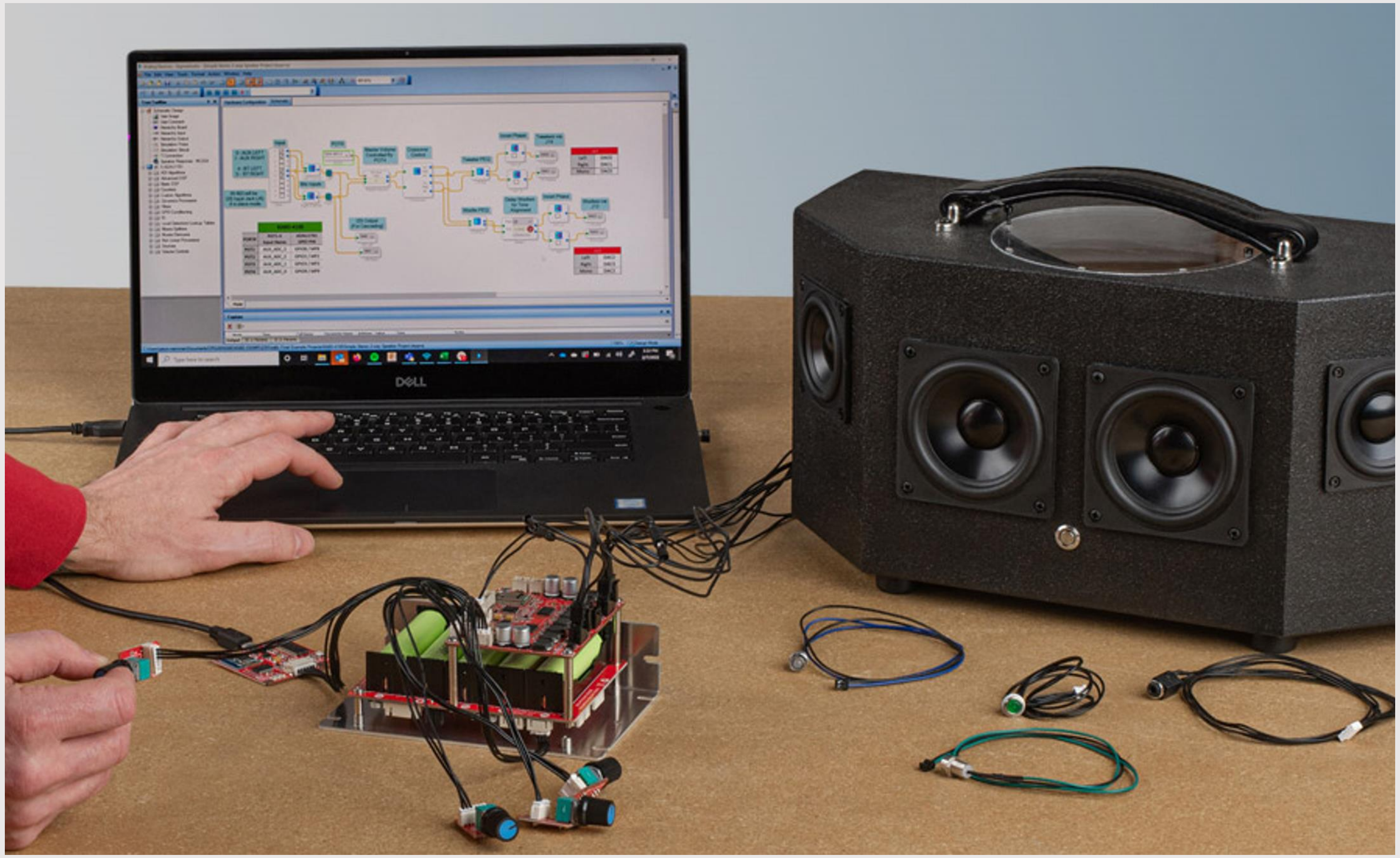
SigmaStudio Programming Guide
for the Dayton Audio KABD Series of Amplifiers



Overview

- This guide will not only teach you the specifics about using SigmaStudio to create projects for a Dayton Audio KABD Amplifier, but it will also teach you DSP concepts to maximize the potential of your next audio project.

- There is variety in the KABD amplifier product line, so this SigmaStudio programming guide is meant to be general purpose for any of the amplifiers in the line, as most of it will be common between them. For more information about your specific amplifier, find its individual user manual or quick start wiring guide on its Dayton Audio product page.

- Analog Device's not only has lots of their own documentation, but the ADAU1701 processor used in the KABD series is also common, which means that many questions you have might have already been answered. A quick google search with "your question + ADAU1701" or "your question + SigmaStudio" will usually provide many useful results.

Dayton Audio's SimgaStudio Supported Products

This guide will apply to all products in Dayton Audio's KABD series, which are listed below. This guide also mostly will apply to the older DSPB line of products, however the Bluetooth and external potentiometer features are not available in the DSPB line.

- **KABD-230**
- **KABD-250**
- **KABD-4100**
- **KABD-430**
- **DSPB-250 (Legacy)**
- **DSPB-100 (Legacy)**
- **DSPB-K**

KABD-250

KABD-4100

KABD-430

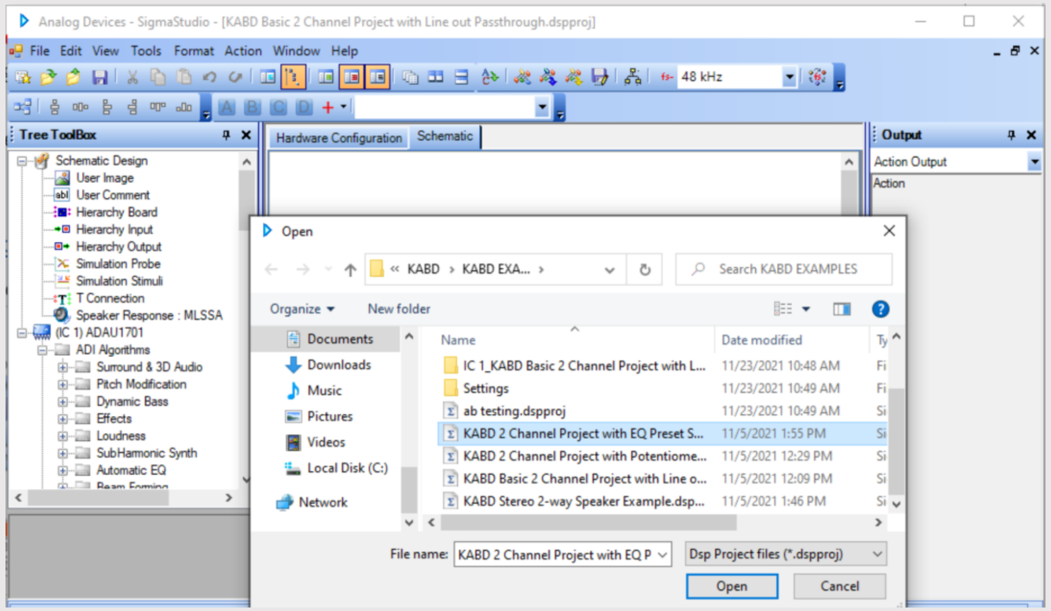
What do I need to connect my KABD Amplifier to SigmaStudio?

- Although your KABD amplifier houses its own ADAU1701 DSP chip, to communicate with your PC, a USBi programmer is required. Dayton Audio has two compatible programmers – the DSPB-ICP1 and the KPX. The programmer can be thought of as the bridge between your PC and your DSP chip so that they can communicate.
- **To connect your KABD Amplifier to SigmaStudio, you will need:**
  - A Windows PC / Laptop with available USB Port
  - The free SigmaStudio download
  - A KABD Amplifier and power supply
  - A compatible USBi programmer and USB cable
    - Dayton Audio DSPB-ICP1 (Micro-USB)
    - Dayton Audio KPX (USB Type C)

Connecting a KABD amplifier to SigmaStudio with an ICP1 or KPX programmer (Usbi)

- Notes before starting
  - **The following steps need to be completed in the exact order below for consistent connection to SigmaStudio.**
  - Your programmer only needs to be connected while programming, it can be removed once programming is complete, and can be used to program another amplifier.

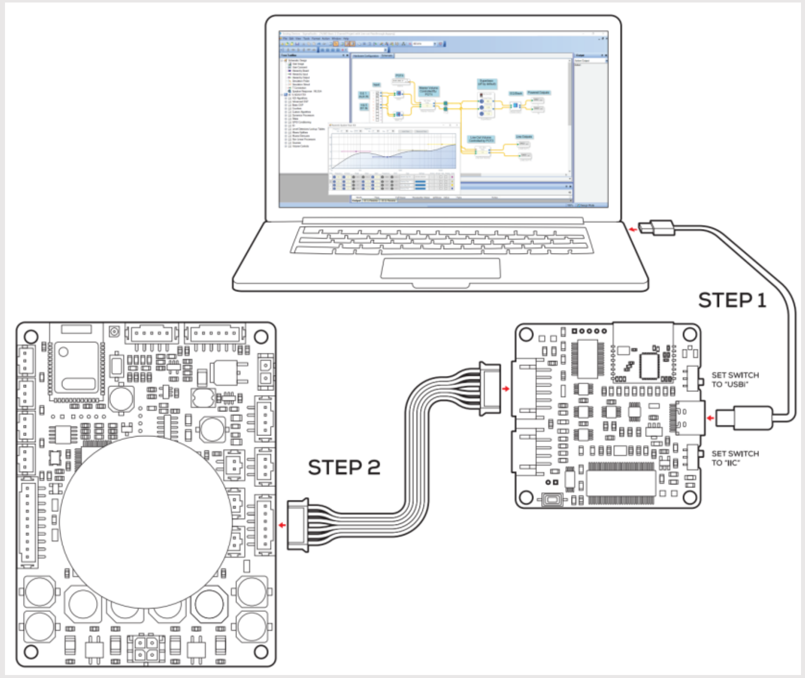
1. Prepare your KABD Amplifier and programmer for connection
   - A. Power your KABD amplifier and connect power supply appropriate for your specific amplifier. Optionally connect speakers and connect to Bluetooth.
   - B. Make sure the switches on your programmer are set properly for SigmaStudio (USBi) programming.
     - For the **KPX** programmer, the two switches should be set to USBi mode and IIC (I2S).
     - For the **ICP1** programmer, set the only switch to "Program".
2. Download and open SigmaStudio, and then load an example project from Dayton Audio's website by pressing File -> Open and then finding the file you downloaded. The file extension should be '.dspproj'.



Name	Date modified
IC 1_KABD Basic 2 Channel Project with L...	11/23/2021 10:48 AM
Settings	11/23/2021 10:49 AM
ab testing.dspproj	11/23/2021 10:49 AM
KABD 2 Channel Project with EQ Preset S...	11/5/2021 1:55 PM
KABD 2 Channel Project with Potentiome...	11/5/2021 12:29 PM
KABD Basic 2 Channel Project with Line o...	11/5/2021 12:09 PM
KABD Stereo 2-way Speaker Example.dsp...	11/5/2021 1:46 PM

File name: KABD 2 Channel Project with EQ P
Dsp Project files (*.dspproj)

Connecting a KABD amplifier to SigmaStudio with an ICP1 or KPX programmer

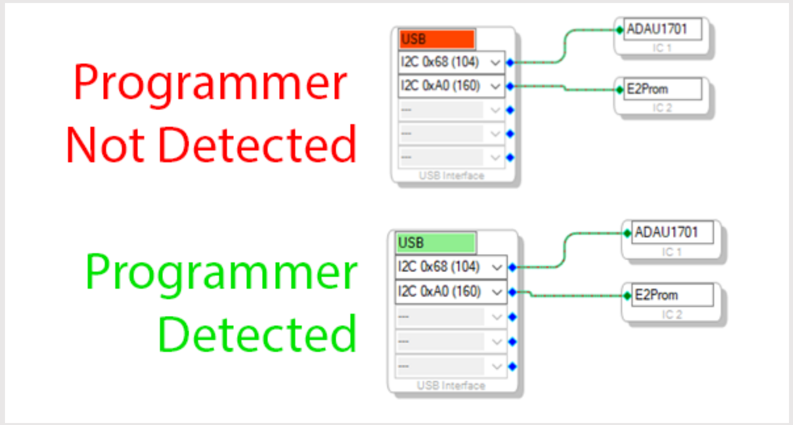


4.Plug your KPX or ICP1 programmer into your PC with an appropriate USB cable.

- It **should NOT be connected** to your KABD amplifier yet with the 6-pin cable. Only to the PC.
- You will notice the "USB" Block will turn green. This means that SigmaStudio can see your programmer. (it does not indicate connection to the KABD amplifier yet, ONLY PC connection to the programmer).
- If you do not see the "USB" block turn green like in the image, make sure your USB cable is functional and connected well on both ends or try another cable. If that does not fix the problem, try reinstalling SigmaStudio to ensure drivers have been correctly installed (they install automatically when SigmaStudio is installed, you should NOT have to install them manually)

5. After the programmer is connected to your PC, then connect your KABD amp's programming port to the programmer using the 6 pin cable included with the programmer.

