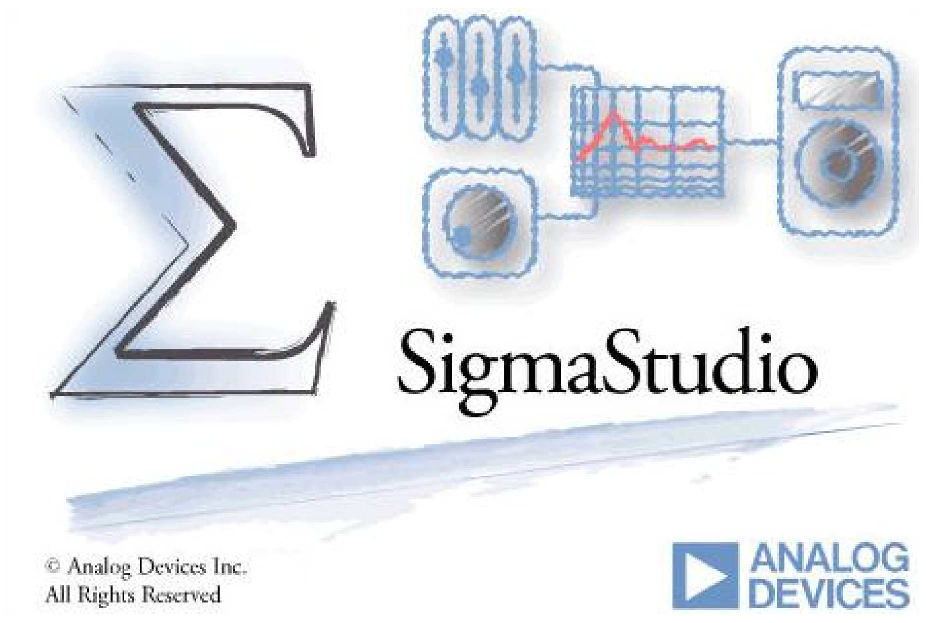
# SigmaStudio

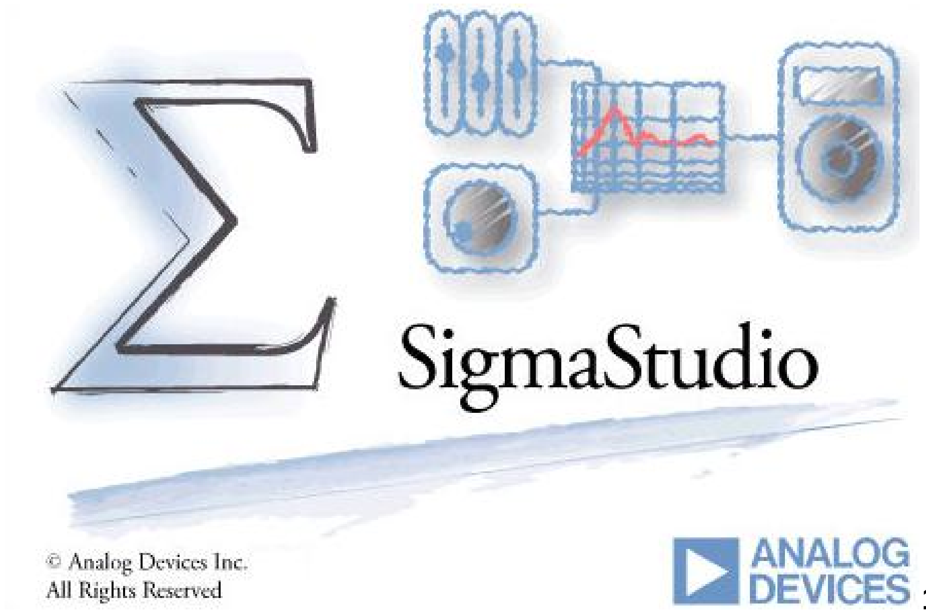


Welcome to The SigmaStudio™ Graphical Development Tool. This is a tool specifically developed for SigmaDSP™ audio processors. Familiar audio processing blocks can be wired together as in a schematic, and the compiler generates DSP-ready code and a control surface for setting and tuning parameters. This tool allows engineers with no DSP code writing experience to easily implement a DSP into their design and yet is still powerful enough to satisfy the demands of experienced DSP designers.

This manual is organized with the intent that you follow the Getting Started section first, and then peruse the rest of the material depending on your application needs. The main sections of this manual are:

- Getting Started
- Hardware Configuration
- SigmaStudio System
- ToolBox
- Algorithm Information
- Basic Sigma DSP Architecture
- Tutorials

# Table Of Contents



1

**Getting Started** 2

AD1940 rev B Evaluation Board Setup 3
AD1940/1 revA Evaluation Board Setup 5
USB Serial Converter Setup 7
Parallel Port 9
Program window 10
Hardware Configuration 11
Schematic 12
Numeric Formats 13
USB Serial Convertor Communication Channel 15

**Hardware Configuration** 16

Capture Output Data 17
Flash Downloader 22
Register Control Tab 23
Register Read/Write Window 27

**SigmaStudio System** 29

Sampling Rate Considerations 29
Toolbar Window Options 31
Linking your Project 33
Link/Compile/Download your Project 35
Generating Parameter File 37
Allow Realtime AB Testing Between Projects 40
Freeze Selected Schematic 41
System Implementation 42
Filter Table Generator 53

**Toolbox** 55

Using Cells 56
System 60

BaseBoardCell .......... 61
BoardInCell .......... 62
BoardOutCell .......... 63
ProbeCell .......... 64
Speaker Simulation .......... 65
StimuliCell .......... 66
TConnectionCell .......... 67
Terminal .......... 68
User Comment .......... 69
User Image .......... 70
Examples .......... 71
Basic DSP .......... 75
AB in CD out Condition .......... 76
AB in/out Condition .......... 77
Absolute Value .......... 78
DC Input Entry .......... 79
Delay .......... 80
Divide .......... 81
DSP Readback .......... 83
Feedback .......... 84
Linear Gain .......... 85
Multiply .......... 86
Signal Add .......... 87
Signal Invert .......... 88
Examples .......... 89
Dynamics Processors .......... 92
Peak Detect (gain) .......... 95
Peak Detect (no gain) .......... 96
RMS Detect (display) .......... 97
RMS Detect (gain) .......... 98
RMS Detect (no gain) .......... 99
RMS Detect (no: gain, hold, decay) .......... 100
Examples .......... 101
Advanced DSP .......... 105
Adaptive Mixer Dual (graph) .......... 106
Adaptive Mixer Single (graph) .......... 108
RMS Table .......... 110
State Variable Filter (Q input) .......... 111
Examples .......... 112
Mixers/Splitters .......... 114
Eight Input Cross Mixer .......... 115
Multiple Control Mixer .......... 116
Multiple Control Splitter .......... 117
Signal Merger .......... 118
Single Control Mixer .......... 119
Single Control Splitter .......... 120
Three Input Cross Mixer .......... 121
Two Input Cross Mixer .......... 122
Examples .......... 123
Filters .......... 125
All-Pass 1st Order .......... 126
All-Pass 2nd Order .......... 127
DC Block Filter .......... 129
De-emphasis Filter .......... 130
General 1st Order .......... 131
General Purpose Filter .......... 132

- Medium Size Eq ........ 138
- Pink Filter ........ 143
- Small Size Eq ........ 144
- State Variable Filter ........ 146
- Text-In Filter (linked) ........ 147
- Text-In filter (unlinked) ........ 150
- Tone Control(With Index lookup) ........ 152
- Examples ........ 153

I/O ........ 157
- Input ........ 158
- Output ........ 159
- AD1953 Output ........ 159
- GPIO input ........ 161
- GPIO output ........ 162
- Ext interface in ........ 163
- Ext interface out ........ 164
- Aux ADC Input ........ 165
- Examples ........ 166

Level Detectors ........ 168
- Level Detector Designer ........ 169
- Seven Band Level Detector ........ 173
- Single Level Detector ........ 174
- Example ........ 175

ADI Algorithms ........ 177
- Dynamic Bass Boost ........ 178
- Dynamic Enhancement ........ 180
- Loudness Ctrl (Lower End) ........ 182
- Midnight Mode ........ 184
- Phat Stereo ........ 186
- Reverb ........ 187
- Vocal Chorus ........ 188

Volume Controls ........ 189
- Adjustable Volume Control ........ 191
- Mono Switch 1xN ........ 192
- Mono Switch Nx1 ........ 193
- Multiple Volume Control ........ 194
- Single slew ext vol ........ 195
- Single SW slew vol ........ 196
- Single Vol (shared) ........ 197
- Single Volume Control ........ 200
- Stereo Switch 2xN ........ 201
- Stereo Switch Nx2 ........ 202
- Surround Sound Volume Control ........ 203
- Example ........ 205

Sources ........ 207
- Beep Sources ........ 208
- Sweep Up ........ 209
- Tone (lookup/sine) ........ 210
- White Noise Source ........ 211
- Example ........ 212

GPIO conditioning ........ 214
- Index look up table ........ 215
- Push and Hold ........ 217
- up down increment ........ 218
- Example ........ 219

Example Schematics ........ 221

**Algorithm Information** 222

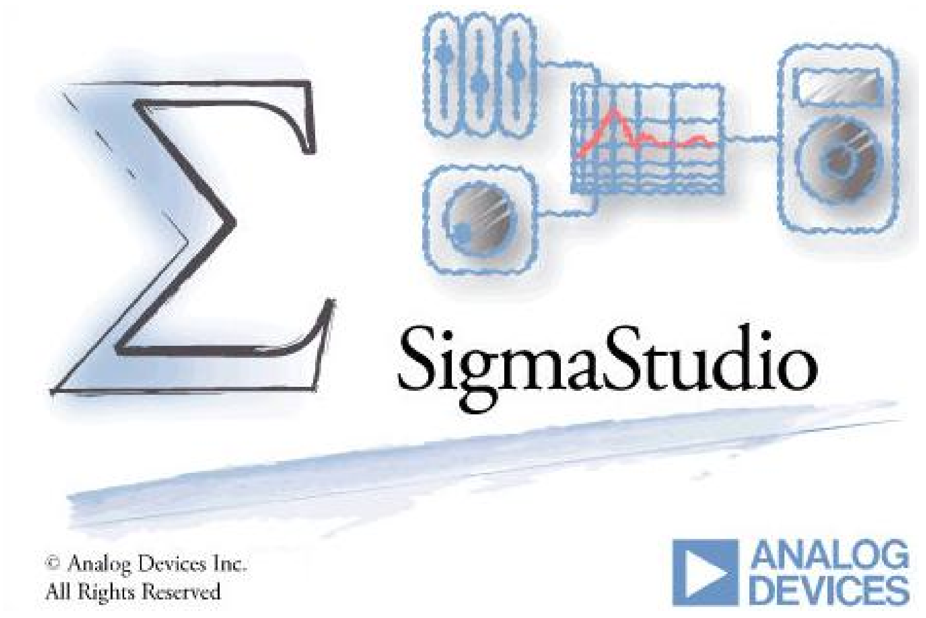
EQ Algorithm .......... 223
AD1940/AD1941 Volume Controls .......... 225
AD1953 Volume Controls .......... 226
General 1st Order Filters .......... 227
General 2nd Order Filters .......... 228
All Pass Filters .......... 234
State Variable Filters .......... 236
De-emphasis Filter .......... 237
Pink Filter .......... 238
Level Detector Algorithms .......... 239

**Basic SigmaDSP Architecture** 241

Program RAM Execution .......... 242
Parameter Ram and Data Memory .......... 243
Safeload Registers .......... 246
Target Slew RAM .......... 249
Core Control Register .......... 252

**Tutorials** 253

Stereo Audio with Volume Control and 5-band EQ .......... 254
Probe and Stimulus Blocks .......... 258



Welcome to The SigmaStudio™ Graphical Development Tool. This is a tool specifically developed for SigmaDSP™ audio processors. Familiar audio processing blocks can be wired together as in a schematic, and the compiler generates DSP-ready code and a control surface for setting and tuning parameters. This tool allows engineers with no DSP code writing experience to easily implement a DSP into their design and yet is still powerful enough to satisfy the demands of experienced DSP designers.

This manual is organized with the intent that you follow the Getting Started section first, and then peruse the rest of the material depending on your application needs. The main sections of this manual are:

- Getting Started
- Hardware Configuration
- SigmaStudio System
- ToolBox
- Algorithm Information
- Basic Sigma DSP Architecture
- Tutorials

# Getting Started

In order to start using SigmaStudio there are a few necessary steps to configure your system. The following is an outline description to help you get started. There are pages which describe each step in detail so that you can be successful in setting up SigmaStudio for your needs.

- First ensure that your board is setup for the correct connections that you will be using by checking the Evaluation Board Setup page for some typical input/output connections
- Next connect the USB Serial Converter to your board and to your USB connection of your processor. You can also communicate to the computer via Parallel Port connection
- Now you are ready to load the software and launch SigmaStudio which brings you to the Program Window
- From the Program Window you can open a project or create a new project and select the Hardware Configuration tab to setup your configuration on the screen.
- Now you can click on the Schematic tab and begin to construct your design

If you would like more information about the Board you are using, please see the Analog Devices website and navigate to your part's datasheet for more information:

http://analog.com/sigmadsp

# AD1940 rev B Evaluation Board Setup

Depending on your application and needs, there are different configurations that can be set on the evaluation board. The following are examples of typical input/output configurations:

Six-Channel Analog Input/Analog Output

S/PDIF Input/Analog & S/PDIF Output

I$^2$S Input/ I$^2$S Output

**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 designed in the SigmaStudio software. Default outputs for the input 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 slave. The following list explains how the switches and jumpers need to be set up for this mode. Any component setting not mentioned in this list can be considered “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 designed in the SigmaStudio software. Default outputs for the input 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 explains how the switches and jumpers need to be set up for this mode. Any component setting not mentioned in this list can be considered “don't care.”

- SW2 - Position 0
- SW9 - Position 0
- LK1 - Connected
- LK2/3 - Disconnected

- LK5/14 - SPDIF
- SW10 - Selects S/PDIF source from either optical or RCA
- S1 - Position 2 up (1), all other positions down (0)

**I$^2$S Input/ I$^2$S Output**

This mode is intended to be used to connect to external devices, such as ADCs, DACs, or compressed audio (such as AC-3) decoders. 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 designed in the SigmaStudio software. Default outputs for the input 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 explains how the switches and jumpers need to be set up for this mode. Any component setting not mentioned in this list can be considered “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)

# AD1940/1 revA Evaluation Board Setup

Depending on your application and needs, there are different configurations that can be set on the evaluation board. The following are examples of typical input/output configurations:

SPDIF Input, Analog Output

Analog Input, Analog Output

$I^2$ Input, $I^2$ Output

**SPDIF 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 – "CO-AX" or "Optical" - depends on source
- LK2 – left (0)
- LK1/3/10/11/13 – all in left jumper position
- The remaining switches and jumpers are “don’t care”

Input will come from the SPDIF receiver to AD1940 SDATA_IN0 and will output on whatever outputs (SDATA_OUTx) are designed in the software. Default outputs 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 are “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 outputs for the input is SDATA_OUT0/1/2, or Analog Output 0-5

**$I^2$ S Input, $I^2$ 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 are “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 designed in the software. Default outputs 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.

# USB Serial Converter Setup

Analog Devices' USB Adapter (EVAL-ADUSB1) board is used to interface between the PC's USB port and evaluation boards' control port connections. In the case of the AD1953, AD1954, AD1940 and AD1941 evaluation boards, the USB Serial Converter connects directly to the DB25 connector on the board. Once installed, the use of the USB adapter is transparent to SigmaStudio. A USB connection just needs to be selected when a new project is started as is described in the Hardware Configuration page.

The USB Adapter is normally powered from the computer's USB port. In the case where the adapter is being used in stand-alone mode to load new DSP programs, it can also be powered with +5V and ground on test points TP1 and TP2.

The following steps need to be taken to use the USB adapter with SigmaStudio:

1. Plug the USB 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 un-check 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 the browse button and find the ftd2xx.inf file in the USB drivers sub-directory of your SigmaStudio installation. The default installation location is C:\Program Files\Analog Devices Inc\Sigma Studio\USB drivers

5. In XP click Continue Anyway if you are prompted with a dialog box saying the software hasn’t passed Windows Logo testing.

Using the USB Adapter

The functions of the following parts on the USB Adapter are:

- S1 - Load a program that is saved in flash to the SigmaDSP.

- S2 - Select which of 8 SigmaDSP programs will be loaded when S1 is pressed.
- S3 - Reset the USB Adapter board.
- TP1 - +5V connection for cases where the USB adapter isn't powered through the PC's USB port.
- TP2 - Ground connection for cases where the USB adapter isn't powered through the PC's USB port.
- J3 - When a jumper is present on this header, writing to the flash memory (for storing SigmaDSP programs) is enabled.
- D4 - This LED indicates that the USB board is powered.

SigmaDSP program and parameter files can be saved to flash memory on the USB Adapter using the Flash Downloader tool in SigmaStudio.  Each program can be loaded to the SigmaDSP by setting S2 to the appropriate switch setting and pressing S1.

Note: For more information please see the Analog Devices website and navigate to your board's datasheet:

http://analog.com/sigmadsp

## Parallel Port

Another way to establish communication from your computer to the eval board, rather than using USB Serial Converter is via the parallel port. After connecting the cable to the board and your computer follow these steps if you encounter problems establishing a connection.

Troubleshooting: Check Port Settings in Windows

1. Right Click **My Computer** on the desktop
2. Click the **Hardware** tab
3. Click **Device Manager**
4. Click the plus sign beside **Ports (COM & LPT)** in the list
5. Double click your Printer Port (LPT1)
6. On the **Port Settings** tab make sure **Filter Resource Method** is set to **Never use an interrupt**.
7. Make sure **Enable legacy Plug and Play** detection is checked and the **LPT Port Number** is set to **LPT1**.
8. On the **Resources** tab make sure one of the **Input/Output Range** settings is set to 0378-037F. If it isn't, un-check **Use automatic settings** and change the **Settings based on** field until you see the Input/Output Range set correctly.

Reboot and try to communicate with the Eval board again. If it still isn't working the port may be disabled in the system BIOS.

Check Parallel Port settings in system BIOS

Accessing BIOS is different depending on system manufacturer, and sometimes also different depending on model. Below are possible ways:

- Dell - press **F2** immediately upon boot
- HP/Compaq - press **F10** immediately upon boot
- Generic PC - press **Delete** immediately upon boot

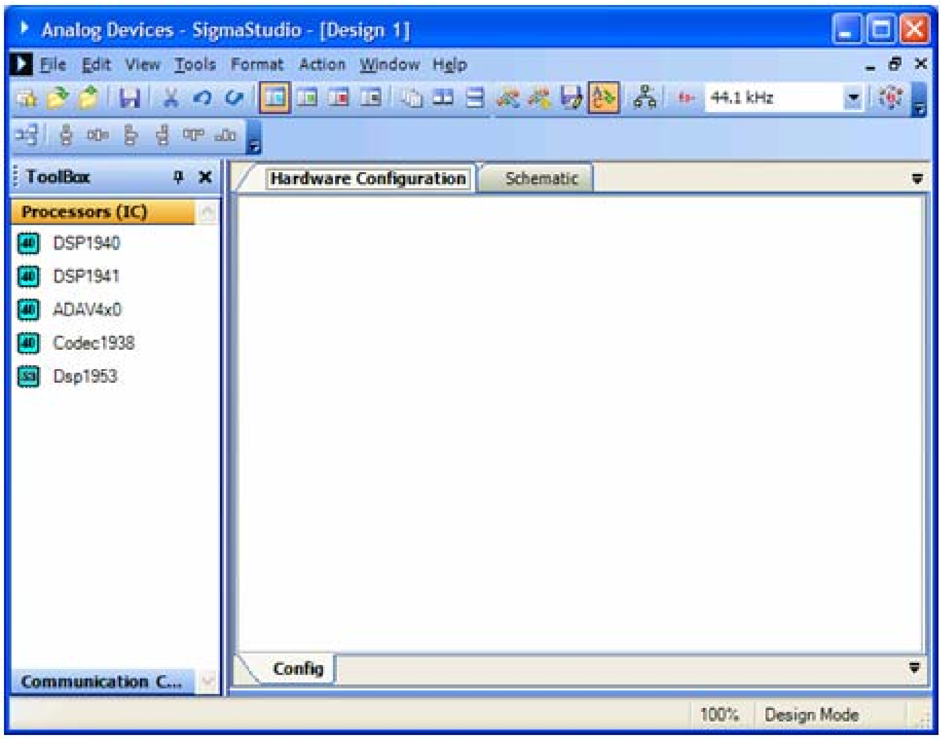
Once you enter BIOS, find the Parallel port setting and make sure it is set to EPP/ECP.

Exit BIOS making sure you save settings. Directions are usually available on screen on how to exit.

*We always recommend the usage of the USB Serial Converter rather than Parallel port .*

## Program window

When you first launch SigmaStudio you will see a gray screen with a menu and icon toolbar. From the **File** menu select either to **Open** an existing project or start a **New Project**. The following screen will appear for a new project.



By default the program opens with the Hardware Configuration tab open. This workspace allows you to choose which DSP board you will use.

In order for proper communication between your DSP board and your computer, you need to ensure that you have correctly set up the USB Serial Converter Setup connection.

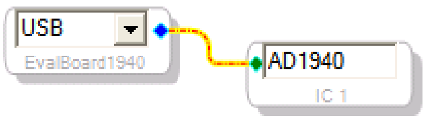
The other tab in the program window is for opening the Schematic workspace. This tab allows access to the system cells and also the algorithm cells for the DSP board selected.

# Hardware Configuration

The Hardware Configuration workspace allows you to set up communication between your hardware components. Follow the simple steps below to establish a connection between your DSP processor and hardware via a communication channel.

1. Having selected the Hardware Configuration tab, select the SigmaDSP for your application from the left column of **Processors (IC)** and drag it into the blank workspace of the Hardware Configuration.
2. Click on the **Communication Channels** menu.
3. Again select your installed board and drag it into the blank workspace. Note: The Communication Channel menu lists the names with the prefix "EvalBoard" and then the number of the board. These communication channels will still work for the same board types on target platforms that don't use the Evaluation Board setup. There is also a communication channel, USBSerialConv that is a generic use channel. See the USB Serial Convertor Communication Channel page for more information.
4. Click on the arrow on the cell box and select whether your connection is **Parallel** or **USB**. Connect the two cells by drawing a wire between the communication cell and the processor cell.

For a USB connection using the AD1940, the cells in your workspace should look something like this:

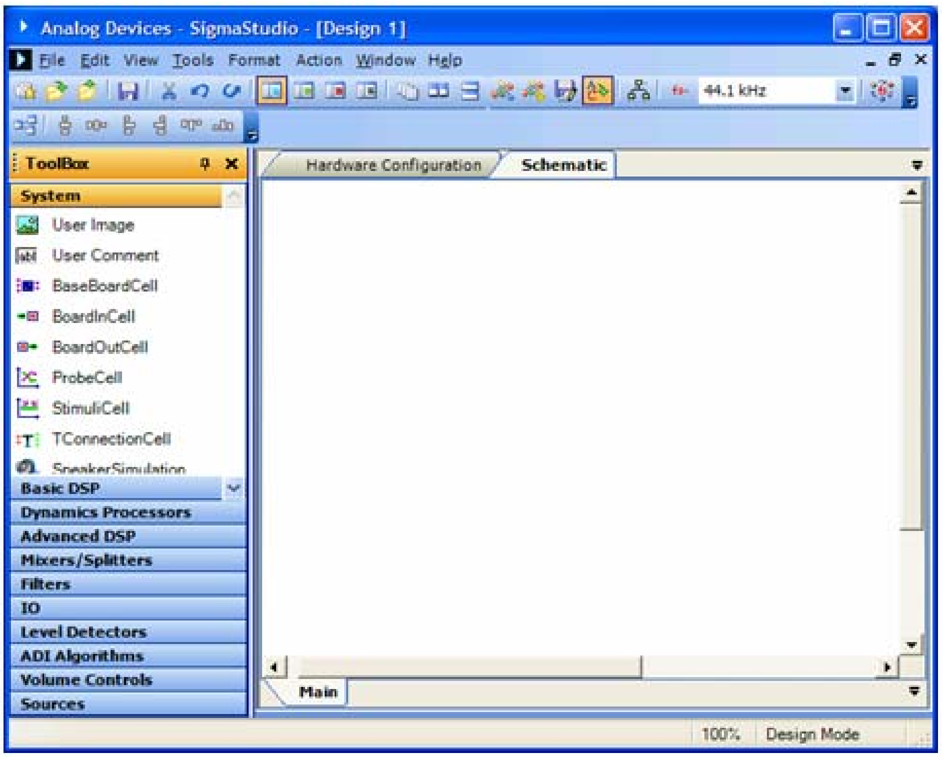


Note: The color of the drop-down menu of the communication cell indicates whether a proper channel has been established. If you haven't setup your hardware for USB connection the background color will be red. This configuration can be done in USB Serial Converter Setup.

See the Hardware Configurations section of this manual to learn about the more advanced operations that can be done under the Hardware Configurations tab.

# Schematic

Now that your hardware is setup you are ready to begin construction of your DSP design. This is done in the workspace of the Schematic tab in SigmaStudio. As can be seen below the toolbox is on the left with access to various libraries containing cells that can be dragged and dropped into the blank workspace on the right.



You have now finished the Getting Started portion of the help manual and are ready to begin designing. The following sections in this manual will describe the features and algorithms of the Toolbox in the Schematic workspace. If you are ready to try to setup a simple application, you can skip to the Tutorials and follow the examples for Stereo Audio with Volume Control and 5-band EQ and Probe and Stimulus Blocks. Otherwise, you can learn about each of the individual cells and their functions under the ToolBox section, or get more detailed information about the Algorithms driving the cells. You can also take a look at the Hardware Configurations section for advanced operations and control of the hardware, learn more about SigmaStudio functions in the SigmaStudio System section, or see more in-depth information about microcontroller considerations under the Basic SigmaDSP Architecture.

## Numeric Formats

It is common in DSP systems to use a standardized method of specifying numeric formats. Fixed point numbers are specified by an A.B format, 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. The format is as follows:

Numerical Format: 5.23

Range: –16.0 to (+16.0 − 1 LSB)

Examples:

1000 0000 0000 0000 0000 0000 0000 = –16.0

1110 0000 0000 0000 0000 0000 0000 = –4.0

1111 1000 0000 0000 0000 0000 0000 = –1.0

1111 1110 0000 0000 0000 0000 0000 = –0.25

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

0000 0000 0000 0000 0000 0000 0000 = 0.0

0000 0010 0000 0000 0000 0000 0000 = 0.25

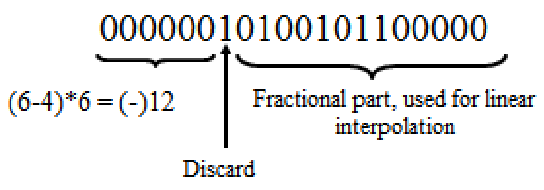
0000 1000 0000 0000 0000 0000 0000 = 1.0 (0 dB full scale)

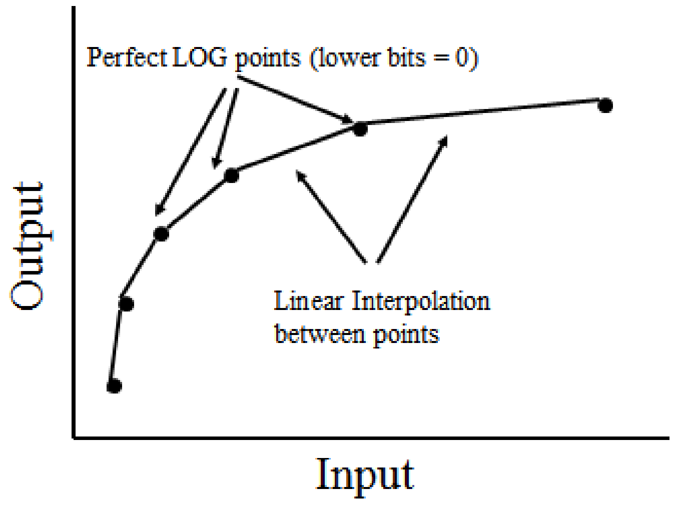
0010 0000 0000 0000 0000 0000 0000 = 4.0

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

Concept for dB conversion:

1. Count the number of leading 0's (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.




Perfect LOG points (lower bits = 0)
Linear Interpolation between points
Output
Input

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:

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

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 then integer part of this result, and convert 83886 to hex, and you will get 0x00147AE

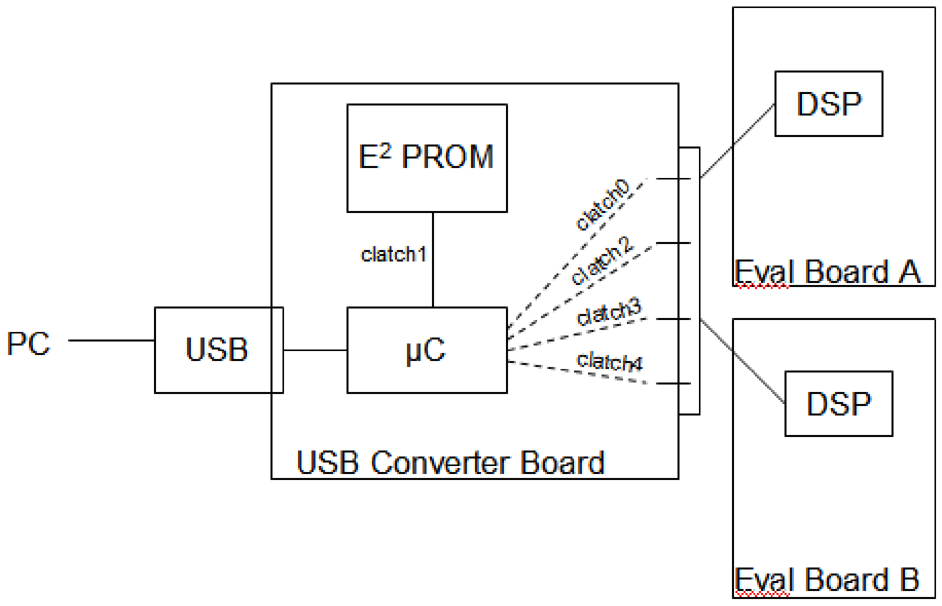
The output of some blocks is a 5.19 number. The formula to determine the dB output value from the readback value is the following:

dB_value = 96.32959861 * (readback_value / $2^{19}$ - 1)

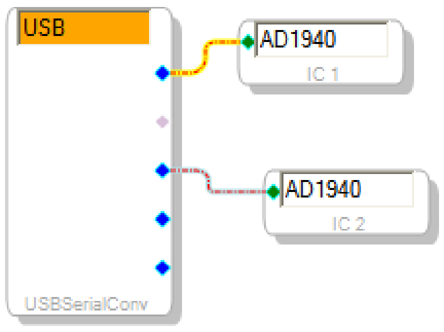
## USB Serial Convertor Communication Channel

The USB Serial Convertor Communication Channel cell offers connections for multiple processors. All connections made to this cell must be for SPI data. This channel offers the flexibility of connecting 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 convertor EPROM. This is represented on the cell by a grayed pin. The pins from top to bottom are clatch0, clatch1, clatch2, clatch3, and clatch4. The diagram below shows one possible configuration of connecting boards to different pins.

Note: All pins need not be connected, also you can choose the order to connect pins to suit your own needs.



In SigmaStudio:



# Hardware Configuration

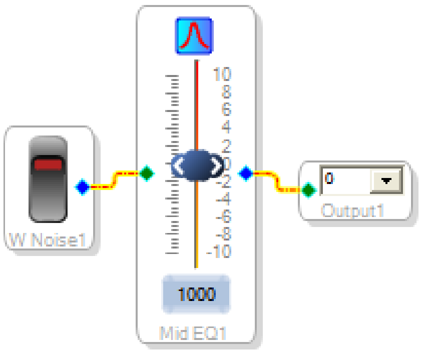
Under the Hardware Configuration tab you have access to many controls of the Hardware. The following pages describe the functionality of these tools. If you have not yet set up the communication channel for your board please look at the Hardware Configuration page under the Getting Started section of this manual.

- Capture Output Data
- Flash Downloader
- Register Control Window
- Register Read/Write Window

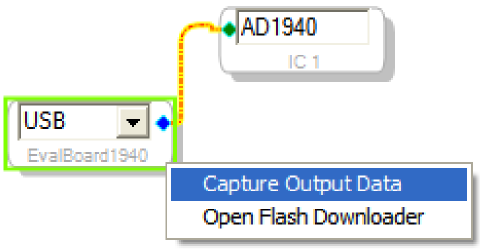
## Capture Output Data

The capture window allows you to see the exact data SigmaStudio is sending to the board. When it initially opens it is empty. When the "Link-Compile-Download" button is pressed you will see code that is downloaded to the board. The capture window allows you to see the exact coefficients, parameter addresses, and parameter data being sent to the board in real time. Whenever you make adjustments to sliders or control knobs in your schematic, and it has been compiled, you can see the data being updated in the Capture Output Data window.

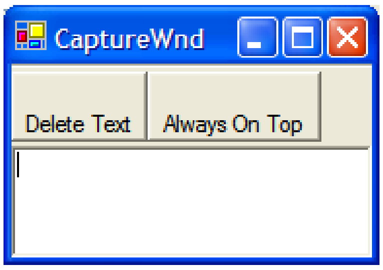
The following is an example for what you can expect to see in the Capture Output Data window using the AD1940. For the purpose of this example, the sample schematic seen to the right, using a White Noise Source, Medium Size EQ Filter, and Output cell.



Click on the **Hardware Configuration** tab, and then access the Capture Output Data window, by right-clicking the Communication Channels cell and selecting **Capture Output Data**, as seen to the right.



The Capture Window will now appear, with a blank screen. You have the option to click **Always On Top** so that you can work on SigmaStudio while being able to constantly see the data window.



At this point you can click back to the Schematic tab of your workspace and compile your project. You will now see data being sent to the Capture Output Data window:

If you scroll back to the top of the Capture window you will see the order in which data is sent to the DSP. For the AD1940, the first set of code that is sent is to mute the board. You can see that the parameter address for muting is 2642 and the values being sent for the data are 0x00 0x00.

```
Block Mode Write

Time  8:10:49 - 357ms

Parameter Name=

Parameter Address= 2642

Parameter Data:

0X00 , 0X00 ,
```

The following block of code is the Program block at address 1024. The Program Data for the Program block code is represented in 5 bytes of HEX values.

```
Block Mode Write

Time  8:10:49 - 377ms

Parameter Name=

Parameter Address= 1024

Parameter Data:

0XFF , 0XF2 , 0X01 , 0X20 , 0X01 ,

0X00 , 0X08 , 0X00 , 0XE2 , 0X01 ,

0X00 , 0X08 , 0X00 , 0XC0 , 0X01 ,

0X00 , 0X00 , 0X00 , 0X00 , 0X01 ,

...
```

The following block of code is the Parameter

```
Block Mode Write
```

Data which is where you will see updates, once you make real-time changes to sliders, control knobs, etc. on your compiled schematic.

```
Time  8:10:55 - 253ms

Parameter Name=

Parameter Address= 0

Parameter Data:

0X00 , 0X12 , 0X38 , 0XB3 ,

0X00 , 0X00 , 0X00 , 0X00 ,

0X00 , 0X88 , 0X9E , 0X18 ,

0X0F , 0X5D , 0XE0 , 0X07 ,

...
```

The following block of code for the AD1940 is the Slew Ram data at address 2560.

```
Safeload Write

Time  8:10:55 - 814ms

Parameter Name=

Parameter Address= 2560

Parameter Data:

0X00 , 0X00 , 0X00 , 0X00 , 0X00 ,

0X00 , 0X00 , 0X00 , 0X00 , 0X00 ,

0X00 , 0X00 , 0X00 , 0X00 , 0X00 ,

0X00 , 0X00 , 0X00 , 0X00 , 0X00 ,

...
```

Finally the last block you will see is the unmute block. You can see that the code reaches address 2642, as it did with the mute command, but now the values in the Parameter Data have changed to unmute the DSP chip.

```
Block Mode Write

Time  8:10:55 - 994ms

Parameter Name=

Parameter Address= 2642

Parameter Data:

0X02 , 0X00 ,
```

At this point the Capture Window is filled with a lot of data, so you can click on **Delete Text** to continue without having the rest of the information still visible. Now that your

```
Safeload Write

Time  8:52:16 - 785ms

Cell Name= Mid EQ1

Parameter Name= EQ1940Single101
```

still visible. Now that your project is compiled, try moving the EQ slider to see the values being updated in the Capture Window and being sent to the DSP.

**Cell Name** - Same name as appears on the visual cell in SigmaStudio.

**Parameter Name** - The name describes the filter, which DSP, number of inputs, and numbers the frequency band.

**Parameter Address** - The address to which data is being sent.

**Parameter Value** - The decimal filter coefficient for that filter band.

**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.

```
Parameter Address= 2

Parameter Value   = 1.02249395847321

Parameter Data:

0X00 , 0X82 , 0XE1 , 0X15 ,


Safeload Write

Time  8:52:16 - 795ms

Cell Name= Mid EQ1

Parameter Name= EQ1940Single111

Parameter Address= 3

Parameter Value   = -1.90359771251678

Parameter Data:

0XFF , 0X0C , 0X56 , 0XE9 ,


Safeload Write

Time  8:52:16 - 806ms

Cell Name= Mid EQ1

Parameter Name= EQ1940Single121

Parameter Address= 4

Parameter Value   = 0.900589466094971

Parameter Data:

0X00 , 0X73 , 0X46 , 0X84 ,


Safeload Write

Time  8:52:16 - 837ms

Cell Name= Mid EQ1

Parameter Name= EQ1940Single131

Parameter Address= 5

Parameter Value   = 1.90359771251678

Parameter Data:
```

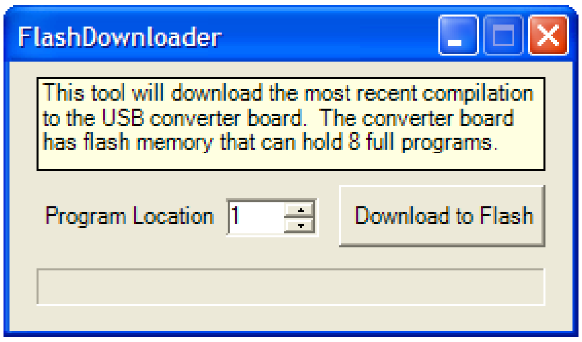
```
0X00 , 0XF3 , 0XA9 , 0X17 ,


Safeload Write

Time  8:52:16 - 879ms

Cell Name= Mid EQ1

Parameter Name= EQ1940Single141

Parameter Address= 6

Parameter Value   = -0.923083424568176

Parameter Data:

0XFF , 0X89 , 0XD8 , 0X67 ,
```

# Flash Downloader

The Flash Downloader lets you store compiled projects in the Flash RAM on the USB converter board. There are 8 locations available. Once a program is stored in the flash RAM, use the rotary switch on the SerialBoard to choose desired program and press the Program Load push button on the USB converter board to download the data. This allows very quick comparison between different projects. You can Link-Compile-Download each project to the Flash RAM and then switch between them easily.

