.....	186
Mute.....	376
<b>N</b>	
name.....	82
N-Channel.....	222
New Features .....	11
No Inputs are Defined.....	78
No Slew (Standard).....	376
Not enough DSP Resources for this Algorithm .....	288
Numeric Formats .....	118
<b>O</b>	
On/Off Switch.....	364
Output.....	296
Overview .....	1, 5
<b>P</b>	
Parameter Ram.....	443
Passthru.....	437
Paste.....	124
Peak (gain) .....	224
Peak (no gain).....	226
Phat-Stereo.....	167
Pin.....	82
Pinking .....	271
pins .....	78, 82
Print.....	60
Printing .....	60
Probe and Stimulus Blocks .....	457
Processors (ICs/DSPs) .....	78
Program RAM .....	442
Program Window.....	36, 38
Push and Hold.....	284
<b>R</b>	
RC-type .....	375, 376
Reduce.....	82
Register Control Window.....	64
Register Read/Write .....	68
Register Read/Write Window .....	68
Remove.....	82
Remove Filter.....	258
Remove Segment.....	92
Reverb.....	168
RMS (gain) .....	227
RMS (no gain) .....	229
RMS Detect (Display).....	231
RMS Detect (no gain,hold,decay) .....	233
RMS Table .....	305
Rotary Encoder .....	285
RTA to Output .....	437
<b>S</b>	
Safeload .....	446
Safeload Registers .....	446
Safeload Write.....	101
Sawtooth Wave .....	365
Schematic .....	45
Schematic Blocks .....	82
Schematic Settings.....	129
Schematic Tab .....	36, 45
Schematic Terminal.....	141
select.....	82
Settings .....	129
Seven Band Level Detector.....	307
Show .....	120
Signal Add.....	187
Signal Invert .....	188
Signal Merger.....	342
Simulation Probe .....	142
Simulation Stimuli.....	143
Single Control Mixer .....	343
Single Control Splitter.....	344

Single Level Detector.....	308	Tolerance Analyzer .....	189
Single slew ext vol .....	378	Tone (lookup/sine).....	370
Single SW slew vol .....	379	Toolbars .....	53
Single Vol (shared) .....	380	Toolbox .....	38, 47
Single Volume Control .....	383	Tracking Filter .....	278
slew.....	242	Tree ToolBox.....	47
Slew RAM .....	449	Triangle Wave .....	371
Slider.....	82	TruBass.....	329
Software Debounce .....	286	TruSurround XT .....	330
Sources.....	361	Tutorial .....	5, 453, 454
SPDIF Input .....	298	Tutorials .....	453
SPDIF output .....	298	Type .....	258
Speaker Response		<b>U</b>	
MLSSA .....	144	Unconnected pins found.....	78
Splitters.....	333, 410	Up/Down + INTF .....	386
Square Wave .....	366	Up/Down Control w/ Lookup Table	
.....	386	.....	386
SRS CS Auto .....	328	Up/Down Control, Index Output ..	287
SRS CS Auto LITE.....	328	Up/Down/Simultaneous + INTF ..	386
SRS CS II .....	328	USB .....	3, 26
SRS TruBass .....	329	USB driver.....	3, 26
SRS TruSurround XT.....	330	USB Interfaces .....	26
State Machine .....	358	USB Serial Converter .....	26, 30
State Variable .....	272	USBi .....	26, 34
State Variable (Q input) .....	273	USBSerialConv .....	26
Step .....	82	User Comment .....	146
Stereo Mixer .....	345	User Image.....	147
Stereo Switch 2xN .....	359	<b>V</b>	
Stereo Switch Nx2 .....	360	Value .....	82
Stop Watch .....	196	Value Hold.....	191
Stop Watch w/External Reset ....	200	VCO .....	372
Sub Type .....	258	View .....	120
Subharmonic Synth (general) .....	317	Vocal Chorus.....	169
Subharmonic Synth (low freq)....	318	Voltage Controlled Delay.....	192
Surround Sound Volume Control	384	voltage-controlled oscillator .....	372
SW (RCtype Slew) .....	376	Volume Control and 5-Band EQ..	454
Sweep (Log) with on/off switch ...	367	Volume Controls.....	374, 420
Sweep (lookup/sine) .....	368	<b>W</b>	
System.....	136, 417	White Noise Source.....	373
<b>T</b>		Wire .....	92
T Connection .....	145	Workspace Windows .....	50
Target/Slew RAM.....	449	WOW .....	331
Text-In (linked).....	274	WOW HD .....	332
Text-In (unlinked) .....	276		
Timer - w/ External Reset .....	205		