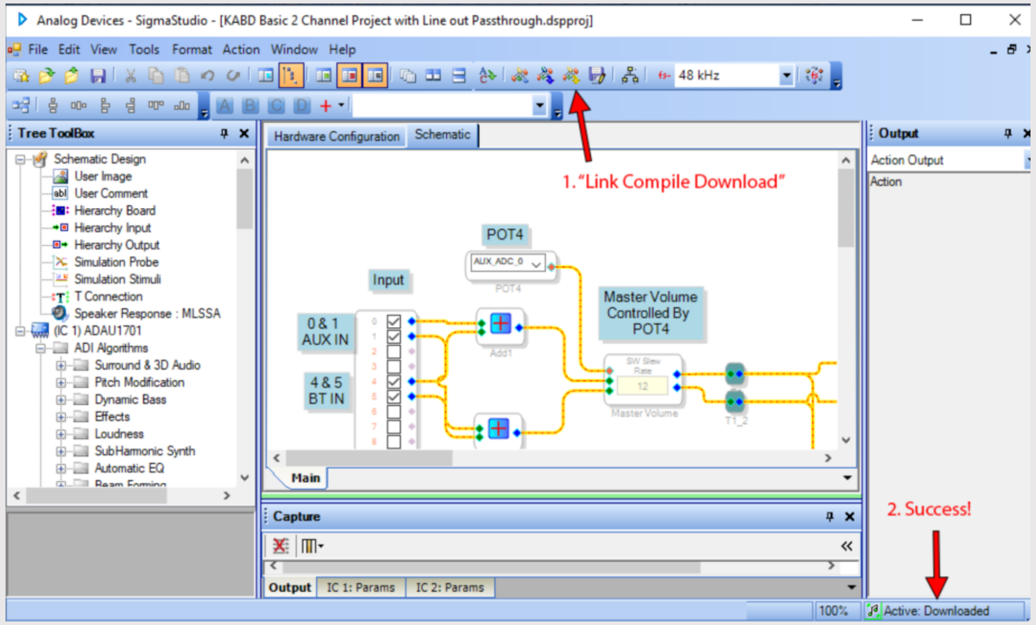
6. If these steps were completed in the correct order, you should now be ready to run an example project by following the next guide.



How to run custom DSP presets / programs

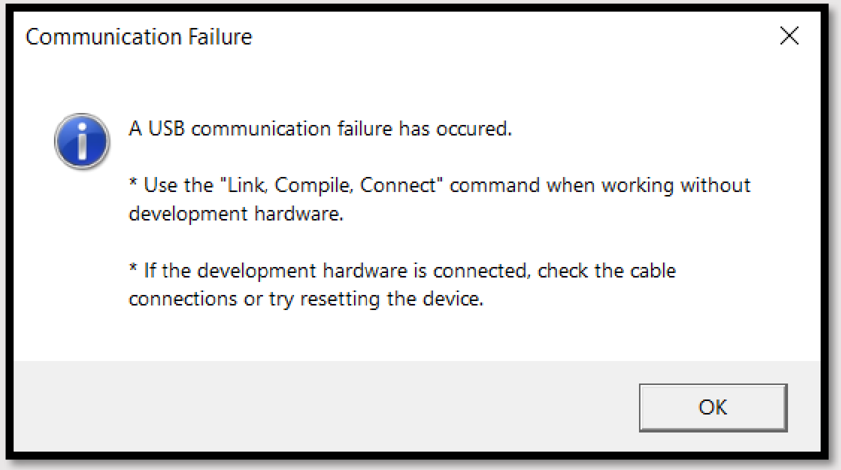
After you have successfully connected your KABD to your programmer, and your programmer to your PC, you are ready to run custom DSP programs created with SigmaStudio on your KABD amplifier.

1. To test that you can run custom programs, it is best to start by opening an example project from the Dayton Audio website with file -> open. The file type you need is '.dspproj', and numerous examples can be found as a pack from your KABD amplifier's product page.
2. When ready, press the "Link Compile Download" button in the toolbar.
3. You will then see "active downloaded" in the bottom right corner and hear your custom program. If you get a communication failure, see the next page for troubleshooting
4. Once you have successfully downloaded a custom program to your device, this program will only live in temporary memory while the device is on. To save it to permanent (non-volatile) memory, see the "burn to E2Prom" section. This will allow the custom program to be re-loaded after a power cycle.



Communication Failure!

- If you see the "communication failure" error after pressing "link compile download" button when trying to run programs, do not fret. In most cases, it is very easy to prevent.
- The typical cause of this error is that the programmer was connected first to the KABD, and then to the computer (the wrong order). To prevent this error, **connect your programmer to your PC first, and then connect your programmer to your KABD amplifier (the KABD must be powered on).**
- This error will also occur if you have forgotten to plug in power to your KABD.
  - If you have your programmer connected to the PC, and the programmer to your KABD, lights will flash on the KABD which might make you forget to power the KABD separately. If you have done this, unplug your programmer from the PC and your KABD and try again.
- If your 'USB' block is not green, you will also see this error because it indicates your PC cannot communicate with your programmer. (See previous sections)


Communication Failure

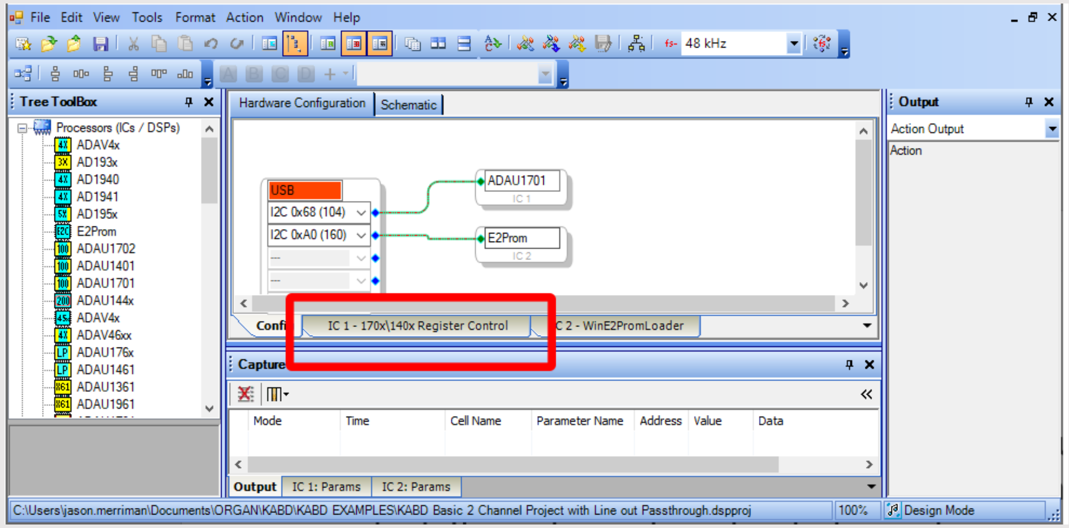
A USB communication failure has occured.

* Use the "Link, Compile, Connect" command when working without development hardware.

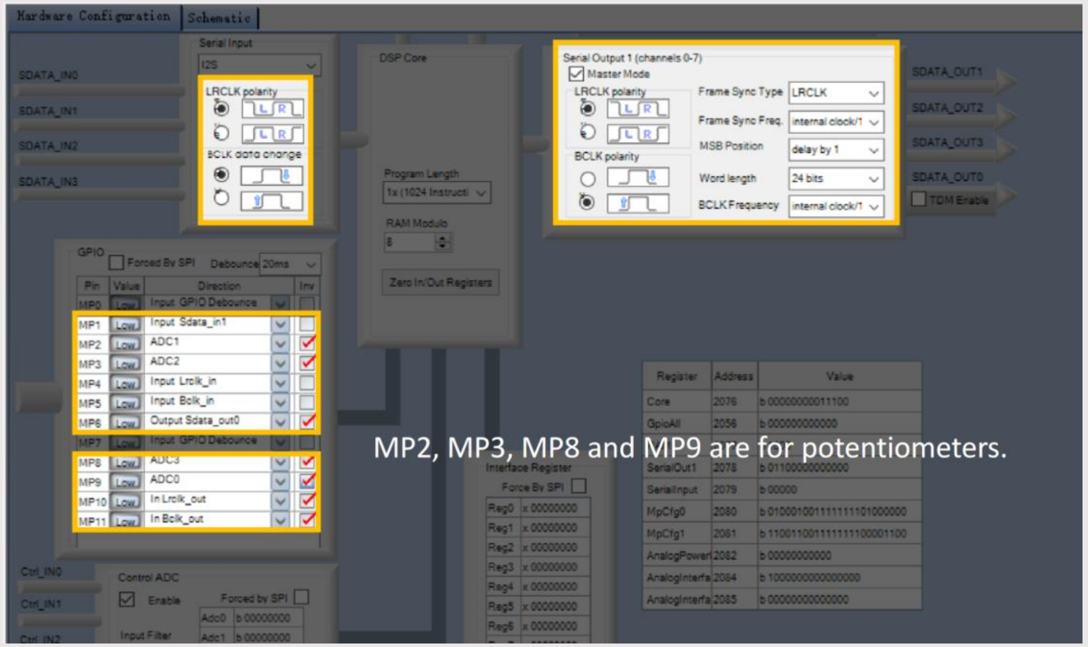
* If the development hardware is connected, check the cable connections or try resetting the device.

Hardware Configuration specific to KABD amplifiers

- The KABD amplifiers require additional configuration for new projects to allow for Bluetooth input (Bluetooth Modules are connected via I2S (digital) to the ADAU1701 directly) and also for potentiometer control via the ADUAU1701's ADC inputs (we need to configure multipurpose pins to ADC input mode). **If you have downloaded a custom program from the KABD product page, this has been done for you and this can be skipped.**
- Enter the "Register Control" tab near the bottom of your screen.
- Match all of your settings to the ones in the screenshot, or copy them from an example project for the KABD series.



C:\Users\jason.merriman\Documents\ORGAN\KABD\KABD EXAMPLES\KABD Basic 2 Channel Project with Line out Passthrough.dspproj



Serial Input: I2S

LRCLK polarity

BCLK data change

Program Length: 1x (1024 Instructi

RAM Modulo: 8

Serial Output 1 (channels 0-7)

☑ Master Mode

LRCLK polarity

BCLK polarity

Frame Sync Type: LRCLK

Frame Sync Freq.: internal clock/1

MSB Position: delay by 1

Word length: 24 bits

BCLK Frequency: internal clock/1

GPIO ☐ Forced By SPI Debounce 20ms

Pin	Value	Direction	Inv
MP0	Low	Input GPIO Debounce	☐
MP1	Low	Input Sdata_in1	☐
MP2	Low	ADC1	✓
MP3	Low	ADC2	✓
MP4	Low	Input Lrclk_in	☐
MP5	Low	Input Bclk_in	☐
MP6	Low	Output Sdata_out0	✓
MP7	Low	Input GPIO Debounce	☐
MP8	Low	ADC3	✓
MP9	Low	ADC0	✓
MP10	Low	In Lrclk_out	✓
MP11	Low	In Bclk_out	✓