In order to use the Flash Downloader:

1. Compile your project that you wish to download.
2. Click on the **Hardware Configuration** tab of the workspace
3. Right-click the Communication Channels Cell
4. Select **Open Flash Downloader**
5. Enter the Program Location to be stored
6. Click on **Download to Flash**



This tool will download the most recent compilation to the USB converter board. The converter board has flash memory that can hold 8 full programs.

Program Location 1

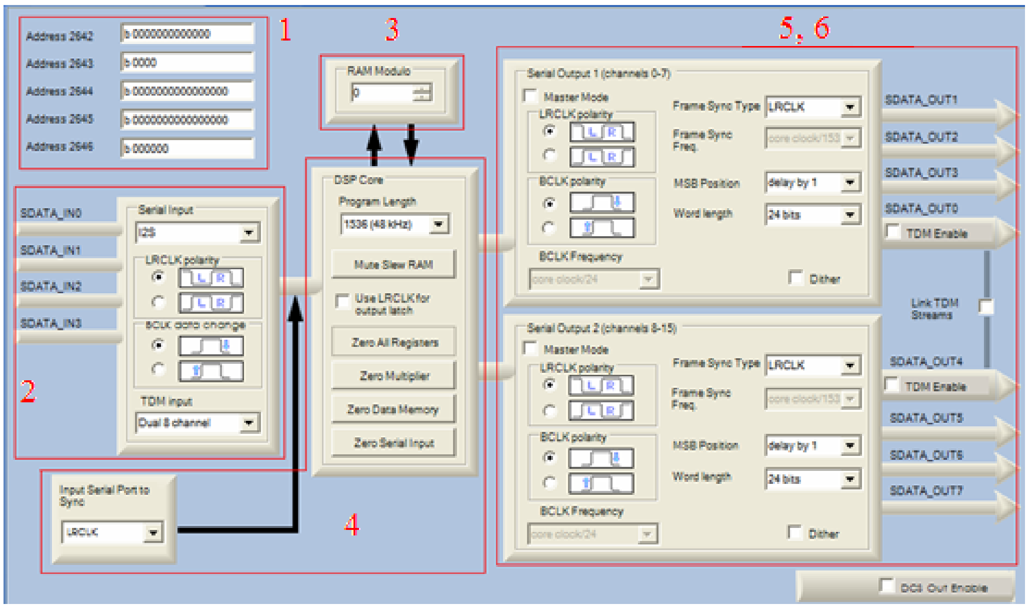
Now you project is accessible in the Program Location you entered by selecting that location on the USB Serial Converter Board and pushing the Program Load button.

Note: The Program Location number listed on the FlashDownloader pop-up window begins counting from 1, whereas the rotary switch on the USB Converter board begins with 0.

# Register Control Tab

The Register Control Window allows access to internal registers and DSP core settings.  It is divided into 6 sections.  Whenever you make changes, if your project is in Ready-Download mode the changes are immediately sent to the board. You can access the Register Control Window by clicking on the **Hardware Configuration** tab at the top of the workspace, and the clicking on the **Register Control** tab at the bottom of the Configuration workspace. The following describes each of the 6 sections of the Register Control tab, using the AD1940 as an example. Note: For more information please see the Analog Devices website and navigate to your part's datasheet:

http://analog.com/sigmadsp


Address 2642 b 0000000000000
Address 2643 b 0000
Address 2644 b 0000000000000000
Address 2645 b 0000000000000000
Address 2646 b 000000

RAM Modulo: 0

Serial Input: I2S
TDM input: Dual 8 channel
Input Serial Port to Sync: LRCLK

DSP Core
Program Length: 1536 (48 kHz)
Use LRCLK for output latch: ☐

Serial Output 1 (channels 0-7)
Master Mode: ☐
Frame Sync Type: LRCLK
Frame Sync Freq.: core clock/153
MSB Position: delay by 1
Word length: 24 bits
BCLK Frequency: core clock/24
Dither: ☐

Serial Output 2 (channels 8-15)
Master Mode: ☐
Frame Sync Type: LRCLK
Frame Sync Freq.: core clock/153
MSB Position: delay by 1
Word length: 24 bits
BCLK Frequency: core clock/24
Dither: ☐

TDM Enable: ☐
Link TDM Streams: ☐
TDM Enable: ☐
DCS Out Enable: ☐

### 1. Internal Registers

This area lists addresses 2642 through 2646 and their current status.  Whenever you make changes in the other areas of the window you will see the results here.  If you know the register bit locations you wish to change, then you can change them here also 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 Serial Input Control Register (address 2646)**

Controls control of 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/AD1941 Datasheet pages 27-29.
- **BCLK Polarity (Bit 3)** - This bit controls on which edge of the bit clock the input data is clocked. Data is clocked on the falling edge of BCLK_IN when this bit is set to 0, and on the rising edge when this bit is set at 1.
- **LRCLK Polarity (Bit 4)** - When set to 0, the left channel data on the SDATA_Inx pins is clocked when LRCLK_IN is low; and the right input data clocked when LRCLK_IN is high. When set to 1, this is reversed. In TDM mode, when this bit is set to 0, data is clocked in starting with the next appropriate BCLK edge following a falling edge on the LRCLK_IN pin. When set to 1 and running in TDM mode, the input data is valid on the BCLK edge following a rising edge on LRCLK_IN.
- **TDM Input (Bit 5)** - Setting this bit to 0 puts the AD1940/AD1941 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 in 16-channel TDM input mode, input on pin SDATA_IN2.

**3. Ram Modulo - affects 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 data-words 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/AD1941's DSP core.

- **Program Length (Bits 1:0)** - These bits set the length of the internal program. The default program length is 1536 instructions, but the program length can be shortened by factors of 2 to accommodate sample rates higher than 48 kHz.
- **Input Serial Port to Sequencer Sync (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/AD1941 may be used to decimate an incoming signal by some integer factor. In this case, it is desirable to synchronize the sequencer to a sub-multiple 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 click-free 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 is normally 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.  This means the AD1940/AD1941 will not pass a signal until a 1 is written to this bit. This is intended to prevent inadvertent noises from accruing during the power-up sequence.
- **Use LRCLK for Output Latch (Bit 10)** - Normally, data is transferred from the DSP core to the serial output registers at the end of each program cycle. In some cases (e.g., when output sample rate is set to some multiple of 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 this 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 only functional if slew RAM locations are used in the custom program design.

### 5,6. Serial Output 1 (address 2644) and Serial Output 2 (address 2645)

The output control registers give the user control of clock polarities, clock frequencies, clock types, and data format.  In all modes except for the right-justified modes (MSB delayed by 8, 12, or 16), the serial port accepts an arbitrary number of bits up to a limit of 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.

- **Word Length (Bits 1:0)** - These bits set the word length of the output data-word. Options are 16-bits, 20bits, or 24bits.
- **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/AD1941 is always MSB first.
- **Frame Sync 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 Sync 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.

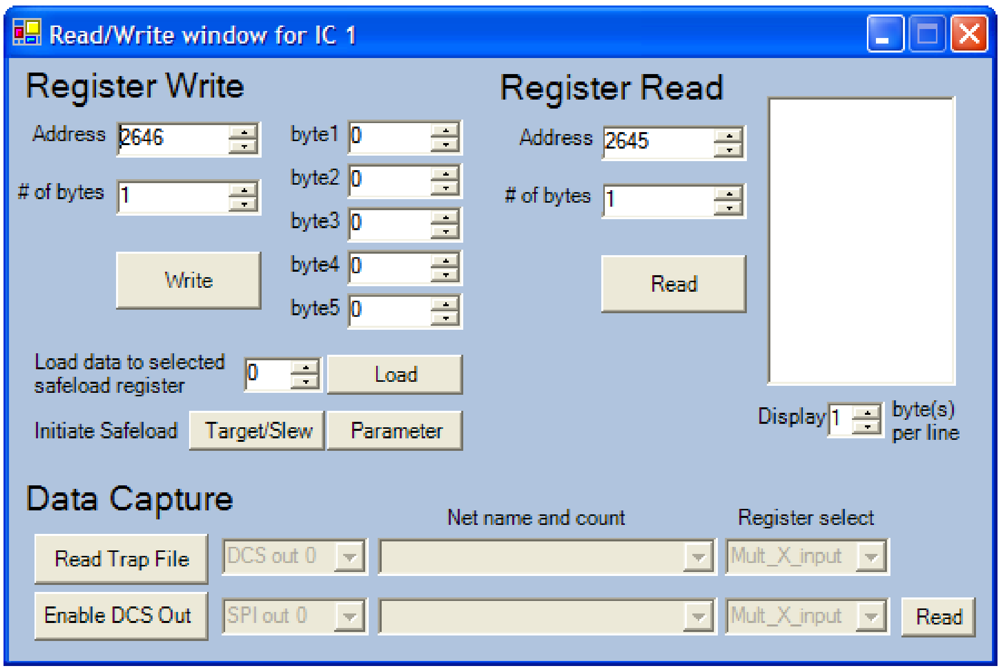
- **Master/Slave (Bit 11)** - This bit sets whether the output port is a clock master or slave. The default setting is slave; on power-up, Pins BCLK_OUTx and LRCLK_OUTx are set as inputs until this bit is set to 1, at which time they become clock outputs.
- **BCLK Polarity (Bit 12)** - This bit controls on which edge of the bit clock the output data is clocked. Data is clocked on the falling edge of BCLK_OUTx when this bit is set to 0, and on the rising edge when this bit is set at 1.
- **LRCLK Polarity (Bit 13)** - When set to 0, the left channel data is 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 is 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.

## Register Read/Write Window

The Register Read/Write window gives you to access to tools that allow you to read and write bytes of data from/to the DSP, and also perform Data Capture. In order to use the Register Read/Write Window:

1. Click on the **Hardware Configuration** tab
2. Right-click the Processor cell displaying which DSP you are using
3. Select **Show Register Read/Write Window**

Doing this will bring up the following window:



Read/Write window for IC 1

Register Write

Address 2646

# of bytes 1

byte1 0

byte2 0

byte3 0

byte4 0

byte5 0

Load data to selected safeload register 0

Register Read

Address 2645

# of bytes 1

Display 1 byte(s) per line

Data Capture

Net name and count

Register select

DCS out 0

SPI out 0

Mult_X_input

Mult_X_input

**Register Read/Write**

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

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

Note: Only 28 bits of the 32 available are used, so the 4 most significant bits 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. Also it is important to keep in mind that the Register Read is actually reading parameters back from the DSP, whereas the Capture Output Data and *.params file are reading the parameters being sent to the DSP.

**Data Capture**

Allows any node in the signal processing flow to be sent to control port-readable registers or to a digital output pin, provided it is 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). The "Enable DCS Out" button sets this bit for you.

- Read the generated trap.dat file with the "Read Trap File" button. This 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 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. Available options are:
    - Mult_X_input - Multiplier X Input
    - Mult_Y_input - Multiplier Y Input
    - MAC_out - Multiplier-Accumulator Output
    - Accum_fback - Accumulator Feedback
- Click the "Read" button to see the data. The captured data is in 5.19 twos complement data format. The 4 LSBs are truncated from the internal 5.23 data-word.

# SigmaStudio System

The following pages help describe the basic system functions of the SigmaStudio software:

- Sampling Rate Considerations
- Toolbar Window
- Linking your Project
- Link/Compile/Download your Project
- Generating Parameter File
- Allow Realtime AB testing between projects
- Freeze Selected Schematic
- System Implementation
- Filter Table Generator

## Sampling Rate Considerations

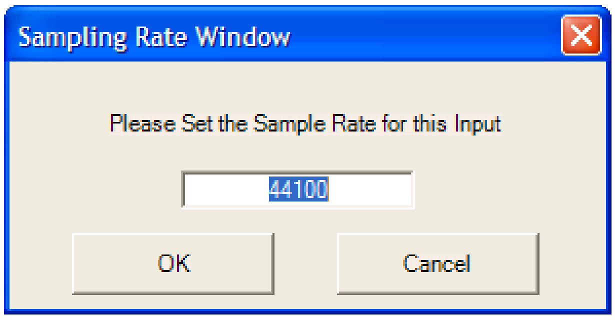
The number of instructions available on the DSP (MIPS) will vary depending on the selected sampling frequency. The AD1940/1941 DSPs can perform:

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

The sample rate should be set by the Program Length bits in the Core Control Register. SigmaStudio defaults to 44.1Hz sampling rate, if you want to use any other sampling rate it should be selected prior to the design from the New Item Sample Rate drop-down menu from the Toolbar. Now any blocks added to your schematic will be set to that particular sampling rate.

You can also change the sampling-rate mid design by changing the sample rate on the inputs (right-click all input or source cells and select

Set Sample Rate). The figure included below is the window that allows you to change the sample rate on the input and source cells.

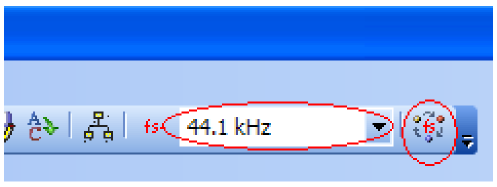


Sampling Rate Window

Please Set the Sample Rate for this Input

44100

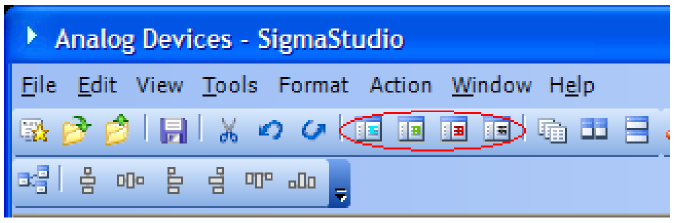
Note: for the rest of the schematic to retain the changes made to the inputs for sampling rate, you must click on the Propagate Sampling rate button.



## Toolbar Window Options

From the Toolbar menu you have access to 4 window buttons which are circled below. Each button allows you to bring up or hide one of the following windows:

- ToolBox
- DSP Resources
- Output
- Spy Window


Analog Devices - SigmaStudio
File Edit View Tools Format Action Window Help

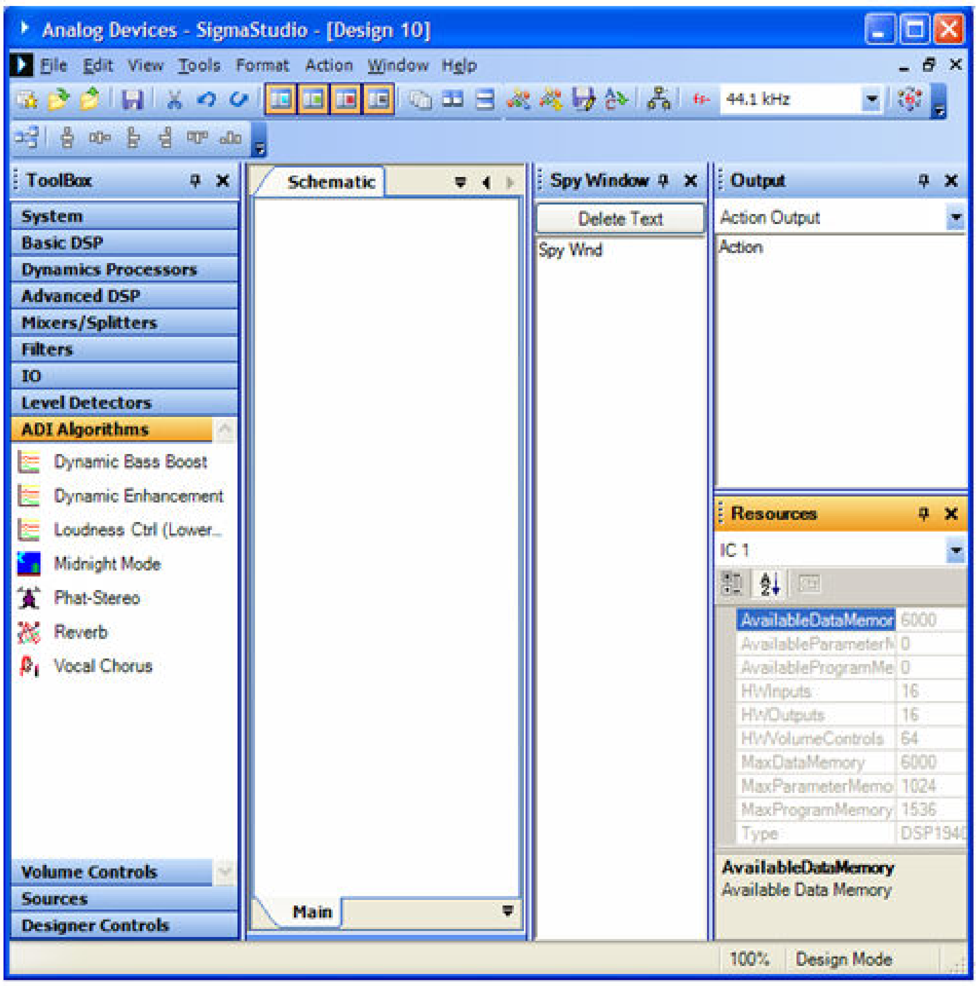
**ToolBox -** The ToolBox window contains all the basic building blocks to construct your design. When you click on a library topic the list expands to show you the available cells to be dragged and dropped into the Schematic workspace. Check the ToolBox page to learn about the individual libraries and cells available from the ToolBox.

**DSP Resources** - The DSP Resources window allows you to view the available resources you have left depending on your project complexity. You are able to monitor resources such as program RAM, parameter RAM and other hardware-specific resources.

**Output** - The Output window is a quick way to view the files generated upon linking, compiling, and general action in the schematic workspace. This is a faster alternative to having to open the IC 1 folder to check the output files generated upon compilation. For more information about those files see Compile/Downloading your Project.

**Spy Window** - The Spy window is a quick way to view parameter data information without having to Capture Output Data or Generate a Parameter File.

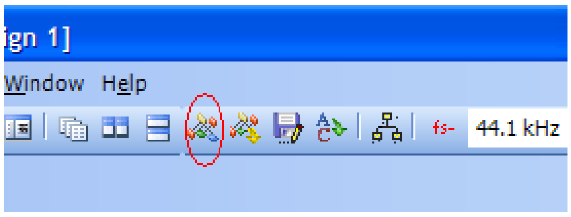
The figure below shows the workspace with all the windows open:



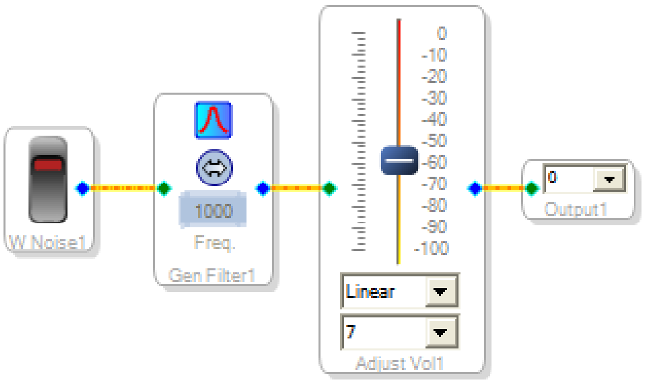
| | |
|---|---|
| AvailableDataMemor | 6000 |
| AvailableParameterN | 0 |
| AvailableProgramMe | 0 |
| HWInputs | 16 |
| HWOutputs | 16 |
| HWVolumeControls | 64 |
| MaxDataMemory | 6000 |
| MaxParameterMemo | 1024 |
| MaxProgramMemory | 1536 |
| Type | DSP194 |

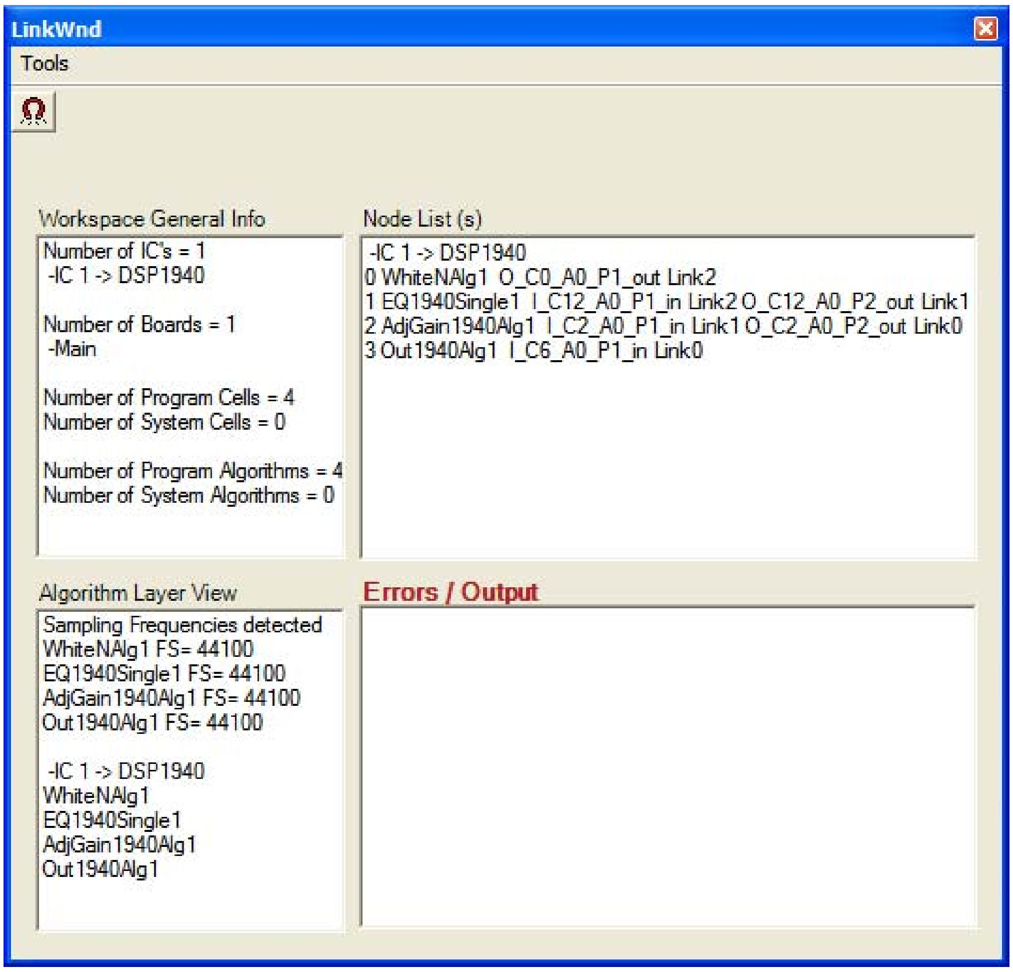
# Linking your Project

Linking your project is the step before compiling/downloading your project to the DSP. Upon linking, you cannot control your design, but you will be presented with a window giving you information about your design. Any errors in your design will be reported as well as information about the algorithms used in your design and general system information. The figure below shows you the Link button circled:



For the following sample schematic, this is what to expect to see in the Link Window:





```
Number of IC's = 1
 -IC 1 -> DSP1940

Number of Boards = 1
 -Main

Number of Program Cells = 4
Number of System Cells = 0

Number of Program Algorithms = 4
Number of System Algorithms = 0
```

```
 -IC 1 -> DSP1940
0 WhiteNAlg1  O_C0_A0_P1_out Link2
1 EQ1940Single1  I_C12_A0_P1_in Link2 O_C12_A0_P2_out Link1
2 AdjGain1940Alg1  I_C2_A0_P1_in Link1 O_C2_A0_P2_out Link0
3 Out1940Alg1  I_C6_A0_P1_in Link0
```

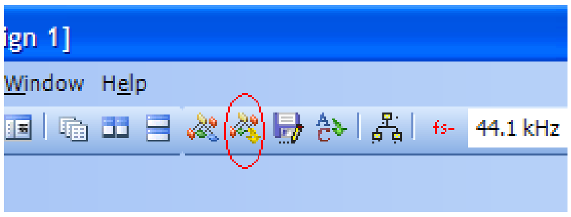
```
Sampling Frequencies detected
WhiteNAlg1 FS= 44100
EQ1940Single1 FS= 44100
AdjGain1940Alg1 FS= 44100
Out1940Alg1 FS= 44100

 -IC 1 -> DSP1940
WhiteNAlg1
EQ1940Single1
AdjGain1940Alg1
Out1940Alg1
```

# Link/Compile/Download your Project

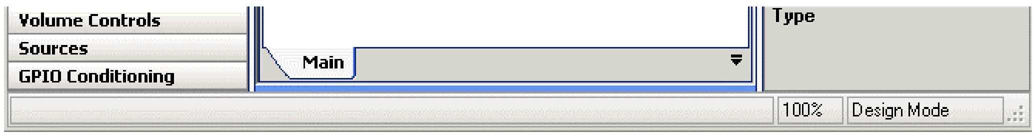
Clicking on the Link/Compile/Download button on the SigmaStudio toolbar of your project, actually sends your design code from your schematic, to the DSP. When it is 100% Ready indicated by the green bar below the workspace, your program is now responsive to real-time changes made to sliders, control knobs etc. in order to actively control the parameters in your design. Upon compiling your project, there are certain files that are generated. The list of files and their descriptions are as follows:

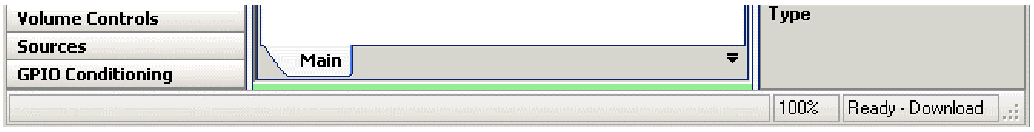
- hex_program_data.dat - load file for downloading code using microcontroller in hex format
- program_data.dat - load file for downloading code using ADI loader
- hex_program_simdata.dat - load file for downloading code using microcontroller, with no commas, x's, or spaces, useful format for Verilog 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.

The figure below shows you the Link/Compile/Download button circled:



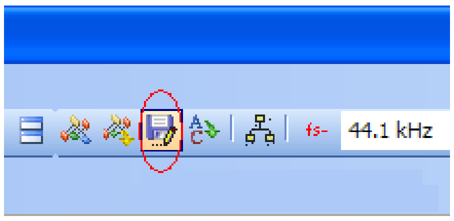
The figures below shows you how the blue bar below the work space turn green and indicates 100% Ready when successful, on Clicking the Link/Compile/Download button on the SigmaStudio toolbar of your project.



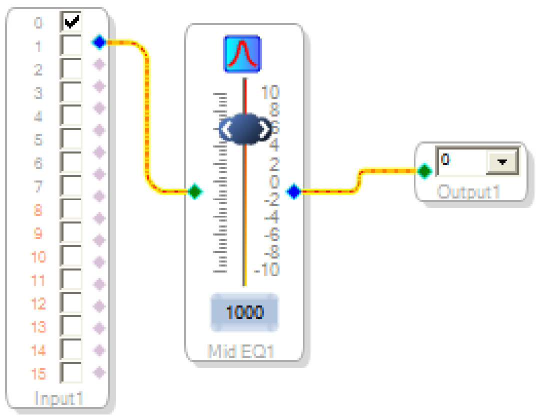

Volume Controls
Sources
GPIO Conditioning
Main
Type
100%
Ready - Download

## Generating Parameter File

Generating a Parameter File allows you to see the data that is sent to the DSP upon compilation. This file is very helpful for finding the parmaters and addresses to integrate into microcontroller code. After your project is compiled you can select the Generate Parameter File icon. You need to compile your project every time you wish to generate the parameter file in order to have the accurate up-to-date information for your current project. You will then be prompted to enter a file destination for a ***.params** file. This file can then be opened in any basic text editor for analysis. Upon the creation of this ***.params** file, another file, ***.hex** is also generated. This file contains just the list of the parameter values in the ***.params** file without the Parameter names and addresses.



For the following sample schematic, this is what to expect in your *.params file:



Cell Name = Mid EQ1

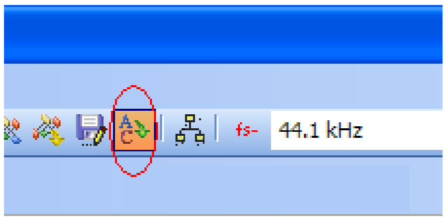
```
Parameter Name    = EQ1940Single101

Parameter Address = 0

Parameter Value   = 1.03425681591034

Parameter Data :

0X00 , 0X84 , 0X62 , 0X87 ,


Cell Name   = Mid EQ1

Parameter Name    = EQ1940Single111

Parameter Address = 1

Parameter Value   = -1.9115926027298

Parameter Data :

0X0F , 0X0B , 0X50 , 0XEF ,


Cell Name   = Mid EQ1

Parameter Name    = EQ1940Single121

Parameter Address = 2

Parameter Value   = 0.896903276443481

Parameter Data :

0X00 , 0X72 , 0XCD , 0XBA ,


Cell Name   = Mid EQ1

Parameter Name    = EQ1940Single131

Parameter Address = 3

Parameter Value   = 1.9115926027298

Parameter Data :

0X00 , 0XF4 , 0XAF , 0X11 ,


Cell Name   = Mid EQ1

Parameter Name    = EQ1940Single141

Parameter Address = 4
Parameter Value   = -0.931160092353821

Parameter Data :

0X0F , 0X88 , 0XCF , 0XBF ,


Parameter data for: IC 1 (Hexadecimal format starting
at parameter address 0)

See also C:\Program Files\Analog Devices Inc\Sigma
Studio\sshelp\exmprobo.hex file

0X00 , 0X84 , 0X62 , 0X87 ,

0X0F , 0X0B , 0X50 , 0XEF ,

0X00 , 0X72 , 0XCD , 0XBA ,

0X00 , 0XF4 , 0XAF , 0X11 ,

0X0F , 0X88 , 0XCF , 0XBF ,


Parameter data for: IC 1 (Binary format starting at
parameter address 0)

00000000 10000100 01100010 10000111

00001111 00001011 01010000 11101111

00000000 01110010 11001101 10111010

00000000 11110100 10101111 00010001

00001111 10001000 11001111 10111111
```

Note: The *.hex file only contains the following:

```
0X00 , 0X84 , 0X62 , 0X87 ,

0X0F , 0X0B , 0X50 , 0XEF ,

0X00 , 0X72 , 0XCD , 0XBA ,

0X00 , 0XF4 , 0XAF , 0X11 ,

0X0F , 0X88 , 0XCF , 0XBF ,
```

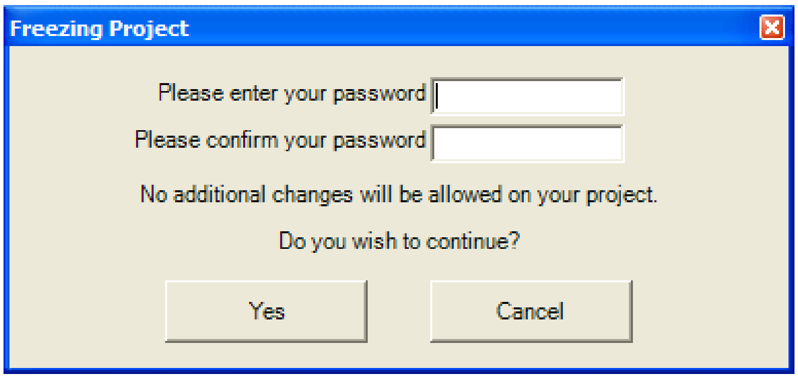
# Allow Realtime AB Testing Between Projects

Use of the Realtime AB Testing icon, allows to switch back and forth between any number of open projects that are compiled, without having to recompile each one again, when you bring the window to the front. Without this enabled you must recompile a project every time you bring it to the front, even if you have not made any changes to the schematic. This is very useful to have so that you can do realtime testing between multiple projects.



## Freeze Selected Schematic

Freezing your project allows you to secure a current design so that no additional changes or edits can occur. This can safe-guard your design against accidental changes and also give you private protection and control. When you click on the Freeze Selected Schematic icon, and window appears asking for your password:



Freezing Project

Please enter your password

Please confirm your password

No additional changes will be allowed on your project.

Do you wish to continue?

Enter a password and click Yes if you wish to continue. To make any further changes to your design, you will have to re-click the Freeze button and enter your password to be able to make modifications to the design.

# System Implementation

After you have finished designing your schematic and successfully compiled your project, you are ready to integrate your design into microcontroller code. The basic steps you take are as follows:

1. Obtain parameters and addresses from SigmaStudio
2. Parse file with parameter information
3. Integrate into microcontroller code

In order to access the parameters and addresses from SigmaStudio, you must have compiled your project first and then generated a parameter file. For more information on this procedure check the Generating Paramater File page. Once you have done this you will now have two files available to you. Note you can also view a sample *.params and *.hex file on the Generating Parameter File page.

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

The following code is an example of integrating your parameters into microcontroller code. The highlighted functions in the main function are also included:

```
    #include "prog_data.h"

    #include "param_data.h"

    #include "comms.h‟

    #include ‟regvals.h”


    #define byte unsigned char


    //Microcontroller Code MAIN program

    int main(void)

    {


SpiInit(); // setup SPI port – this is hardware-specific
Mute();

//Write a program to DSP

SpiWrite(PROG_START_ADR, PROGRAM_DATA, PROG_LENGTH*6);

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

//Write a parameter file

SpiWrite(PARAM_START_ADR, PARAM_DATA, PARAM_LENGTH*4);

Wait(...);


UnMute();


//Write a single value (0.242) to specified address (43)

ParamWrite(43, 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) load the data to each of the five safeload

// locations, then initiate the safeload

SafeloadMultipleParamWrite(5, 1.999, 0);

SafeloadMultipleParamWrite(6, -.988, 1);

SafeloadMultipleParamWrite(7, 1.999, 2);

SafeloadMultipleParamWrite(8, -.999888, 3);

SafeloadMultipleParamWrite(9, 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(20); //Wait 20us
    }

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

    //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 buffer has been emptied (i.e. write has completed) – this is hardware-specific

```
    //Microcontroller Code - functions, write a single parameter

    void ParamWrite(short address, float param)

    {

byte param_hex[4] = {0x00, 0x00, 0x00, 0x00} ;

//convert decimal paramater 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

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);

        }
```

Note: AFTER  you call this function, then call the `InitSafeloadParam` function once

```
    // 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

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

//Fill in you 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 buffer has been emptied (i.e. 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;


safe_addr_lo=(byte)safe_addr;  //get low byte of register/RAM address

//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;


param_addr_lo=(byte)param_addr;  //get low byte of parameter address

//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;


safe_param_lo=(byte)safe_param;  //get low byte of register/RAM address

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


SPI_TX_REG = 0x00; //Write to DSP at chip address 0

SPI_TX_REG = safe_param_hi; //Write high byte of Safeload address

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;

  param223 = param_dec * (1 << 23); //multiply decimal number by 2^23

  param227 = param223 + (1 << 27); //convert to positive binary


  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 created using the data from the Generate Parameter File. They should look like the following:

```
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 ,

  ...

  }
```

Note: The length of the array is 1024 coeffs * 4 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 ,

...