Control ADC ☑ Enable Forced by SPI ☐

Input Filter

Adc0	b 00000000
Adc1	b 00000000

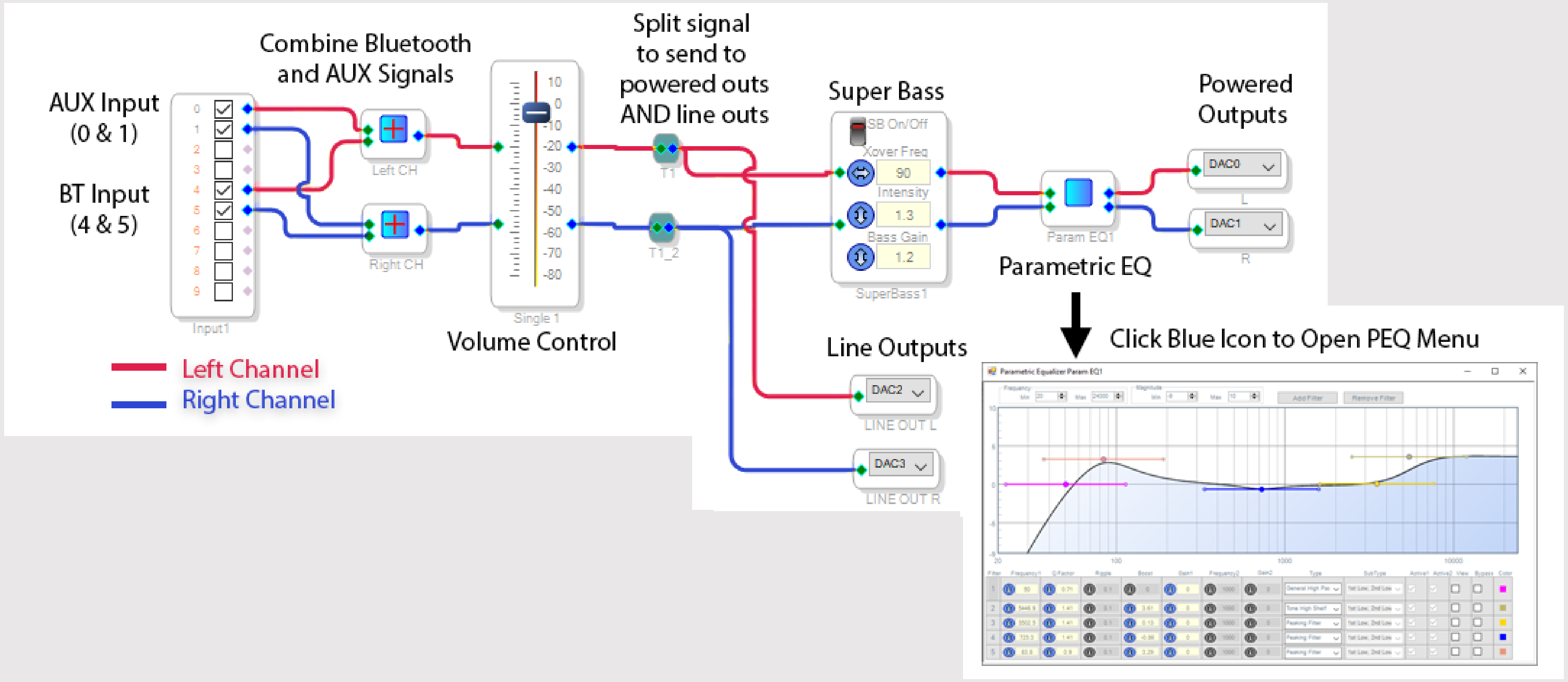
Reg0	x 00000000
Reg1	x 00000000
Reg2	x 00000000
Reg3	x 00000000
Reg4	x 00000000
Reg5	x 00000000
Reg6	x 00000000

Register	Address	Value
Core	2076	b 0000000011100
GpioAll	2056	b 000000000000
SerialOut1	2078	b 0110000000000
SerialInput	2079	b 00000
MpCfg0	2080	b 010001001111111101000000
MpCfg1	2081	b 110011001111111100001100
AnalogPowerl	2082	b 0000000000
AnalogInterfa	2084	b 1000000000000000
AnalogInterfa	2085	b 0000000000000

MP2, MP3, MP8 and MP9 are for potentiometers.

A basic example!

Below is a basic example project with extra labeling. Bluetooth and AUX inputs are combined, then sent to a basic volume control. From there, the left and right channels are split using T-blocks between powered outputs and line outputs. The powered output then has an optional super bass algorithm applied, then a few bands of PEQ are applied. We then send the processed output to the DAC0 and DAC1 outputs, which represent the connected speakers. The only processing applied to the line out is the master volume control, but we could easily add another parametric EQ block and filter our line out separately.

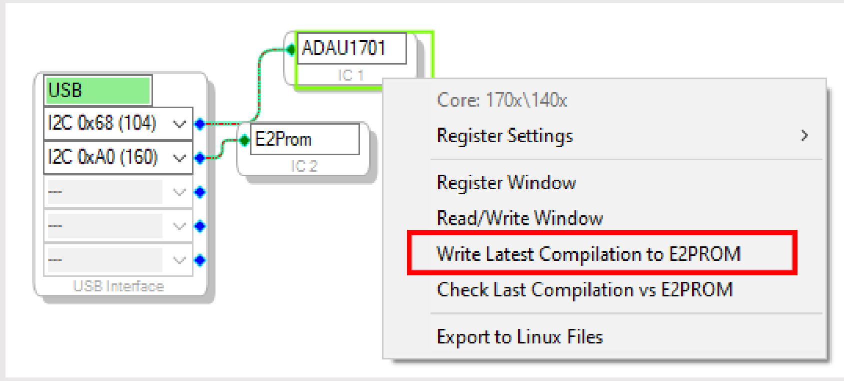


Writing to E2Prom - Saving programs to long term memory

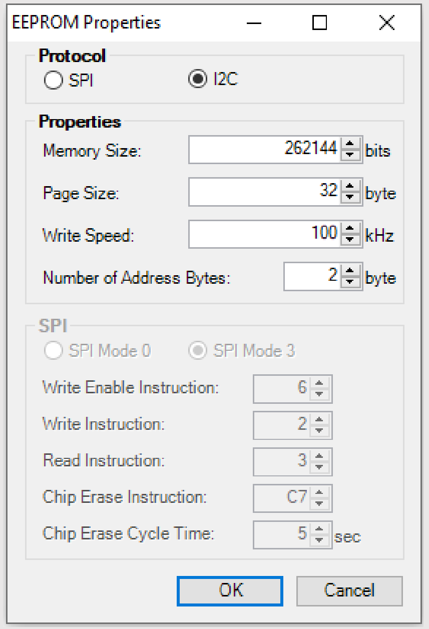
- When actively programming your amplifier with SigmaStudio, you will hear the changes you make in real time. However, this custom program is only in volatile (temporary) memory, which means it will be lost when the amplifier loses power. To prevent this loss, we burn the program to non-volatile memory that will be loaded by the DSP chip every time it boots up.
- This section assumes you have already pressed 'link compile download' and have a program you are ready to burn already loaded onto your amplifier.
- **IMPORTANT – This step will erase the stock configuration of the board! This means that the default functions of POT1-4 will reprogrammed or removed, depending on the SigmaStudio project that is loaded.**

STEPS

1. Right click your ADAU1701 block in the "hardware configuration" tab and press "write latest compilation to E2Prom.



2. In the window that opens, make sure the settings match our screenshot (they are defaults), and press OK. If there is a failure, make sure your KPX programmer is set to IIC (I2C) mode, or your ICP1 programmer is set to '1' for program mode.
3. If completed correctly, you will find that your DSP settings will remain even after a power cycle.



EEPROM Properties

Protocol
SPI [✓ I2C]

Properties

Memory Size:	262144	bits
Page Size:	32	byte
Write Speed:	100	kHz
Number of Address Bytes:	2	byte

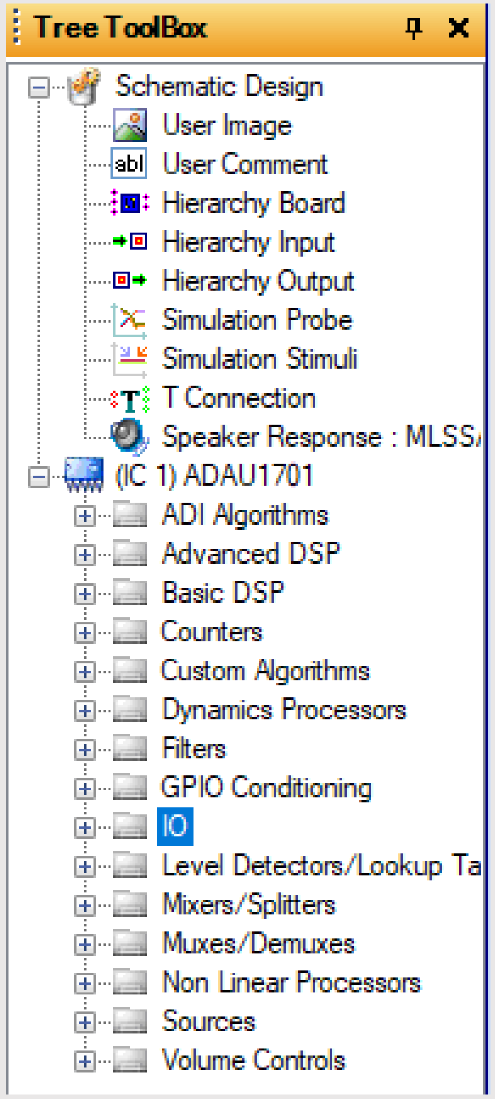
SPI
SPI Mode 0 [✓ SPI Mode 3]

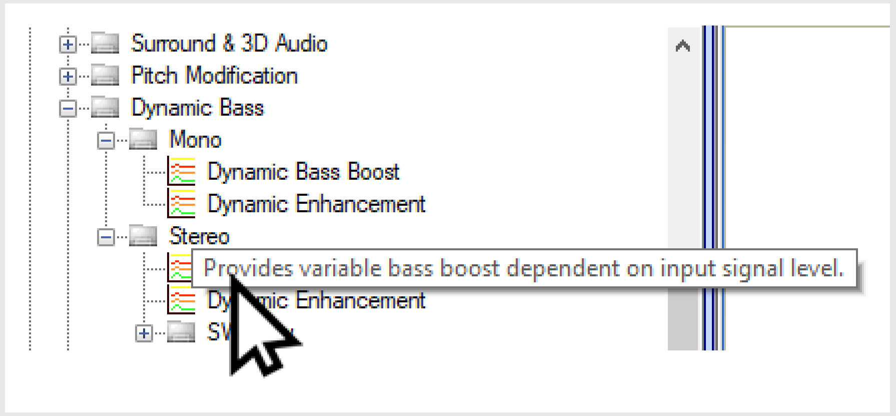
Write Enable Instruction:	6	
Write Instruction:	2	
Read Instruction:	3	
Chip Erase Instruction:	C7	
Chip Erase Cycle Time:	5	sec

OK Cancel

The SigmaStudio Toolbox

- Found on the left side of your screen, The SigmaStudio ToolBox is key to creating SigmaStudio programs. Here you will find all of your inputs, outputs, filters, and any other signal processing algorithm you can think of.
- Each item in the toolbox is considered a "block" and can be easily drag & dropped into your project. They are the "building blocks" of your custom programs, and they make SigmaStudio easy to use.
- Throughout this tutorial, each block that is used or described will include a screenshot of where to find it in the toolbar.
- With your cursor, hover over any item in the toolbox for a short description of its function.
- We recommend fully exploring the toolbar yourself, but this guide will give you numerous examples of useful blocks. You might get project inspiration by clicking through the toolbox.



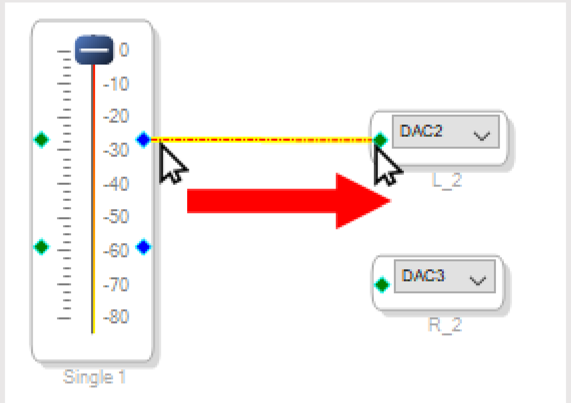


Provides variable bass boost dependent on input signal level.

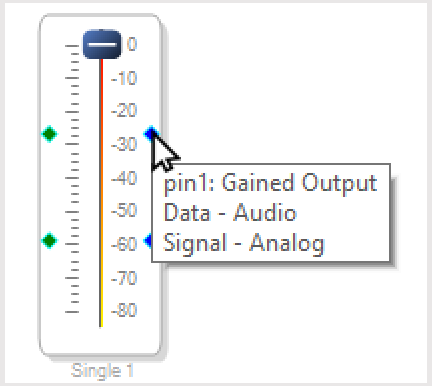
The SigmaStudio DSP “Block”

- Every SigmaStudio project is composed as a series of “blocks” that represent inputs, outputs, filters or DSP algorithms or functions. These blocks are connected with simple ‘wires’, and make SigmaStudio easy to use, as it allows you to make advanced projects without knowing the gritty details of how the DSP works and also make it easy to follow the Signal Flow from left to right (usually) of a project.
- Most blocks will have input pins on the left (green pins) and output pins on the right (blue pins). Some blocks also have external control pins (orange). For example, your potentiometer (ADC) blocks will connect to these orange pins for external volume control or filter control. If a block does not have an orange pin, it cannot be externally controlled without advanced techniques and hardware external to the KABD amplifiers.
- Blocks can simply be connected together by connecting the output of one block to the input of another block by clicking and dragging. This is possible to do with a touch screen, but we recommend a mouse or trackpad.
- You will see numerous examples of blocks in this guide.

Click and drag from pin to pin to connect blocks



Hover over an individual pin for more information about the pin



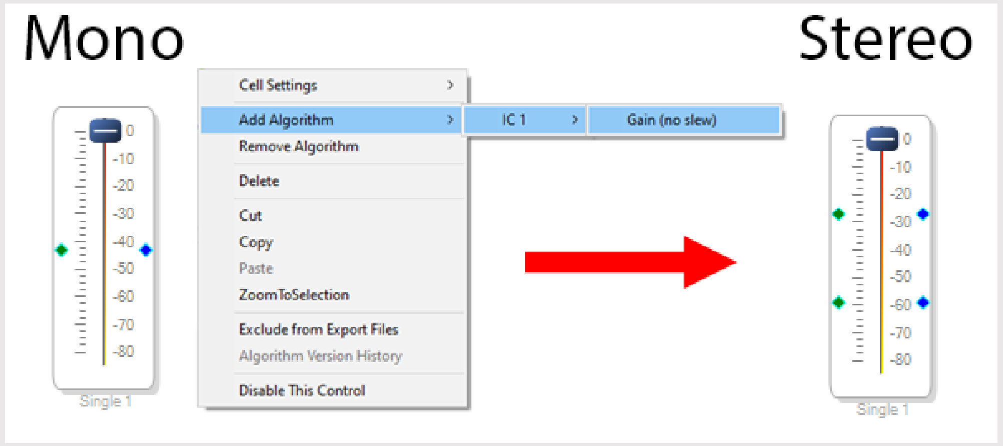
pin1: Gained Output
Data - Audio
Signal - Analog

Block Configuration and options

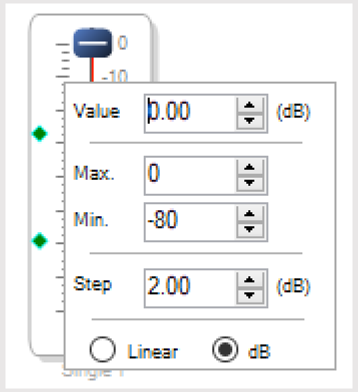
Once a block has been dragged into your project, most blocks can be configured beyond what is on the screen by right clicking.

If we click the inner part of some blocks, it will bring up additional configuration for the block. Many blocks do not have this, but some require pressing an icon within the block.

If we right click on the outer edge, we get a context menu that allows us to grow a block, disable it, etc. Different blocks have different options.



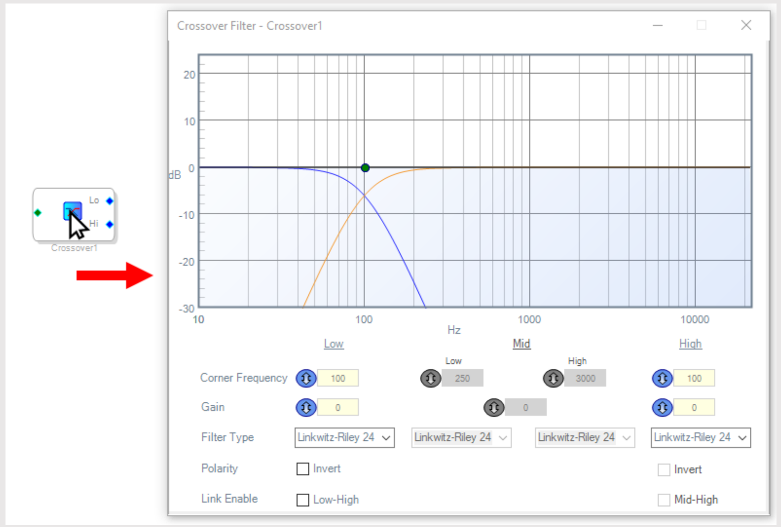
If we right click the inner part of the volume control block, it allows us to set min/max values



Value	0.00	(dB)
Max.	0	
Min.	-80	
Step	2.00	(dB)

Linear [✓ dB]

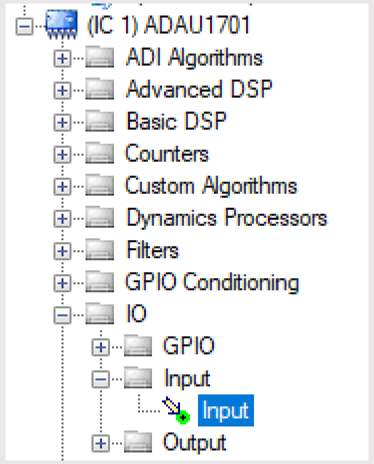
If we press the blue button within the crossover block, it opens the crossover menu.

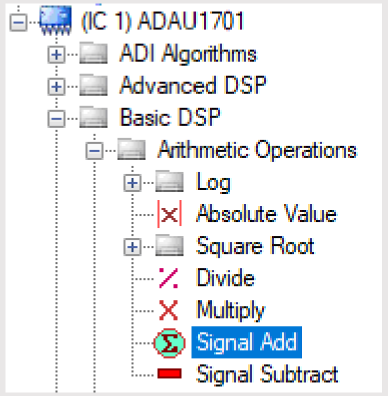


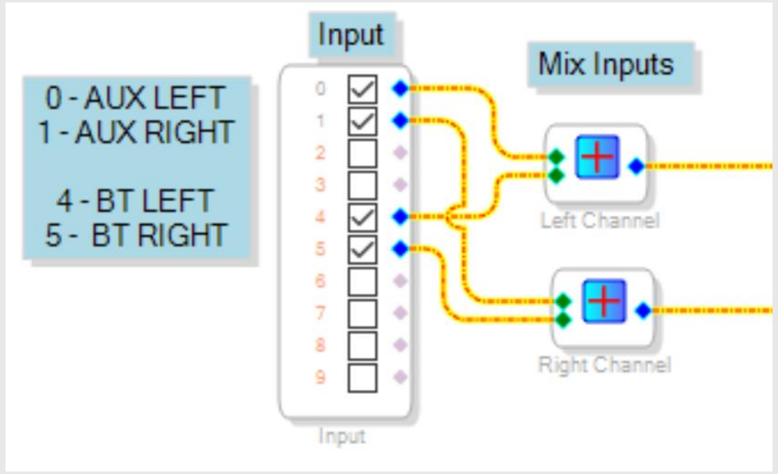
	Low	Mid Low	Mid High	High
Corner Frequency	100	250	3000	100
Gain	0	0		0
Filter Type	Linkwitz-Riley 24	Linkwitz-Riley 24	Linkwitz-Riley 24	Linkwitz-Riley 24
Polarity	☐ Invert			☐ Invert
Link Enable	☐ Low-High			☐ Mid-High

Input Routing

- Each KABD amplifier has an AUX input and a Bluetooth input. The AUX input is connected to the ADAU1701's ADC inputs (0 and 1), and the BT input is connected to I2S (digital) input labeled 4 and 5 on the input block.
  - For a KABD-4100 configured in slave mode, Bluetooth is disabled and IN4 & IN5 instead is the I2S input jack (J6)
- To the right, you can see our standard configuration. Using the "Input" block and the "Signal Add" block, we add the left AUX input (0) to the left BT input (4), and the same for the right inputs (1 and 5). This allows the user to seamlessly move between BT and AUX inputs without touching the speaker/amplifier.
- Advanced users could switch between AUX and Bluetooth mode with a physical switch and multiplexer block using a multiplexer block described later in this document.

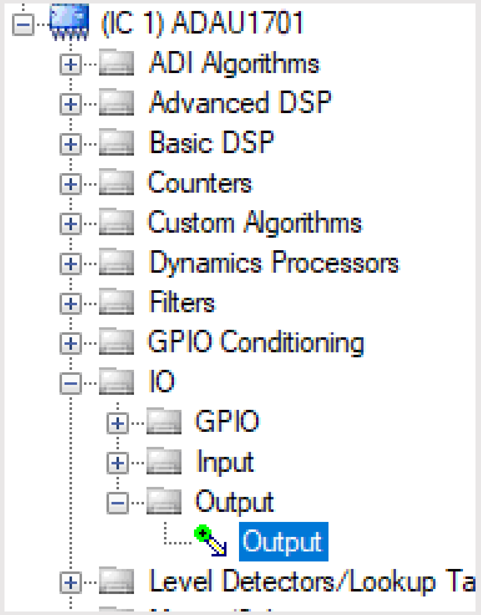


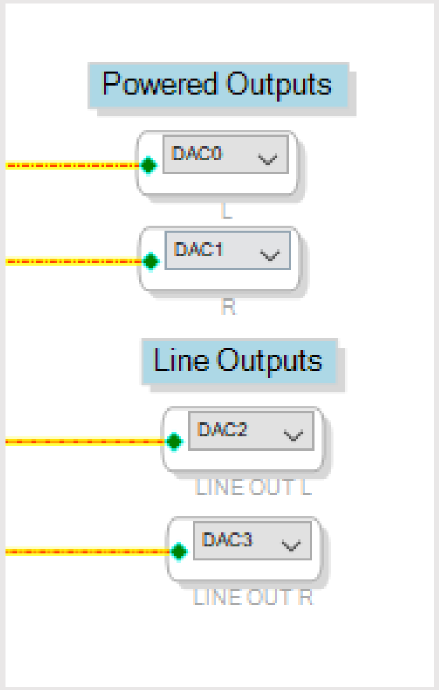


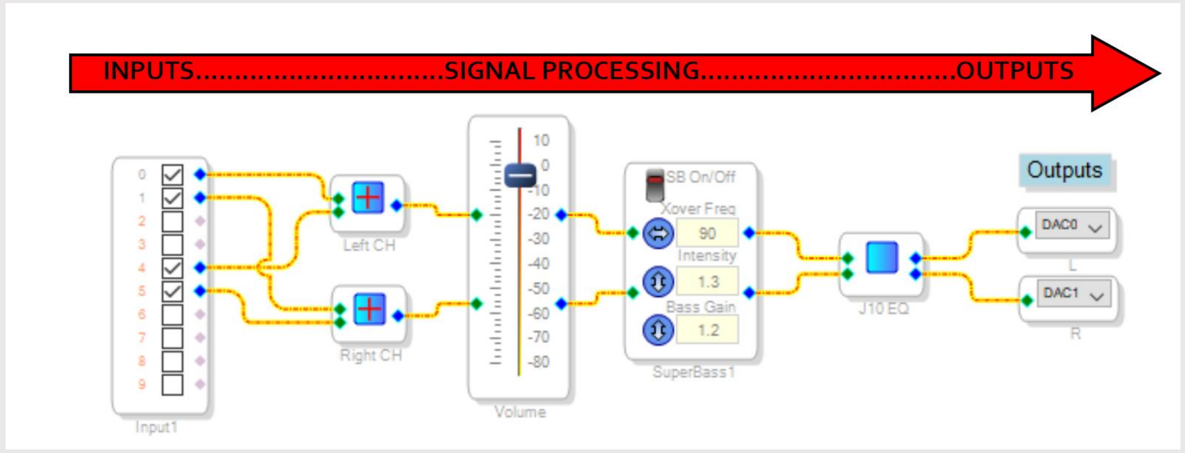


The Output Block

- The output block allows you to send your DSP'd signal to the powered outputs of your KABD, or even to line outputs.
- Typically at the end of a program, output blocks are placed after all of the desired processing blocks
- There are two types of outputs available. DAC outputs, and digital outputs.
    - The ADAU1701 DSP has 4 channels of DAC output built in (DAC0-3), and these outputs are directly connected to the outputs on your KABD. For the KABD-250, this means the powered outputs and the line outputs. For the KABD-4100 or KABD-430 it means all 4 channels of powered output.
    - Available on the KABD-430 and KABD-4100, thee are I2S Digital Output Ports on the board. These audio can be sent to these via DIG0 and DIG1 (see next pages)