}
```

Note: The length of the array is 2560 instructions * 6 bytes (for ADAV4xx)

The Microcontroller code register header file should look like the following:

```
#define CORE_CONTROL_REG 0x041C

#define SER_OUT_REG 0x041E

#define SER_IN_REG 0x041F


#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


#define PROG_START_ADR 1024

#define PROG_LENGTH 2560

#define PARAM_START_ADR 0

#define PARAM_LENGTH 1024
```

The 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:

```
    static MyBlockEnable

    //toggle the block to turn ON (no clicks) using target/slew RAM

    {

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 taken from the .params file in SigmaStudio – for example:

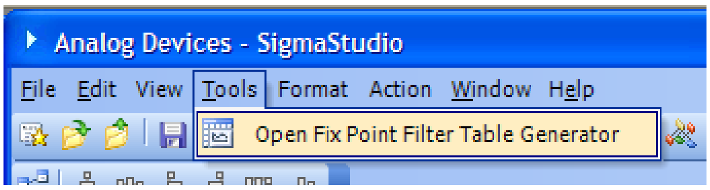
```
Parameter Name    = myblock40015

Parameter Address = 4096

Parameter Data    = 0 40 80 0 0
```

Routine to write to target RAM via safeload:

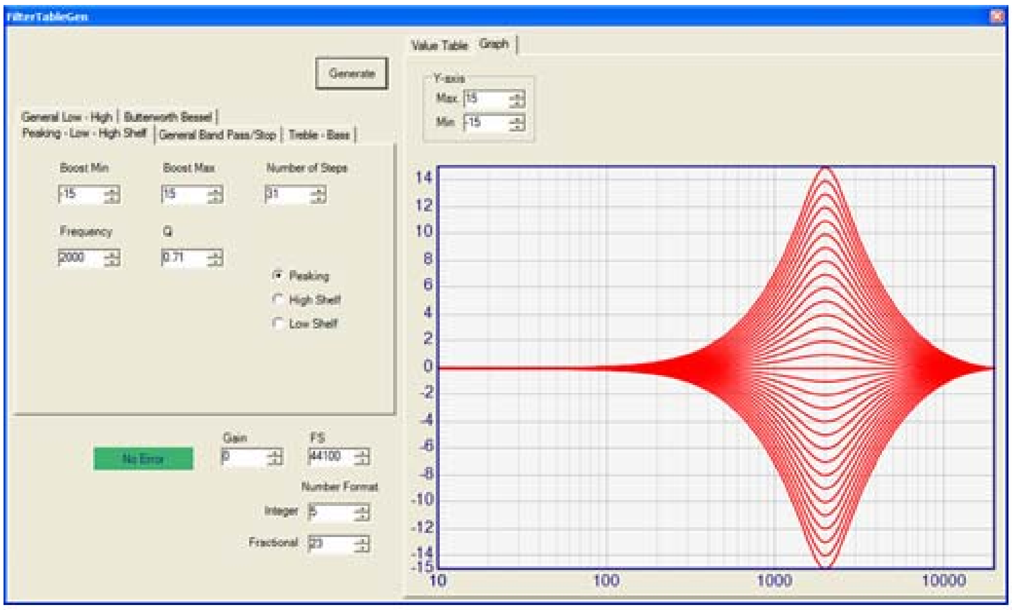
```
    static SafeloadTargetRAM(TargetRamAddress, TargetRamData) {

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

    }
```

# Filter Table Generator

The Filter Table Generator allows you to easily set coefficients so that a fixed frequency can change between different levels. This is a useful for applications such as an equalizer slider. You can access the Filter Table Generator by clicking on the **Tools** menu-bar and selecting **Open Fix Point Filter Table Generator**:


Analog Devices - SigmaStudio
File
Edit
View
Tools
Format
Action
Window
Help
Open Fix Point Filter Table Generator

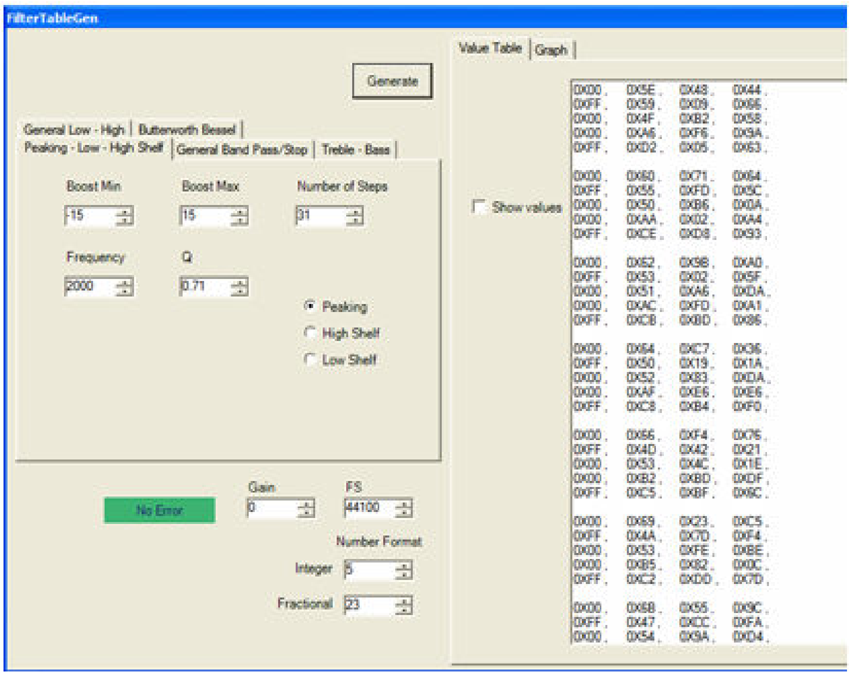
The table generator tool allows the user to set all filter parameters, and then specify the number of steps and the step size. This example shows a 2 kHz peaking filter with Q=0.71 and a boost/cut of 15 dB with 1 dB steps.



Boost Min: -15
Boost Max: 15
Number of Steps: 31
Frequency: 2000
Q: 0.71
[✓ Peaking] High Shelf Low Shelf
Gain: 0
FS: 44100
Integer: 5
Fractional: 23
Max: 15
Min: -15

As well as showing a frequency response plot of the filters, the tool also generates a list of coefficients. These coefficients can be displayed in both decimal and Parameter

RAM-ready hexadecimal values. These hex values are sign extended from the fourth bit.



Boost Min: -15
Boost Max: 15
Number of Steps: 31
Frequency: 2000
Q: 0.71
[✓ Peaking] High Shelf Low Shelf
Gain: 0
FS: 44100
Number Format
Integer: 5
Fractional: 23
☐ Show values

```
0X00 ,  0X5E ,  0X48 ,  0X44 ,
0XFF ,  0X59 ,  0X09 ,  0X66 ,
0X00 ,  0X4F ,  0XB2 ,  0X58 ,
0X00 ,  0XA6 ,  0XF6 ,  0X9A ,
0XFF ,  0XD2 ,  0X05 ,  0X63 ,

0X00 ,  0X60 ,  0X71 ,  0X64 ,
0XFF ,  0X55 ,  0XFD ,  0X5C ,
0X00 ,  0X50 ,  0XB6 ,  0X0A ,
0X00 ,  0XAA ,  0X02 ,  0XA4 ,
0XFF ,  0XCE ,  0XD8 ,  0X93 ,

0X00 ,  0X62 ,  0X9B ,  0XA0 ,
0XFF ,  0X53 ,  0X02 ,  0X5F ,
0X00 ,  0X51 ,  0XA6 ,  0XDA ,
0X00 ,  0XAC ,  0XFD ,  0XA1 ,
0XFF ,  0XCB ,  0XBD ,  0X86 ,

0X00 ,  0X64 ,  0XC7 ,  0X36 ,
0XFF ,  0X50 ,  0X19 ,  0X1A ,
0X00 ,  0X52 ,  0X83 ,  0XDA ,
0X00 ,  0XAF ,  0XE6 ,  0XE6 ,
0XFF ,  0XC8 ,  0XB4 ,  0XF0 ,

0X00 ,  0X66 ,  0XF4 ,  0X76 ,
0XFF ,  0X4D ,  0X42 ,  0X21 ,
0X00 ,  0X53 ,  0X4C ,  0X1E ,
0X00 ,  0XB2 ,  0XBD ,  0XDF ,
0XFF ,  0XC5 ,  0XBF ,  0X6C ,

0X00 ,  0X69 ,  0X23 ,  0XC5 ,
0XFF ,  0X4A ,  0X7D ,  0XF4 ,
0X00 ,  0X53 ,  0XFE ,  0XBE ,
0X00 ,  0XB5 ,  0X82 ,  0X0C ,
0XFF ,  0XC2 ,  0XDD ,  0X7D ,

0X00 ,  0X6B ,  0X55 ,  0X9C ,
0XFF ,  0X47 ,  0XCC ,  0XFA ,
0X00 ,  0X54 ,  0X9A ,  0XD4 ,
```

# Toolbox

The toolbox contains all the basic building blocks to construct your design. When you click on a library topic the list expands to show you the available cells to be dragged and dropped into the Schematic workspace. Depending on which DSP you are using, certain cells may not be available.

There are some basic usage tips for the cells found on the Using Cells page that will help you get the most out of your design.

Below are the library tabs in the ToolBox containing cells

- System
- Basic DSP
- Dynamic Processors
- Advanced DSP
- Mixers/Splitters
- Filters
- IO
- Level Detectors
- ADI Algorithms
- Volume Controls
- Sources
- GPIO conditioning

# Using Cells

In order to construct your DSP design you need to drag and drop cells from your ToolBox into the Schematic workspace of your project. The following are certain tips that will help you get the most out of making your design.

- Selecting Cells
- Creating Aliases
- Adding Algorithms
- Growing Algorithms
- Editing Cell Parameters

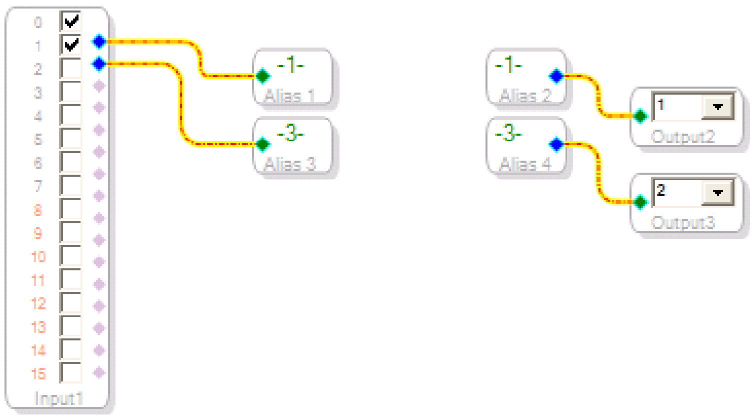
## Selecting Cells

To select a cell, in order to either delete or move it, you must click somewhere on the border of the cell or on its label.  Usually left-clicking the cell label name is the easiest.   Holding down the SHIFT or CTRL keys allows you to click and select multiple cells. Successful selection of a cell will be indicated by a light-green outline of the cell.

To select multiple cells quickly without pressing SHIFT or CTRL click and drag a box around the cells. Note: this will select every component that is completely enclosed in the box, otherwise it will not be selected.

## Creating Aliases

To organize your project visually on the Schematic workspace, it might be helpful to create an alias to provide a "jump" in the signal flow. Right-click on any cell's blue pins and select **Alias.** You must click directly on the pin to get the menu to open.  When you click the Alias button two blocks will appear, the input and output. These blocks are already connected, but visually they allow a jump in the signal flow.  Connect a wire from the blue pin of your cell to the green pin of the Alias 1 block. Then from the Alias 2 block connect a wire from the blue pin the green pin of your next cell in the signal flow. The figure below shows a simple input cell using an alias to reach the output cell.



## Adding Algorithms

Some cells require that you add an algorithm for the cell to function and have input/output pins. For example when you drag and drop a Volume Control cell into the workspace, it is empty. You need to right-click the empty cell select Add Algorithm > IC N , and then you have your choice of gain with or without slew. Once you do this, the cell appears as it is meant to be used with its slider and input/output pins. From this point you can right-click the cell somewhere on the border of the cell or on its label to add another algorithm or remove an algorithm. By selecting Add Algorithm, you effectively add another set of I/O pins. Also it is important to note that by Adding algorithms, you have the option of selecting a new algorithm for computation. (If you have multiple DSP boards, adding an algorithm also lets you select the algorithm for a different board). In the same Volume Control example, once you have your default cell, when you add an algorithm you have the choice of selecting slew or no slew regardless of what algorithm your default cell is using.

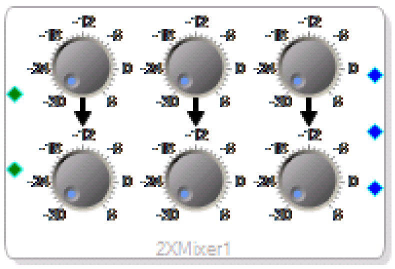
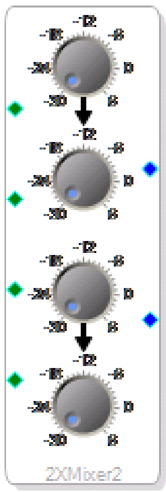
It is important to right click on the border of the cell or on its label to get to the pop-up menu for selection of adding an algorithm. Otherwise by right clicking in the center of the cell you get another pop-up window that allows you to enter the values for the parameters of that cell.

Note: By adding an algorithm not only are you choosing the method of computation for this cell, but you are also selecting a particular DSP association. This is important if you have multiple DSP boards connected. By adding algorithms to cells you can have shared cells that communicate to multiple DSP boards.

## Growing Algorithms

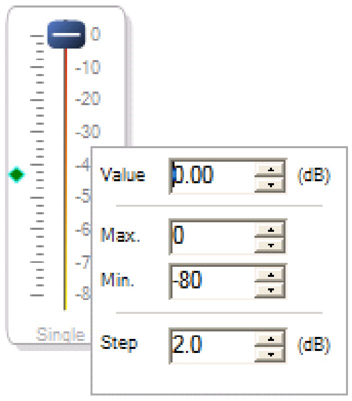
Growing Algorithms is different from Adding Algorithms in the sense that growing builds upon the existing algorithm of the cell whereas Adding Algorithms, adds a new algorithm to the cell. Growing algorithms maintains the same method of computation for the algorithm and also maintain the same DSP association.

The easiest way to see this distinction is with the Mixer cells. For example when you drag and drop a Two Input Cross Mixer into the workspace and you right-click the cell somewhere on the border of the cell or its label, you have the options: Grow Algorithm, Add Algorithm, Remove Algorithm. When you select Grow Algorithm you are choosing the extra number of columns (output mixes) for your cell. If you select Add Algorithm you are effectively adding another Two Input Cross Mixer to your cell. Look below for the diagrams showing the distinction between growing and adding algorithms.

| **Grow Algorithm** | **Add Algorithm** |
|---|---|
|  Method to get this particular cell: Right Click original cell and select: **Grow Algorithm** > **1. 2 Channel X Mixer** > **2** |  Method to get this particular cell: Right Click original cell and select: **Add Algorithm** > **IC N** > **2 Channel X Mixer** |
| As can be seen from the picture, this cell still only has 2 inputs, but now has three outputs. These three output correspond to the three vertical set of control level knobs. By growing the algorithm, you can have three different mixes of the same two inputs. | As can be seen from this picture, this cell has two separate mix algorithms that allow for two inputs to be mixed down to one output. |

## Editing Cell Parameters

Certain cells have edit boxes, sliders, control knobs and other devices that allow you to edit the parameters they control. On the cell you can either move the devices with your mouse or you can right click the middle of the cell to open the pop-up window that allows you to enter precise values for the parameters. The figure below shows the edit window that you get when you click in the middle of the cell of the Single Volume Control cell.



Value 0.00 (dB)

Max. 0

Min. -80

Step 2.0 (dB)

# System

The System library in the ToolBox allows you basic control of your design layout and response. These cell functions are not driven by DSP algorithms and do not use any instructions, yet they are useful in analyzing and organizing your design. The System cells can be considered helper cells that don't own a DSP. The algorithms for these cells are built-in and don't require you to Add an algorithm because they are not associated with a specific DSP. The association is made to the DSP at link time after you have made the wire connections. The system cells then acquire the algorithm for the specific DSP and will not release this association until it is completely disconnected from the DSP and the relationship to the particular DSP is broken. The following list are the available cells under the System tab.

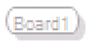
- User Image
- User Comment
- BaseBoardCell
- BoardInCell
- BoardOutCell
- ProbeCell
- StimuliCell
- TConnectionCell
- SpeakerSimulation
- Terminal

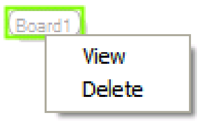
The following pages contain more specific information about the individual cells. Also take a look at the System Example page to see a sample schematic using some of the System cells.

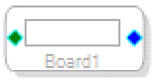
## BaseBoardCell

As your program becomes more and more populated with cells, it becomes necessary to have extra workspace area. The BaseBoardCell creates a new workspace page that you can use to organize your program flow. Each board is accessible by a tab at the bottom of the program window, with the original board labeled "Main." In order to use this cell:

1. Drag and drop into the workspace (currently the cell has no pins)
2. Right click on the cell and select **View** or click on the tab at the bottom of the program window labeled **Board1** in order to view the new workspace
3. Drag and drop a BoardInCell and a BoardOutCell into the Board1 workspace
4. Return to the Main page and wire your system to use the BaseBoardCell (now with pins) as a black-box in the Main page.







Note: When you are using multiple BaseBoardCells with their own BoardIn and BoardOut cells, you can keep track of which board cells belong to each other by placing your mouse above the input/output pins for a tooltip.

## BoardInCell

The BoardInCell is used to add an input pin to your BaseBoardCell. This allows you to bring the signal flow from the main page into the BaseBoardCell workspace.

Note: When you are using multiple BaseBoardCells with their own BoardIn and BoardOut cells, you can keep track of which board cells belong to each other by placing your mouse above the input/output pins for a tooltip.

## BoardOutCell

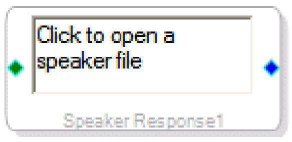
The BoardOutCell is used to add an output pin to your BaseBoardCell. This allows you to take the signal flow from your BaseBoardCell workspace back to the main page.

Note: When you are using multiple BaseBoardCells with their own BoardIn and BoardOut cells, you can keep track of which board cells belong to each other by placing your mouse above the input/output pins for a tooltip.

## ProbeCell

The ProbeCell is used in conjunction with the StimuliCell to plot the Frequency Response of the system you configure. Click on the Probe cell to bring up the blank Frequency Response graph. Then click on the Stimulus cell to graph the curve for your system. See the Second Tutorial for more information.

## Speaker Simulation



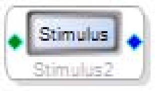
The Speaker Response algorithm uses the MLSSA (Maximum Length Sequence System Analyzer) standard. MLSSA is the industry standard loudspeaker measurement system, for more information see www.mlssa.com In order to use this cell:

1. Drag and drop into the workspace
2. Click in the text area of the cell
3. Load your speaker data, by selecting the appropriate file and clicking **Open**

   Note: Speaker data files are ASCII text in MLSSA format. Header information is contained inside quotes on the first two lines. Data is then delimited by commas and by line.

## StimuliCell

The StimuliCell is used in conjunction with the ProbeCell to plot the Frequency Response of the system you configure. After bringing up the Frequency Response graph by clicking on the Probe cell, click on the Stimulus window to graph the curve for your system. See the second tutorial for more information.



## TConnectionCell

The TConnection is used for splitting signal, and can split multiple times. It can split the signal as many times as you need. It is different than the splitters algorithms found under the Mixers/Splitters tab, in that it does not contain any DSP code and doesn't use any DSP resources.

## Terminal

The Terminal cell is used as a sink for the signal flow when an actual output is not necessary for your application. This is useful to connect the signal flow from a cell that is not being sent to output. If this is not done, the compiler will give you an error for a non connected pin.

## User Comment

SigmaStudio allows you the option of writing text in the Schematic workspace. This has no DSP code or function involved. In order to use this cell:

1. Drag and drop into the workspace
2. Click on displayed text
3. Enter your desired text
4. Click outside text box to return to the workspace

## User Image

SigmaStudio allows you the option to add an image to your Schematic workspace. This has no DSP code or function involved. In order to use this cell:

1. Drag and Drop into the workspace
2. Right-click the displayed image
3. Browse and find desired image
4. Click **OK**

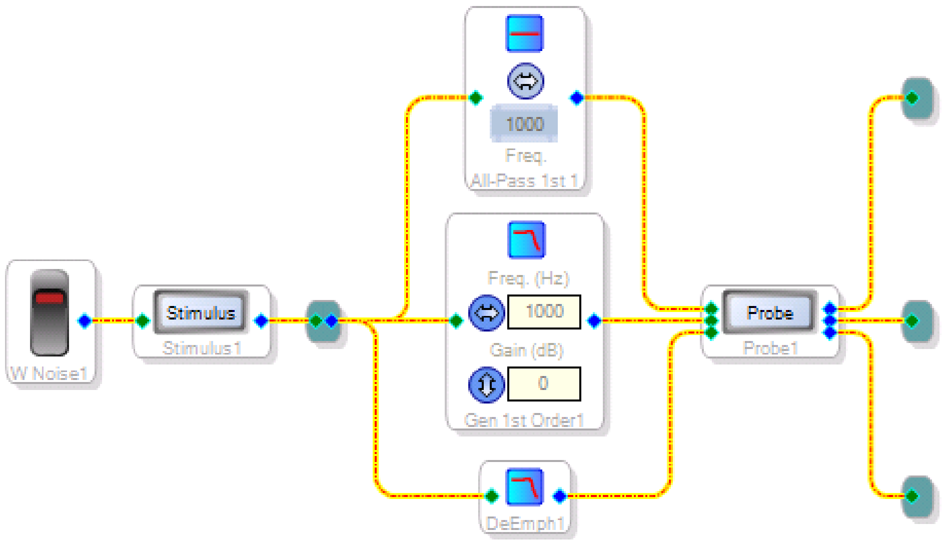
## Examples

### System Example

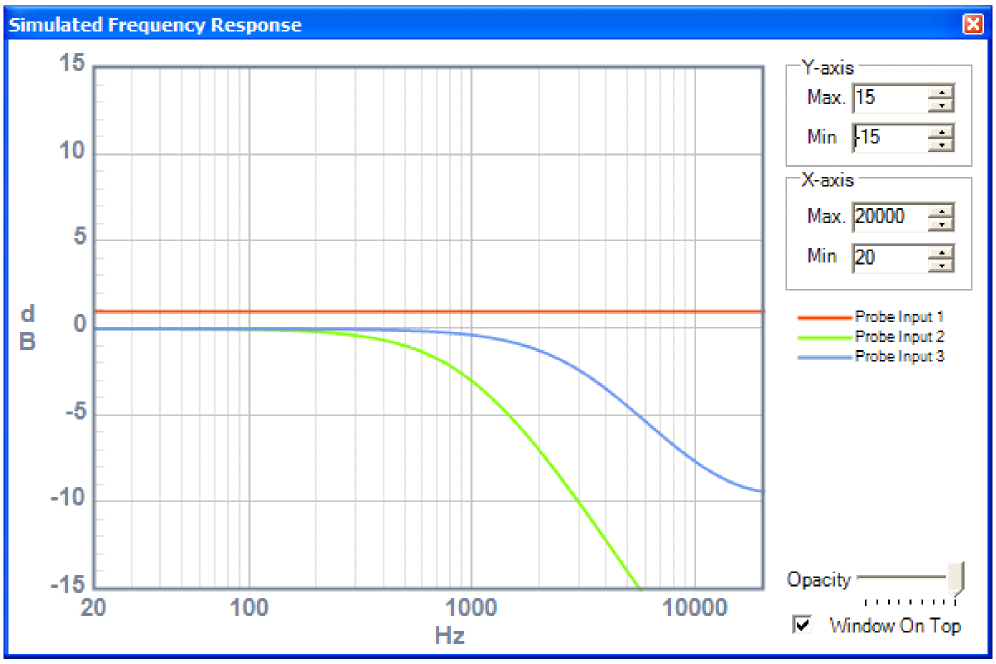
#### Example 1.1

The following example uses StimulusCell, ProbeCell, TConnectionCell, and Terminals to produce the Frequency  Response of an All Pass, General 1st Order, and De-emphasis Filter.

Schematic:



Frequency Response:



Y-axis
Max. 15
Min -15

X-axis
Max. 20000
Min 20

Probe Input 1
Probe Input 2
Probe Input 3

[✓] Window On Top

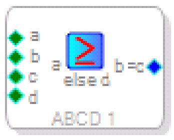
# Basic DSP

The Basic DSP library of the ToolBox gives you access to common DSP functions for your application. These basic building blocks give you fundamental control of DSP algorithms for your task. The following is a list of the available cells under this library.

- Absolute Value
- AB in/out Condition
- AB in CD out Condition
- DC Input Entry
- Linear Gain
- Multiply
- Delay
- Signal Add
- Signal Invert
- Feedback

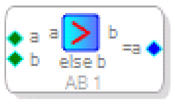
The following pages contain more specific information about the individual cells. Also take a look at the Basic DSP Examples page to see a sample schematic using some of the Basic DSP cells.

## AB in CD out Condition



The AB in CD out Condition cell allows you to compare the sample by sample amplitude value of two incoming signals (AB) and output the sample of one of two new signals (CD) depending on the condition. Click on the blue icon in the cell 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 your output sample is D.

## AB in/out Condition



The AB in/out Condition cell allows you to compare the sample by sample amplitude value of two incoming signals (AB) and output the sample of the signal meeting the condition. Click on the blue icon in the cell 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 your output sample is B.

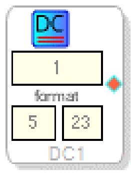
Note: You will need to recompile your project every time you select a different condition.

## Absolute Value

The Absolute Value cell converts all negative components of the input signal to positive values. Note: By right-clicking the cell you can add multiple pins by selecting: **Add Algorithm** > **IC N** > **Absolute Value**

## DC Input Entry

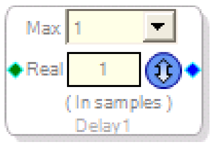
The DC Input cell allows you to specify a DC value that can be used to apply to a signal through any variety of functions (adder, mixer, multiplier, etc.). To enter a value, click on the text window of the cell.



Note:The value that you specify is a 5.23 value (5 bits for the integer 23 bits for the decimal). The accepted values for this block are therefore in the range: [-16, 15.999]. you can specify the format with which you want the value to be entered. The default is a 5.23 value (5 bits for the integer 23 bits for the decimal).

## Delay

The Delay cell, using the $Z^{-a}$ algorithm adds a variable delay into the signal flow, between 1 and the maximum available data memory on the DSP.



The maximum allowable delay for a particular DSP depends on the available data memory (please refer the appropriate data sheet of the DSP to know the maximum available data RAM).

The Max value allocates memory for the the DSP since it is a compiler directive and modifies the assembly code. If memory space is being used for other cells (i.e. Filters,delay. Any time you change the Max value you must re-compile and download the program to Compressors, etc.)then the maximum delay available will depend on whatever is left of the data RAM.

## Divide

The division cell allows you to divide two incoming signals. The division is performed using the Newton Raphson Iteration method. In order to use this cell:

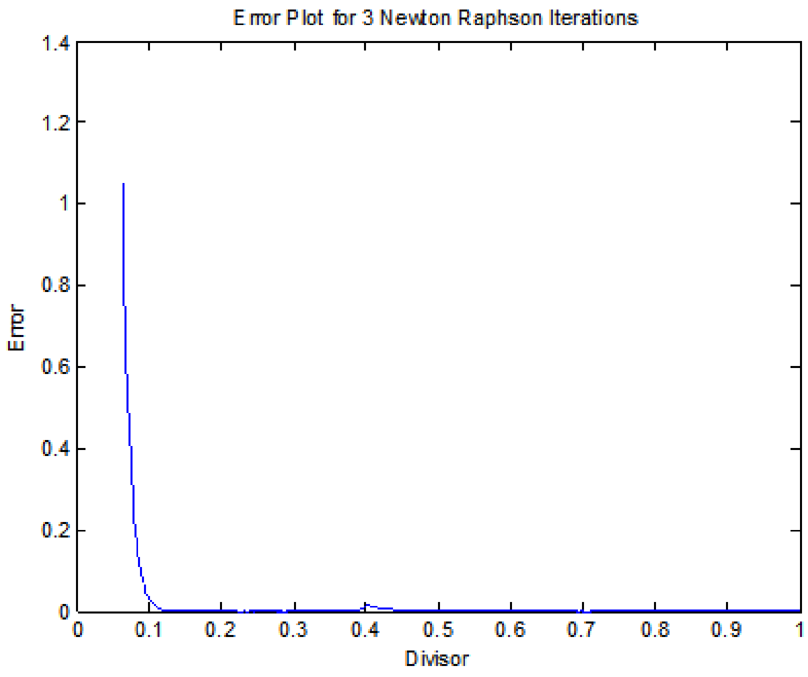
1. Drag and drop into the workspace
2. Right click the cell and select **Add Algorithm** > **IC N** >
   - **Newton Raphson 3 Iterations**
   - **Newton Raphson 4 Iterations**
3. Connect your signals to the pins of the cell so that you are performing the division: pin1 / pin2

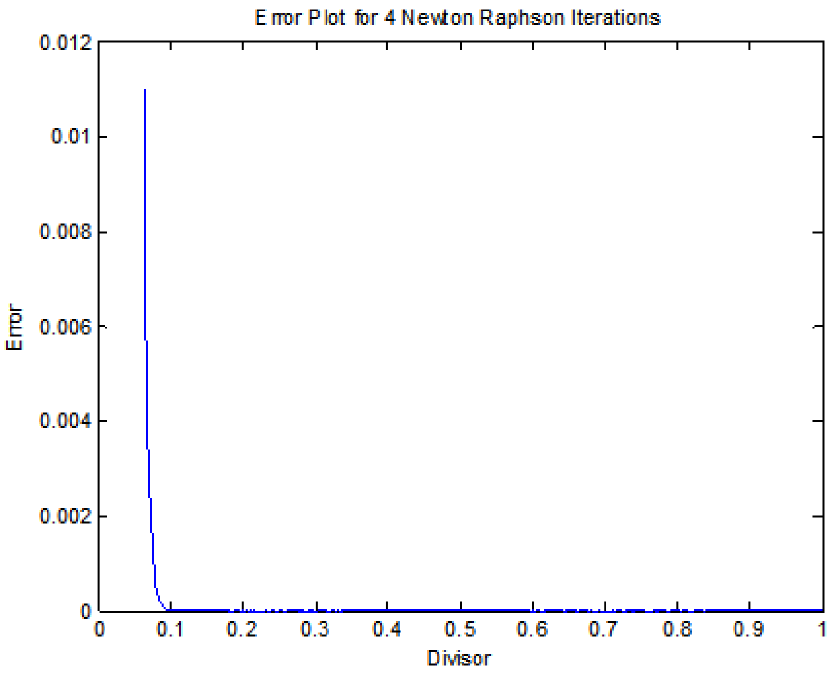
The Newton Raphson Iteration is performed according to the equation:

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

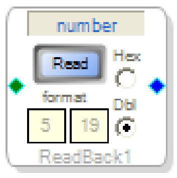
This cell allows you the option to select the precision of the algorithm, whether to compute 3 or 4 iterations. There is a tradeoff between saving number of instructions and accuracy of the division computation for divisors (values of pin2) less than 0.1.

Below are error graphs plots for both the 3 and 4 iteration algorithms:


Error Plot for 3 Newton Raphson Iterations
Error
Divisor
1.4
1.2
1
0.8
0.6
0.4
0.2
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1


Error Plot for 4 Newton Raphson Iterations
Error
Divisor
0.012
0.01
0.008
0.006
0.004
0.002
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

## DSP Readback



The DSP Readback cell allows you to read values back from the DSP at any point in your schematic design. The number displayed on the screen is the data value sent back from the DSP, considering all the cells to the left of the DSP Readback cell. Every time you click on the **Read** button, this value will be updated with the latest value from the DSP.

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

## Feedback

The feedback algorithm is used to generate a delay in the signal path and re-route signal to an input occurring earlier in the signal path. It is the only module that has the green Input on the right side and the blue output on the left side in order to show the reverse signal flow. It is important to note that this block must be used if a feedback path is required in a SigmaDSP design.

## Linear Gain

The Linear Gain cell scales the signal by the value specified in the text window. In order to use this cell

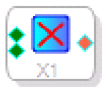
1. Drag and drop into the workspace
2. Right-click the cell and select **Add Algorithm** > **IC N** >
   - **Gain (slew)**
   - **Gain (no slew)**
3. Enter the desired scale factor into the text window of the cell

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

Note: You have the option between choosing a slew or no slew algorithm. Using the slew ram will ramp the signal gradually from original to target value.  Not using the slew ram will immediately jump the signal from original to target. For more information on target/slew ram, see the Target Slew RAM page under the Basic Sigma DSP Architecture section of the manual.

## Multiply

The Multiply cell multiplies two signals together. This is could be used for modulation or other algorithms were a multiply operation is needed.. By right-clicking the cell you can add more pins by selecting **Add Algorithm** > **IC N** > **Multiplication**



Care must be taken using this block to avoid computational overflows when multiplying signals together.

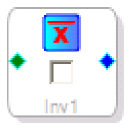
## Signal Add

The Signal Adder cell takes the input signals and directly adds them together.  No other modification of the signal is done.  Care must be taken using this block to avoid clipping when adding the signals together.

If automatic gain reduction is needed to avoid clipping using the Signal Merger cell.

## Signal Invert

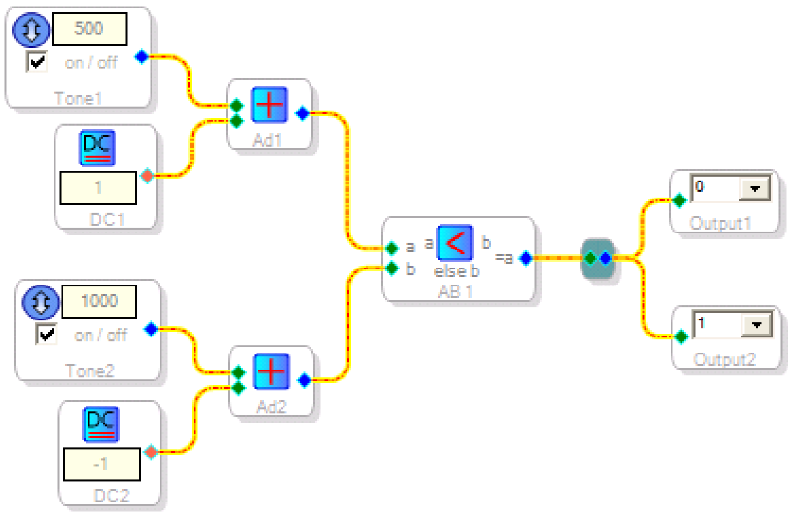
The Signal Invert cell takes the incoming signal and outputs the inversion. This is equivalent to a 180 degree phase shift, completely inverting the signal making the positive components of the waveform negative, and the negative components positive. The cell is enabled when you check the box.



## Basic DSP Examples

### Example 2.1

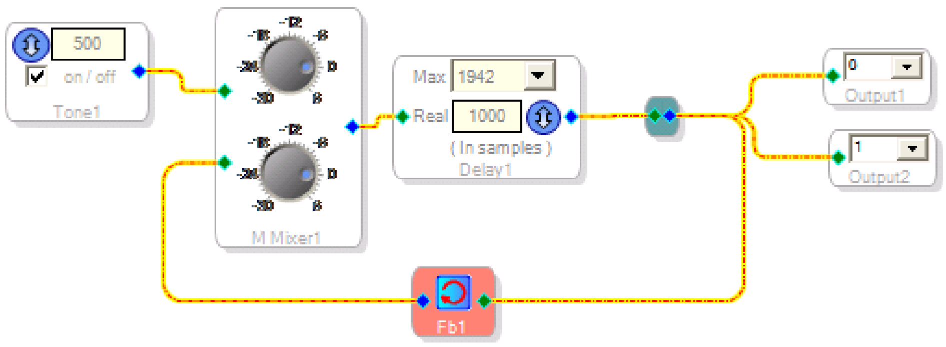
This schematic uses the DC Input cell, Adders, and an AB in/out Condition cell to compare the level of two input Tone(lookup/sine) sources that have a DC value added to them. The output of the Condition cell is split with a TConnection and sent to Output.



For the current configuration, the condition will evaluate false and signal B will be sent to the output. If you click on the condition icon in the AB in/out Condition cell so that is shows greater than, and recompile, the condition will evaluate true and the A signal will be sent to output.

### Example 2.2

This schematic shows how the output from the Delay cell is split at a TConnection, to be sent to Output and also fed into the Feedback and then back into the 2nd input of the Multiple Control Mixer.



# Dynamics Processors

The Dynamics Processors library of the ToolBox gives you access to a variety of different cells that affect the dynamic range of the signal. The cells offered in this library have a variety of options to help accomplish your desired goals. The main applications that these cells are used for is to program the cells to function as compressors or expanders.

Compressors are used to compress the dynamic range of the signal passing through. The output gain is determined by comparing the input signal amplitude to a threshold level. Below this threshold all audio remains unaffected, but whenever the input signal exceeds this threshold the output level is turned down by the ratio you determine. For example if the ratio you set is 1.6:1, for every 1.6dB increase above the threshold point, the output only increases 1dB. Therefore compressors make loud sounds quieter which is effectively compressing the dynamic range. More information on how to use compressors is given with the descriptions of the individual cells.

Expanders are used to expand the dynamic range of the signal passing through. This is done by operating only below the set threshold point in order to make the quiet sounds quieter. For example the ratio you set is 2:1 and there is a step decrease in input level of 10dB the expander attenuates the output by 20dB. This is typically used in noise reduction applications as can be seen here since the expander achieved an improvement of 10dB in noise reduction.

All of the cells available under the Dynamics Processors library have many common features. Depending on your application you will need to choose a cell with the correct parameters. Below are definitions of these parameters, and then following that is the description of what each cell contains.

| | | |
|---|---|---|
| **Post Gain** | | Controls post processing gain to increase the overall signal level attenuated by a compressor |
| **RMS TC (db/s)** | | Controls the Time Constant (TC) setting the attack time of the compressor. The RMS time constant determines how rapidly the gain will adapt to abrupt changes in the input signal level. The faster the TC the faster the compressor responds to the signal |

exceeding the threshold.

The time constant is calculated from the value you enter in dB/sec according to the following equation:

```
Timeconstant=1.0-
(dbperseconds]/(10.0*fs))
```

**Hold (ms)**

Controls the amount of time the compressor holds its setting once activated

The hold time is calculated from the value you enter in milliseconds, according to the following formula:

```
hold=fs*milliseconds/1000
```

**Decay (db/s)**

Controls the rate at which the compressor returns to full volume after the signal has returned to below threshold levels.

The decay time is calculated from the value you enter in dB/sec according to the following equation:

```
decay=dbperseconds/(96.0*fs)
```

**Soft Knee**

With soft knee enabled the compressor eases into the compression ratio, making a less abrupt change between the compressed and non-compressed signal. If not activated the default is a Hard Knee which begins to compress right after the threshold is passed.

**Show Graph**

Opens the Compression Curve window which allows you to visually represent the compression ratio. You can drag and draw points on the graph to get your desired Compression Curve. By right-clicking the graph you also have the option to add or remove points or set the compression curve to flat.

**/Detect**

This option is given when you right click the empty cell to select the algorithm for the compressor. When using the detect algorithm the red pin acts at a detect circuit. The compressor is only active when signal is detected at the red input pin.

Note: If you are not using the detect algorithm, the input signal to the compressor acts as the detect signal itself. If you have a stereo signal the detect signal is computed as the average between left and right channels (L+R)/2.

The following is a list of available cells under this library:

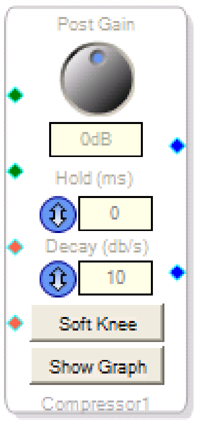
- Limiter
- RMS Detect (no:gain, hold, decay)
- RMS Detect (gain)
- RMS Detect (no gain)
- Peak Detect (gain)
- Peak Detect (no gain)
- RMS Detect (display)

The following pages contain more specific information about the individual cells. Also take a look at the Dynamics Processors Example page to see a sample schematic using some of the Dynamics Processor cells.