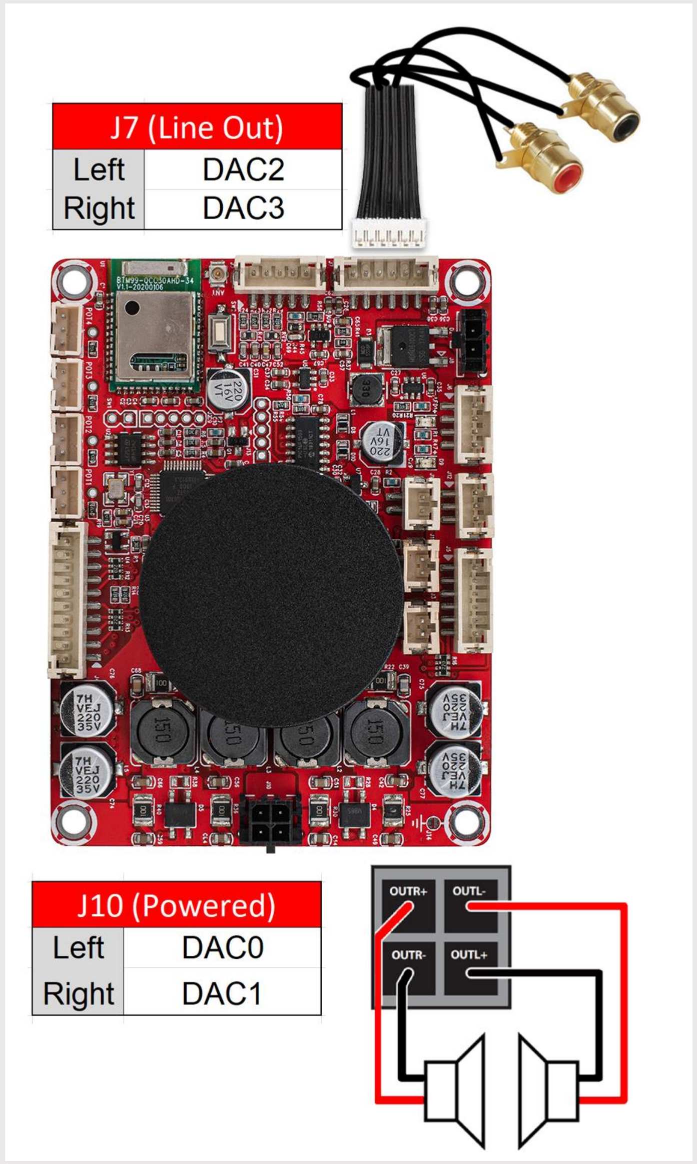




Output Routing – KABD-250 and KABD-230

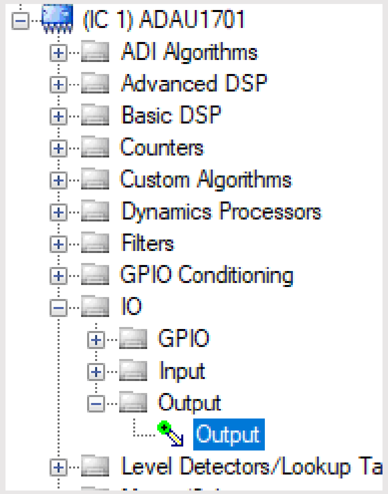
- Use the “Output” Block to send your DSP’d signal to your KABD’s powered or line outputs.
- Output blocks are usually end the ‘end’ of your program, placed after DSP blocks

  INPUT -> DSP Block -> DSP Block -> OUTPUT
- The KABD-250 and KABD-230 have 2 powered output channels (J10) and 2 line level output channels (J7).
  - These outputs are all directly attached to the ADAU1701’s DAC outputs.
- Powered outputs (J10) and Line outputs (J7) can have completely separate DSP, or none at all.

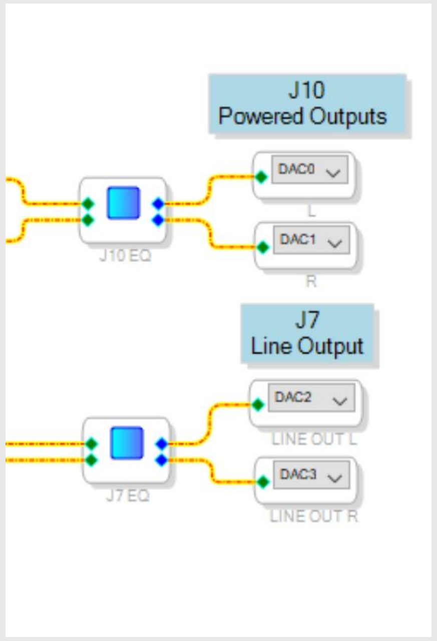


J7 (Line Out)	
Left	DAC2
Right	DAC3

J10 (Powered)	
Left	DAC0
Right	DAC1

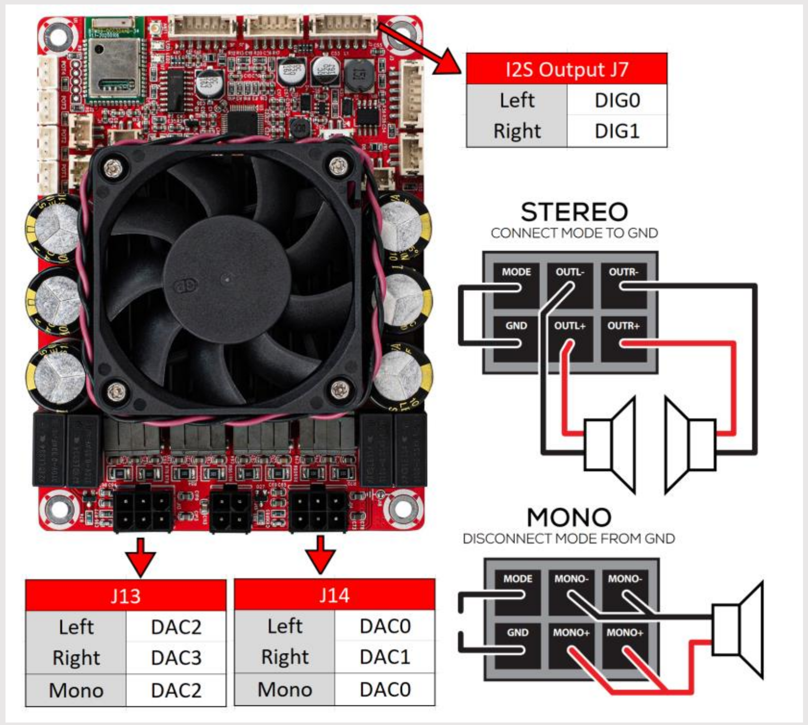


(IC 1) ADAU1701
- ADI Algorithms
- Advanced DSP
- Basic DSP
- Counters
- Custom Algorithms
- Dynamics Processors
- Filters
- GPIO Conditioning
- IO
  - GPIO
  - Input
  - Output
    - Output
- Level Detectors/Lookup Ta



Output Routing – KABD-4100

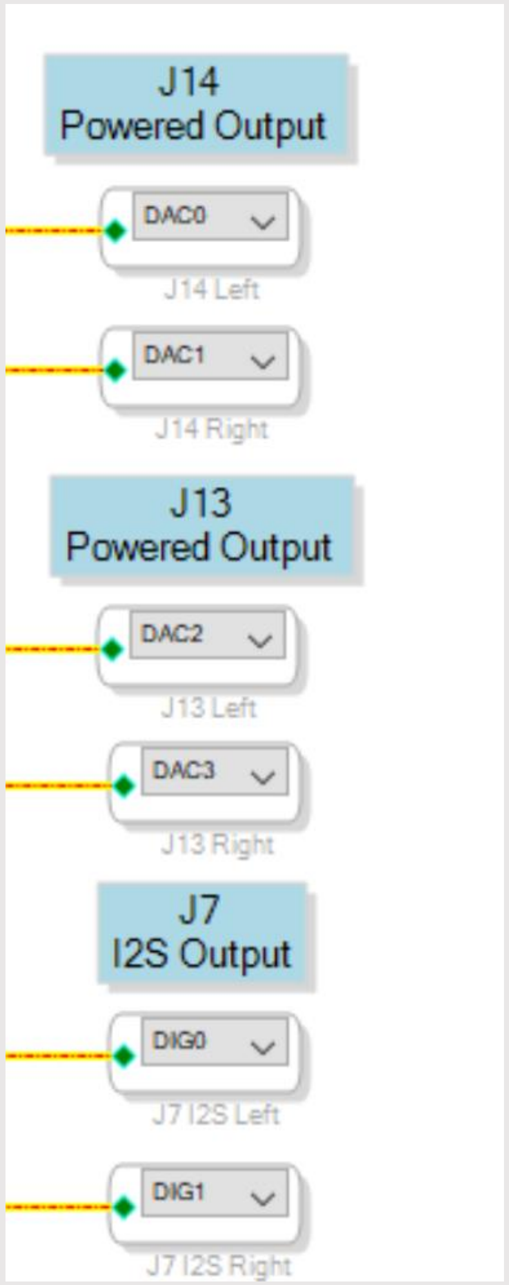
- The KABD-4100 has 2 stereo speaker jacks for a maximum of 4 channels of powered output. Each output jack can also be bridged for a more powerful output. The powered outputs are DAC0, DAC1, DAC2 and DAC3.
  - These outputs are all directly attached to the ADAU1701's DAC outputs.
- When J14 is in bridged mode, use DAC0 as your output. When J13 is in bridged mode, use DAC2 as your output.
- The KABD-4100 has an I2S Output Port that is used for cascading 2 KABD-4100 devices together. For this to work, the audio signal must be routed to DIG0 and DIG1 outputs.



I2S Output J7	
Left	DIG0
Right	DIG1

J13	
Left	DAC2
Right	DAC3
Mono	DAC2

J14	
Left	DAC0
Right	DAC1
Mono	DAC0



J14 Powered Output
DAC0 — J14 Left
DAC1 — J14 Right

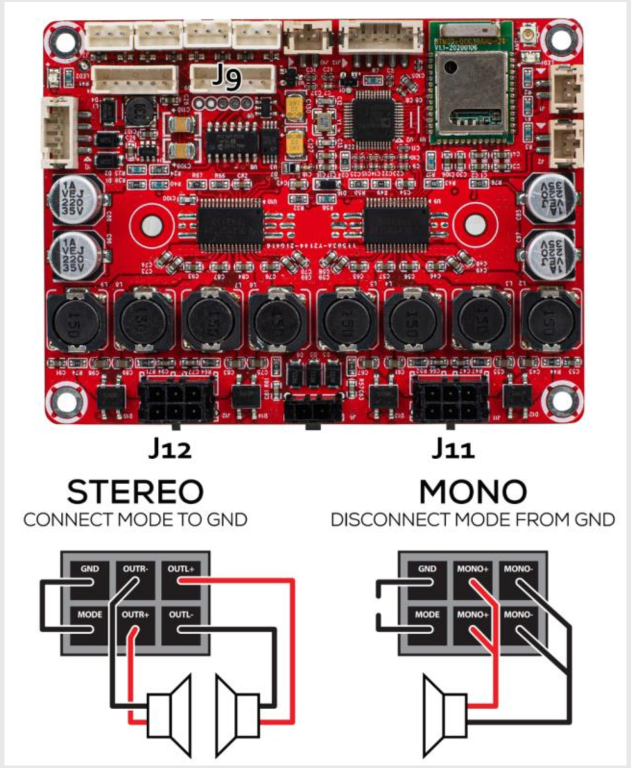
J13 Powered Output
DAC2 — J13 Left
DAC3 — J13 Right

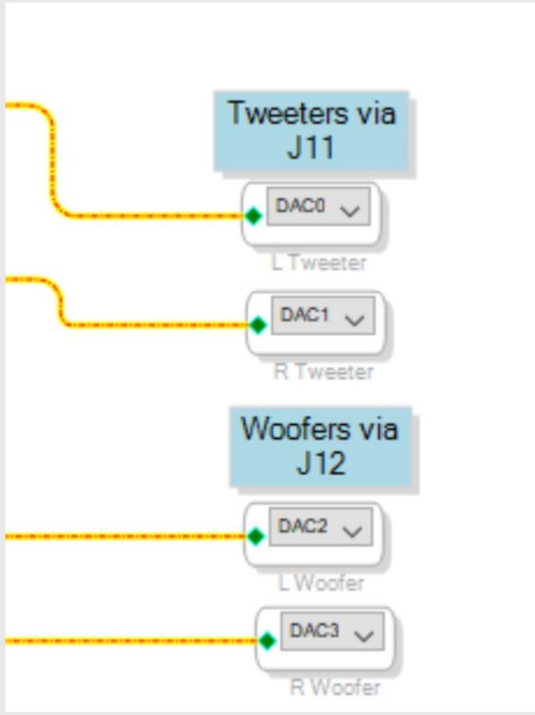
J7 I2S Output
DIG0 — J7 I2S Left
DIG1 — J7 I2S Right

Output Routing – KABD-430

- The KABD-430 has 2 stereo speaker jacks for a maximum of 4 channels of powered output. Each output jack can also be bridged for a more powerful output. The powered outputs are DAC0, DAC1, DAC2 and DAC3.
  - These outputs are all directly attached to the ADAU1701's DAC outputs.
- When J11 is in bridged mode, use DAC0 as your output. When J12 is in bridged mode, use DAC2 as your output.
- The KABD-430 has an I2S Output Port as J9. For this to work, the audio signal must be routed to DIG0 and DIG1 outputs.

I2S Output J9	
Left	DIG0
Right	DIG1



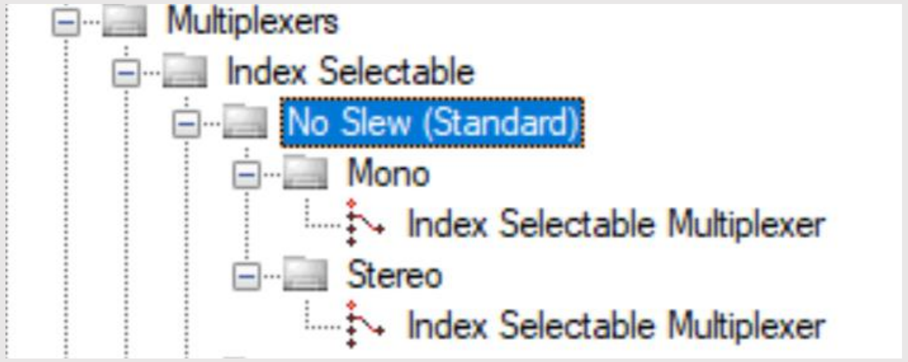


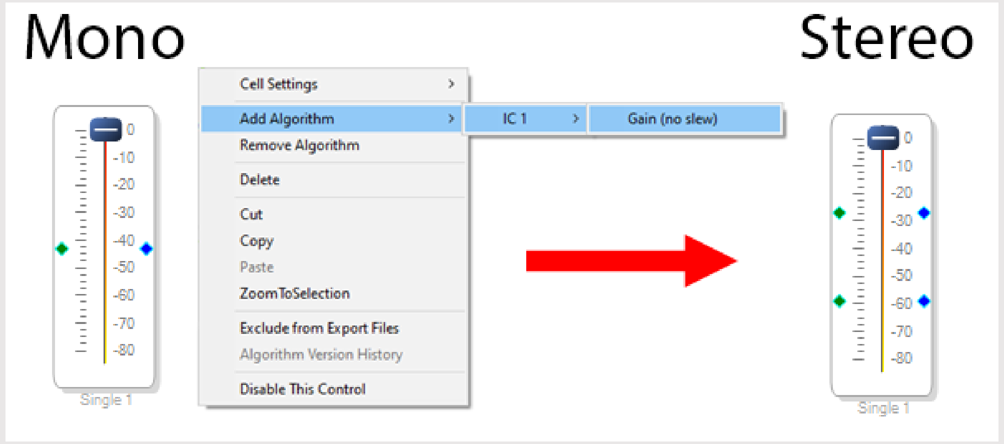
J11	
Left	DAC0
Right	DAC1
Mono	DAC0

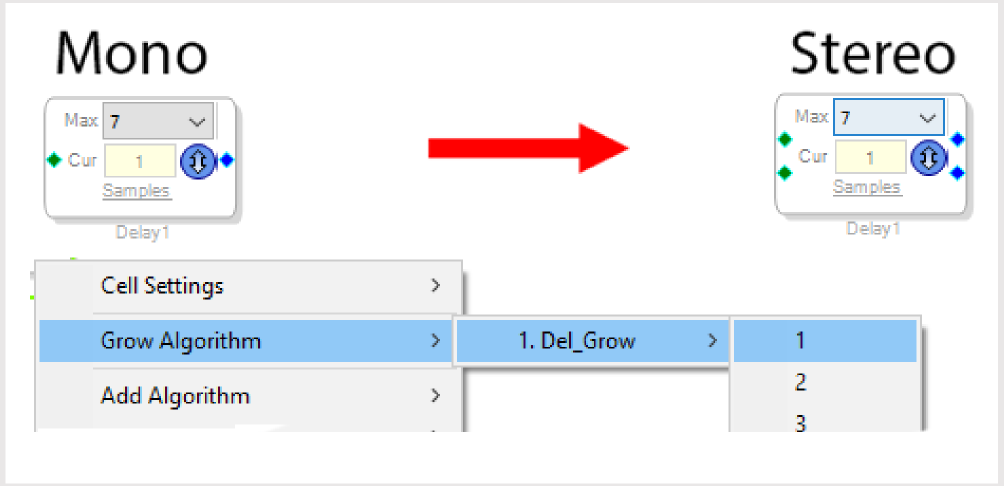
J12	
Left	DAC2
Right	DAC3
Mono	DAC2

Turning a Mono Block into A Stereo Block

- Some blocks have specific stereo and mono versions of themselves. Other blocks might be only mono by default, but by right clicking and pressing ‘add algorithm’ or ‘grow algorithm’ can turn these blocks into stereo and beyond.
- For some blocks, ‘add algorithm’ will have other effects.

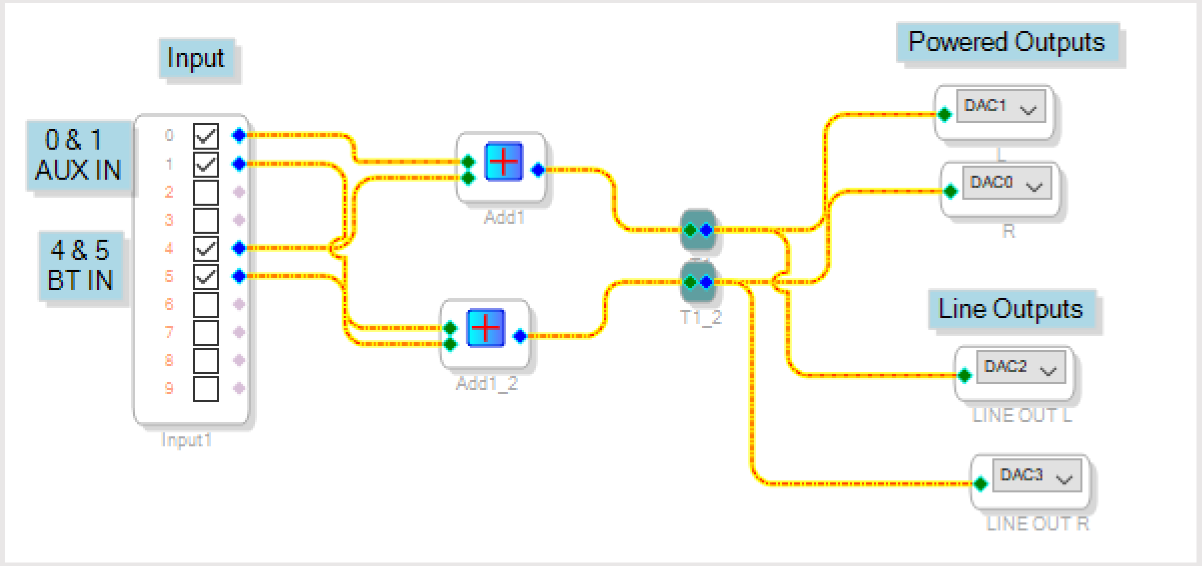


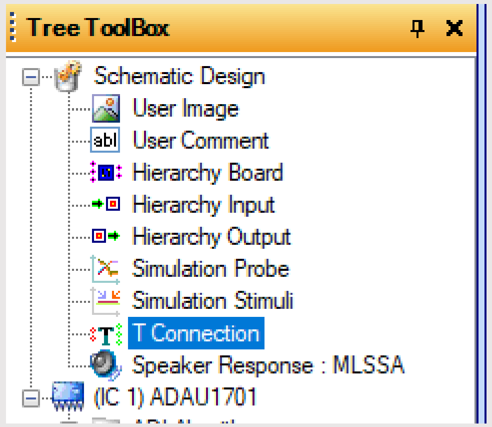




T-Connection Block (Splitting signals)

- Normally, you can only connect a single output pin to a single input pin. To get around this, we use the "T Connection" block which allows us to connect a signal line to as many other inputs as we want.
- The basic example below shows us splitting our left and right channels to the powered outputs and the line outputs.
- The T Connection block can be thought of like a Y splitter cable.



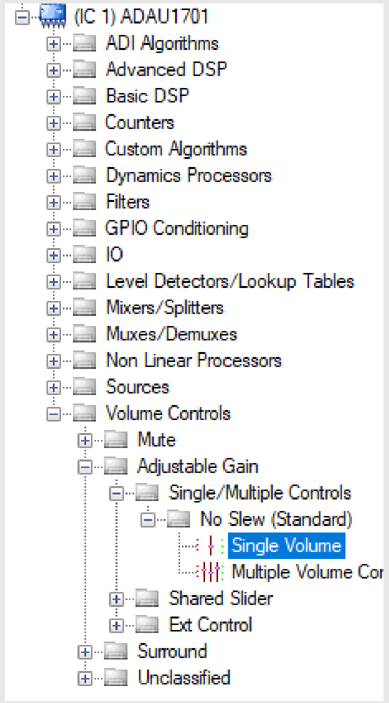


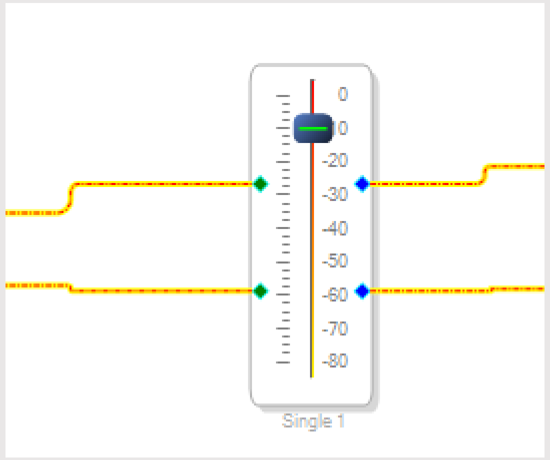
Volume Controls

- There are numerous types of volume controls via SigmaStudio. The two types that are most common are -

Graphic Volume Controls

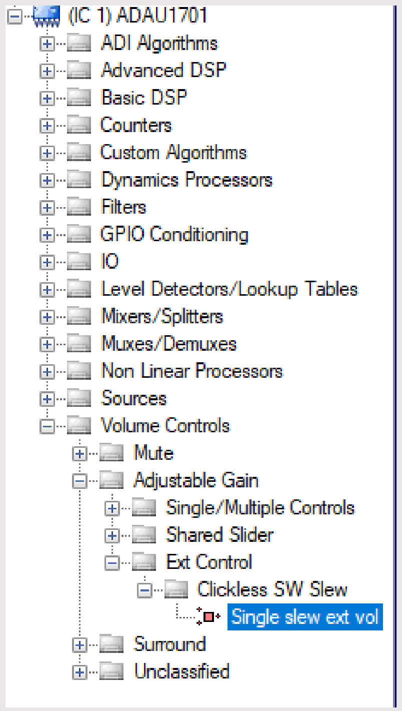
- Configurable within SigmaStudio itself by dragging the slider.
- Can be further configured / by right clicking, allowing max and min limits, growing from mono to stereo, etc.
- Available in slew or no slew types (see later section for description)

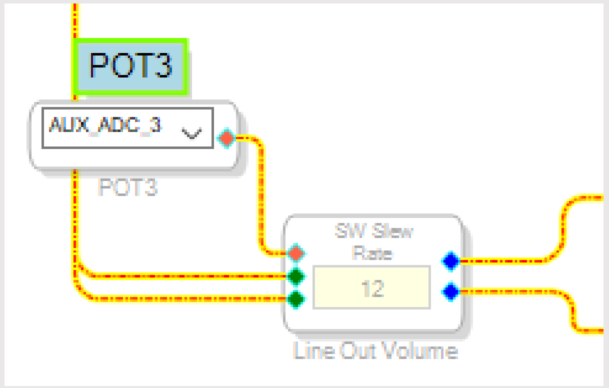


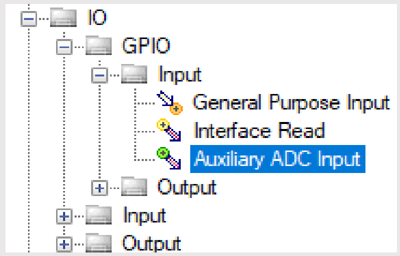


External (Hardware) Volume Controls

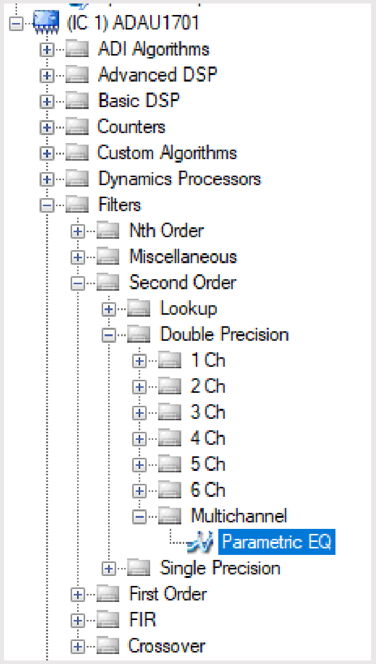
- Controllable via external potentiometers
- Must connect an ADC input (this represents the ports POT1-4)
  - "AUX_ADC_3" below represents input of the POT3 port on a KABD amplifier.
- Right click to grow the block to have stereo ins/outs
- The number in the box is the slew rate