Note: All cells, except for the Limiter contain access to a Compression Curve window via the Show Graph Button, as described above. It is important to note though, that currently the algorithm supports input values only up to 6dB. Any input values exceeding 6dB will go back to a linear compression ratio. This is to say that if you have a compression curve mapped out up to 6dB it will be ignored after the 6dB input mark, and return to a linear compression ratio 1:1. The Limiter however solves this problem as it supports internal gains higher than 6dB and will attenuate the signal accordingly.

## Peak Detect (gain)

This cell uses a Peak Compressor with the ability to control Post Gain, Hold, Decay, Soft Knee vs. Hard Knee, and the Compression Curve. The Peak Compressor operates on any signal rising above the threshold no matter how fast the transient. Note that peak compression no longer has a setting for the Time Constant. Peak compression is beneficial in stopping any signals from overloading the input no matter how quickly it exceeds the threshold, but can sometimes be less natural sounding than RMS compression. To use this cell:

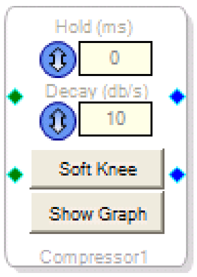


1. Drag and drop into the workspace
2. Right click the cell and select the algorithm for your application
   - **Stereo 1 Peak w/ gain**
   - **Stereo Separate Detect w/ gain**
3. Set the parameters to fit your application and click on show graph and drag, add, and remove points on the graph in order to achieve your desired compression ratio curve.

Note: For the picture included to the right, the Stereo Separate Detect w/ gain was selected. Unlike other Detect algorithms, for the Peak compression, you need two separate signals to activate the stereo compression.

## Peak Detect (no gain)

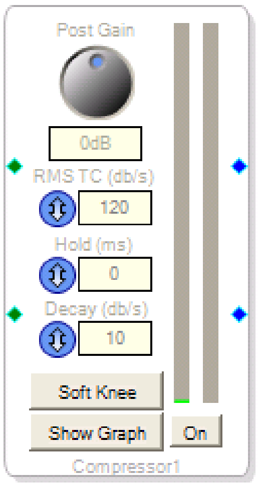
This cell functions exactly as the Peak Detect (gain) cell, however it does not include the control knob for post gain control. See the Peak Detect (gain) page description for more details.



Note: For the picture included to the right the algorithm selected is Stereo 1 Peak (no gain)

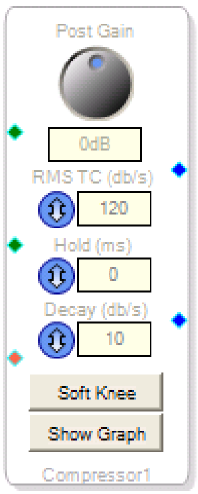
## RMS Detect (display)

This cell functions exactly like the RMS Detect (gain) with the Stereo 1 RMS algorithm, except that it comes with a real-time display. The left display bar shows the input signal level and the right display bar shows the output signal level. The On button allows you to toggle between having the display on or off.



## RMS Detect (gain)

This cell uses a RMS compressor with the ability to control Post Gain, RMS TC, Hold, Decay, Soft Knee vs. Hard Knee, and the Compression Curve. RMS works on a wider average than Peak compressors, thus allowing some fast loud transients to pass through without compression, but operating more on longer segments that exceed the threshold. To use this cell:



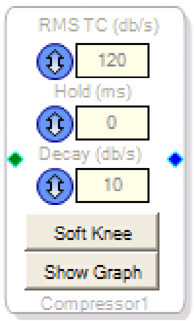
1. Drag and drop into the workspace
2. Right click the cell and select the algorithm for your application
   - **Stereo 1 RMS**
   - **Stereo 1 RMS/Detect**
   - **Mono 1 RMS**
   - **Mono 1 RMS/Detect**
3. Set the parameters to fit your application and click on show graph and drag, add, and remove points on the graph in order to achieve your desired compression ratio curve.

Note: For the picture included to the right, this cell was chosen to have the /Detect algorithm, shown by the red pin. In order for this compressor to be active there needs to be a signal connected to this pin.

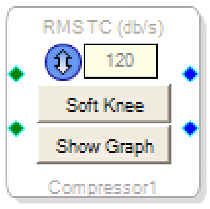
## RMS Detect (no gain)

This cell functions exactly as the RMS Detect (gain) cell, however it does not include the control knob for post gain control. See the RMS Detect (gain) page description for more details.

Note: For the picture included to the right, the Mono Algorithm was chosen hence there is only one set of input/output pins.



## RMS Detect (no: gain, hold, decay)



This cell uses a RMS compressor with the ability to control RMS TC, Soft Knee vs. Hard Knee, and the Compression Curve. RMS works on a wider average than Peak compressors, thus allowing some fast loud transients to pass through without compression, but operating more on longer segments that exceed the threshold. To use this cell:

1. Drag and drop into the workspace
2. Right click the cell 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 on show graph and drag, add, and remove points on the graph in order to achieve your desired compression ratio curve.

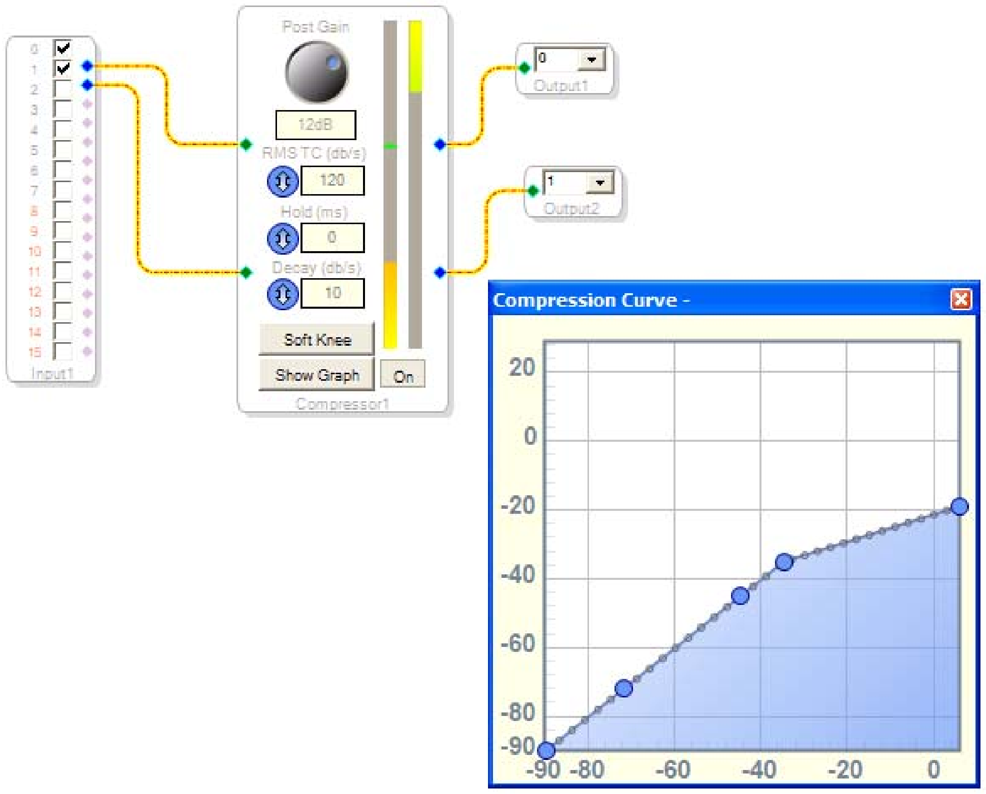
## Dynamics Processors Example

### Example 3.1

The following example uses a RMS Detect (display) cell along with input/output cells for audio flow. What is important to note about this example is the Compression Curve window included with the schematic, the relative levels of input/output on the display bar, and the Post Gain knob.

Here the threshold is set at approximately 35 dB on the input signal level axis. The compression ratio from there on is set to approximate 2.7:1 from the Compression Curve. (This is determined by finding the slope of the line drawn above the threshold, taking the inverse and comparing to the unit slope interval below the threshold). Thus for any input signal below 35 dB, the output level will remain unaffected. However when the input signal exceeds this threshold, for every 2.7dB it increases the output will only increase 1 dB.

The display bars indicate the relative levels of the input and output signals. The green bar on the input display shows the level of the last value before the current level. In this case the compression of the audio sample significantly lowered the overall level of the output signal. Thus it was necessary to turn up the Post Gain by 12dB to increase the overall level of the output signal.


Post Gain 12dB
RMS TC (db/s) 120
Hold (ms) 0
Decay (db/s) 10

Note: This example shows a very simple compression curve: a linear ratio constructed using only one of the graph points. This Compression Curve graph allows you to develop more complex Compression Curves, but for the purpose of this example, there is only one active graph point.

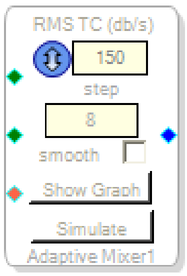
## Advanced DSP

The Advanced DSP library of the ToolBox gives you access to cells that have advanced functionality in DSP applications. The following is a list of the available cells under this library:

- Adaptive Mixer Dual (graph)
- Adaptive Mixer Single (graph)
- RMS Table
- State Variable Filter
- State Variable Filter w/Q

The following pages contain more specific information about the individual cells. Also take a look at the Advanced DSP Example page to see a sample schematic using some of the Advanced DSP cells.

## Adaptive Mixer Dual (graph)



The Adaptive Mixer Dual (graph) is an advanced method of mixing two signals based on a third control signal. The two green pins designate the two input signals you wish the mix together and the third orange pin is for the control signal. Note: usually orange pins indicate that this channel is not for true audio, but this is not the case for this cell. The orange pin here indicates that the input control signal will be converted to a RMS average value, eliminating the need to use the RMS table for this application. The RMS table value is used to determine the scale factors for the signals to be mixed. This cell is very similar to the Adaptive Mixer Single (graph), but allows more advanced control over the scale factors for the signals to be mixed. In the Single case, one scale factor is determined as the compliment of the other, whereas with the Dual control, you can select and change the curves for both signals to be mixed.

**RMS TC (db/s)** -Controls the Time Constant (TC) setting the attack time of the RMS average computation. The RMS time constant determines how rapidly the gain will adapt to abrupt changes in the input signal level.

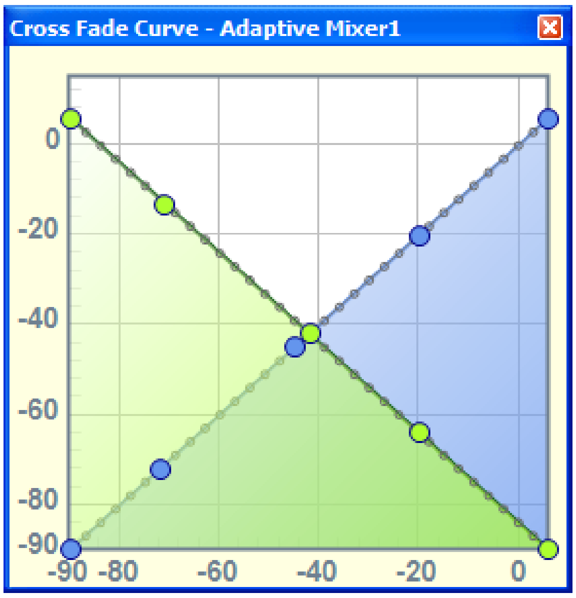
**Step** - Controls the slew ram rate in terms of how quickly the algorithm ramps to the next value. The step is usually calculated by:

Step = (Target Data - Slew Data) / Number of Steps

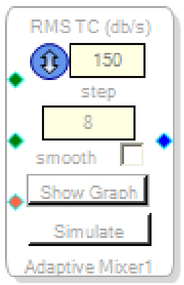
Here, you have control over this value and the ramp will be calculated according to $2^{-step}$

**Smooth** - This checkbox determines whether the ratio curve of the levels will be exactly linear from graph point to graph point, or if it will be smoothed out.

**Show Graph** - This button allows you to bring up the mix ratio window. For the case of the Dual mixer, you have control over the level of both the 1st and 2nd pins. You can change the curve by moving, adding, or removing graph points. The RMS table value gives you the x-value on this graph and the corresponding y-values will be the scale factors for the output levels of the mix between the 1st and 2nd pins. The graph can be seen below:


Cross Fade Curve - Adaptive Mixer1
0
-20
-40
-60
-80
-90
-90
-80
-60
-40
-20
0

## Adaptive Mixer Single (graph)



The Adaptive Mixer Single (graph) cell is an advanced method of mixing two signals based on a third control signal. The two green pins designate the two input signals you wish the mix together and the third orange pin is for the control signal. Note: usually orange pins indicate that this channel is not for true audio, but this is not the case for this cell. The orange pin here indicates that the input control signal will be converted to a RMS average value, eliminating the need to use the RMS table for this application. The RMS table value is used to determine the scale factor for the signals to be mixed.

**RMS TC (db/s)** -Controls the Time Constant (TC) setting the attack time of the RMS average computation.  The RMS time constant determines how rapidly the gain will adapt to abrupt changes in the input signal level.

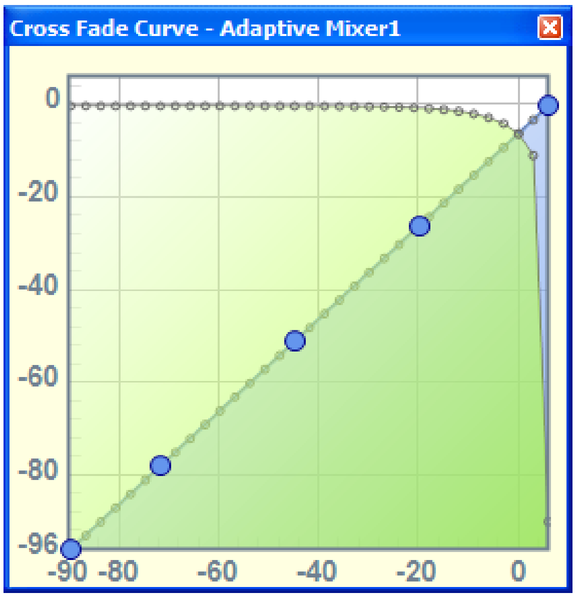
**Step** - Controls the slew ram rate in terms of how quickly the algorithm ramps to the next value. The step is usually calculated by:

Step = (Target Data - Slew Data) / Number of Steps

Here, you have control over this value and the ramp will be calculated according to $2^{-step}$

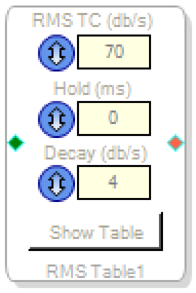
**Smooth** - This checkbox determines whether the ratio curve of the levels will be exactly linear from graph point to graph point, or if it will be smoothed out.

**Show Graph** - This button allows you to bring up the mix ratio window. For the case of the Single mixer, you only have control over the level of the 2nd input pin. You can change the curve  by moving, adding, or removing graph points. The output curve scale value for the 1st pin is automatically calculated as the compliment of the 2nd pin. The RMS table value gives you the x-value on this graph and the corresponding y-value will be the scale factors for the output levels of the mix between the 1st and 2nd pins. The graph can be seen below:



**Simulate** - The Simulate button brings up a frequency response graph that shows the mix of the Transfer Function between the 1st and 2nd pins, depending on the RMS table value.

## RMS Table

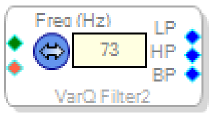


The RMS table takes an input signal and outputs the RMS average values of the signal. The output of this cell is an orange pin to denote that the output of this cell is not a true audio signal. One of the primary uses for this cell is to use it in conjunction with the State Variable Filter, in order to control the filter Q.

There are three parameters available for control through the edit boxes/ control arrows. Below are descriptions of their functions. There is also a **Show Table** button which allows you to view a table of the average output.

| | | |
|---|---|---|
| **RMS TC (db/s)** | | Controls the Time Constant (TC) setting the attack time of the RMS average computation. The RMS time constant determines how rapidly the gain will adapt to abrupt changes in the input signal level. |
| **Hold (ms)** | | Controls the amount of time the RMS average holds its current output setting once activated. |
| **Decay (db/s)** | | Controls the rate at which the RMS average returns to the original output setting after the signal has returned to below threshold levels. |

## State Variable Filter (Q input)



The State Variable Filter cell allows for simultaneous access to three different filter types: lowpass, highpass and bandpass. See the State Variable Algorithms page for more information about the calculations and parameters for this algorithm.

As can be seen from the figure on the right, this cell has two input pins and three output pins. The standard green pin is for your signal input and the orange pin is for the value to control the filter Q. The Q affects the sharpness of the cut/boost at the cutoff frequency.  It is common to control the filter Q of the State Variable filter by either sending a DC input value to this pin, or using it with the RMS table to generate Q parameters from your input signal. If you would like to have control over the Q from the cell, please refer to the State Variable Filter.

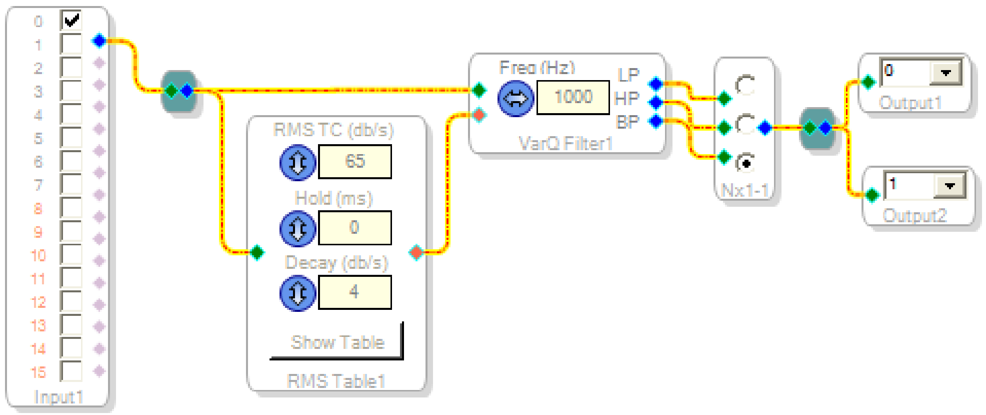
The three output pins allow you to choose between LP, HP, BP filters. The nature of this algorithm is to compute the filter coefficients for all filter types giving you simultaneous access to all of the filters. If your application does not require use of all filter types you can connect the filter output to a terminal cell.

You can control the value of the cutoff/center frequency by using the edit box or the click and drag arrows.

## Advanced DSP Example

### Example 4.1

The following example uses a State Variable Filter (Q input), RMS table, Mono Nx1 switch, TConnection, and Input/Output cells. The Q of the state variable filter is determined by the RMS table and the filter type can be selected using the mono switch.



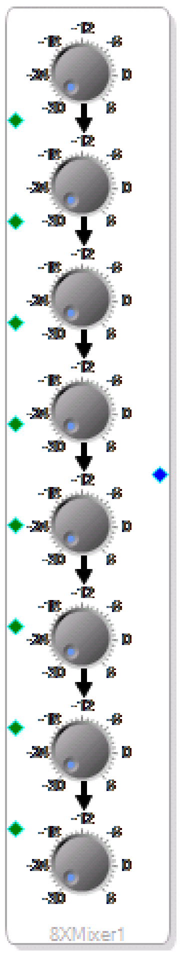
## Mixers/Splitters

The Mixers/Splitters library of the ToolBox gives you access to cells that allow you to combine or split signals in various ways. The following is a list of the available cells under this library:

- Eight Input Cross Mixer
- Three Input Cross Mixer
- Two Input Cross Mixer
- Multiple Control Mixer
- Single Control Mixer
- Signal Merger
- Single Control Splitter
- Multiple Control Splitter

The following pages contain more specific information about the individual cells. Also take a look at the Mixers/Splitters Example page to see a sample schematic using some of the Mixers/Splitters cells.

## Eight Input Cross Mixer



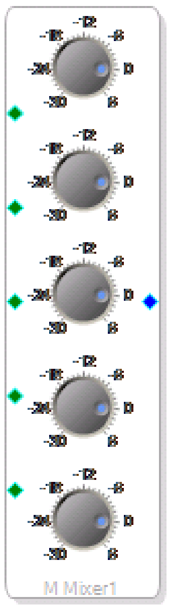
The Eight Input Cross Mixer allows eight input signals to be mixed together with variable gain settings down to one output signal. The cell is ready to use with eight input pins, eight control knobs and one output pin. The eight input signals will be mixed together with whatever gain setting you specify. These gain settings are independent and can be set for each input.

There is the option to both Grow and Add to this algorithm. Growing the algorithm will give extra output pins corresponding to the number of vertical control knob columns. This allows you to make different output mixes of the same inputs. Adding an algorithm will add a separate mixer to the same cell allowing eight new inputs to be mixed to another output. It is important to note that this is a separate mixer even though it appears on the same cell. You can access the Grow or Add Algorithms by right-clicking the cell on the boarder or title of the cell.

Note: By right-clicking the individual control knobs you can specify the parameters of the control knob: Min/Max, current value, and step size of the knob. you can have different parameter settings for all your control knobs.

## Multiple Control Mixer

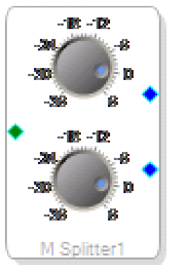
The Multiple Control Mixer allows you to select the exact number of inputs that you wish to mix down with variable gain to one output signal. When you first drag and drop into the workspace, it appears to look like a Two Input Cross Mixer. By right-clicking the cell on the border or the title you can Grow the Algorithm. This allows you to add extra inputs that are mixed to the same output. Once you grow the algorithm, you can also right-click again on the cell and reduce the algorithm. The figure on the right is an example of selecting **Grow Algorithm** > **1. Multiple Ctrl Mixer** > **3**



Note: This cell functions differently than the X- Input Cross Mixers because it gives you exact control over the number of inputs, but does not allow multiple columns for different mixes.

## Multiple Control Splitter

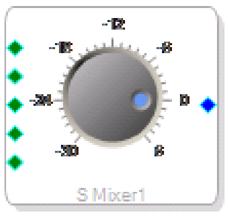
The Multiple Control cell splits one input signal into multiple outputs, each with its own variable gain settings. Each output is scaled independently by its own control knob value. This cell can be grown to select the desired number of outputs. The figure on the right is the default cell.



## Signal Merger

The signal Merger Cell is a mixer that automatically lowers levels proportional to the number of inputs. This helps avoid clipping by sacrificing control of setting independent levels for your inputs. This cell can be grown to select your desired number of inputs. The figure on the right is an example of selecting **Grow Algorithm** > 1**. Single Control Mixer** > **1**

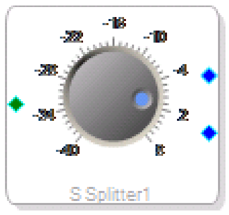
## Single Control Mixer



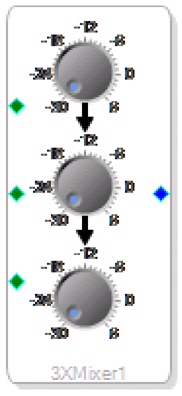
The signal control mixer allows you to mix multiple input signals down to one output with one variable gain control setting for all your inputs. The default is for there to only be two inputs, but by right-clicking the border or title of the cell you can Grow the Algorithm to set the desired number of inputs. By right clicking the control knob you can specify the parameters of the gain control knob. The figure on the right is an example of selecting **Grow Algorithm** > **1. Single Control Mixer** > **3**

## Single Control Splitter

The Single Control cell splits one input signal into multiple outputs. The output gain is scaled by the value specified in the gain control knob and all outputs are affected by the same gain. This cell can be grown to select the desired number of outputs. The figure on the right is the default cell.



## Three Input Cross Mixer

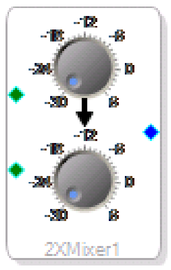


The Three Input Cross Mixer allows three input signals to be mixed together with variable gain settings down to one output signal. This cell is exactly the same in functionality as the Eight Input Cross Mixer, except for the number of inputs. Also selecting Add Algorithm for the Three Input Mixer, adds a separate mixer to the same cell with three new inputs that are mixed to one input (instead of eight new inputs).

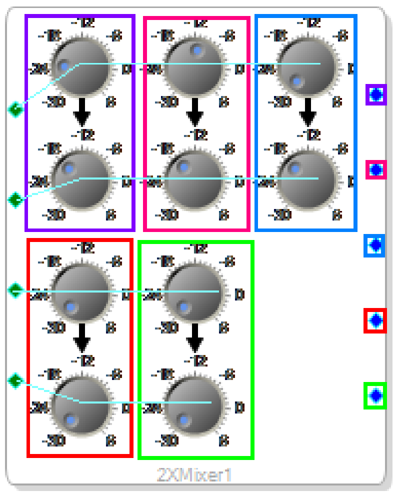
There is option to both Grow and Add an algorithm to this cell. For information about the mapping of inputs/outputs for cells which have multiple algorithms, please see the example included on the Two Input Cross Mixer page.

## Two Input Cross Mixer

The Two Input Cross Mixer allows two input signals to be mixed together with variable gain settings down to one output signal. This cell is exactly the same in functionality as the Eight Input Cross Mixer and the Three Input Cross Mixer, except for the number of inputs. Also selecting Add Algorithm for the Two Input Mixer, adds a separate mixer to the same cell with two new inputs that are mixed to one input (instead of eight or three new inputs).



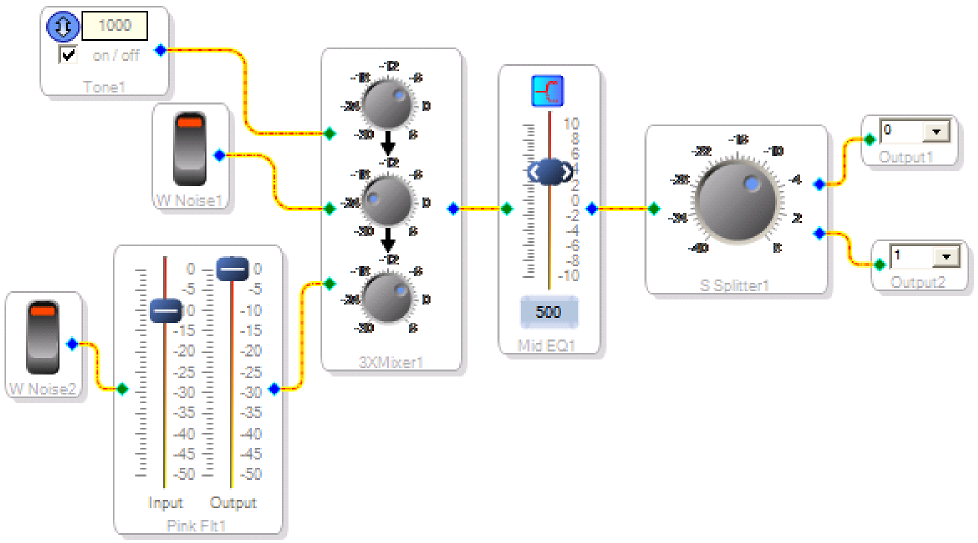
It is important to know the mapping of inputs/outputs for cells in which you have an algorithm that has been both grown and added. Below as an example of a Two Input Cross Mixer that has two algorithms, the first grown by 2, the other grown by 1. The color-coded boxes show the different possible mixes with this configuration and their corresponding output pins. The light-green horizontal line shows which control knobs have control over which input signals.



## Mixers/Splitters Example

### Example 5.1

The following example uses a Three Input Cross Mixer and a Single Control Splitter along with White Noise source, Tone Source, Pink Filter, Mid EQ filter, and two Outputs. This schematic shows a simple way of mixing three different sources with different levels, doing some EQ processing on the frequency content, and then splitting the signals to two outputs, scaled by the same gain.



# Filters

The Filters library of the ToolBox gives you access to numerous Filters/EQs that allow you to shape the frequency content of your signal. Check the Algorithms section for more information about the algorithms driving the filter cells. The following is a list of the available cells under this library.

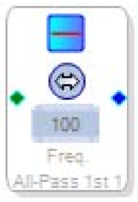
- All-Pass 1st Order
- All-Pass 2nd Order
- De-emphasis Filter
- General 1st Order
- General Purpose Filter
- Medium Size Eq
- Small Size Eq
- Text-In Filter (linked)
- Text-In Filter (unlinked)
- Tone control (with Index lookup)

The following pages contain more specific information about the individual cells. Also take a look at the Filters Example page to see a sample schematic using some of the Filter cells.

Note: Most filters will have the option of selecting between Double and Single Precision computation. For the AD1940 double precision uses 56 bits for each calculation, takes 10 instructions per filter, and should normally be used. Single precision uses 28 bits for calculations, takes 6 instructions per filter, and saves 3 data ram spaces over the double precision algorithm. Single precision should not be used for frequencies below 1/10 the sampling frequency, or for high Q filters.

## All-Pass 1st Order

The All-Pass 1st Order Filter passes all frequencies with equal gain. The filter is a lossless system in that it preserves signal energy, i.e. the magnitude of the signal is preserved. The All-Pass 1st order affects the phase of the signal providing phase compensation from 0 to 90 degrees. Look at the All Pass Algorithm page for information about the algorithm driving this Filter cell.



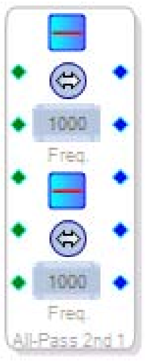
The default cell is ready to use after dragging and dropping into the workspace, but there is an option to either Grow or Add to this algorithm. Growing the algorithm will add another frequency band to the cell, which is equivalent to having two individual filters in series. Adding an algorithm adds another input/output pair to the cell which is equivalent to adding a filter in parallel.

The cutoff frequency is determined from your entry in the text window of the filter. You can also click on the arrows to select a value for the frequency parameter.

Note: For all filters in SigmaStudio, having multiple I/Os on one filter block will share the parameters for the filters and save parameter ram. If coefficients can be shared, optimal design would be multiple inputs/outputs on one block as shown above, rather then a large number of separate blocks.

## All-Pass 2nd Order

The All-Pass 2nd Order Filter passes all frequencies with equal gain.  The filter is a lossless system in that it preserves signal energy, i.e. the magnitude of the signal is preserved.  The All-Pass 2nd order affects the phase of the signal providing phase compensation from 0 to 180 degrees. Look at the All Pass Algorithm page for information about the algorithm driving this Filter cell.



Unlike the All Pass 1st Order cell, the 2nd Order cell is empty and requires that an algorithm be added before it can be used. In order to use this cell:

1. Drag and drop into the workspace
2. Right click the cell and select **Add Algorithm** > **IC N**.
3. From the list select which algorithm meets your needs:
   - **Single - Double Precision**
   - **Single - Single Precision**
   - **Double - Double Precision**
   - **Double - Single Precision**
   - **Triple - Double Precision**
   - **Triple - Single Precision**
   - **Quad - Single Precision**
   - **Quad - Double Precision**
   - **Penta - Double Precision**
   - **Penta - Single  Precision**
   - **Hexa - Double Precision**
   - **Hexa - Single Precision**
4. Select your desired cutoff Frequency, by entering it into the textbox or clicking on the arrows.

For the figure included to the right, the default algorithm is Double - Double Precision, and Triple - Double Precision algorithm was added. Note that the first two pins correspond to top frequency band, and the bottom three pins correspond to the bottom frequency band.

After selecting your default algorithm, there is an option to either Grow or Add to this algorithm. Growing the algorithm will add another frequency band to the cell, which is equivalent to having two individual filters in series. Adding an algorithm adds another input/output pair to the cell which is equivalent to adding a filter in parallel.

Note: For all filters in SigmaStudio, having multiple I/Os on one filter block will share the parameters for the filters and save parameter ram.  If coefficients can be shared, optimal design would be multiple inputs/outputs on one block as shown above, rather then a large number of separate blocks.

## DC Block Filter

The DC-block is a pre-configured filter with set parameters. It is used to remove DC components that may be present in your signal.

The default cell is ready to use after dragging and dropping into the workspace, but there is an option to Add to this algorithm.

The DC block filter is computer according to the following transfer function:

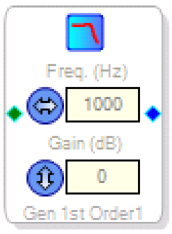
$$H(z) = \frac{1 - z^{-1}}{1 - Rz^{-1}}$$

## De-emphasis Filter

The De-emphasis Filter cell is a pre-configured filter with set parameters. It is used to attenuate the high frequency components that were boosted during Pre-emphasis. Pre-emphasis is a high-frequency boost used during recording. When the De-emphasis Filter is used in conjunction with audio that has been recorded with a Pre-emphasis filter there is improved signal-to-noise ratio.

This cell is ready to use with no need to add an algorithm. This cell does not have the ability be have algorithms grown or added.

## General 1st Order

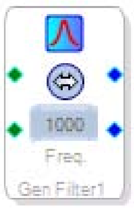


The General 1st Order Filter cell allows you to design 1st order low-pass and high-pass filters. See the General 1st Order Algorithms Page for more information about the algorithms driving this cell.

The default cell is ready to use after dragging and dropping into the workspace, but there is an option to either Grow or Add to this algorithm. Growing the algorithm will add another frequency band to the cell, which is equivalent to having two individual filters in series. Adding an algorithm adds another input/output pair to the cell which is equivalent to adding a filter in parallel.

To select between High Pass and Low Pass, click on the blue icon display the frequency representation of the filters to toggle between the two filter options. This option to toggle can be done in real-time without needing to recompile the project. The cutoff frequency, and gain of the filter is determined from your entries in the text windows of the filter. You can also click on the arrows to select values for those parameters.

## General Purpose Filter



The General Purpose filter is a powerful cell that has access to a variety of 2nd order IIR filter algorithms. See the General 2nd Order Algorithms page for more information about the algorithms driving these cells. Below are descriptions of how to use each of the 2nd order filters available from this cell:

- Peaking EQ
- Shelving EQ: Low/High Shelf
- General: Low/High Pass Filter and Bandpass/Bandstop Filter
- Butterworth: High/Low Pass and Bessel: High/Low Pass
- Tone Control
- IIR Coefficient

In order to use this cell:

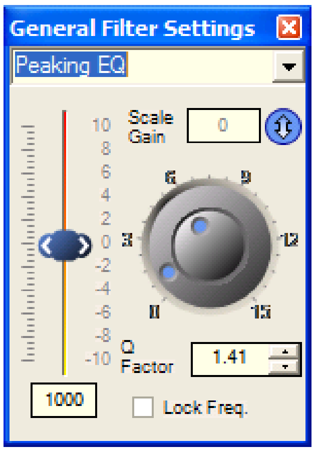
1. Drag and drop into the workspace
2. Right click the cell and select **Add Algorithm** > **IC N**.
3. From the list select which algorithm meets your needs:
   - **Single - Double Precision**
   - **Single - Single Precision**
   - **Double - Double Precision**
   - **Double - Single Precision**
   - **Triple - Double Precision**
   - **Triple - Single Precision**
   - **Quad - Single Precision**
   - **Quad - Double Precision**
   - **Penta - Double Precision**
   - **Penta - Single  Precision**
   - **Hexa - Double Precision**
   - **Hexa - Single Precision**
4. Double click in the middle of the cell to bring up the EQ parameter window.

5. In the parameter window, select which type of filter you want to use from the top drop down menu and enter the parameters accordingly. (The Peaking EQ is the default filter for the General Purpose Filter).

   For the figure included to the right, the default algorithm is Double - Double Precision.

   After selecting your default algorithm, there is an option to either Grow or Add to this algorithm. Growing the algorithm will add another frequency band to the cell, which is equivalent to having two individual filters in series. Adding an algorithm adds another input/output pair to the cell which is equivalent to adding a filter in parallel.

**Peaking EQ**



The Peaking EQ is a filter designed to boost or cut the designated center frequency. Below are the parameters available in this filter and a description of their function.

1. *Center Frequency*: You can enter the center frequency in the text box below the slider bar, or click and drag the horizontal arrows on the slider bar to edit your center frequency. As its name implies the center frequency is the center of the peak boost or cut you create with this filter.

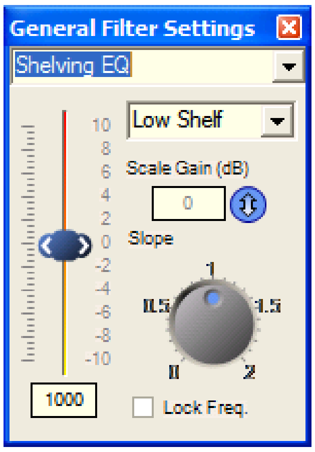
2. *Boost/Cut Gain*: You can edit the filter boost or cut gain by using the slider bar. Negative values will give you a cut at the center frequency and positive values will give a boost at the center frequency.

3. *Q Factor*: You can edit the Q Factor by entering a value in the text box, clicking on the arrows of the text box, or using the concentric control knobs. ( The outer knob controls the integer value, while the inner knob controls the decimal value). The Q factor affects the sharpness of the cut/boost at the center frequency. High Q values will give sharper resolution on the center frequency (smaller bandwidth), but also have a lower magnitude, whereas small Q values will have wider bandwidth with larger magnitude.

4. *Scale Gain*: You can edit the scale gain by using the text box or clicking the arrows to the right of the text box. Scale Gain affects the overall gain of the filter.

5. *Lock Frequency*: This checkbox can be enabled so that the horizontal control of the center frequency on the slider bar is not active.

**Shelving EQ: Low/High Shelf**



Shelving EQ

Low Shelf

Scale Gain (dB): 0

1000

The Shelving EQ is a filter designed to boost or cut all frequencies above (High Shelf) or below (Low Shelf) the cutoff frequency. Below are the parameters available in this filter and a description of their function.

1. *Low Shelf/High Shelf*: You can choose which filter you want to use by selecting Low or High shelf respectively from the drop down menu.

   Low Shelf - cuts or boosts all frequencies below cutoff frequency

   High Shelf - cuts or boosts all frequencies above cutoff frequency

2. *Cutoff Frequency*: You can enter the cutoff frequency in the text box below the slider bar, or click and drag the horizontal arrows on the slider bar to edit your center frequency. As its name implies, the cutoff frequency is the cutoff point between the affected shelf boost/cut of the filter to the unaffected flat response.

3. *Boost/Cut Gain*: You can edit the filter boost or cut gain by using the slider bar. Negative values will give you a cut for all frequencies above (High Shelf) or below (Low Shelf) the cutoff frequency and positive values will give a boost for all frequencies above (High Shelf) or below (Low Shelf) the cutoff frequency.

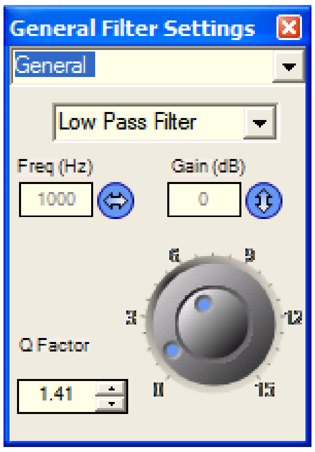
4. *Slope*: You can edit the slope of the filter using the control knob. By right-clicking the control knob, you can enter an exact value

into the pop-up text box window. The slope controls the steepness of the filter and affects the transition band between the affected shelf boost/cut of the filter to the unaffected flat response.

5. *Scale Gain*: You can edit the scale gain by using the text box or clicking the arrows to the right of the text box. Scale Gain affects the overall gain of the filter.

6. *Lock Frequency*: This checkbox can be enabled so that the horizontal control of the center frequency on the slider bar is not active.

**General: Low/High Pass Filter and Bandpass/Bandstop Filter**



Low Pass Filter

Freq (Hz) 1000

Gain (dB) 0

Q Factor 1.41

The General EQ has access to the four basic types of 2nd order filters: Low-Pass, High-Pass, Bandpass and Bandstop. Below are the parameters available in this filter and a description of their function.

1. *Low-Pass/High-Pass/Bandpass/Bandstop*: You can choose which filter you want for your application by selecting the filter from the drop down menu.

   Low Pass - passes all frequencies below cutoff frequency

   High Pass - passes all frequencies above cutoff frequency

   Band Pass - passes frequencies within the bandwidth of the center frequency

   Band Stop -attenuates frequencies within the bandwidth of the center frequency

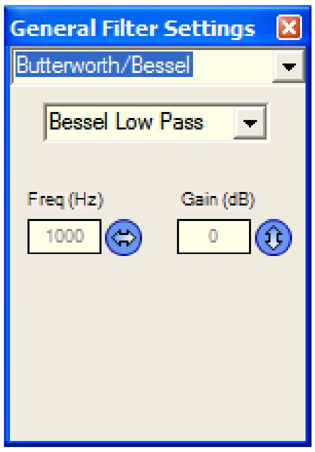
2. *Freq (Hz)*: You can enter the cutoff (High/Low pass) or center (Bandpass/Bandstop) frequency of your filter in the text box or by clicking the arrows to the right of the text box.

3. *Gain*: You can edit the gain by using the text box or clicking the arrows to the right of the text box. This setting affects the overall gain of the filter.

4. *Q factor*: (Only available for Low/High Pass Filters). You can edit the Q Factor by entering a value in the text box, clicking on the arrows of the text box, or using the concentric control knobs. ( The outer knob controls the integer value, while the inner knob controls the decimal value). The Q factor affects the sharpness of the filter slope during the transition band. The higher Q the faster the transition between the passband and the stopband of the filter, but there is a tradeoff with the behavior of the filter in the transition band.

5. *Bandwidth (octaves)*: (Only available for Bandpass/Bandstop Filters). You can edit the Bandwidth in the same manner as the Q factor. The Bandwidth determines the range of frequencies affected around the center frequency and also the sharpness of the transition from the center band to the passband or stopband.

**Butterworth: High/Low Pass and Bessel: High/Low Pass**


General Filter Settings

Butterworth/Bessel

Bessel Low Pass

Freq (Hz): 1000

Gain (dB): 0

The Butterworth/Bessel Filters offer High and Low Pass filters with alternate algorithms for computation of the filter coefficients than the General 2nd order Filters.

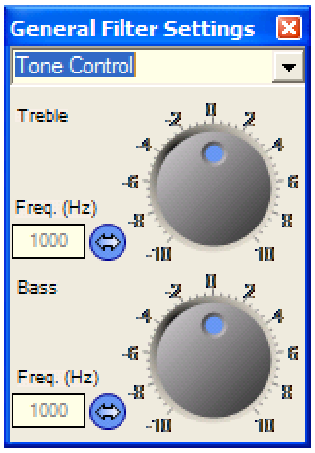
1. *Bessel High/Low Pass, Butterworth High/Low Pass*: You can choose which filter you want for your application by selecting the filter from the drop down menu. The Low Pass and High Pass Filter both have the basic behavior of the General Low/High Pass, but use alternate algorithms for computer of the filter coefficients.

2. *Freq (Hz)*: You can enter the cutoff frequency of your filter in the text box or by clicking the arrows to the right of the text box.

3. *Gain*: You can edit the gain by using the

text box or clicking the arrows to the right of the text box. This setting affects the overall gain of the filter.

### Tone Control



Tone Control

1000

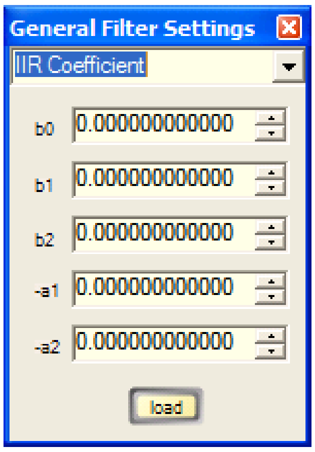
1000

The Tone Control filter is a combination of two 1st order shelving filters, one in the Bass Frequencies (Low Shelf) and one in the Treble Frequencies (High Shelf). This Filter allows you to shape the contour of the frequency response based around the two cutoff frequencies of the shelving filters

1. 1. *Freq (Hz)*: You can enter the cutoff frequency of the shelving filters in the text box or by clicking the arrows to the right of the edit box.

2. 2. *Boost/Cut Gain*: You can enter the value of the boost/ cut for each shelving filter by using the control knob or by right-clicking the knob and entering a value in the pop-up text window.

### IIR Coefficient



IIR Coefficient

b0 0.000000000000

b1 0.000000000000

b2 0.000000000000

-a1 0.000000000000

-a2 0.000000000000

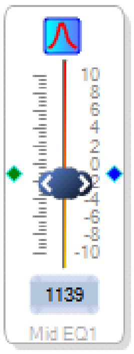
The IIR Coefficient Filter allows creation of your own 1st or 2nd order filter with the ability to specify your own coefficients in the text window. The filter is calculated according to the 2nd order IIR transfer function below:

$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}$$

Note: For a first order filter leave a2 and b2 equal to zero.

After entering your coefficient values click the **load** button to load the filter settings.

## Medium Size Eq



The Medium Size EQ has access to two General 2nd order IIR filters: Peaking EQ and Shelving EQ. The algorithms driving this cell is the same for the other 2nd order filters. This cell offers an alternate cell layout representation and method for controlling parameters which may be beneficial for your application.

As can be seen from the figure on the right, the cell has access to control Frequency, Boost/Cut Gain, and Filter type. By double clicking the middle of the cell, the parameter window is brought up for other key parameters and finer control of editing the parameters.

See the General 2nd Order Algorithms page for more information about the algorithms driving these cells. Below are descriptions of how to use each of the 2nd order filters available from this cell:

- Peaking EQ
- Shelving EQ: Low/High Shelf

In order to use this cell:

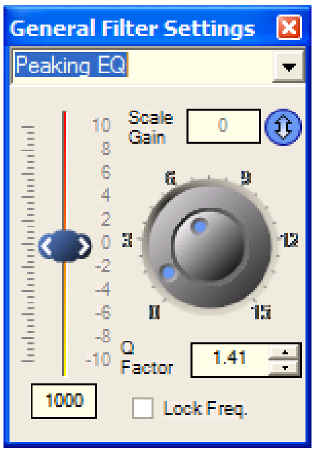
1. Drag and drop into the workspace
2. Right click the cell and select **Add Algorithm > IC N.**
3. From the list select which algorithm meets your needs:
   - **Single - Double Precision**
   - **Single - Single Precision**
   - **Double - Double Precision**
   - **Double - Single Precision**

- **Triple - Double Precision**
- **Triple - Single Precision**
- **Quad - Single Precision**
- **Quad - Double Precision**
- **Penta - Double Precision**
- **Penta - Single  Precision**
- **Hexa - Double Precision**
- **Hexa - Single Precision**

4. Click on the blue filter icon representation to choose whichever filter you want to execute: Peak or High/Low Shelf
5. Double click in the middle of the cell to bring up the EQ parameter window.
6. In the parameter window, select which type of filter you want to use from the top drop down menu and enter the parameters accordingly. (The Peaking EQ is the default filter for the Medium Size EQ).

After selecting your starting algorithm, there is an option to either Grow or Add to this algorithm. Growing the algorithm will add another frequency band to the cell, which is equivalent to having two individual filters in series. Adding an algorithm adds another input/output pair to the cell which is equivalent to adding a filter in parallel.

### Peaking EQ



The Peaking EQ is a filter designed to boost or cut the designated center frequency. Below are the parameters available in this filter and a description of their function.

1. *Center Frequency*: You can enter the center frequency in the text box below the slider bar, or click and drag the horizontal arrows on the slider bar to edit your center frequency. As its name implies the center frequency is the center of the peak boost or cut you create with this filter.

2. *Boost/Cut Gain*: You can edit the filter boost or cut gain by using the slider bar. Negative values will give you a cut at the center frequency and positive values will give a boost at the center frequency.

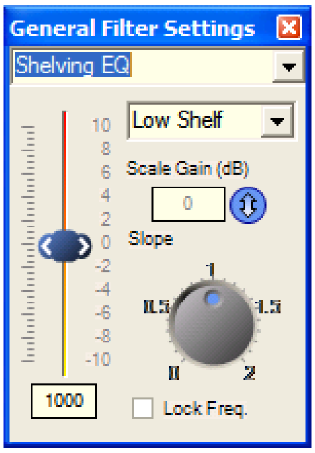
3. *Q Factor*: You can edit the Q Factor by entering a value in the text box, clicking on the arrows of the text box, or using the concentric control knobs. ( The outer knob controls the integer value, while the inner knob controls the decimal value). The Q factor affects the sharpness of the cut/boost at the center frequency. High Q values will give sharper resolution on the center frequency (smaller bandwidth), but also have a lower magnitude, whereas small Q values will have wider bandwidth with larger magnitude.

4. *Scale Gain*: You can edit the scale gain by using the text box or clicking the arrows to the right of the text box. Scale Gain affects the overall gain of the filter.

5. *Lock Frequency*: This checkbox can be enabled so that the horizontal control of the center frequency on the slider bar is not active.

### Shelving EQ: Low/High Shelf

The Shelving EQ is a filter designed to boost or cut all frequencies above (High Shelf) or below (Low Shelf) the cutoff frequency. Below are the parameters available in this filter and a description of their function.



*Low Shelf/High Shelf*: You can choose which filter you want to use by selecting Low or High shelf respectively from the drop down menu.

Low Shelf - cuts or boosts all frequencies below cutoff frequency

High Shelf - cuts or boosts all frequencies above cutoff frequency

2. **Cutoff Frequency**: You can enter the cutoff frequency in the text box below the slider bar, or click and drag the horizontal arrows on the slider bar to edit your center frequency. As its name implies, the cutoff frequency is the cutoff point between the affected shelf boost/cut of the filter to the unaffected flat response.

3. **Boost/Cut Gain**: You can edit the filter boost or cut gain by using the slider bar. Negative values will give you a cut for all frequencies above (High Shelf) or below (Low Shelf) the cutoff frequency and positive values will give a boost for all frequencies above (High Shelf) or below (Low Shelf) the cutoff frequency.

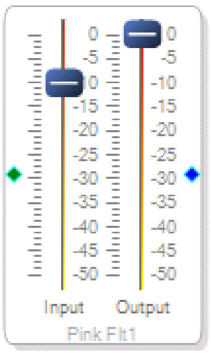
4. **Slope**: You can edit the slope of the filter using the control knob. By right-clicking the control knob, you can enter an exact value into the pop-up text box window. The slope controls the steepness of the filter and affects the transition band between the affected shelf boost/cut of the filter to the unaffected flat response.

5. **Scale Gain**: You can edit the scale gain by using the text box or clicking the arrows to the right of the text box. Scale Gain affects the overall gain of the filter.

6. **Lock Frequency**: This checkbox can be enabled so that the horizontal control of the center frequency on the slider bar is not active.

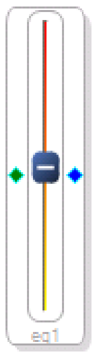
## Pink Filter

The Pink Filter cell takes an input signal and outputs a signal with a 3dB drop per octave. The classical use of this filter is to convert White Noise (flat frequency response) to Pink Noise (-3dB per octave). This can be done by connecting the White Noise Source to the Pink Filter. See the Pink Filter Algorithm page for more information on the algorithm, and also look at the Level Detectors Example, to see the Pink Filter being used in a sample schematic.



After the default algorithm has been established, this cell has the ability to have algorithms added to it. (if you are using more than one DSP board, you will need to add the initial default algorithm for the desired board). Right-click on the cell and select: **Add Algorithm** > **IC N** > **Pink Noise Filter**. This action will add another set of input/output pins for connection.

## Small Size Eq



The Small Size EQ has the same functionality of the Medium Size EQ with an alternate cell layout representation. The Small Size EQ has access to two General 2nd order IIR filters: Peaking EQ and Shelving EQ. The algorithms driving this cell is the same for the other 2nd order filters.

As can be seen from the figure on the right the cell only has access to control the Boost/Cut gain of the filter. For access to the other parameters and finer control, double click the middle of the cell to bring up the parameter window.

See the General 2nd Order Algorithms page for more information about the algorithms driving these cells. Below are descriptions of how to use each of the 2nd order filters available from this cell:

- Peaking EQ
- Shelving EQ: Low/High Shelf

In order to use this cell:

1. Drag and drop into the workspace
2. Right click the cell and select **Add Algorithm** > **IC N**.
3. From the list select which algorithm meets your needs:
   - **Single - Double Precision**
   - **Single - Single Precision**
   - **Double - Double Precision**
   - **Double - Single Precision**
   - **Triple - Double Precision**
   - **Triple - Single Precision**
   - **Quad - Single Precision**

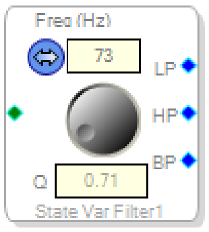
- **Quad - Double Precision**
- **Penta - Double Precision**
- **Penta - Single  Precision**
- **Hexa - Double Precision**
- **Hexa - Single Precision**

4. Double click in the middle of the cell to bring up the EQ parameter window.

5. In the parameter window, select which type of filter you want to use from the top drop down menu and enter the parameters accordingly. (The Peaking EQ is the default filter for the Small Size EQ).

After selecting your default algorithm, there is an option to either Grow or Add to this algorithm. Growing the algorithm will add another frequency band to the cell, which is equivalent to having two individual filters in series. Adding an algorithm adds another input/output pair to the cell which is equivalent to adding a filter in parallel.

## State Variable Filter



The State Variable Filter cell allows for simultaneous access to three different filter types: lowpass, highpass and bandpass. See the State Variable Algorithms page for more information about the calculations and parameters for this algorithm.

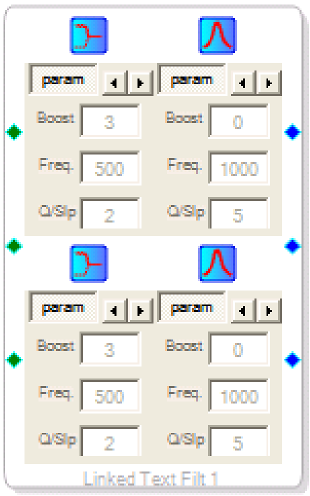
The State Variable Filter w/Q allows for control of the Q value on the cell. The Q affects the sharpness of the cut/boost at the cutoff frequency. You can control the value of the Q by using the control knob, or edit box. The values for Q range from 0.10- 10.0. For external control of the Q, see the State Variable Filter cell.

The three output pins allow you to choose between LP, HP, BP filters. The nature of this algorithm is to compute the filter coefficients for all filter types giving you simultaneous access to all of the filters. If your application does not require use of all filter types you can connect the filter output to a terminal cell.

You can control the value of the cutoff/center frequency by using the edit box or the click and drag arrows.

## Text-In Filter (linked)

The Text-In Filter is an alternative to the General Purpose Filter in that it uses text entry fields to parameter values. The algorithms driving the cells are still the same giving access to a variety of 2nd order IIR filter algorithms. Not all the algorithms offered in the General Purpose Filter are offered in the Text-In filter, but this cell has the advantage that all parameters can be controlled from the cell itself rather than having to open another parameter window.



The linked Text-In filter differs from the (unlinked) version in that it allows simultaneous control of added algorithms. Also when you grow upon an added algorithm, a corresponding parallel filter will be added to match the previous added algorithm.

See the General 2nd Order Algorithms page for more information about the algorithms driving these cells. The available filters with this cell are:

- Peaking EQ
- Shelving EQ: Low/High Shelf
- General: Low/High Pass Filter

In order to use this cell:

1. Drag and drop into the workspace
2. Right click the cell and select **Add Algorithm** > **IC N**.
3. From the list select which algorithm meets your needs:
   - **Single - Double Precision**
   - **Single - Single Precision**
   - **Double - Double Precision**

- **Double - Single Precision**
- **Triple - Double Precision**
- **Triple - Single Precision**
- **Quad - Single Precision**
- **Quad - Double Precision**
- **Penta - Double Precision**
- **Penta - Single  Precision**
- **Hexa - Double Precision**
- **Hexa - Single Precision**

4. Click on the blue filter icon representation to choose whichever filter you want to execute: Peak, Low pass, High pass, Low Shelf, High Shelf.

5. On the param tab enter values for Boost (only Peak and Shelf EQ), Freq, Q/Slope. Click on the right arrow and click on the gain tab. Enter the gain of the filter. These tabs remains the same for each filter, with the exception of the Boost edit box not being available for Low/High Pass Filters.

   **Boost**: Amount of boost/cut applied to designated range of frequencies

   **Freq**: Center of Cutoff Frequency

   **Q/Slope**: Sharpness of frequency response curve

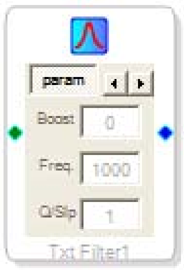
   ***Gain***: overall gain of filter

After selecting your default algorithm, there is an option to either Grow or Add to this algorithm. Growing the algorithm will add another frequency band to the cell, which is equivalent to having two individual filters in series. Adding an algorithm adds another input/output pair to the cell which is equivalent to adding a filter in parallel.

Since this is a linked cell, when you add an algorithm, changing the values in one parameter box, will automatically update the other filter in parallel. Also, after having an added algorithm, if you grow the algorithm, the series filter will automatically add parallel filters to match the previous case. The same is true if you add onto a grown algorithm, the parallel filter will automatically add series filters to match the previous case.

The figure to the right is an example of a default algorithm of Single - Double Precision, grown by 1, with an Added Double - Double Precision. Note that the parameters are the same in the vertical settings because of the linked nature of this cell.

## Text-In filter (unlinked)



The Text-In Filter is an alternative to the General Purpose Filter in that it uses text entry fields to parameter values. The algorithms driving the cells are still the same giving access to a variety of 2nd order IIR filter algorithms. Not all the algorithms offered in the General Purpose Filter are offered in the Text-In filter, but this cell has the advantage that all parameters can be controlled from the cell itself rather than having to open another parameter window.

See the General 2nd Order Algorithms page for more information about the algorithms driving these cells. The available filters with this cell are:

- Peaking EQ
- Shelving EQ: Low/High Shelf
- General: Low/High Pass Filter

In order to use this cell:

1. Drag and drop into the workspace
2. Right click the cell and select **Add Algorithm** > **IC N.**
3. From the list select which algorithm meets your needs:
   - **Single - Double Precision**
   - **Single - Single Precision**
   - **Double - Double Precision**
   - **Double - Single Precision**
   - **Triple - Double Precision**
   - **Triple - Single Precision**
   - **Quad - Single Precision**
   - **Quad - Double Precision**

- **Penta - Double Precision**
- **Penta - Single  Precision**
- **Hexa - Double Precision**
- **Hexa - Single Precision**

4. Click on the blue filter icon representation to choose whichever filter you want to execute: Peak, Low pass, High pass, Low Shelf, High Shelf.

5. On the param tab enter values for Boost (only Peak and Shelf EQ), Freq, Q/Slope. Click on the right arrow and click on the gain tab. Enter the gain of the filter. These tabs remains the same for each filter, with the exception of the Boost edit box not being available for Low/High Pass Filters.

   **Boost**: Amount of boost/cut applied to designated range of frequencies

   **Freq**: Center of Cutoff Frequency

   **Q/Slope**: Sharpness of frequency response curve

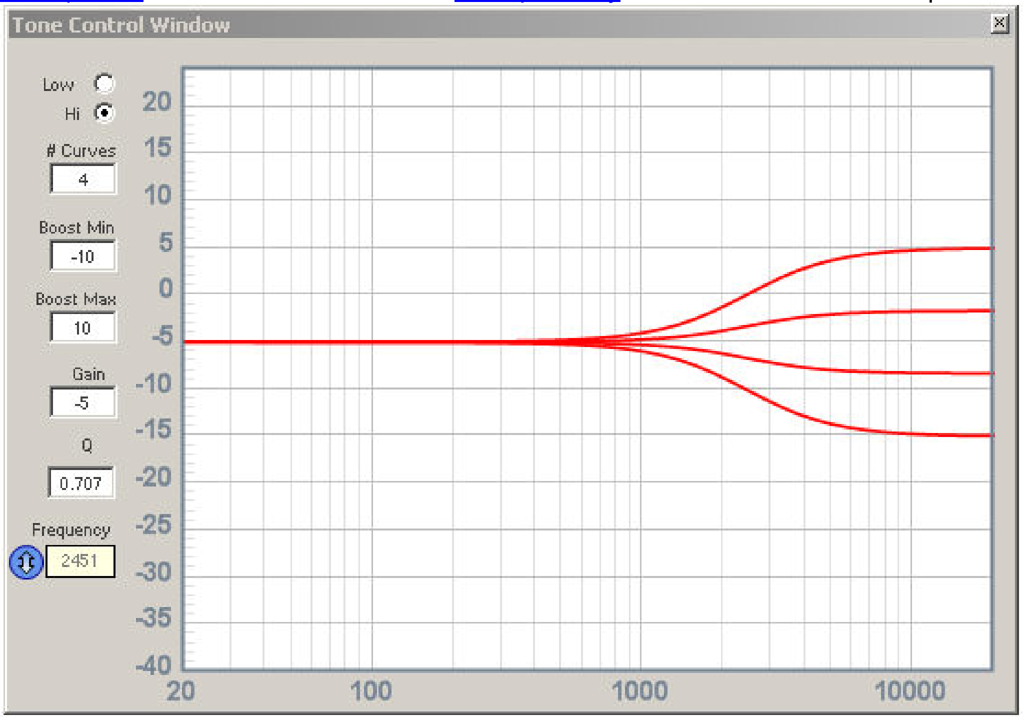
   **Gain**: overall gain of filter

After selecting your default algorithm, there is an option to either Grow or Add to this algorithm. Growing the algorithm will add another frequency band to the cell, which is equivalent to having two individual filters in series. Adding an algorithm adds another input/output pair to the cell which is equivalent to adding a filter in parallel.

The behavior of this cell in terms of growing and adding algorithms is like the other filter cells. A special version of this same filter, the Text-In Filter (linked) forces added algorithms to share the same parameters, and grown algorithms must add a corresponding parallel algorithm.

## Tone Control(With Index lookup)

The tone control cell is simply a biquad filter that stores a set of coefficients in tables internal to the DSP. It takes an input signal and outputs a  tone controlled signal at its output. The cell controls the tone by  attenuating or boosting the signals according to the cutoff frequency and the type of filter (low pass or high pass) of your preference . The frequency response of the filter can be observed from the Tone control window which is shown below .The number of curves can be entered in the text box. The desired frequency response curve can be achieved by appropriate index values from the index look up table cell or the values from the DC input entry cell connected to the Red pin .

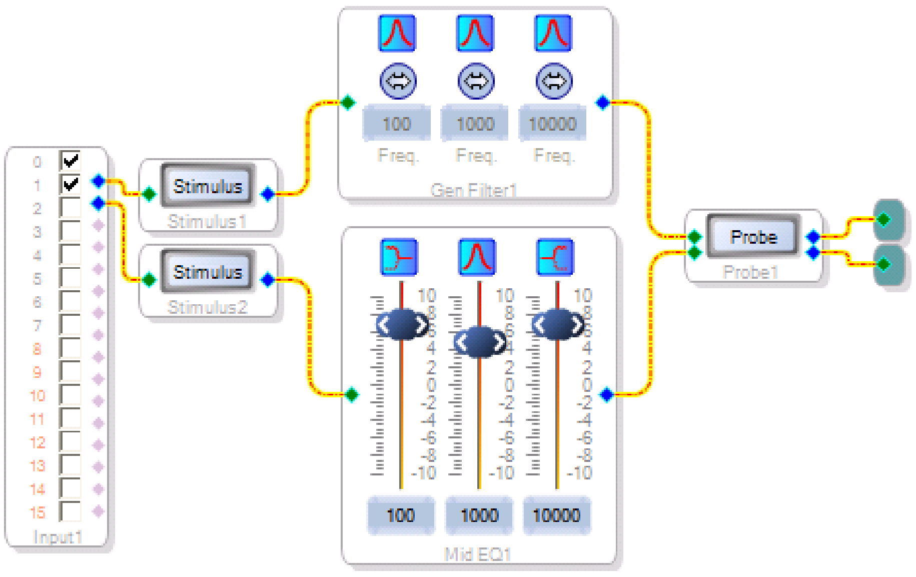

Tone Control Window

Low ○
Hi ●
# Curves: 4
Boost Min: -10
Boost Max: 10
Gain: -5
Q: 0.707
Frequency: 2451

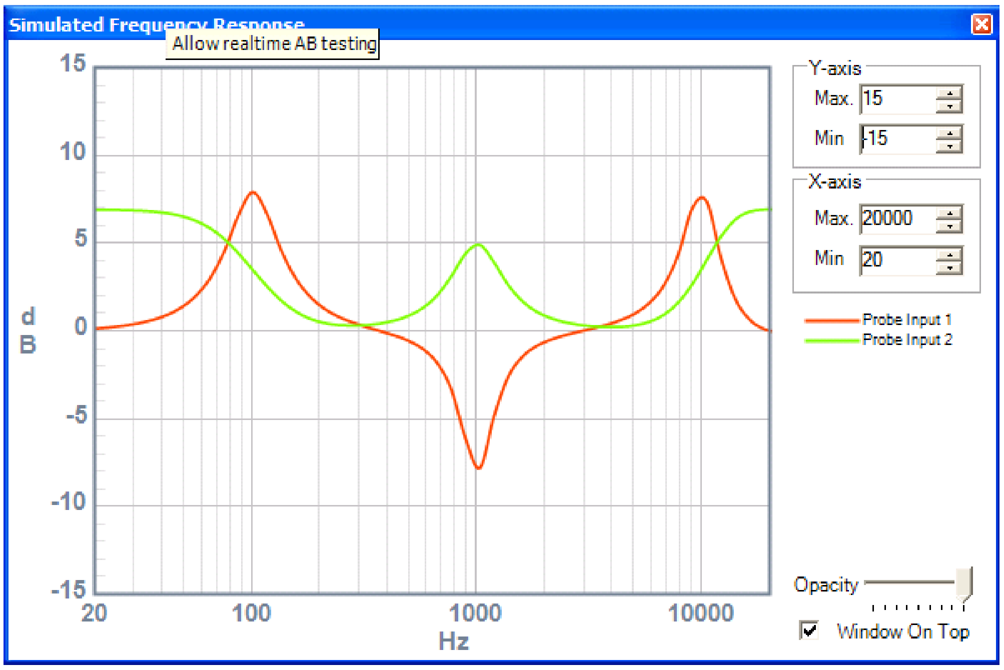
## Filters Example

### Example 6.1

The best way to gain understanding in using the filter cells is to use the Probe and Stimuli cells to plot the Frequency Response of your filter specification. The example below shows a General Purpose Filter with Single - Double Precision algorithm grown by 2 and a Medium Size EQ with the same configuration (Single - Double Precision algorithm grown by 2). This schematic also includes an input block and terminals for completion of signal flow.



This configuration provides the following frequency response graph with the red line representing the General Purpose Filter and the green line representing the Medium Size EQ:


Simulated Frequency Response
Allow realtime AB testing
15
10
5
0
-5
-10
-15
dB
20
100
1000
10000
Hz
Y-axis
Max. 15
Min -15
X-axis
Max. 20000
Min 20
Probe Input 1
Probe Input 2
Opacity
Window On Top

# I/O

The I/O library of the toolbox gives you access to the input/output blocks. These blocks are essential to bring the signal you physically connect to the sigma 100 board to the program and back out to the board's output connections. The following is a list of the available cells under this library.

1. Input
2. Output
3. GPIO input
4. GPIO output
5. Ext interface in
6. Ext interface out
7. Aux ADC Input

8.

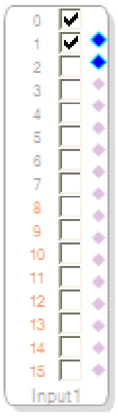
The following pages contain more specific information about the individual cells. Also take a look at the I/O Example  GPIO Example page to see a sample schematic using both I/O cells.

## Input

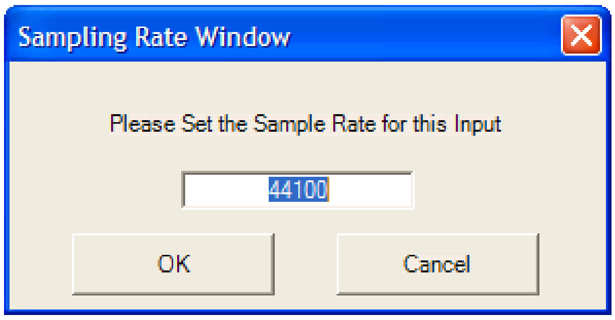
The Input algorithm takes board input and makes it available through the blue output pins. Every DSP has a different input design and the block will only show those inputs that are available. If you have multiple DSP boards connected, you will have to specify which board to use by right-clicking the cell and selecting **Add Algorithm** > **IC N** > *Board*.

- Pins can be enabled/disabled by clicking the check boxes. The default cell will have two pins already enabled for stereo connection.
- Every enabled input must be connected to an output by a checkbox or there will be errors on compilation.
- Right-click the cell name and select **Set Sampling Rate** to bring up the window which allows you to enter the Sampling Rate for the input. The default sampling rate is 44100 Hz

Input cell:



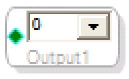
Sampling Rate Window:


Sampling Rate Window

Please Set the Sample Rate for this Input

44100

## Output



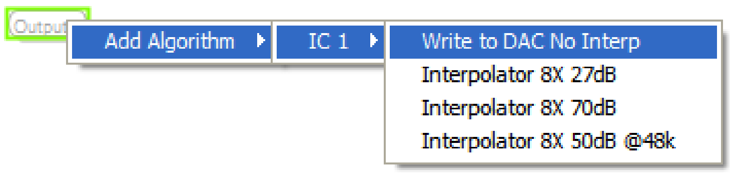
The output algorithm takes the program signal flow and sends it to the board output. Each output block is linked to one output channel. You can select the appropriate output channel you want to send your signal to on the board, from the drop-down menu of the output cell. As you drag and drop more output blocks to your schematic workspace, your number of available output channels from the drop-down menu decreases.

If you have multiple DSP boards connected, you will have to specify which board to use by right-clicking the cell and selecting **Add Algorithm** > **IC N** > *Board*.

Note: The AD1953 has the DACs built-in the DSP core and thus you must choose an interpolation filter for the output.

## AD1953 Output

In order to use output cells with the AD1953 you need to add an algorithm to the cell. As shown in the figure below, by right-clicking the cell you can select which interpolating filter algorithm you would like to implement on the DSP core.



Once you  have selected your output interpolation algorithm, you must select the channel on the cell to be “left”, “right”, or “sub.” It is illegal to have more than one cell writing to the same output channel.

The following are descriptions of the interpolation algorithms:

### Write to DAC No Interp

Writes to the DAC registers with no interpolation. Useful for subwoofer outputs. Distortion above 2kHz will rise due to output slewing. Frequency response at high frequencies will suffer from sinc(x) droop.

### Interpolator 8x 27db

27db DAC interpolation filter. Response flat to 20kHz with a 48kHz sample-rate. Stopband starts at 40kHz (with fs = 48kHz), so images of high-frequency sine waves above 8 kHz will not be attenuated very strongly. Lowest quality, but very low MIPS. Uses 37 instructions.

### Interpolator 8x 70db

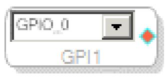
Highest-quality DAC interpolation filter. Response flat to 20kHz with a 44.1kHz sample rate. Uses 80 instructions.

### Interpolator 8x 50db @48k

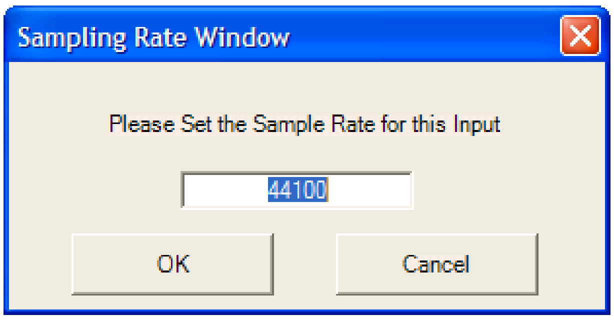
50db DAC interpolation filter. Response flat to 20kHz with a 48kHz sample-rate. Uses 53 instructions.

## GPIO input

The GPIO input algorithm takes the signal flow and sends it to the GPIO conditioning cells through the red pin. There are twelve input channels to the GPIO. You can select the appropriate channel that you want to send your signal from the drop-down menu of the GPIO input cell. As you drag and drop more output blocks to your schematic workspace, your number of available input channels from the drop-down menu decreases.



- Every enabled input must be connected to an output otherwise there will be errors on compilation.
- Right-click the cell name and select **Set Sampling Rate** to bring up the window which allows you to enter the Sampling Rate for the input. The default sampling rate is 44100 Hz


Sampling Rate Window

Please Set the Sample Rate for this Input

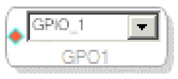
44100

OK Cancel

Note:

This cell will only be useful for DSPs that have GPIO

## GPIO output

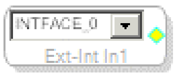


The GPIO output algorithm takes the program signal flow and sends it to the GPIO output. Each output block is linked to one output channel. You can select the appropriate output channel you want to send your signal to on the GPIO, from the drop-down menu of the GPIO output cell. As you drag and drop more GPIO output blocks to your schematic workspace, your number of available output channels from the drop-down menu decreases.

Note:

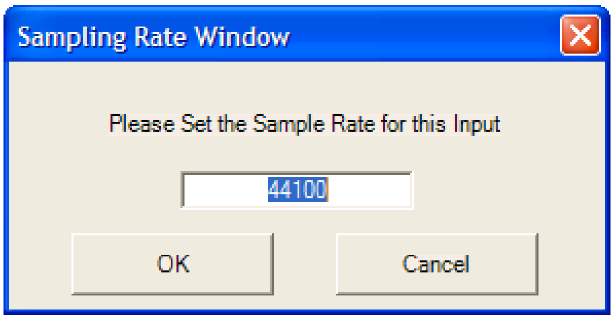
This cell will only be useful for DSPs that have GPIO

## Ext interface in



The Ext interface in algorithm takes the signal from the interface and sends it to the cells in GPIO conditioning library through the green pin. There are eight interfaces, which can be connected to the cells in GPIO conditioning library. You can select the appropriate interface from which it is desired to send the signal, by using the drop-down menu of the Ext interface in cell. As you drag and drop more output blocks to your schematic workspace, your number of available interfaces from the drop-down menu decreases.

- Every enabled input must be connected to an output otherwise there will be errors on compilation.
- Right-click the cell name and select **Set Sampling Rate** to bring up the window which allows you to enter the Sampling Rate for the input. The default sampling rate is 44100 Hz



Sampling Rate Window

Please Set the Sample Rate for this Input

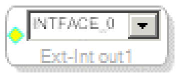
44100

Note:

This cell would be useful for parts that have a self boot mechanism that use external interface registers.

## Ext interface out

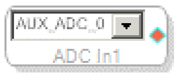
The Ext interface out algorithm takes the program signal flow and sends it to the interface output. Each output block is linked to one interface. You can select the appropriate interface using  the drop-down menu of the Ext interface out cell. As you drag and drop more Ext interface out blocks to your schematic workspace, your number of available interfaces from the drop-down menu decreases.



Note:

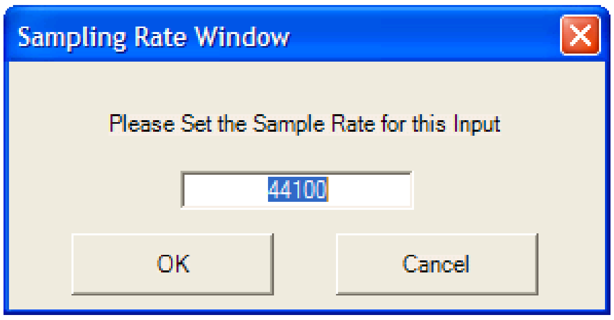
This cell  will only be useful for DSPs that have general purpose inputs or auxiliary adc.

## Aux ADC Input



The Aux ADC Input algorithm takes the digital signal from the auxiliary analog to digital converter and makes it available at the output pin. There are three ADCs. The output can be available from any one of the ADC. Selection of the ADC can be done by using the drop-down menu of the Aux ADC Input cell. As you drag and drop more output blocks to your schematic workspace, your number of available auxiliary ADC from the drop-down menu decreases.