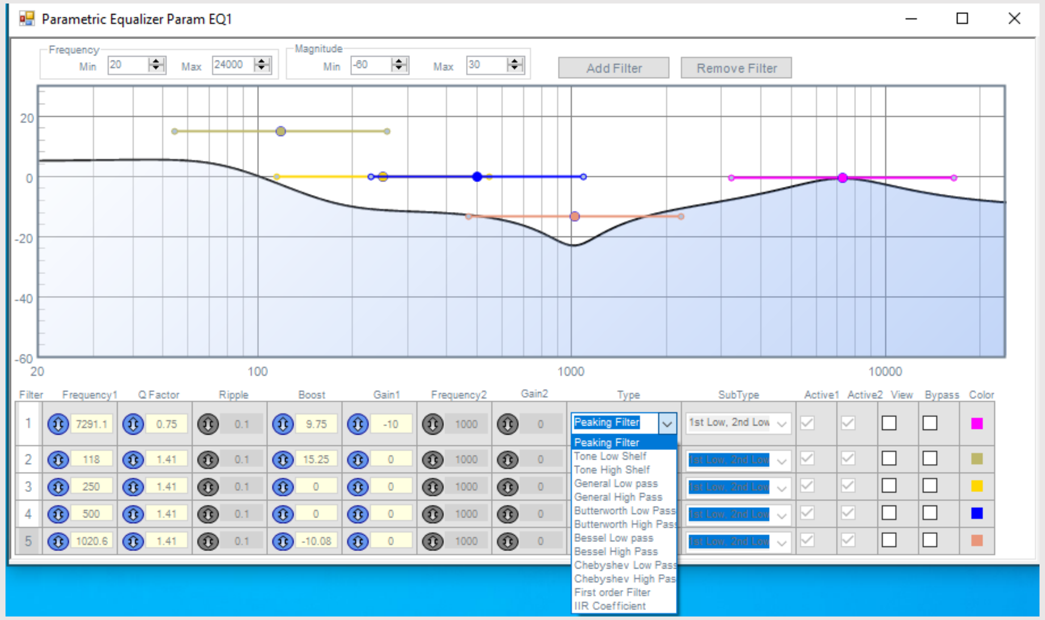




Parametric EQ Block

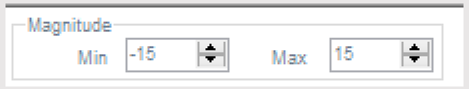


- There are many types of filter blocks, however the parametric EQ block allows for graphic, easy to use and precise adjustment of numerous bands of peaking filters, high/low pass filters, tone shelves and more.
- **Do not press the 'add filter' button while your KABD amplifier is connected to SigmaStudio.** This will result in loud and awful sounds. If you need to add more bands of EQ/filters (using the add filter button), **disconnect your KABD first.**
- Click the blue button in the middle of the block to open the graphic PEQ menu seen to the right.
- There is not necessarily a hard limit on how many bands of EQ you can use, but the more bands you use, the more resources of your DSP chip you are using. It is unlikely you will hit this limit from the PEQ block alone, so feel free to keep adding more until SigmaStudio fails to compile the project (it will warn you).
- The default vertical scale for this block is -60dB to +30dB, which is quite large and will result in dramatic changes in sound by clicking and dragging. For a better representation of your filters, try reducing the scale to something like -15dB to +15dB



Frequency Min 20 Max 24000
Magnitude Min -60 Max 30

Filter	Frequency1	Q Factor	Ripple	Boost	Gain1	Frequency2	Gain2	Type	SubType	Active1	Active2	View	Bypass	Color
1	7291.1	0.75	0.1	9.75	-10	1000	0	Peaking Filter	1st Low, 2nd Low	✓	✓	☐	☐	🟪
2	118	1.41	0.1	15.25	0	1000	0		1st Low, 2nd Low	✓	✓	☐	☐	🟫
3	250	1.41	0.1	0	0	1000	0		1st Low, 2nd Low	✓	✓	☐	☐	🟨
4	500	1.41	0.1	0	0	1000	0		1st Low, 2nd Low	✓	✓	☐	☐	🟦
5	1020.6	1.41	0.1	-10.08	0	1000	0		1st Low, 2nd Low	✓	✓	☐	☐	🟧



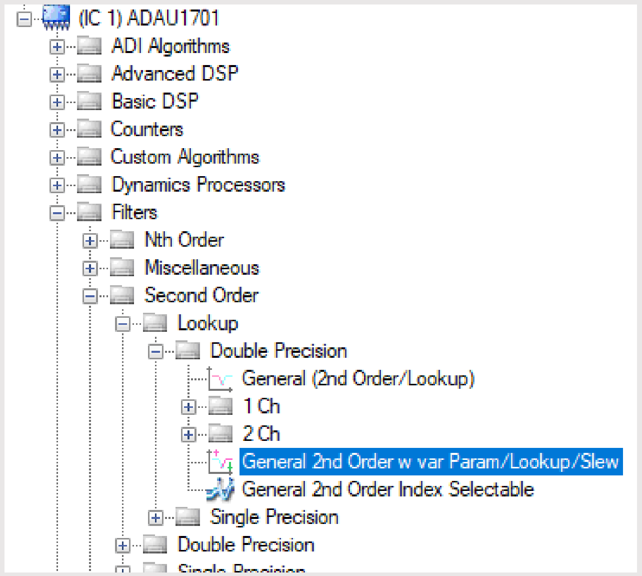
Magnitude
Min -15 Max 15

Potentiometer Controlled Filters

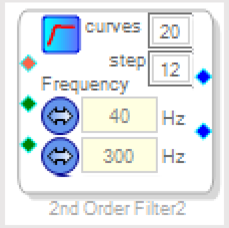
"General 2nd Order w var Param/Lookup/Slew" block

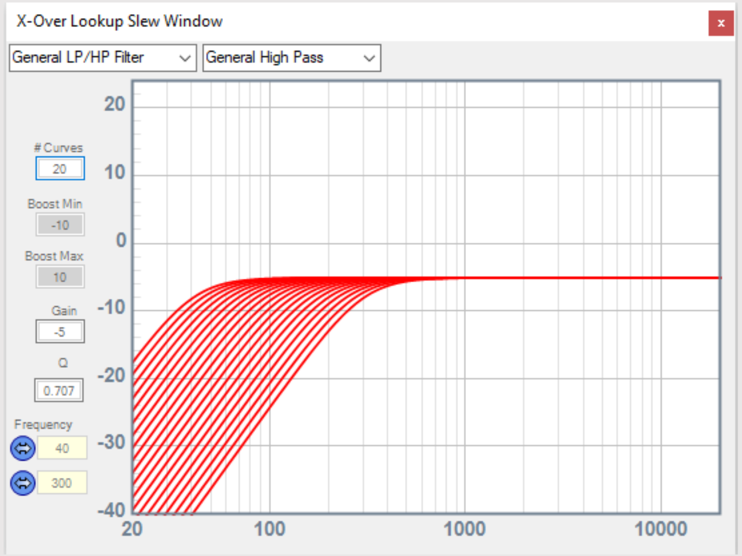
- Although there are multiple blocks that could allow for potentiometer control of filters, we choose our favorite, the

  "General 2nd Order w var Param/Lookup/Slew" block for this guide.
- This filter works by pre-calculating a specific number of filters based on user parameters, and then storing them into the DSP memory (calculating these on the fly is beyond the processing power of the ADAU1701). The block will apply the corresponding filter based on the index (number) applied to the orange pin of the block. So in this example, if we apply 1 to the orange pin, we get a HPF at 40 Hz applied to our signal. If we apply 20 to the orange pin, we get a HPF at 300 Hz.
- Press the blue button within the filter block to open the menu to the right. Adjust the number of curves, filter type, and frequency range or boost range of the filter from here.
- The larger the number of curves used, the more DSP resources are used.



(IC 1) ADAU1701
- ADI Algorithms
- Advanced DSP
- Basic DSP
- Counters
- Custom Algorithms
- Dynamics Processors
- Filters
  - Nth Order
  - Miscellaneous
  - Second Order
    - Lookup
      - Double Precision
        - General (2nd Order/Lookup)
        - 1 Ch
        - 2 Ch
        - General 2nd Order w var Param/Lookup/Slew
        - General 2nd Order Index Selectable
      - Single Precision
    - Double Precision





X-Over Lookup Slew Window

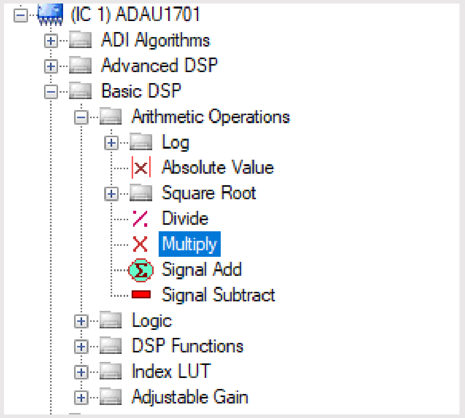
General LP/HP Filter | General High Pass

Curves: 20
Boost Min: -10
Boost Max: 10
Gain: -5
Q: 0.707
Frequency: 40, 300

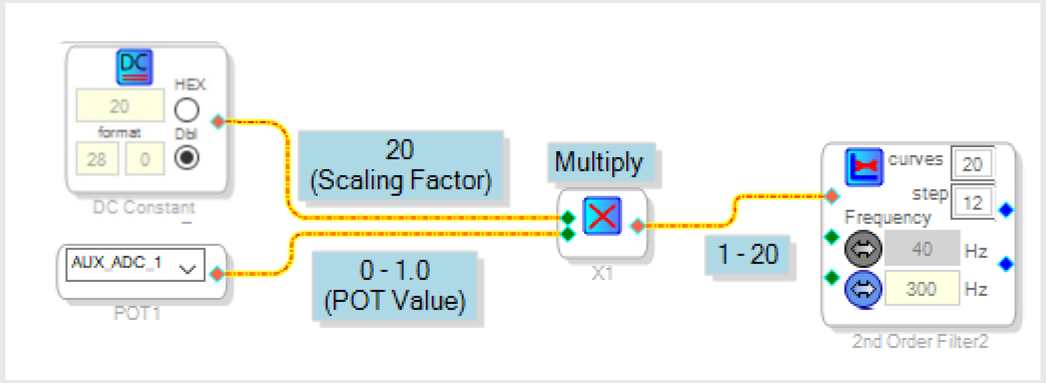
Potentiometer Controlled Filters

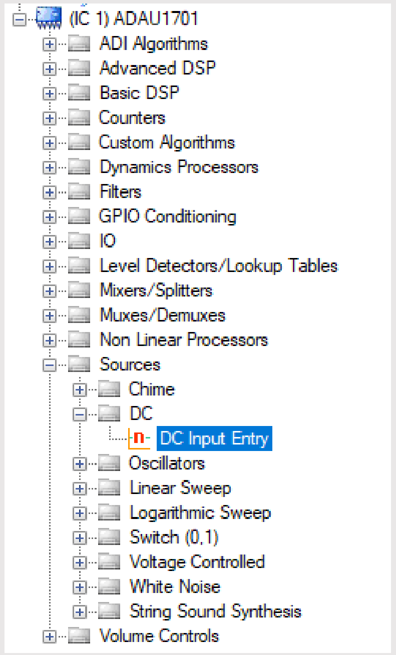
How to create an index value from a potentiometer input based on it's position

- The input from our potentiometer is not a number 1-20, instead, it is a value between 0 and 1. With some simple math blocks, we can scale the value from our potentiometer input to create index values 1-20 that our filter block can work with.
- We use the "DC Input Entry" and set it to 20 to create a scaling factor that matches our number of curves.
- All that's left is simple multiplication. If we multiply 20 (the number of curves we have) times our potentiometer input value (0 – 1.0), we will get a value between 0 and 20, which we can apply directly to the index pin of our filter block (it will round itself).
- This system effectively lets us select a number between 0-20 proportional to how much we turn out potentiometer.