- Every enabled input must be connected to an output otherwise there will be errors on compilations.
- Right-click the cell name and select **Set Sampling Rate** to bring up the window which allows you to enter the Sampling Rate for the input. The default sampling rate is 44100 Hz



Please Set the Sample Rate for this Input

44100

Note:

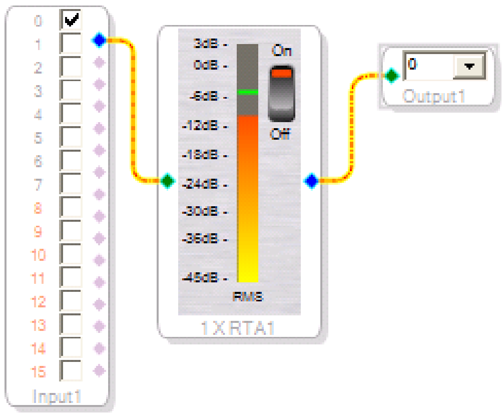
This cell will only be useful for DSPs that have auxiliary ADC.

## Examples

### I/O Example

#### Example 7.1

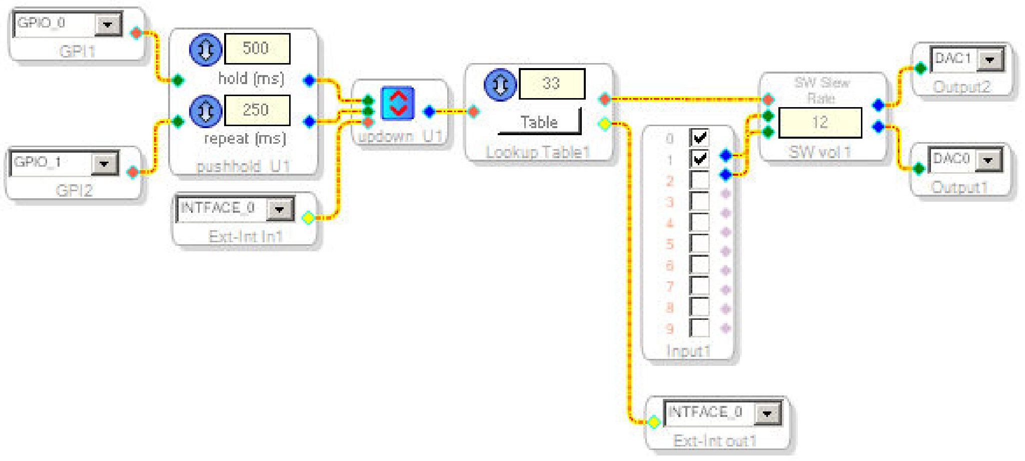
This is a very simple I/O schematic which shows one input channel activated connected to a Single Level Detector to show that signal is being received, and then sent back to the board via the output cell.



## GPIO example

### Example 7.2

This is a simple GPIO I/O schematic diagram. It has two GPIO input cells that drive the push and hold cell . The outputs of these cells are used to select the index from the index look up table .The up/down increment cell is used to either increment or decrement the index based on its input levels .The Ext interface in cell is connected to the red pin of up/down increment cell . The outputs of the look up table are used to drive the Ext interface out and the red pin (the pin used for external representation connection ) of Single slew ext vol. Then the signal is given to two digital to analog convertors which is nothing but the analog output of the board.



Note:

As can be seen from the above schematic, the sigma 100 evaluation board supports only up to two analog inputs and seven digital inputs.

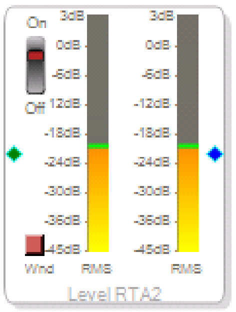
# Level Detectors

The Level Detectors library of the ToolBox give you access to cells which allow for graphical representation of signal level. The following is a list of the available cells under this library:

- Level Detector Designer
- Seven Band Level Detector
- Single Level Detector

The following pages contain more specific information about the individual cells. Also take a look at the Level Detectors Example page to see a sample schematic using some of the Level Detectors cells.

## Level Detector Designer



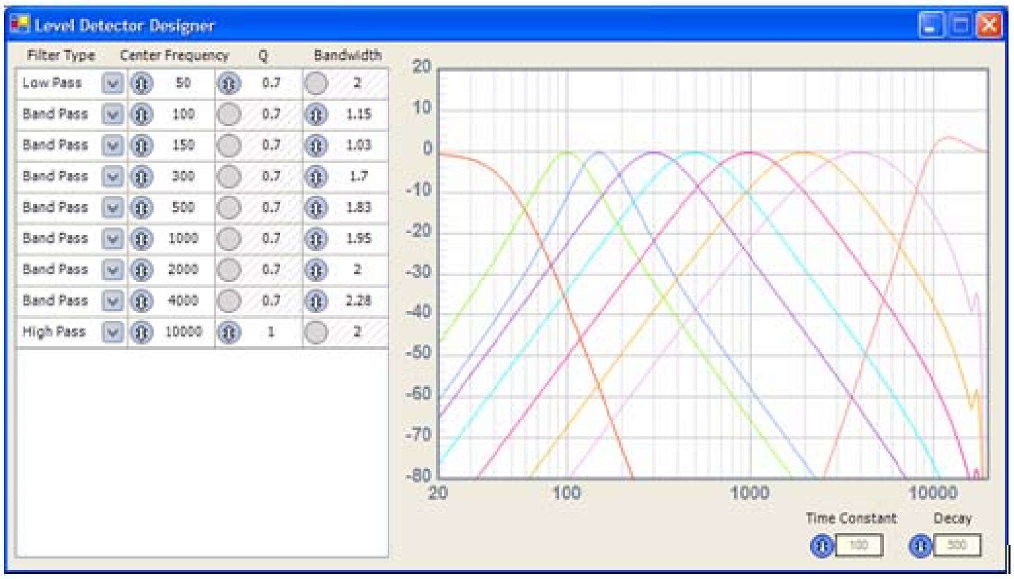
The Level Detector Designer is a unique cell that has the ability to define  your own frequency bands and time constants for the display.

In order to use this cell:

1. Drag and drop into the workspace
2. Right click the cell and select **Add Algorithm** > **IC N**
3. From the list select the algorithm that meets your needs:
   - Pass thru
   - RTA to output
   - Pass thru /minimal reading. Single Precision
   - Pass thru /minimal reading. Double Precision
4. Click on the red **Wnd** button to bring up the designer window so that you can enter the parameters for your design.

After selecting your default algorithm, there is an option to Grow your algorithm. The default cell only contains one color bar indicator, but you can grow your algorithm to account for multiple frequency bands. The figure to the right is an example of the Pass thru algorithm grown by 1.

It is important to understand the parameters in the designer window in order to make the most of your level detector design. The figure below is the designer window from a Pass thru algorithm grown by 8. There are a total of 9 bands  each with the option of choosing Filter type, Center Frequency, and Q or Bandwidth (depending on filter type). There is also an overall Time Constant and Decay value that can be set in ms to control the refresh time of the color bar display.


Level Detector Designer

| Filter Type | Center Frequency | Q | Bandwidth |
|---|---|---|---|
| Low Pass | 50 | 0.7 | 2 |
| Band Pass | 100 | 0.7 | 1.15 |
| Band Pass | 150 | 0.7 | 1.03 |
| Band Pass | 300 | 0.7 | 1.7 |
| Band Pass | 500 | 0.7 | 1.83 |
| Band Pass | 1000 | 0.7 | 1.95 |
| Band Pass | 2000 | 0.7 | 2 |
| Band Pass | 4000 | 0.7 | 2.28 |
| High Pass | 10000 | 1 | 2 |

Time Constant: 100
Decay: 500

Parameters:

**Filter Type**: The Level Detector Designer algorithm allows for Low Pass, High Pass, and Bandpass filters to be set for the individual bands of the color display

**Center Frequency**: You can specify the center/cutoff frequency of the filter you choose.

**Q**: This parameter is available for control with the Low Pass and High Pass filters. The Q determines the sharpness of the filter and is inversely related to the Bandwidth.

**Bandwidth**: This parameter is available for control with the Bandpass filter. The Bandwidth determines the range of frequencies you will affect.

**Time Constant**: The time constant is a value in ms that designates the averaging time of the detector; how rapidly it detects level changes in the signal.

**Decay**: The decay value is another type of time constant that designates the time for the display to adjust from its previous value.

Algorithms:

**Pass thru**

The Pass thru algorithm receives a 5.19 (5 bytes used to represent the integer portion of the value, 19 bytes for the decimal part) number during readback and then converts this value to a decimal value which is represented in decibels. This is the value that is visually seen on the display bar. This is a Pass thru system because the output of this block is the same signal that is being sent into the cell.

### RTA to output

The RTA to output algorithm outputs the RMS level value through the red output pin. This pin is designated in red because there is not actual audio being sent out, just the level values for each frequency band. You will notice that when you grow this algorithm, each pin corresponds to a frequency band.

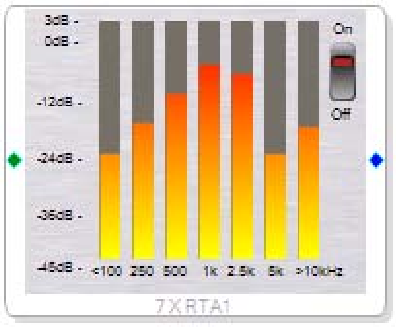
### Pass thru/ minimal reading, Single and Double Precision

The Pass thru/ minimal reading functions in a similar manner to Pass thru, but saves on communication bandwidth, at the expense of losing precision. Pass thru/minimal reading only uses 1 byte for the readback values and then converts this value to a decimal value which is represented in decibels.

Double precision uses 56 bits for each calculation, takes 10 instructions per filter, and should normally be used. Single precision uses 28 bits for calculations, takes 6 instructions per filter, and saves 3 data ram spaces over the double precision algorithm. Single precision should not be used for frequencies below 1/10 the sampling frequency, or for high Q filters.

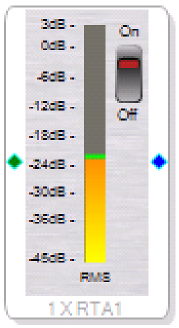
## Seven Band Level Detector

The Seven Band Level Detector cell takes the input signal and represents the level in seven different frequency bands. The frequency bands are <100Hz, 250Hz, 500Hz, 1kHz, 2.5kHz, 5kHz, and >10kHz. The level is represented in decibel value (dB) by the color bars. The Level Detector contains an on/off switch that must be enabled for the display to be viewed.



## Single Level Detector

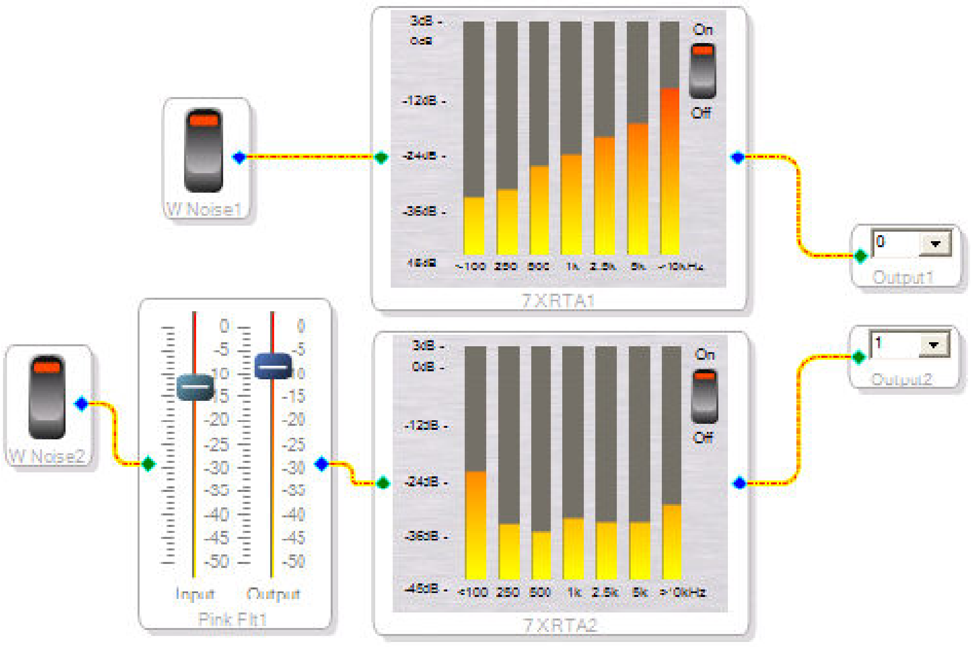
The Single Level Detector responds to the RMS level of the signal, and represents the level in dB with a color bar. The display is refreshed 10 times per second. The green bar is a visual display indicator which has no DSP function; it is a delayed visual track of the max RMS value.



## Level Detectors Example

### Example 8.1

The following example uses two Seven Band  Level Detectors to show the difference in levels between White Noise and Pink Noise (created using White Noise source and Pinking Filter). There are also two outputs for the sake of signal flow. For more information about the difference between the two noise sources, look at the Pink Filter Algorithm page.

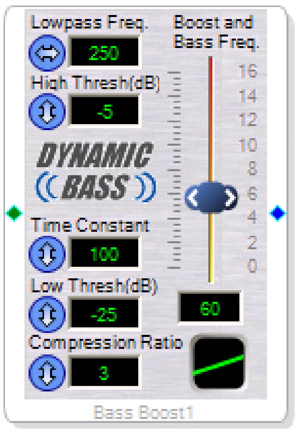


# ADI Algorithms

The ADI Algorithms library of the ToolBox offers access to unique, proprietary, ADI algorithms for specific audio applications. The following is a list of the available cells under this library:

- Dynamic Bass Boost
- Dynamic Enhancement
- Loudness Ctrl (Lower End)
- Midnight Mode
- Phat Stereo
- Reverb
- Vocal Chorus

## Dynamic Bass Boost



The Dynamic Bass Boost cell provides variable bass boost dependent on input signal level. Lower gain levels may require more bass than higher gain levels. The filter dynamically adjusts the amount of bass boost depending on the volume of the input signal, by using a variable Q filter.

The filter calculates and applies a boost between the Threshold and the Mininum Gain (Min Gain) settings. A fixed maximum boost is applied to input levels above Min Gain. Similarly a fixed maximum boost is applied to inputs below Threshold.

There are seven parameters available for control on the cell, which are described below:

**Lowpass Freq.** - You can enter this value in the edit box or use the control arrows. The Lowpass frequency ranges from 20 - 250Hz. Only frequencies lower than the Lowpass Freq. setting are used by the detector for determining the amount of bass boost.

**High Threshold (dB)** - You can enter this value in the edit box or use the control arrows. The High Threshold value ranges from -20 to 10dB and it is the upper threshold of the detector. Detected signals higher than the minimum gain (Min Gain) will not influence the boost calculation, for those signals, a fixed boost will be applied.

**Time Constant** - You can enter this value in the edit box or use the control arrows. The time constant ranges from 0 to 500 milliseconds. It controls the RMS time constant for the detector. Effect attack and release time will get affected by this parameter

**Low Threshold (dB)** - You can enter this value in the edit box or use the control arrows. The Low Threshold value ranges from -100 to -20dB and it is the lower threshold of the detector. Any signal into the detector below Threshold will not influence the boost calculation and receives a fixed boost.

**Compression Ratio** - You can enter this value in the edit box or use the control arrows.

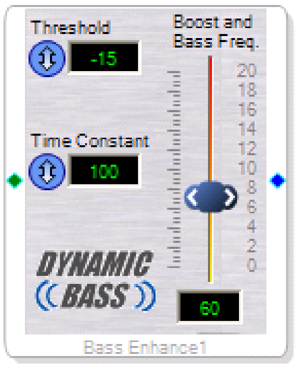
The Compression Ratio ( Better called dynamic boost ratio.) ranges from 1 to 15 and it controls the rate at which bass frequency boost changes from the low threshold to the high threshold.

**Boost** - You can control this value using the boost slider. The boost ranges from

0 to 16dB and it controls the maximum dynamic boost gain applied to the algorithm.

**Bass Freq**. - You can enter this value using 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.

## Dynamic Enhancement



The Dynamic Enhancement cell provides variable bass Enhancement dependent on input signal level. Lower gain levels may require more bass than higher gain levels. The filter dynamically adjusts the amount of bass Enhancement depending on the volume of the input signal, by using a variable Q filter.

The filter calculates and applies a Enhancement above the Threshold setting. A fixed Enhancement is applied to input levels above Threshold Gain.

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

**Time Constant** – The value for the Time Constant will be enter 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.

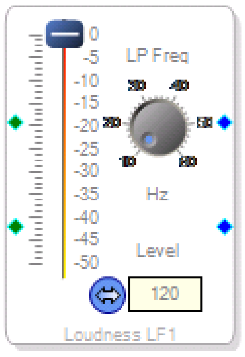
**Threshold (dB)** - The value for the Threshold will be entered in the edit box or use the updown arrows to enter the value. Threshold value ranges from -24 to -20dB . Any signal into the detector below Threshold will not influence the boost calculation and receives a fixed Enhancement.

**Boost** – The Boost will be control using the boost slider. The boost ranges from

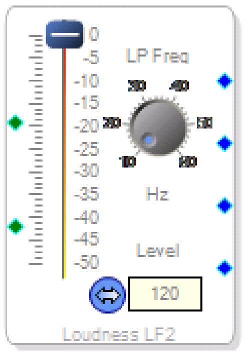
-24dB to 20 dB and it controls the maximum dynamic Enhancement gain applied to the algorithm.

**Bass Freq.** – The Bass Frequency value will be enter 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.

## Loudness Ctrl (Lower End)



The Loudness Ctrl 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.



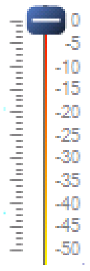
There are two default algorithms that can be used with this cell:

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

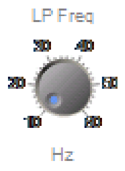
They both function the same, except the latter algorithm contains another pair of stereo outputs that allow the signal to pass through unaffected by the low-end loudness boost, but it will be affected by the overall volume level slider. The figure in the bottom right shows the two extra output pins below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

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

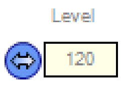
1. The volume slider controls the output gain of the entire signal and is also the control factor for the loudness algorithm. At low levels, the loudness algorithm needs to boost the low frequencies more than at higher frequencies. Note that at 0dB despite what the Level for the loudness algorithm may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



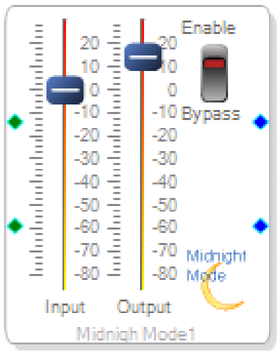
2. The LP Frequency control knob allows you to edit the cutoff frequency of the low-pass filter. The default value of 10 Hz approximates the Fletcher-Munson curve.  Increased frequency values will provide greater bass frequency bandwidth gain.



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 id dependant on the signal level.



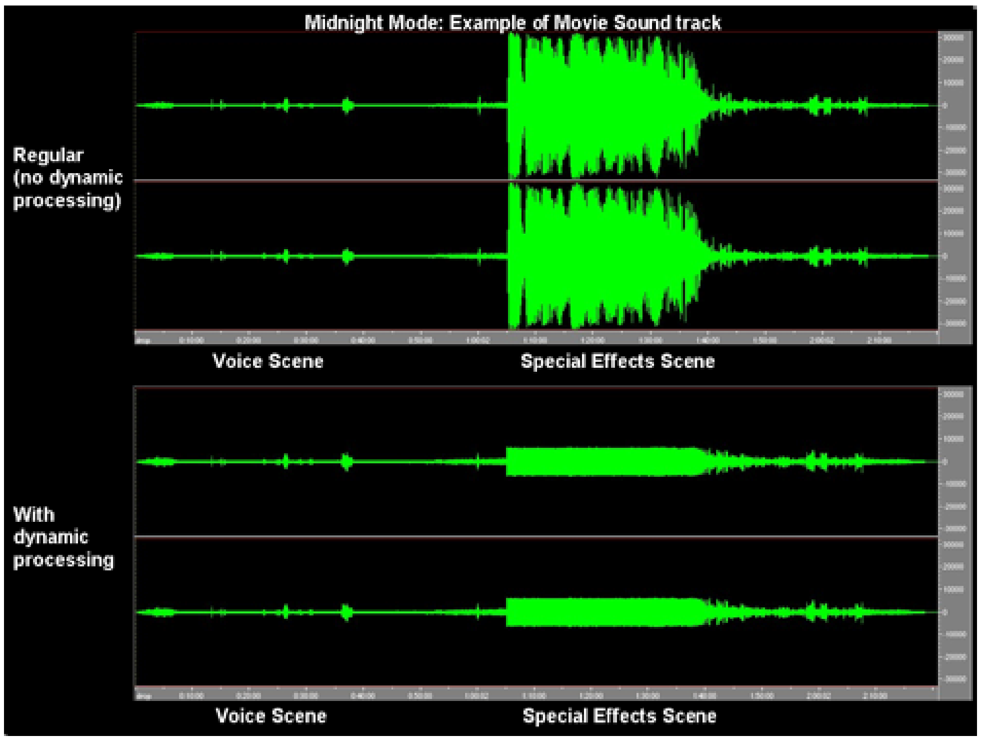
## Midnight Mode



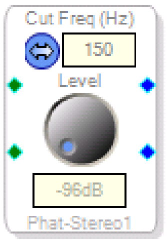
Midnight Mode is a dynamic processing algorithm primarily used for movie sound tracks.  A classic example for using Midnight Mode would for quiet listening environments, where loud sounds can disturb people near the sound source. By Enabling Midnight Mode, high volume passages  are attenuated, and very low levels are raised, while dialog remains unchanged. This allows for higher overall listening volumes and better clarity for the intended listeners, while avoiding disturbing others who may be nearby. Depending on the input level of the source, loud peaks such as explosions in a movie soundtrack will be dynamically attenuated, while softer sections will be increased in level to aid in clarity. Essentially the dynamic range of the soundtrack will be compressed to not disturb other people nearby the source, while maintaining the effect and intelligibility of the soundtrack for the active listeners.

- This algorithm defaults to bypassed - click the toggle switch to enable Midnight Mode.
- The Input slider works as a volume control, changing the input gain to Midnight Mode.  If the cell is currently "Bypassed" it works purely as a volume control.
- The Output slider controls the output post gain.

Below is a figure showing the same clip from a movie soundtrack with Midnight Mode bypassed and enabled. Notice the unchanged volume for the low level signals (voice)


Midnight Mode: Example of Movie Sound track
Regular (no dynamic processing)
Voice Scene
Special Effects Scene
With dynamic processing
Voice Scene
Special Effects Scene

## Phat Stereo



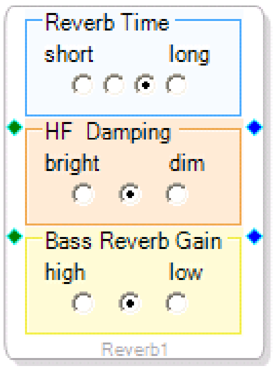
Phat Stereo™ is a spreading algorithm that uses stereo cross-coupling to simulate surround sound in stereo speakers. Since the ear is responsive to interaural phase shifts below 1 kHz, this increase in phase shifts results in a widening of the stereo image. This 3D enhancement provides an enriched surround field designed for headphones or stereo speakers.

There are two parameters available for control on the cell, which are described below:

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

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

## Reverb



The Reverb algorithm accomplishes the effect of simulating the natural reverberation of an echoic room, and mixing it back with the original sample. Reverberation is the total sound field remaining in a room once the original source is not active. This cell allows you to control the following three parameters:

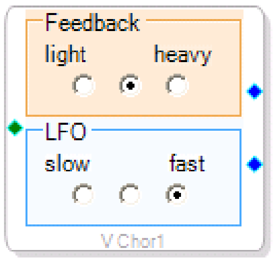
**Reverb Time** - You have control over the Reverb Time settings (short - long) with radio buttons. The Reverb time is the amount of time it takes for the reverberations to decay. This is quantified by measuring how long it takes the Sound Pressure Level of the signal to decay to one-millionth of its original value (which equals a 60dB reduction).

**HF Damping** - You have control over the HF Damping settings (bright - dim) with radio buttons. HF damping is a control measure of the filters, which control the brightness of the reverb reflections.

**Bass Reverb Gain** -You have control over the Bass Reverb Gain settings (high- low) with radio buttons. Bass Reverb Gain controls the gain of the reverb reflections that are attributed the low-frequencies.

After the default algorithm has been established, this cell has the ability to have algorithms added to it. (if you are using more than one DSP board, you will need to add the initial default algorithm for the desired board). Right-click on the cell and select: **Add Algorithm** > **IC N** > **Reverb**. This action will add another pair of stereo inputs/outputs.

## Vocal Chorus



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

This algorithm is intended to be used and optimized for human voice.

- The Feedback radio button settings (light to heavy), determine how much of the delayed signal to mix back with the un-delayed signal.
- The LFO radio button 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 has the ability to have algorithms added to it. (if you are using more than one DSP board, you will need to add the initial default algorithm for the desired board). Right-click on the cell and select: **Add Algorithm** > **IC N** > **Chorus**. This action will add another input pin and pair of output pins for connection.

# Volume Controls

The Volume Controls library of the ToolBox gives you access to numerous cells that contain switches and slider controllers that affect the volume of your signal. Check the Algorithms section for more information about the algorithms driving the Volume Control cells. The following is a list of the available cells under this library.

- Adjustable Volume Control
- Mono Switch 1xN
- Mono Switch Nx1
- Multiple Volume Control
- Single slew ext vol
- Single SW slew vol
- Single Vol (shared)
- Single Volume Control
- Stereo Switch 2xN
- Stereo Switch Nx2
- Surround Sound Volume Control

The following pages contain more specific information about the individual cells. Also take a look at the Volume Controls Example page to see a sample schematic using some of the Volume Control cells.

Any of the Volume Controls cells that contain a volume slider can be edited to specify max/min values and step sizes for the slider. In order to access this window, right-click the slider and edit the values in the pop-up window.

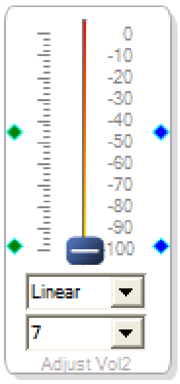
Note: With certain version DSP boards, you will have the option for a slew ram algorithm which ramps to the target volume.

**AD1940/AD1941**: Single Volume Control utilizing 1 slew RAM location on the DSP.  It is set permanently to a linear slew ramp and a time constant of 8 (more info). 64 volume controls are available.

**AD1953**: Single Volume Control utilizing one SPI register per channel for hardware volume adjustment.  Ramp type defaults to fast and takes 512 frames to reach zero from a gain of 1.0 (more info).  AD1953 - 8 volume controls are available.  AD1954 - 3 volume controls are available.

For more information about this, see the Algorithms section of the manual.

## Adjustable Volume Control



The Adjustable Volume Control allows you to control which slew curve to use for the cell, and the time constant for the slew algorithm. The first drop down menu on the cell allows you to choose between: Linear, Const dB, and RC-type. The second drop down menu allows you to choose a value between 0 (fastest ramp) -15 (slowest ramp) for the time constant.

**Linear** - Slews to target using a fixed step size. The ramp ranges from 6.75ms to 213.4ms

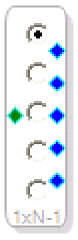
**Constant dB** - Slews to target using the current value to calculate the step size.  The resulting curve has a constant rise and decay when measured in dB. The ramp ranges from 6.1ms to 1.27s

**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 6.1ms to 1.27s

The default cell already comes with the Gain slew algorithm, but you can add pins for input/output by right-clicking the boarder or title of the cell and Adding an algorithm. There is only one slider control for the input/output pins and the algorithm selected with time constant value, will apply the same to all inputs.

The figure to the right is an example of the default cell with 1 added algorithm.

## Mono Switch 1xN



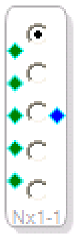
The mono switch 1xN allows the input signal to be routed to one of many possible output paths. The radio button control switch selects which output path the signal is sent to. In order to use this cell:

1. Drag and drop into the workspace
2. Right click the cell and select **Add Algorithm** > **IC N**
   - **Mono/Slew**
   - **Mono**

The cell is now ready to use with a 1x2 connection default. This algorithm can be grown, by right-clicking the cell, to select your desired number of output paths. The figure to the right is the default cell grown by 3.

Note: The Mono/Slew algorithm uses N Hardware Volume Controls in the target/slew RAM to ramp the source volume up/down rapidly when selected. This algorithm avoids "clicking" when switching between the output paths.

## Mono Switch Nx1



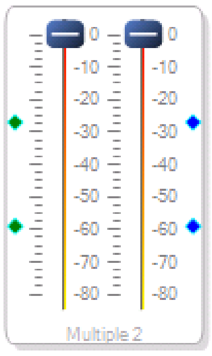
The mono switch Nx1 allows for selection between multiple inputs. The radio button control switch selects which input pin will be sent to the output. In order to use this cell:

1. Drag and drop into the workspace
2. Right click the cell and select **Add Algorithm** > **IC N**
   - **Mono/Slew**
   - **Mono**

The cell is now ready to use with a 2x1 connection default. This algorithm can be grown, by right-clicking the cell, to select your desired number of inputs. The figure to the right is the default cell grown by 3.

Note: The Mono/Slew algorithm uses N Hardware Volume Controls in the target/slew RAM to ramp the source volume up/down rapidly when selected. This algorithm avoids "clicking" when switching between the output paths.

## Multiple Volume Control



The Multiple Volume Control allows for gain adjustments to be made individually to each of its inputs. Every input pin has its own corresponding volume control. In order to use this cell:

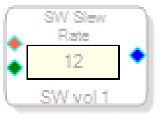
1. Drag and drop into the workspace
2. Right click the cell and select **Add Algorithm** > **IC 1**
   - **Gain (slew)**
   - **Gain (no slew)**

The AD1940/AD1941 includes support for volume controls that use a target/slew RAM location to auto-ramp from one value to a desired final value. For the Multiple Volume Control the slew curve is set to be linear ramp with time constant 8.

The cell is ready to use with the default one input/output pin connection. By right-clicking on the boarder or title of the cell, you can Add to the algorithm to have your desired number of input/output pairs with individual volume control. The figure to the right is the default cell grown by 1.

Note: Right clicking the center of the cell will allow you to enter specific parameters for the volume controller in the pop-up menu.

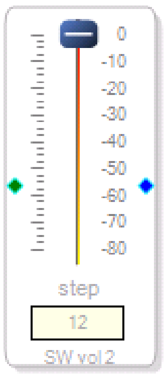
## Single slew ext vol



The Single Slew ext vol cell allows for external control of the volume gain. This cell would be appropriate for applications such as physical external control communication to a 8-10bit ADC. Connect your regular signal flow in the schematic workspace to the green pin, and use the red pin for your external representation connection. The text window allows you to specify a slew rate time constant value between 1-23, which is a larger range than the Adjustable Volume Control cell.

If you have more than one DSP board connected, you will have to right-click the cell to add the algorithm. After the default cell is selected you have the ability to Add and Grow to this cell. For an example of the difference between adding and growing with volume control cells, look at the Single Vol (shared) page.

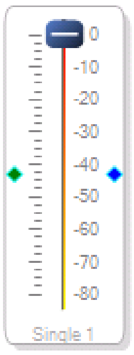
## Single SW slew vol



The Single SW slew vol uses the same algorithm for volume control as the Single slew ext vol, but is a software (SW) application rather than external. Here you directly connect your signal flow from the schematic workspace to the cell and have control of the volume from the software side (like the other sliders in this Volume Control library). However you have the option for entering the slew rate 1-23 as with the Single slew ext vol.

If you have more than one DSP board connected, you will have to right-click the cell to add the algorithm. After the default cell is selected you have the ability to Add and Grow to this cell. For an example of the difference between adding and growing with volume control cells, look at the Single Vol (shared) page.

## Single Vol (shared)

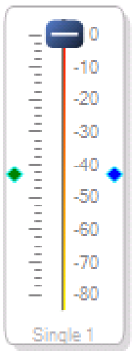


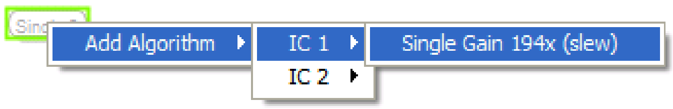
The Single Vol (shared) cell is a volume slider that is programmed with the slew algorithm. This is important to note because if your DSP model does not support the slew algorithm, you must use the Single Volume Control which is direct write and has the option for the algorithm with no slew. However, if your DSP does support the slew algorithm, it is recommended that you use this cell as opposed to the Single Volume Control because it is more powerful. This cell has the ability to Add and Grow algorithms which is very important especially in applications for which multiple DSP boards will be used. For this reason the shared volume control allows you to grow upon a particular algorithm already selected for a DSP.

The following is a step-by-step example showing the difference between an added and grown algorithm with two DSPs. It is important to note however that there is no visual indication of the difference between a grown and added algorithm on the cell itself. The only visual indication of different DSP boards is the different color wires used to connect cells.

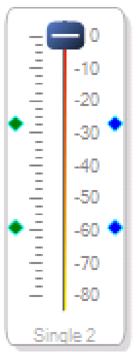
1. Drag and drop the cell into the workspace. It will be an empty cell as seen to the right if you have multiple DSP boards connected.

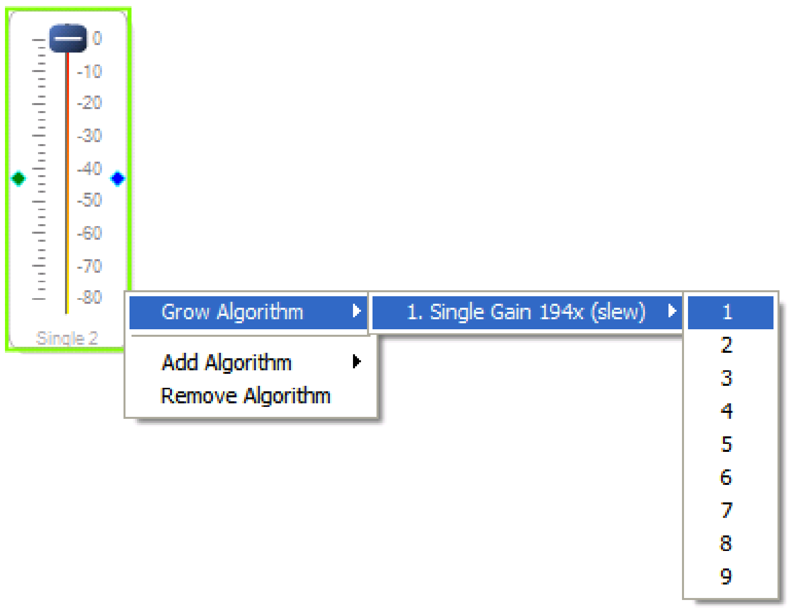
2. Right-click the cell and select **Add Algorithm** > **IC 1** > **Single Gain xxxx (slew)**. Note: If you only have one DSP connected when you drag and drop the cell into the workspace, it does not necessitate that you add an initial algorithm.



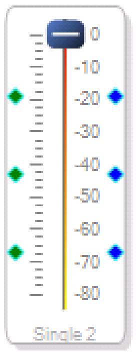


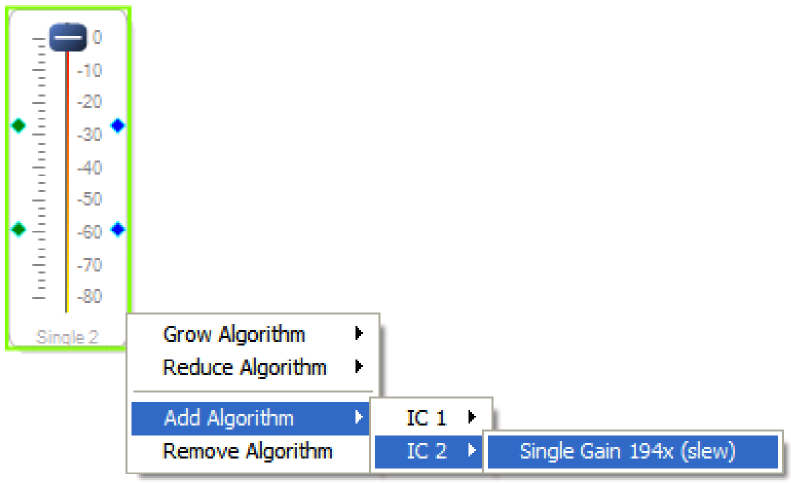
3. Right click the border or title of the cell and select **Grow Algorithm** > **1. Single Gain xxxx (slew)** > **1**. This adds another set of input/output pins with the same DSP board and algorithm as the initial algorithm selected in step 2.



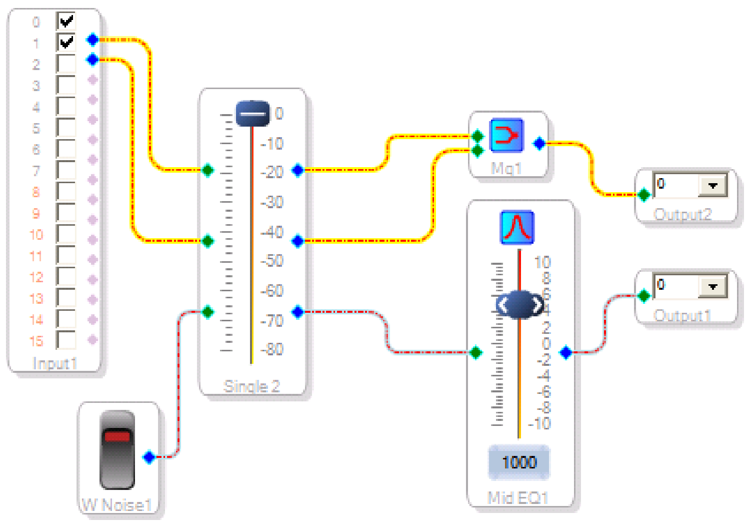


4. Right click the border or title of the cell and select **Add Algorithm** > **IC 2** > **Single Gain xxxx (slew)** > **1**. This adds another set of input/output pins corresponding to a different DSP board. Now other cells that you wish to connect to these pins, must either be independent cells which are not associated with any DSP, or they must have the same DSP algorithm associated with the cell.



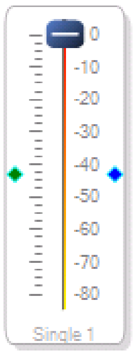


Now you have the top two pins associated with the DSP under IC 1 and then bottom pin is associated with the DSP under IC 2. In order to wire this cell to other cells, you must maintain the same DSP association; the software will not allow you to connect pins from one DSP to another. This will be indicated by different color pins for the different DSP boards as seen below:



## Single Volume Control

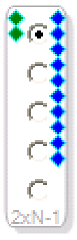
The Single Volume Control cell is a volume slider that has access to both the slew and no slew algorithm. By right-clicking the boarder or title of the cell, you have the ability to Add algorithms which adds pins controlled by the same volume slider, regardless of what each individual pin algorithm may be using. Each pin can be associated with a different DSP and/or algorithm (slew or no slew) while being controlled by the same volume slider. It is important to be careful when having multiple algorithms selected for the same cell because there is no visual indication of the



Right-clicking the center of the cell allows you to edit specific parameters for the control slider in the pop-up menu.

This cell does not have the ability to grow, but for that option use the Single Vol (shared) cell.

## Stereo Switch 2xN



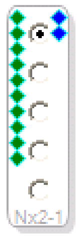
The stereo switch 2xN allows the input stereo signal to be routed to one of many possible stereo output paths. The radio button control switch selects which stereo output path the signal is sent to. In order to use this cell:

1. Drag and drop into the workspace
2. Right click the cell and select **Add Algorithm** > **IC N**
   - **Stereo/Slew**
   - **Stereo**

The cell is now ready to use with a 2x4 connection default. This algorithm can be grown, by right-clicking the cell, to select your desired number of output paths. The figure to the right is the default cell grown by 3.

Note: The Stereo/Slew algorithm uses N Hardware Volume Controls in the target/slew RAM to ramp the source volume up/down rapidly when selected. This algorithm avoids "clicking" when switching between the output paths.

## Stereo Switch Nx2



The stereo switch Nx1 allows for selection between multiple stereo inputs. The radio button control switch selects which stereo input pins will be sent to the stereo output. In order to use this cell:

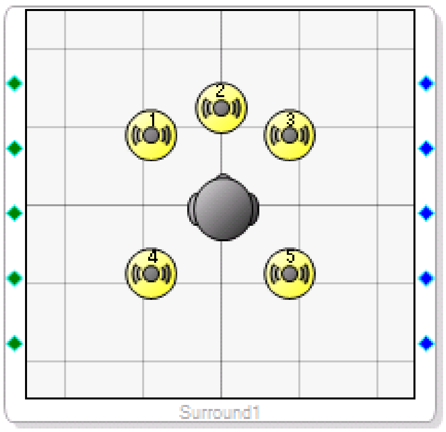
1. Drag and drop into the workspace
2. Right click the cell and select **Add Algorithm** > **IC N**
   - **Mono/Slew**
   - **Mono**

The cell is now ready to use with a 2x1 connection default. This algorithm can be grown, by right-clicking the cell, to select your desired number of inputs. The figure to the right is the default cell grown by 3.

Note: The Stereo/Slew algorithm uses N Hardware Volume Controls in the target/slew RAM to ramp the source volume up/down rapidly when selected. This algorithm avoids "clicking" when switching between the output paths.

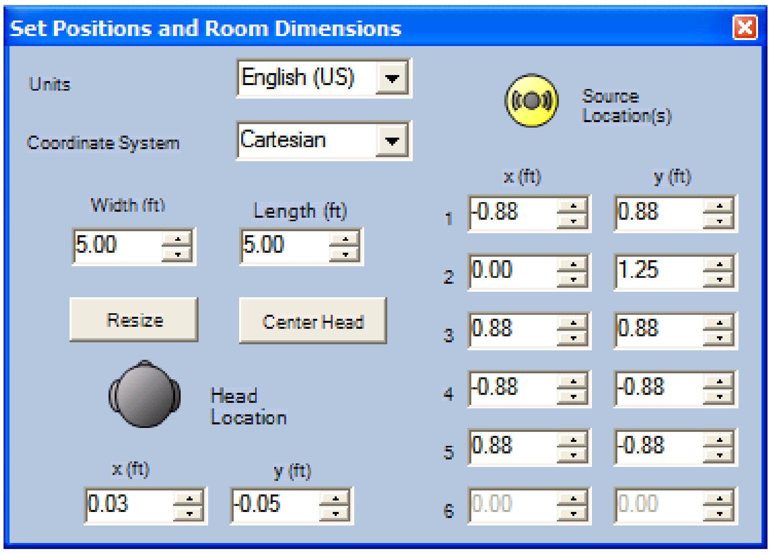
## Surround Sound Volume Control

The Surround Sound Volume Control cell enables positioning of surround source elements relative to a listener head. The default cell has one source centered in front of the listener head. By right-clicking the boarder or title of the cell you can grow the algorithm to have up to 6 sources. The figure to the right is the default algorithm grown by 4 to achieve the typical 5.1 surround speaker setup.



Both source and head can be clicked and dragged to any position on the room grid. By right-clicking the center of the cell, you can select **Set Positions**. The options available for control are:

- Units ( English(US) or Metric)
- Coordinate System ( Cartesian or Polar )
- Width and Length of the simulated room
- Head Location
- Individual Source Locations



Set Positions and Room Dimensions

Units: English (US)

Coordinate System: Cartesian

Width (ft): 5.00

Length (ft): 5.00

Head Location x (ft): 0.03

Head Location y (ft): -0.05

| | x (ft) | y (ft) |
|---|---|---|
| 1 | -0.88 | 0.88 |
| 2 | 0.00 | 1.25 |
| 3 | 0.88 | 0.88 |
| 4 | -0.88 | -0.88 |
| 5 | 0.88 | -0.88 |
| 6 | 0.00 | 0.00 |

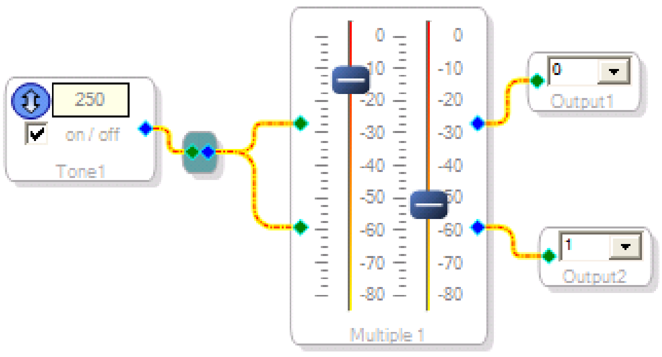
Tips:

- Resize button must be pressed to accept room dimension changes
- Sources locations are grayed out and un-selectable if that source is not currently added to the module.

## Volume Controls Example

### Example 10.1

The following is a simple example using a Multiple Volume Control with one added algorithm, Tone Source, TConnection, and two Outputs. The purpose of this example is to show how you can achieve panning effects (left to right image placement) by level differences between the two outputs. With the current placement of the volume sliders the tone has shifted from the center toward the left side of the sound-scape.



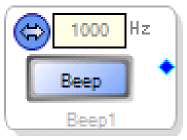
# Sources

The Sources library of the ToolBox gives you access to multiple sound source generation cells. The cells available under this library are:

- Beep Sources
- Sweep(lookup/sine)
- Tone (lookup/sine)
- White Noise Source

The following pages contain more specific information about the individual cells. Also take a look at the Sources Example page to see a list of Example pages containing the Sources cells.

## Beep Sources



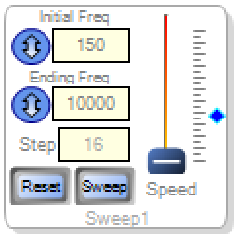
The Beep Sources cell generates test tones using an internal oscillator. You have control over the frequency of the test tone, duration and on/off control. You can edit the frequency value by typing in the text window or using the control arrows. The test tone is only activated when you have the **Beep** button depressed with your mouse arrow.

After the default algorithm has been established, this cell has the ability to have algorithms added to it. (if you are using more than one DSP board, you will need to add the initial default algorithm for the desired board). Right-click on the cell and select: **Add Algorithm** > **IC N** > **Beep variable gain**. This action will add another output pin for connection.

This algorithm also allows you control over the sampling rate of the oscillator. Right-click on the cell and select **Set Sampling Rate**. The Sampling Rate Window will pop-up allowing you to enter a value in the text box to control the sampling rate.

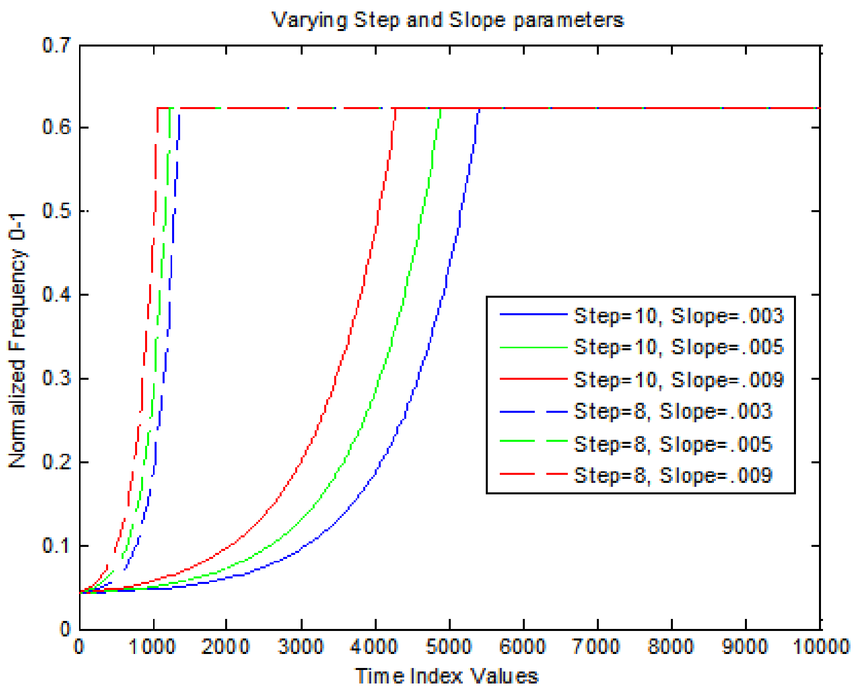
Note: The output has inconsistent gain and for most purposes the Sine Lookup tone generator should be used.  The Beep algorithm does give better THD for testing purposes.  Also, the oscillator becomes unstable over about 7kHz.  The output is never limited and in certain circumstances can cause overloading of the audio circuitry and unusual output results (ex: 12kHz beep combined with input audio in a mixer module).  ADI recommends use of the Sine Lookup in most circumstances.  If the Beep Sources algorithm is required insert a Dynamics Processor used as a compressor to limit the signal.

## Sweep Up

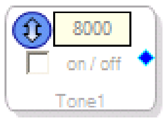


The Sweep Up cell generates a sine-tone sweep from the initial frequency to the ending frequency. The step size and slope are closely related and help determine the speed of the sweep. In order to use this cell, enter the desired starting and ending frequency, and step size in the cell's edit boxes. Depending on the step size you entered, a valid slope range is computed, and controllable via the slope slider bar. Clicking on the **Sweep** button will commence the sweep from the initial to ending frequency. The tone will be held at the ending frequency until you click the **Reset** button to return to the initial frequency.

The following plot shows the correlation for sweep plots with varying step sizes and slopes.



## Tone (lookup/sine)



The Tone (lookup/sine) cell generates a sine tone from a lookup table, with constant gain level at different frequencies. You have control over the frequency of the test tone, and on/off control. You can edit the frequency value by typing in the text window or using the control arrows. The sine tone is activated when the on/off checkbox is marked.

This algorithm also allows you control over the sampling rate of the oscillator. Right-click on the cell and select **Set Sampling Rate**. The Sampling Rate Window will pop-up allowing you to enter a value in the text box to control the sampling rate.

## White Noise Source

The White Noise Source cell generates a white noise signal which contains equal amounts of all audible frequencies in the signal. It is a random time varying signal that can be useful for testing equipment. The cell is controlled by a simple toggle switch for on/off control.

After the default algorithm has been established, this cell has the ability to have algorithms added to it. (if you are using more than one DSP board, you will need to add the initial default algorithm for the desired board). Right-click on the cell and select: **Add Algorithm** > **IC N** > **White Noise**. This action will add another output pin for connection.

This algorithm also allows you control over the sampling rate of the white noise generation. Right-click on the cell and select **Set Sampling Rate**. The Sampling Rate Window will pop-up allowing you to enter a value in the text box to control the sampling rate.

## Sources Example

Please look at the following example pages for other ToolBox libraries that show sample schematics using the Source cells.

- Basic DSP Example
- Level Detectors Example
- Mixers/Splitters Example
- Volume Controls Example

# GPIO conditioning

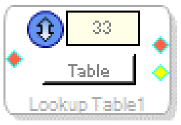
The GPIO conditioning library of the Toolbox give you access to the following cells. The following pages contain the more specific information about the Individual cell.

- Index lookup table
- Push and Hold
- Up/Down increment

Also take a look at the GPIO conditioning example page to see a sample schematic using  GPIO conditioning cells .

## Index look up table

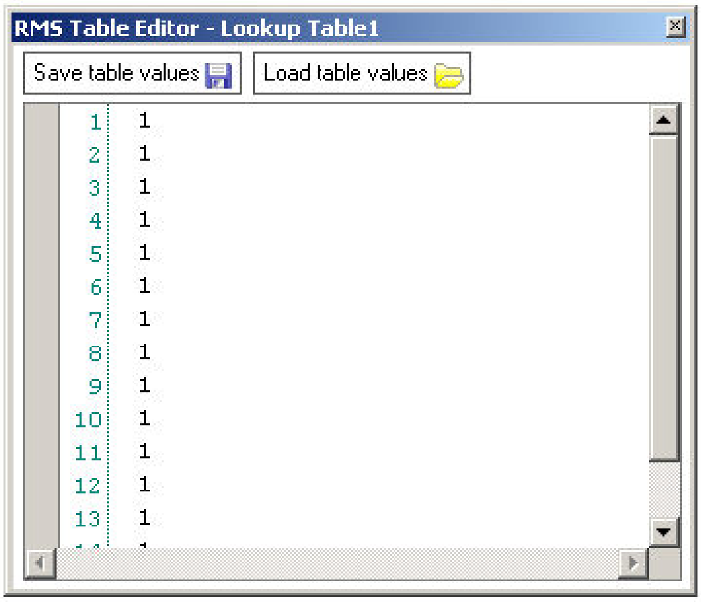
The index look up table cell is used to access the values in the table format. The values in the table are accessed by a table index. To use this cell:



1. Drag and drop into the workspace
2. Right click the cell and select **Add Algorithm** > **IC N**
3. From the list select the algorithm that meets your needs:

- **LUT w/Min and Max**
- **LUT w/Interface out, Min and Max**

4. Click the TABLE button to open the table editor window. The table is shown in the figure below.



```
 1  1
 2  1
 3  1
 4  1
 5  1
 6  1
 7  1
 8  1
 9  1
10  1
11  1
12  1
13  1
```

There are two methods to enter the value in the table.

- One method is the user can enter the value directly in to the table and save the values using **Save table value** option.
- Another method to enter the value is by using the **Load table**

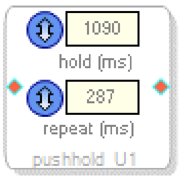
**value** from the file.

5. The maximum entries in the table are controlled by entering the value in the spin text

Note: For the picture included to the right, the ***LUT w/Interface out, Min and Max*** was selected and the green output pin is used for connecting the interface cell. If you don't need to select an interface ,you can select.

***LUT w/Min and Max***

## Push and Hold



This cell can be used when the signal has to be hold after a certain period and repeat the hold once for a particular time interval. To use this cell:

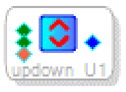
1. Drag and drop into the workspace
2. Right click the cell and select the algorithm for your application
   - **push_hold**
   - **push/hold 2-in 2-out**
   - **push/hold with two-button mute**
3. Set the parameters to fit your application

Note: For the picture included to the right, push_hold was selected. If we want to mute the interface when both the inputs are in phase, then we can use push/hold with two-button mute .The interface is connected to the red pin .

| | |
|---|---|
| **Hold (ms)** | Determines the time at which signal has to be hold. The hold time in milliseconds is entered in edit box. |
| **Repeat (ms)** | Controls the amount of time interval between two hold events. The repeat time in milliseconds is entered in edit box |

## up down increment

The up down increment cell is used as a index to the index look up table. The index is either incremented or decreased based on the two input sample levels and the output of the interface connected to it. In order to use this cell:



1. Drag and drop into the workspace
2. Right click the cell and select **Add Algorithm** > **IC N**
3. From the list select the algorithm that meets your needs:
   - **updown  with INTF input**
   - **updown w/o INTF input**
   - **updown w/o INTF input ,with mute**

I

Note:

For the picture included to the right, the ***updown  with INTF input*** was selected and the red input pin is used for connecting the interface cell.

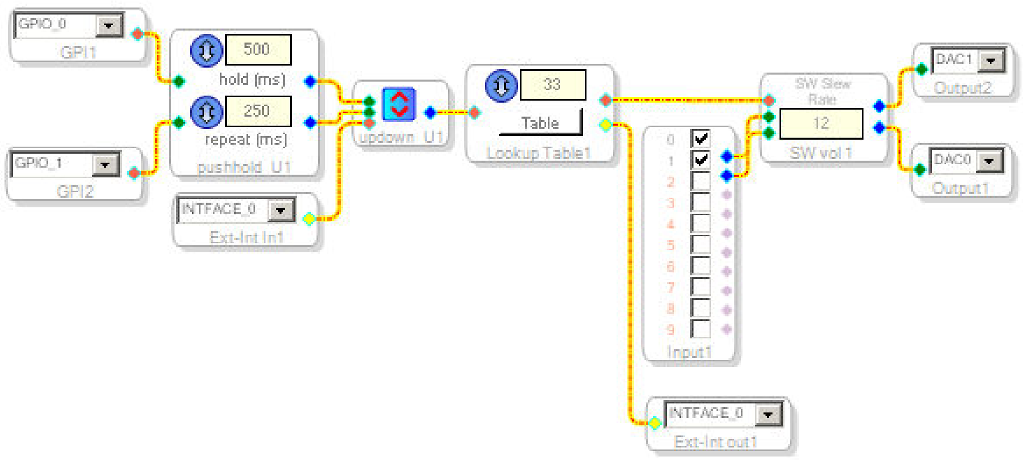
If you don't  need the  interface value in deciding the index,  then you must select ***updown w/o INTF input .***

If you want to select the first index in the index look up table then you must select ***updown w/o INTF input ,with mute.*** The red pin is used to apply  the mute. If  logical '0' is applied to the red pin ,then the index is selected based on the two input sample levels. Other wise the  first index in the index look up table is selected

## GPIO conditioning example

### Example 11.1

This is a simple GPIO I/O schematic diagram. It has two GPIO input cells that drive the push and hold cell . The outputs of these cells are used to select the index from the index look up table .The up/down increment cell is used to either increment or decrement the index based on its input levels .The Ext interface in cell is connected to the red pin of up/down increment cell . The outputs of the look up table are used to drive the Ext interface out and the red pin (the pin used for external representation connection ) of Single slew ext vol. Then the signal is given to two digital to analog convertors which is nothing but the analog output of the board.



Note:

As can be seen from the above schematic, the sigma 100 evaluation board supports only up to two analog inputs and seven digital inputs.

## Example Schematics

System Example

Basic DSP Example

Dynamics Processors Example

Advanced DSP Example

Mixers/Splitters Example

Filters Example

I/O Example

Level Detectors Example

Volume Controls Example

GPIO conditioning Example

# Algorithm Information

The following pages contain technical information on the algorithms driving the cells.

- EQ
- AD 1940/1 Volume Controls
- AD 1953/4 Volume Controls
- General 1st Order Filters
- General 2nd Order Filters
- All Pass Filers
- State Variable Filter
- De-emphasis Filter
- Pink Filter

# EQ Algorithm

The Small and Medium Size EQ cells use multiple biquad filters based on Robert Bristow-Johnson's work here.

Common Variables:

- `A  = 10^(boost/40)`
- `ω0 = 2*pi*f0/Fs`
- `alpha = sin(ω0)/(2*Q)                              (case: Peaking)`
- `      = sin(ω0)/2 * sqrt((A + 1/A)*(1/S - 1) + 2)  (case: 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:

$$H(s) = A * \frac{As^2 + \frac{\sqrt{A}}{Q}s + 1}{s^2 + \frac{\sqrt{A}}{Q}s + A}$$

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
```

For all 3 filters all 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 in software before the parameters are written to memory.

## AD1940/AD1941 Volume Controls

The AD1940/AD1941 includes support for volume controls that use a target/slew RAM memory space to auto-ramp from one value to a desired final value. The ramping type and speed can be controlled by the user in the case of the Adjustable Volume Control. The default case for the other cells is Constant dB slew with a time constant of 8.

**Time Constant**

The Time Constant is a 4 Bit value and ranging from 0 (fastest) to 15 (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 Math**

Step = $2^{13} / 10^{2 * (\text{tconst} - 5)/20}$

The result of the equation is normalized to 5.23 data format. This gives a time constant range from 6.75ms to 213.4ms (-60dB relative to 0dB full scale).

**Constant dB and RC-type (Exponential) Update Math**

Exponential math is accomplished by shifts and adds with a range from 6.1ms to 1.27s (-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 Update Math**

Constant time math is accomplished by adding a step value that is calculated after each new target is loaded. The equation for this step size is:

Step = (Target Data - Slew Data) / Number of Steps

Number of Steps = $2^{\text{tconst} + 6}$

Number of Steps ranges from 64 (tconst = 0) to 8196 (tconst = 7)

## AD1953 Volume Controls

The AD1953/AD1954 includes eight separate SPI registers available to control the volume. These registers are special in that they include automatic digital ramp circuitry for click-less volume adjustment. Gains from +2.0 to -2.0 are possible. The default value is 1.0. It takes 1024 audio frames to adjust the volume from 2.0 down to 0; in the normal case where the max volume is set to 1.0, it will take 512 audio frames for this ramp to reach zero.

It is important to note that the Mute command is the same as setting the volume to zero, except that when the part is unmuted, the volume returns to its original value. These volume ramp times assume that the AD1953 is set for the fast volume ramp speed. If the slow setting is selected, it will take 9192 audio frames to reach zero from a setting of 2.0. Correspondingly, it will take 4096 frames to reach 0 volume from the normal setting of 1.0.

AD1953 - 8 volume controls are available

AD1954 - 3 volume controls are available

Note: The Single and Multiple Volume Controls are set permanently to a fast volume ramp speed.

## General 1st Order Filters

$$H(z) = \frac{b_0 + b_1 z^{-1}}{1 + a_1 z^{-1}}$$

The General 1st Order cell has two options for 1st order filter algorithms.

Common Variables:

- `ω0 = 2*pi*f0/Fs`
- `gainLinear = 10^(gain/20)`

**Low Pass:**

Transfer Function:

$$H(s) = \frac{1}{s^2 + \frac{s}{Q} + 1}$$

Coefficients:

```
a1 =  2.7^-ω0
b0 =  gainLinear * (1.0 - a1)
b1 =  0
```

**High Pass:**

Transfer Function:

$$H(s) = \frac{1}{s^2 + \frac{s}{Q} + 1}$$

Coefficients:

```
a1 =  2.7^-ω0
b0 =  gainLinear * a1
b1 = -a1 * gainLinear
```

Note: In the actual implementation on the DSP, when the coefficients are stored in parameter RAM, **a1** needs to be inverted. This is done automatically in software before the parameters are written to memory.

# General 2nd Order Filters

$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}$$

The General Purpose and Text Entry cells have multiple options for biquad filters based on Robert Bristow-Johnson's work here.

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(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
```

**Low Pass:**

Transfer Function:

$$H(s) = \frac{1}{s^2 + \frac{s}{Q} + 1}$$

Coefficients:

```
alpha = sin(ω0)/(2*Q)

a0 =   1 + alpha

a1 =  -2*cos(ω0)

a2 =   1 - alpha

b0 =  (1 - cos(ω0)) *
gainLinear / 2

b1 =   1 - cos(ω0)  *
gainLinear

b2 =  (1 - cos(ω0)) *
gainLinear / 2
```

**High Pass:**

Transfer Function:

$$H(s) = \frac{1}{s^2 + \frac{s}{Q} + 1}$$

Coefficients:

```
alpha = sin(ω0)/(2*Q)

a0 =   1 + alpha

a1 =  -2*cos(ω0)

a2 =   1 - alpha

b0 =  (1 + cos(ω0)) *
gainLinear / 2

b1 = -(1 + cos(ω0)) *
gainLinear

b2 =  (1 + cos(ω0)) *
gainLinear / 2
```

**Band Pass:**

Transfer Function:

$$H(s) = \frac{s}{s^2 + \frac{s}{Q} + 1}$$

Coefficient:

```
alpha = sin(ω0) * sinh( ln(2)/2 *
        bandwidth * ω0/sin(ω0)

        a0 =   1 + alpha

        a1 =  -2*cos(ω0)

        a2 =   1 - alpha

        b0 =   alpha * gainLinear

        b1 =   0

        b2 =  -alpha * gainLinear
```

**Band Stop:**

Transfer Function:

$$H(s) = \frac{s^2 + 1}{s^2 + \frac{s}{Q} + 1}$$

Coefficients:

```
alpha = sin(ω0) * sinh( ln(2)/2 *
        bandwidth * ω0/sin(ω0)

        a0 =   1 + alpha

        a1 =  -2*cos(ω0)

        a2 =   1 - alpha

        b0 =   1 * gainLinear

        b1 =  -2*cos(ω0) * gainLinear

        b2 =   1 * gainLinear
```

**Butterworth Low Pass:**

Transfer Function:

$$H(s) = \frac{1}{s^2 + \sqrt{2}s + 1}$$

Coefficients:

```
alpha = sin(ω0) / 2.0 *
1/sqrt(2)


a0 =   1 + alpha

a1 =  -2*cos(ω0)

a2 =   1 - alpha

b0 =  (1 - cos(ω0)) *
gainLinear / 2

b1 =   1 - cos(w0)  *
gainLinear

b2 =  (1 - cos(ω0)) *
gainLinear / 2
```

**Butterworth High Pass:**

Transfer Function:

$$H(s) = \frac{s^2}{s^2 + \sqrt{2}s + 1}$$

Coefficients:

```
alpha = sin(ω0) / 2.0 *
1/sqrt(2)


a0 =   1 + alpha

a1 =  -2*cos(ω0)

a2 =   1 - alpha

b0 = -(1 + cos(ω0)) *
gainLinear / 2

b1 = -(1 + cos(ω0)) *
gainLinear

b2 = -(1 + cos(ω0)) *
gainLinear / 2
```

**Bessel Low Pass:**

Transfer Function:

$$H(s) = \frac{3}{s^2 + 3s + 3}$$

Coefficients:

```
alpha = sin(ω0) / 2.0 * 1/sqrt(3)


a0 =   1 + alpha

a1 =  -2*cos(ω0)

a2 =   1 - alpha

b0 =  (1 - cos(ω0)) * gainLinear
/ 2

b1 =   1 - cos(w0)  * gainLinear

b2 =  (1 - cos(ω0)) * gainLinear
/ 2
```

**Bessel High Pass:**

Transfer Function:

$$H(s) = \frac{s^2}{s^2 + 3s + 3}$$

Coefficients:

```
alpha = sin(ω0) / 2.0 *
1/sqrt(3)


a0 =   1 + alpha

a1 =  -2*cos(ω0)

a2 =   1 - alpha

b0 = -(1 + cos(ω0)) * gainLinear
/ 2

b1 = -(1 + cos(ω0)) * gainLinear

b2 = -(1 + cos(ω0)) * gainLinear
/ 2
```

Note: For all 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.

# All Pass Filters

All Pass 1st Order and 2nd Order algorithm coefficient calculations.

Common Variables

- `ω0 = 2*pi*f0/Fs`
- `alpha = sin(ω0)/(2 * Q)`
- `gainLinear = 10^(gain/20)`

**All Pass 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^-ω0` <br> `b0 = -gainLinear * a1` <br> `b1 =  gainLinear` |

**All Pass Second Order:**

| | |
|---|---|
| Transfer Function: (laplace) | $H(s) = \frac{s^2 - \frac{s}{Q} + 1}{s^2 + \frac{s}{Q} + 1}$ |

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(ω0)  / a0

a2 = ( 1 - alpha) / a0

b0 = ( 1 - alpha  ) / a0 * gainLinear

b1 = (-2 * cos(ω0)) / a0 * gainLinear

b2 = ( 1 + alpha  ) / a0 * gainLinear
```

# State Variable Filters

The state variable filter has the advantage of offering lowpass, highpass, bandpass, and bandreject all within one block. This algorithm offers the advantage over the biquad section filters because only one coefficient is needed rather than the five coefficients for the biquad sections. For the algorithm including Q, there is an additional coefficient for the control of Q.

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:**

$$H(z)_{HP} = \frac{(1-z^{-1})}{f}$$

## De-emphasis Filter

The De-emphasis Filter is used to attenuate the high frequency components that were boosted during Pre-emphasis. Pre-emphasis is a high-frequency boost used during recording. When the De-emphasis Filter is used in conjunction with audio that has been recorded with a Pre-emphasis filter there is improved signal-to-noise ratio.

Common Variables

- `sampling frequency:      Fs`
- `de-emphasis factor:      a = exp(-2 * pi * f / Fs)`

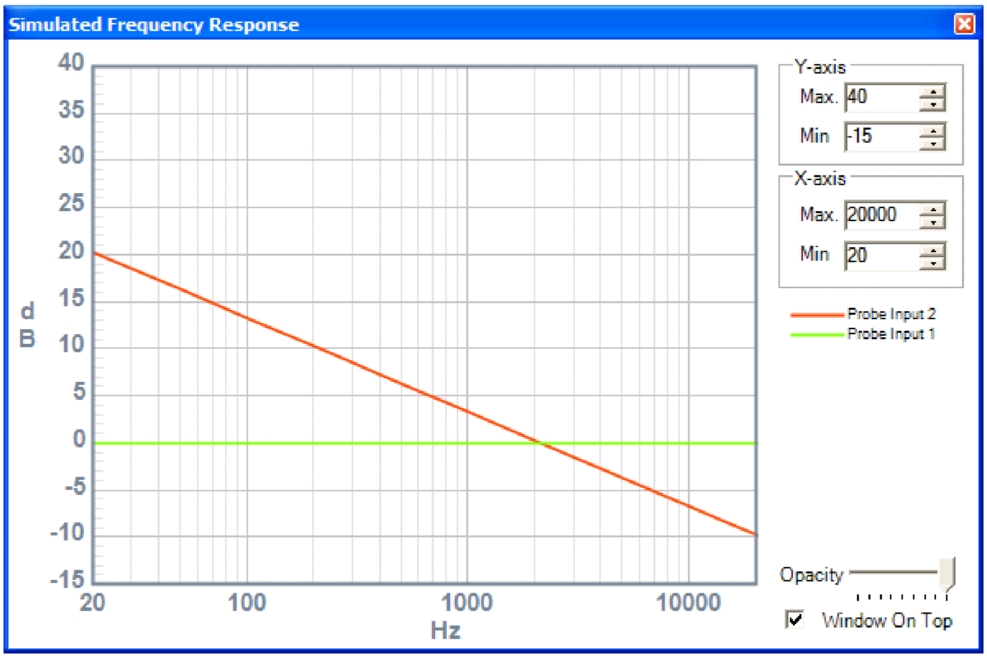
The algorithm is then computed recursively by:

$y_1 = x_1$

$y_i = x_i + a * y_{i-1}$

# Pink Filter

The Pink Filter takes an input source and outputs a signal with a 3dB drop per octave. The classical use of this filter is to convert White Noise (flat frequency response) to Pink Noise (-3dB per octave). White noise contains equal energy across the frequency spectrum whereas Pink Noise contains equal energy per octave. The graph below (generated using simulation stimulus and probe) show the two curves of White (green line) and Pink Noise (red line).



Y-axis
Max. 40
Min -15

X-axis
Max. 20000
Min 20

☑ Window On Top

# Level Detector Algorithms

The Level Detector cells in SigmaStudio are driven by an algorithm to detect the RMS level of the signal, and represents the level 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 output of the level detector is a 5.19 number. Depending on what algorithm you have selected for the level detector, there is one of two formulas to determine the dB value of the output, from the 5.19 value. The default formula for all the level detectors to determine the dB output value is the following:

dB_value = 96.32959861 * (readback_value / $2^{19}$ - 1)

The minimal reading algorithm uses the following formula to determine the dB output value:

```
dB_value = 96.32959861 * (readback_value / 256 - 1)
```

Note: The Level Detector Designer cell is the only level detector cell that requires you add one of the algorithms described below.

### Pass thru

The Pass thru algorithm receives a 5.19 (5 bytes used to represent the integer portion of the value, 19 bytes for the decimal part) number during readback and then converts this value to a decimal value which is represented in decibels. This is the value that is visually seen on the display bar. This is a Pass thru system because the output of this block is the same signal that is being sent into the cell.

### RTA to output

The RTA to output algorithm outputs the RMS level value through the red output pin. This pin is designated in red because there is not actual audio being sent out, just the level values for each frequency band. You will notice that when you grow this algorithm, each pin corresponds to a frequency band.

### Pass thru/ minimal reading, Single and Double Precision

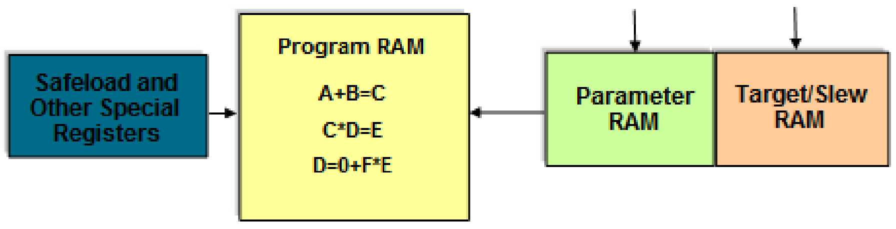
The Pass thru/ minimal reading functions in a similar manner to Pass thru, but saves on communication bandwidth, at the expense of losing precision. Pass thru/minimal reading only uses 1 byte for the readback values and then converts this value to a decimal value which is represented in decibels.

Double precision uses 56 bits for each calculation, takes 10 instructions per filter, and should normally be used. Single precision uses 28 bits for calculations, takes 6 instructions per filter, and saves 3 data ram spaces over the double precision algorithm. Single precision should not be used for frequencies below 1/10 the sampling frequency, or for high Q filters.

# Basic SigmaDSP Architecture

The following describes the Basic Sigma DSP Architecture:

- Program RAM - stores functionality and execution information. The program RAM dictates the signal flow and chain of operations (+,-,x,/, etc) which are stored for execution.
- Parameter RAM and Data Memory - The Parameter RAM stores parameter information (either from the user or as a result of an internal core operation). The Data Memory stores serial information (usually audio samples or the result of internal core operations)
- Safeload registers - registers that help to load parameters smoothly
- Target/Slew RAM - registers that help update parameters smoothly
- Additional Registers- data capture registers, DSP core control registers, Serial Input & Output control registers

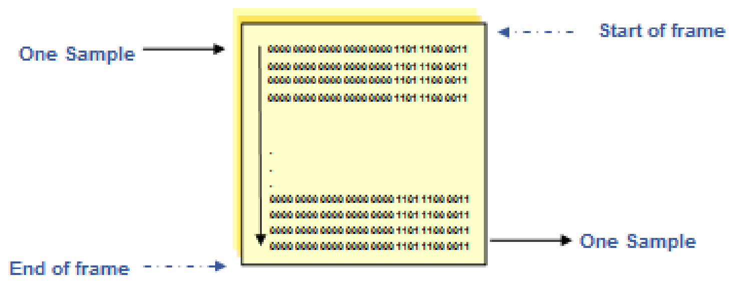


Note: This section of the manual is a general summary of AD1940 Features and Microcontroller considerations. For more details concerning each topic see the ADI website and navigate to your part's datasheet:

http://analog.com/sigmadsp

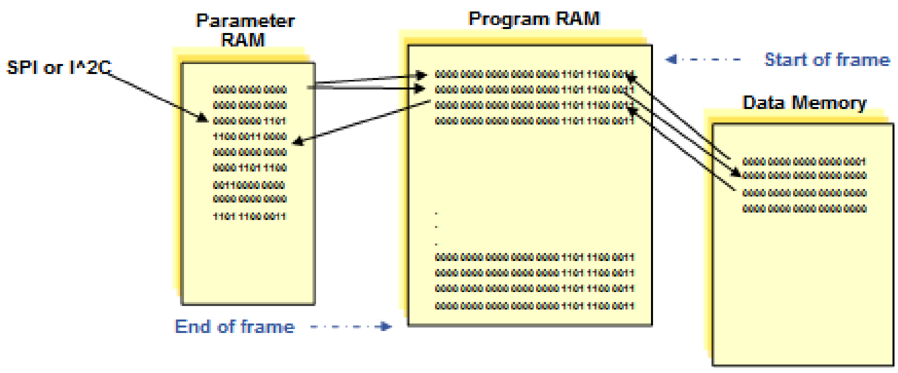
## Program RAM Execution

It is important to understand how the Program RAM is executed. Generally, the SigmaDSP core works like a fixed program executing each one of its instructions (up to 1536 instructions for the 1940) every new audio sample. This  execution period is also known as a FRAME.



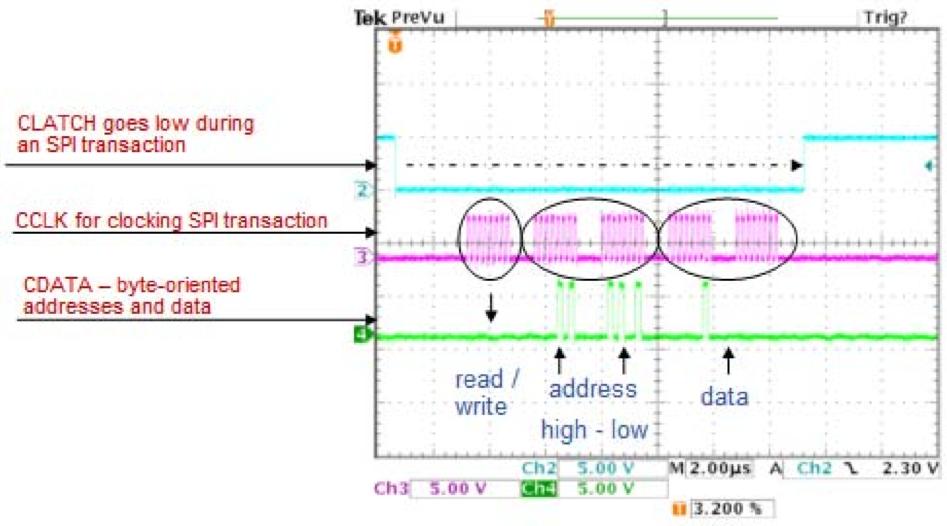
## Parameter Ram and Data Memory

- The Program RAM takes the information contained in the Parameter RAM and the Data Memory to process it and produce results.
- The Parameter Memory is updated through the SPI or I$^2$C port
- The Data Memory is usually updated every frame



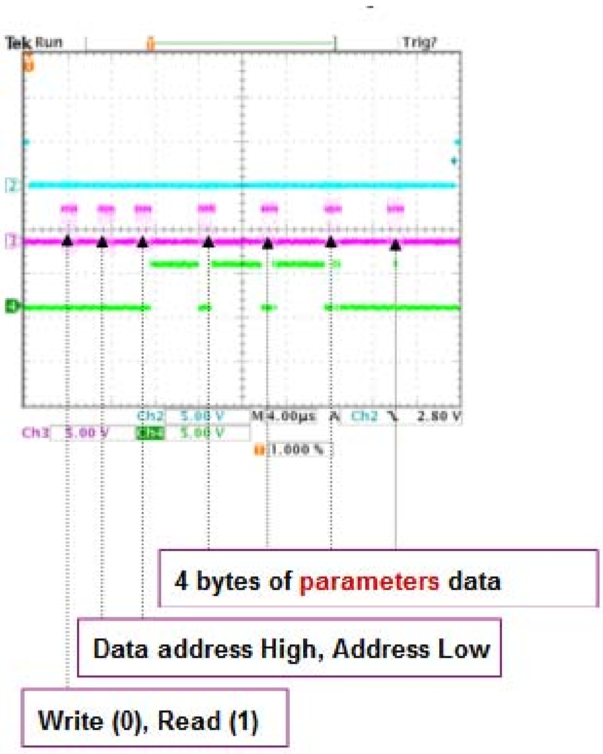
Probably one of the most time consuming parts when developing an application is to get the communication set properly. The following picture should be used as a starting point and as an example for getting basic SPI communication. The picture illustrates a simple mute operation on the AD1940.

Notice that the word shown consists of: 1 byte for writing, 2 bytes of the address and 2 bytes for the data. The operation shown will mute the DSP core.

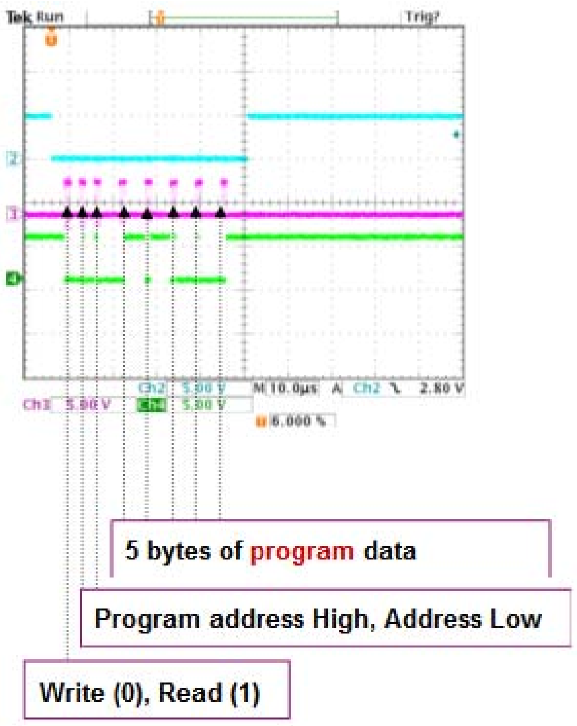


In order to write directly to each one of them:

**Parameter Data:**

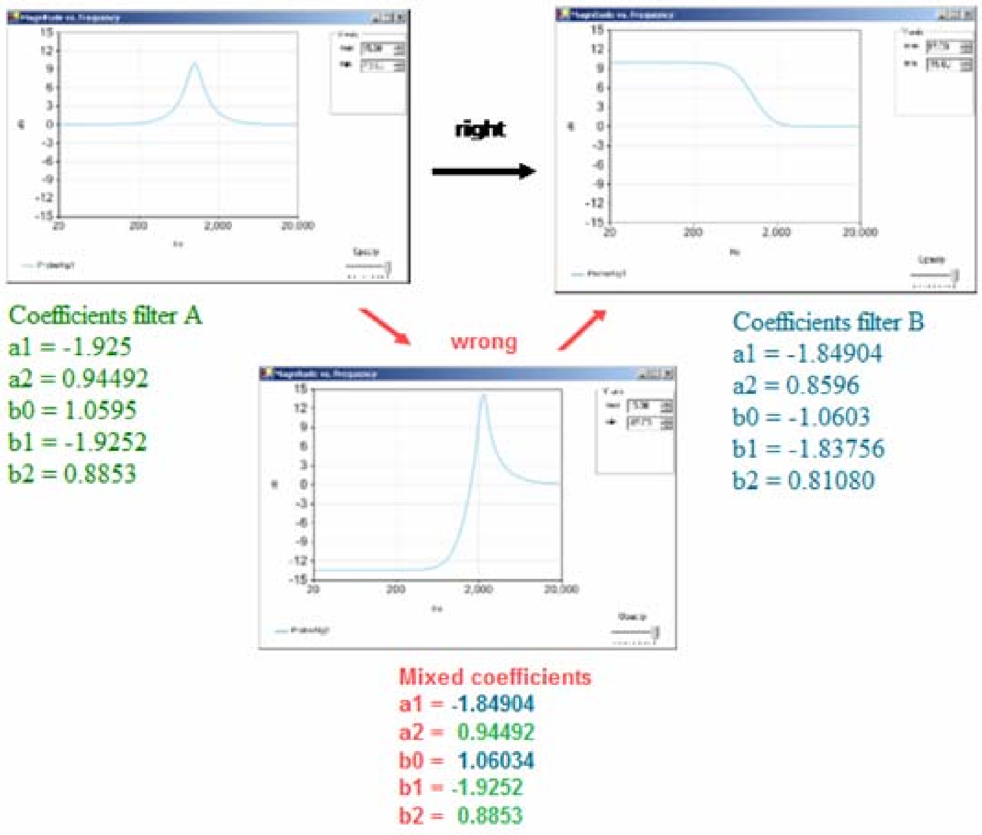


**Program Data:**


Tek Run
Trig?
Ch3 5.00 V
Ch2 5.00 V
Ch4 5.00 V
M 10.0µs A Ch2 2.80 V
6.000 %
5 bytes of program data
Program address High, Address Low
Write (0), Read (1)

## Safeload Registers

Safeload Registers are very important in implementing filter coefficients because of the ability to update coefficients in a safe manner. One of the most important elements in signal processing is a filter. Many filters are implemented in the DSP using coefficients. When coefficients are changed the transition has to be performed simultaneously on all coefficients. If the transition is not made at the same time, an undesirable response (such as audible pops or clicks) will be produced in some cases. The filter may become unstable or undesirable and the output signal becomes harmful to the system because of its unpredictability. Below is a picture showing filter coefficients being updated in the "right" manner, and in a "wrong" manner, with mixed coefficients producing an undesirable frequency response.



Coefficients filter A
a1 = -1.925
a2 = 0.94492
b0 = 1.0595
b1 = -1.9252
b2 = 0.8853

Coefficients filter B
a1 = -1.84904
a2 = 0.8596
b0 = -1.0603
b1 = -1.83756
b2 = 0.81080

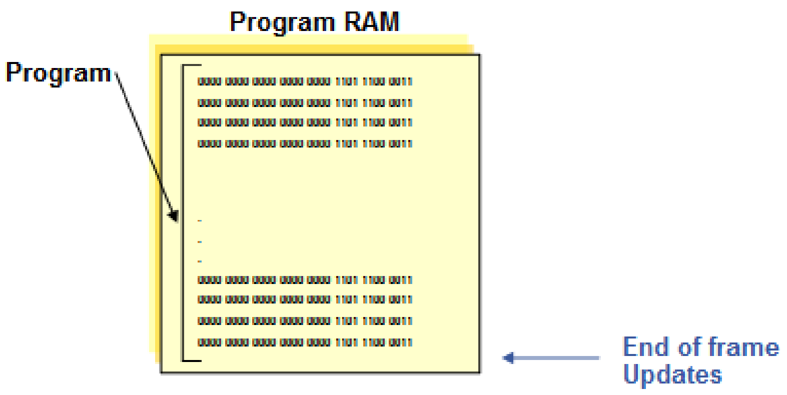
Mixed coefficients
a1 = -1.84904
a2 = 0.94492
b0 = 1.06034
b1 = -1.9252
b2 = 0.8853

The solutions we have available to solve this update problem are the following:

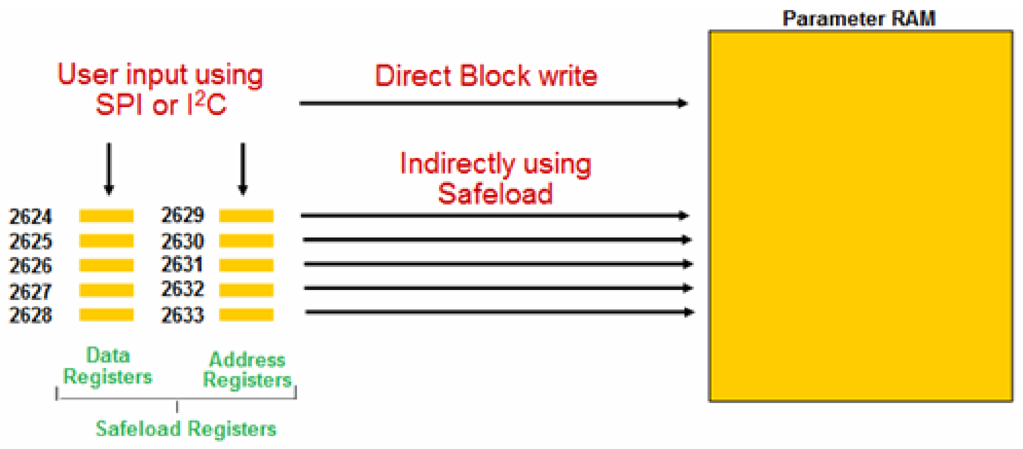
- We can mute the DSP, write directly into the filter's coefficient space in parameter memory and then unmute the DSP to prevent unstable results. But this causes problems:

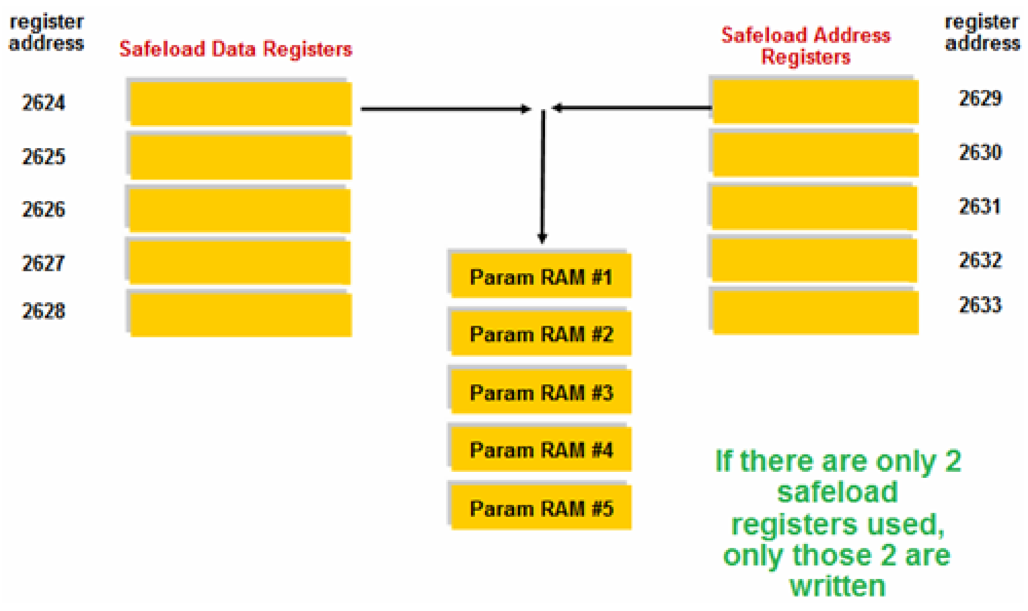
  - Microcontroller intensive – more control port writes to the SigmaDSP.
  - Causes audio glitches every time we change the filter.
  - Audio output will not be continuous

- Use the Safeload Registers which are specifically design 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. Note: The addresses used in the figure below are specific to the AD1940.



The following figures help describe what and where the Safeload Registers are:



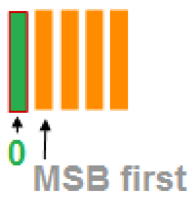


It is important to note that even though the length for parameter RAM is only 4 bytes, we need to reserve space for 5 bytes of data in the Safeload registers. The reason for this is that there are other kinds of registers called Target/Slew RAM registers that require 5 bytes of data. Thus you need to keep in mind the format of your parameter data within the 5-byte space in the safeload registers:

How to format the data:

byte 0 = 0x00

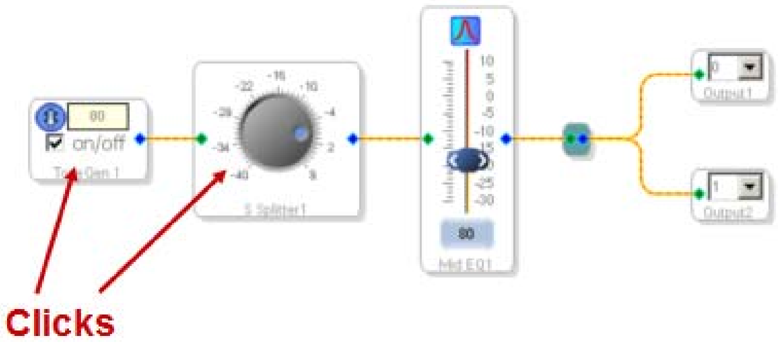
byte 1-4 = parameter data



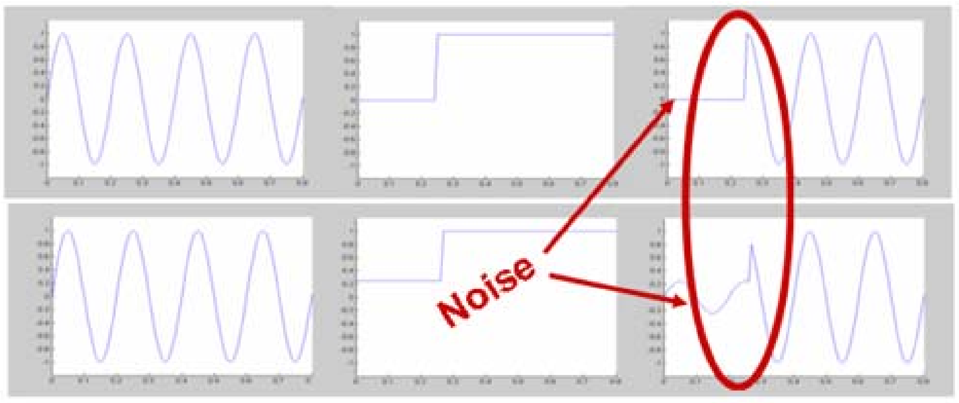
## Target Slew RAM

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. These noises are mostly unwanted. See the sample schematic below showing possible real-time changes that can produce clicks/pops:



The problem with this noise arises from an audio signal getting scale by a step function, which causes an unwanted response. The step function contains an impulse (Dirac Delta function d(k)). It can be proven that the impulse d(k) is a frequency limited white noise. From an audio perspective, this noise is unwanted in most applications. See figure below for a graphic representation of this signal noise caused by the step function:

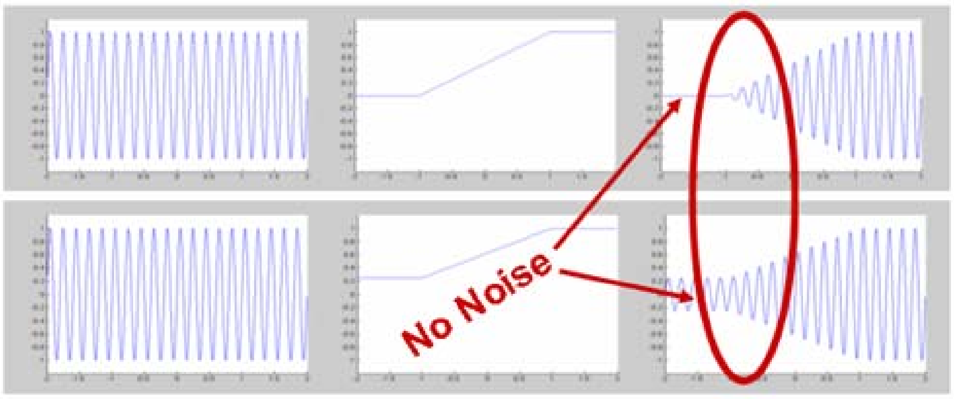


There are a couple of possible solutions to consider for this noise problem:

- 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 the target/slew RAM to automatically do the required ramping with one simple command without the need to mute the DSP. See the result of using the target/slew RAM below:

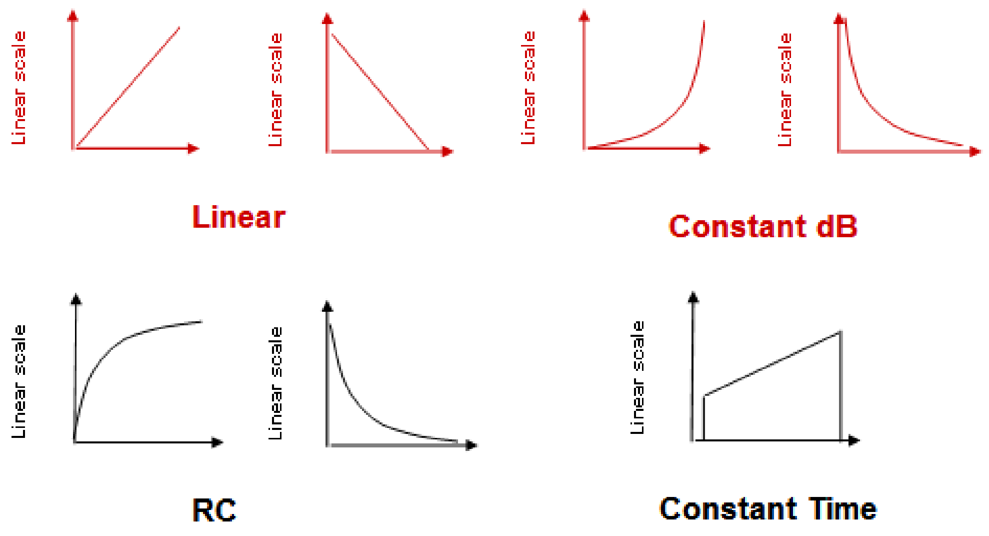


SigmaStudio's volume control and mux blocks both use the slew RAM to cleanly ramp volume from one level to another. Thus for the all the volume control cells, you have the option of using target/slew RAM as your algorithm to prevent the noise inherent in switching between subsequent levels.

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. When measured in dB it is constant up and down.
- RC type – Uses the difference between target and current values to calculate step size.
- Constant type – Values change in a fixed amount of steps linearly

For more information see the AD1940/1941 Volume Control Algorithms page.

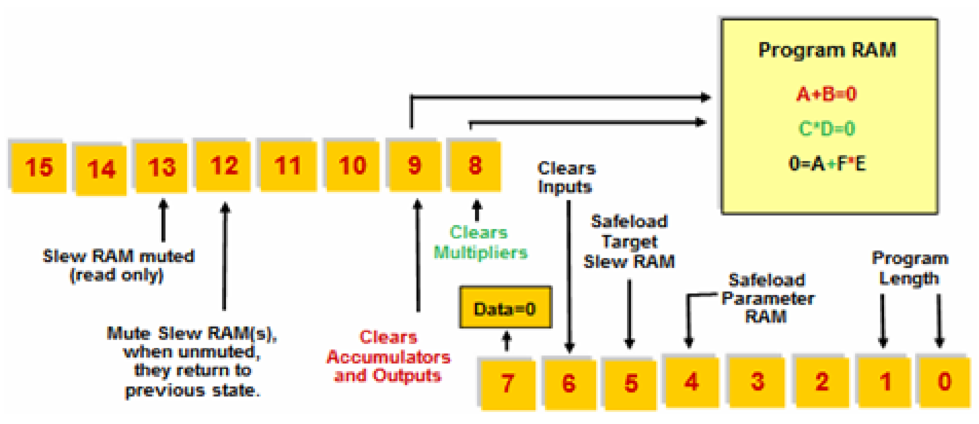


Although the Target/Slew RAM is beneficial for eliminating unwanted noise, there are some drawbacks to this feature:

- 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 this will take possibly scarce program RAM (MIPS) and wouldn't be so optimized.
- Limited number of them; the 1940 currently has 64 target/slew RAM locations that should only be used when a parameter or set of parameters are going to get modified at runtime in the final application.

# Core Control Register

The Core Control register is a 2-byte register that affects the DSP core. This register is necessary for properly running the SigmaDSP by allowing us to easily set important core functions.



Note: For more details concerning the Core Control Register, see the ADI website and navigate to your part's datasheet:

http://analog.com/sigmadsp

# Tutorials

In this section, you will find two simple examples to help you get started using SigmaStudio. The first tutorial shows you 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 in order to view output response of selected algorithms.

# Stereo Audio with Volume Control and 5-band EQ

In order to get started with this tutorial, make sure that you have followed all the steps in the Getting Started portion of this help manual. This ensures that you have all proper configurations set up for your board and that your hardware and communication have been established for your application. Note: This example is specific for the AD1940 since the volume control is using slew RAM, and there are 16 inputs.

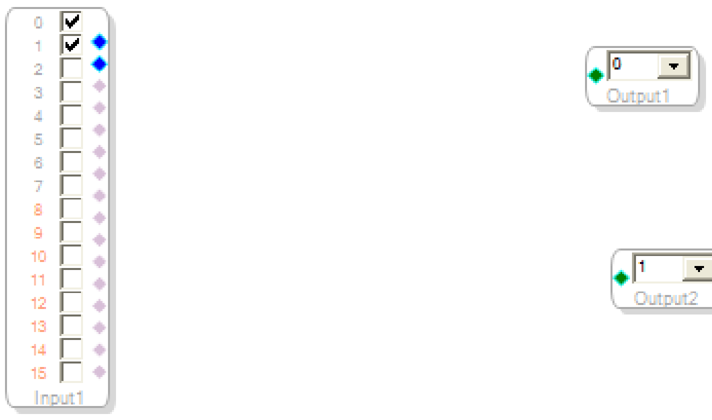
After setting up the Hardware Configuration you should now be in the Schematic workspace of SigmaStudio.

The purpose of this tutorial is to show you how to set up input/output connections, add a volume control, add 5-band EQ,  and wire your system and compile.

Setting up I/O

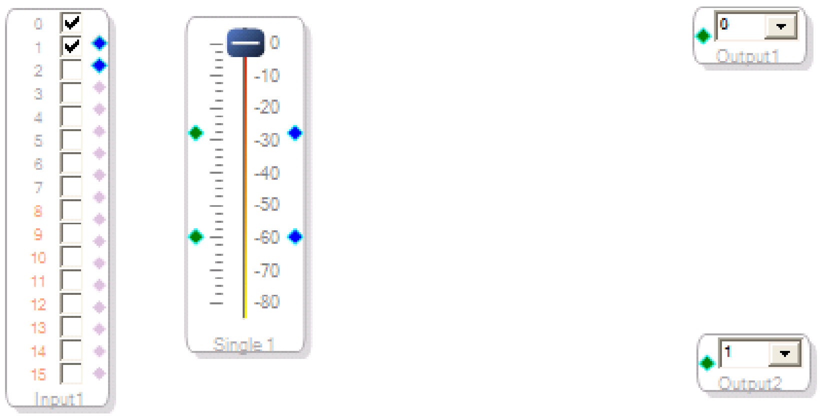
1. Click on the **IO** tab of the **ToolBox**
2. Drag and drop the **Input** cell into the workspace
3. By default two channels are already activated for you by the check marks in boxes for channels 0 and 1
4. Click and drag on an **Output** cell
5. Since this tutorial example is in stereo, repeat step 4 to have two output channels.

Note: Currently you are not ready to hear any output because nothing has been connected or compiled. You have just set the layout for input/output and your workspace should look something like the figure below.



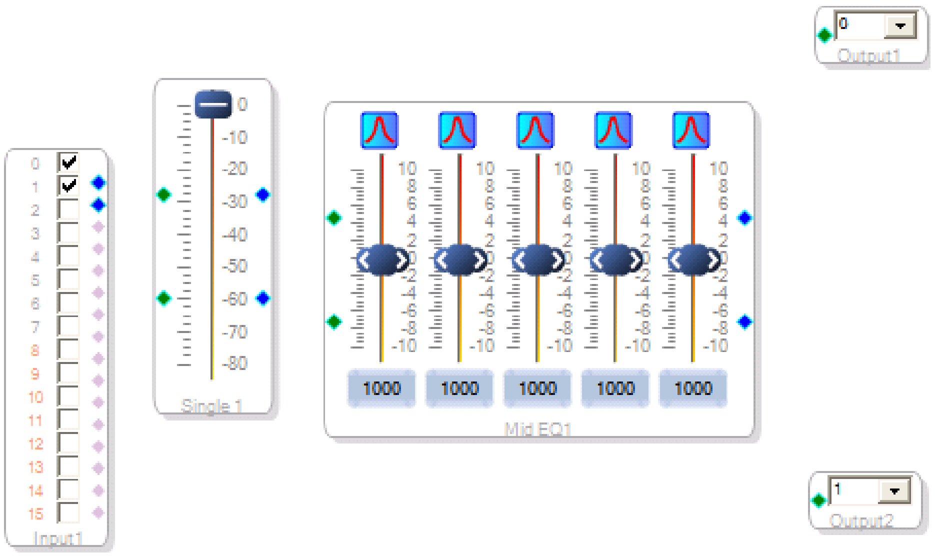
Adding Volume Control

1. Click on the **Volume Controls** tab of the **ToolBox**
2. Drag and Drop the **Single Volume Control** onto your workspace
3. Currently all you see is a simple block that says **Single 1**. Right click on this cell and select **Add Algorithm** > **IC 1** > **Gain (slew)** You should now see the slider cell on your workspace. For more information on volume controllers and the algorithms see Volume Controls page.
4. In order to make this application stereo, right click the slider again and repeat step 3. You now will have two sets connection nodes on your slider as in the figure below. (Again you are not ready to hear output currently).



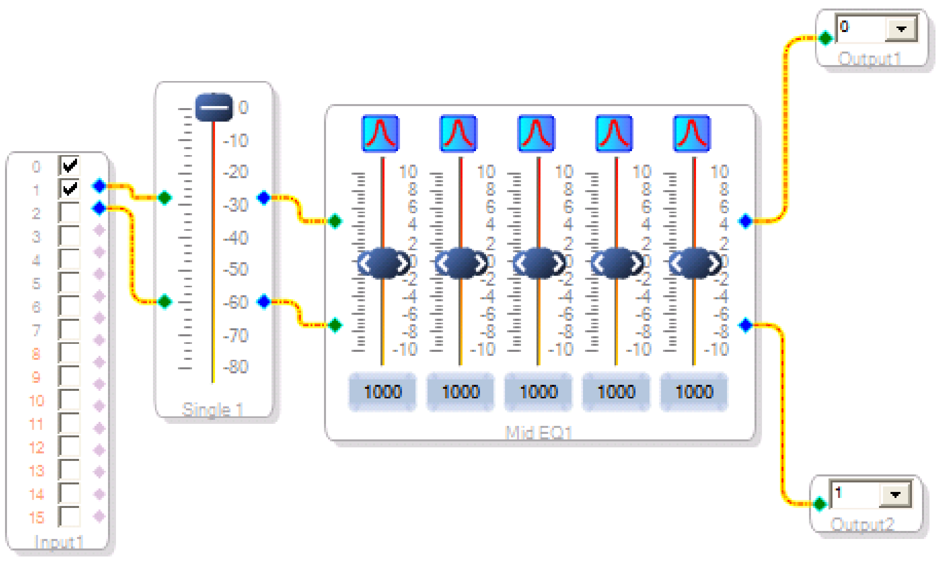
## Adding 5-band EQ

1. Click on the **Filters** tab of the ToolBox
2. Drag and drop the Medium Size Eq onto your workspace.
3. Currently all you see is a simple block that says **Mid EQ1**. Right click on this cell and select **Add Algorithm** > **IC 1** > **Double - Double Precision**
4. By selecting **Double** we are already in stereo but currently we only have one band. To increase your bands right click the cell and select **Grow Algorithm** > **1. Double - Double Precision** > **4**. Your screen should look something like the figure below. For more information on filters and the algorithm for this EQ cell see Filters page.



## Wire System and Compile

1. At this point we have all the building blocks for the tutorial. Using the left-mouse button start from left to right and connect the nodes from blue to green (output of one cell to the input of the next cell).
2. Compile your project by pressing the **Link Compile Download** button on the toolbar, or selecting the **Action** menu and choosing **Link Compile Download**



Note: Once you have successfully compiled your project the blue color bar indicator at the bottom of the screen will turn bright green and the description will read 100% Ready- Downloaded. At this point you are no longer in design mode, and you have real-time control of the volume sliders and EQ controls and can hear the changes you make on the screen. To learn more about basic control of the slider configuration the sliders see the Volume Control and Filters pages.

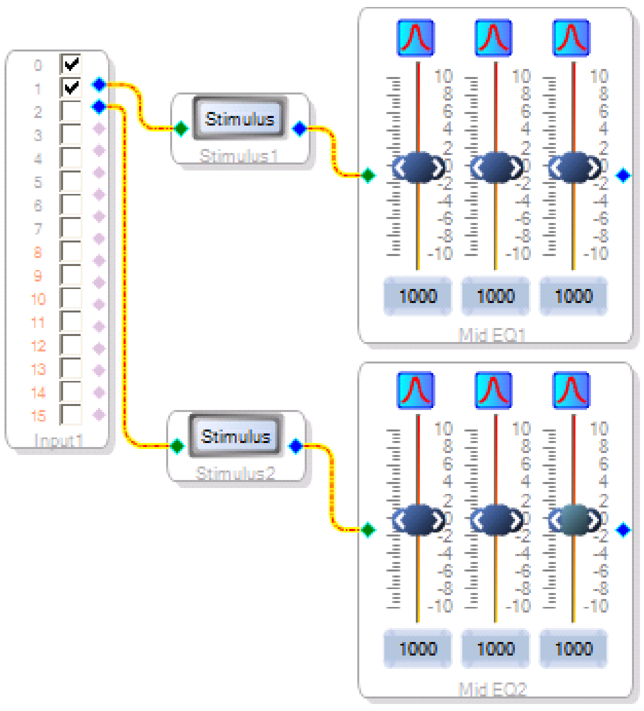
# Probe and Stimulus Blocks

In order to get started with this tutorial, ensure that you have followed all the steps in the Getting Started portion of this help manual. This ensures that you have all proper configurations set up for your board and that your hardware and communication have been established for your application.

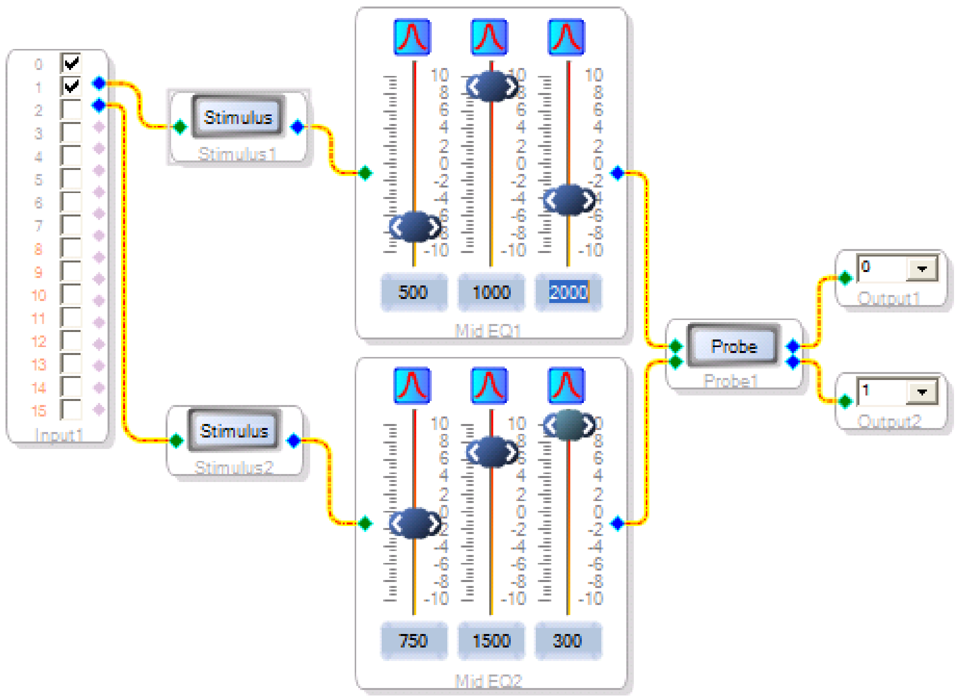
After setting up the Hardware Configuration you should now be in the Schematic workspace of SigmaStudio.

Also you should complete the first tutorial in order to understand the input/output and EQ cells that are used in this tutorial. The purpose of this tutorial is to show you how to make use of probe and stimulus blocks in order to monitor frequency response of the filter controls you set.

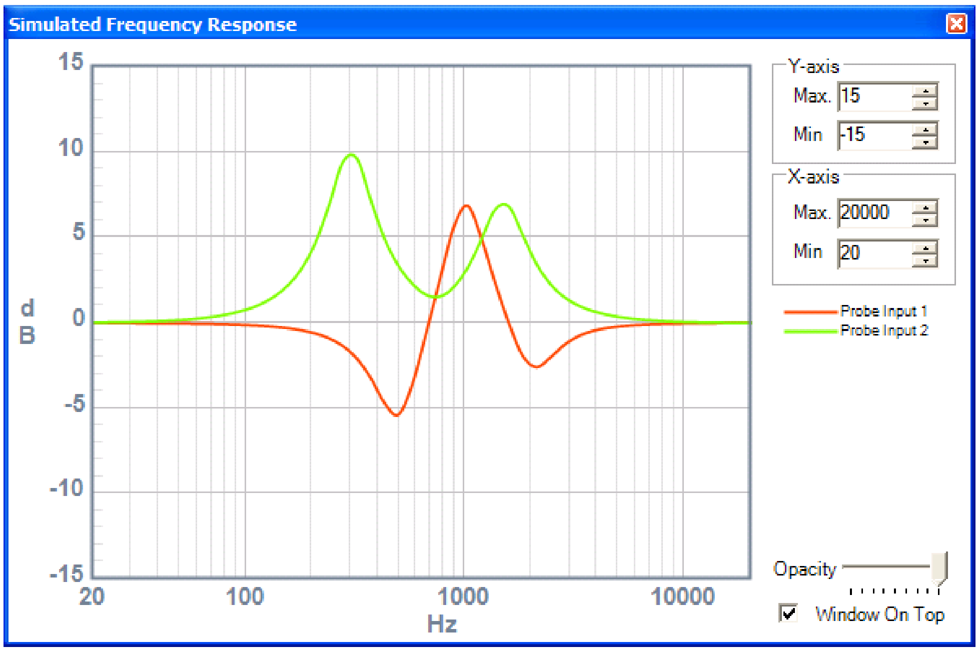
1. Insert an **Input** block from the **IO** tab of the ToolBox
2. Click on the **System** tab of the toolbox
3. Drag and drop two **StimuliCell** blocks to the workspace
4. Connect wires from the input channels to the StimuliCell block.
5. Insert two **Medium Size Eq** cells from the **Filters** tab of the ToolBox
6. For each EQ cell, right click on the cell and select **Add Algorithm** > **IC 1** > **Single - Double Precision**
7. For each EQ cell right click again and select **Grow Algorithm** > **1. Single - Double Precision** > **2**
8. Connect wires from the output of the StimuliCell block to the input of the EQ cell. Your workspace should look something like the following figure:



9. Click on the **System** tab of the ToolBox
10. Drag and Drop one **ProbeCell** block onto your workspace
11. Right click the cell and select **Add Pins**
12. Connect wires from the output of the EQ to the input of the ProbeCell
13. Click on the **IO** tab of the ToolBox
14. Drag and Drop two **Output** blocks into your workspace
15. Connect wires  from the output of the probes to the input of the Output block. Your workspace should look like the following figure:



16. Compile your project by pressing the **Link Compile Download** button on the toolbar or selecting the **Action** menu and choosing **Link Compile Download**
17. Click on the **Probe** cell to bring up the Simulated Frequency Response window. There will be nothing displayed here
18. Click on the **Stimulus** cell buttons to bring up the Frequency Response graph on the curve. You can now view in real-time any changes you make to the EQ parameters. The Frequency Response for the above parameter configuration is given by this graph:



Simulated Frequency Response

Y-axis
Max. 15
Min -15

X-axis
Max. 20000
Min 20