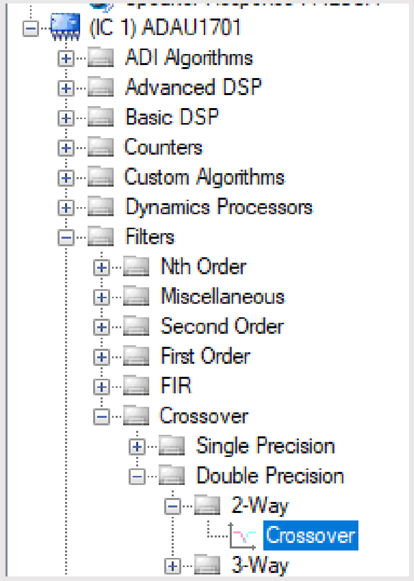
(IC 1) ADAU1701
- ADI Algorithms
- Advanced DSP
- Basic DSP
  - Arithmetic Operations
    - Log
    - Absolute Value
    - Square Root
    - Divide
    - Multiply
    - Signal Add
    - Signal Subtract
  - Logic
  - DSP Functions
  - Index LUT
  - Adjustable Gain



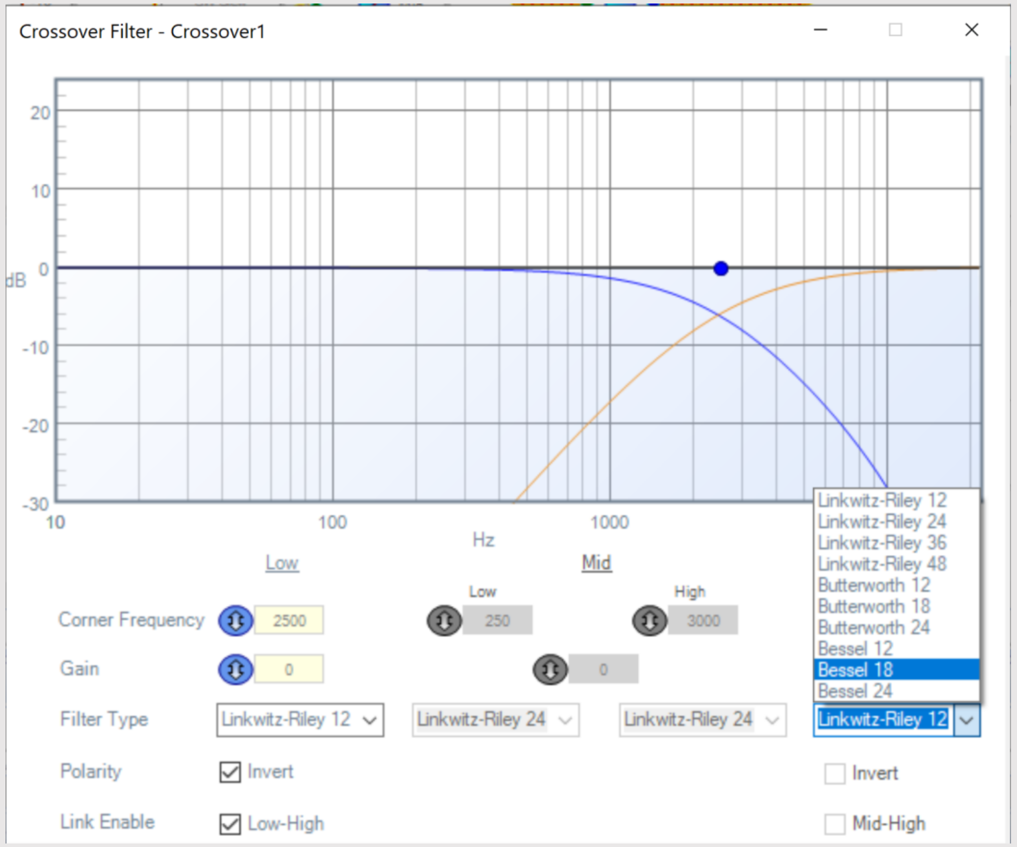


(IC 1) ADAU1701
- ADI Algorithms
- Advanced DSP
- Basic DSP
- Counters
- Custom Algorithms
- Dynamics Processors
- Filters
- GPIO Conditioning
- IO
- Level Detectors/Lookup Tables
- Mixers/Splitters
- Muxes/Demuxes
- Non Linear Processors
- Sources
  - Chime
  - DC
    - DC Input Entry
  - Oscillators
  - Linear Sweep
  - Logarithmic Sweep
  - Switch (0,1)
  - Voltage Controlled
  - White Noise
  - String Sound Synthesis
- Volume Controls

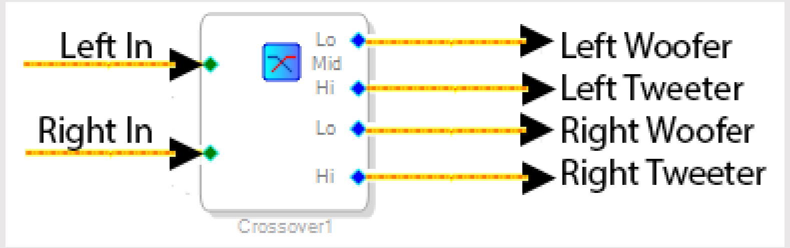
The Crossover Block



- The crossover block provides a convenient interface for doing crossover work, and even shows the effects of phase and frequency response changes due to varying filter alignments and slopes.
- Simply provide your input on the left side of the block, and the output of the block will split your signal into high and low signals.
- You will need to right click -> add algorithm to make this block stereo.
- KABD-250/230 – with the two channels boards, this could be useful if you're making a mono 2-way system, or you have an external amplifier connected to make a stereo 2-way system.
- KABD-4100/430 – with the 4 channel boards, this block is incredibly useful for designing/making 2-way or 2-way systems. **No need to build a new circuit to try a new crossover. Just click and drag or type in some numbers.**

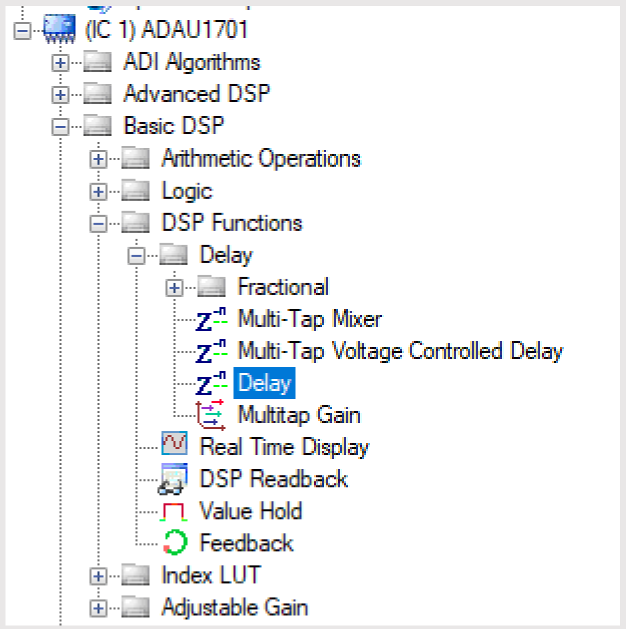


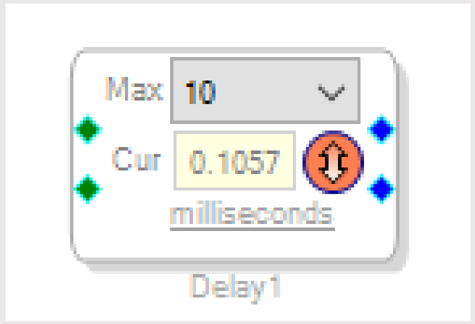
	Low	Mid Low	Mid High	High
Corner Frequency	2500	250	3000	
Gain	0	0		
Filter Type	Linkwitz-Riley 12	Linkwitz-Riley 24	Linkwitz-Riley 24	Linkwitz-Riley 12
Polarity	✓ Invert			Invert
Link Enable	✓ Low-High			Mid-High

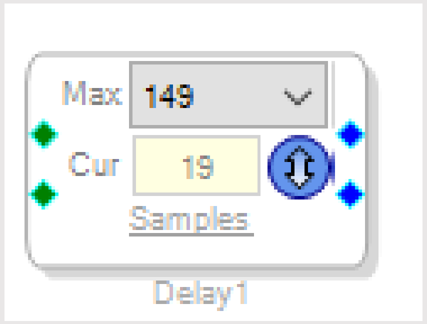


Delays

- The delay block is especially useful in systems containing drivers with different acoustic centers, such as a woofer that sits back more in the z-axis than the tweeter. By careful use of delay, time alignment can be achieved, which in most cases can dramatically improve sound in the crossover region between the drivers. A good professionally designed crossover **always** accounts for this.
- The delay block might also be useful when using the line output on the KABD-250/230 to connect a powered subwoofer, to integrate the subwoofer's sound better with the main speakers.
- The units of the delay block can be either milliseconds or samples. Click the underlined label on the block to change units.

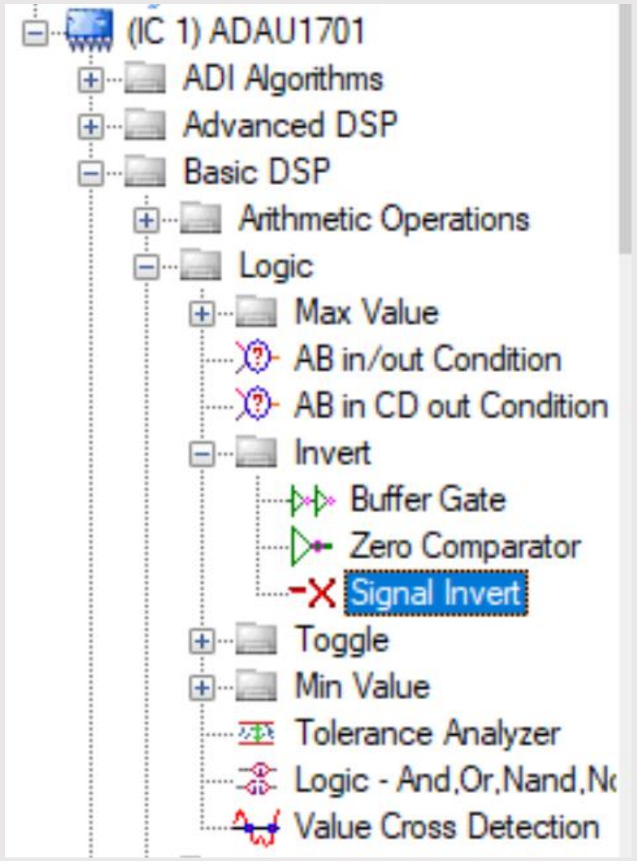


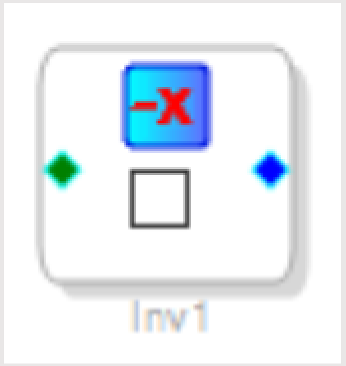




Signal Invert (Phase Invert)

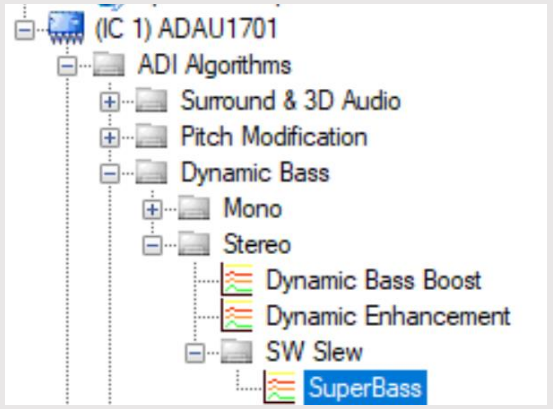
- The Signal Invert block can be useful for quickly inverting the phase of a speaker connected to your KABD amplifier. Simply click in the check box.
- Phase inversion can be useful when measuring speakers, such as checking for reverse null.
  - For example, when a tweeter and a woofer are phase aligned very well, when you invert the phase of one of the drivers, you should see a deep null in the measured frequency response at your crossover frequency.
- This can also be useful if you are not quite sure if you hooked up your speakers right. Perhaps you invert the phase of one of your drivers and everything sounds much better.
  - This likely indicates you hooked up one of your drivers out of phase. The nice thing about DSP is that you can leave the phase inverted and you don't *have* to physically fix it (but maybe you should anyway!).
  - It could also indicate the phase relationship between your drivers is not good to begin with. We recommend using a calibrated measurement microphone while designing speakers to help prevent this. Ultimately you can experiment with the delay block to improve the phase relationship and also your crossover point.

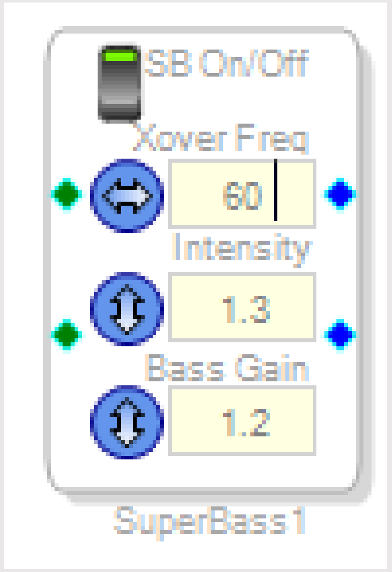




Bass Processing Algorithms – Super Bass

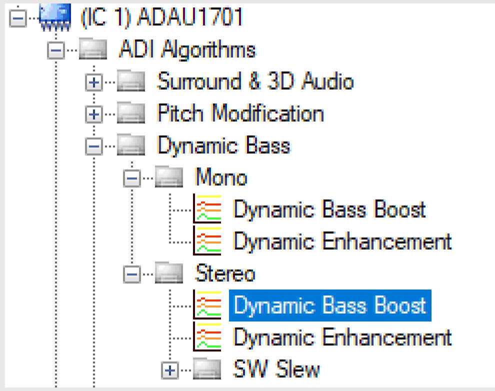
- The super bass algorithm from Analog Devices uses principles of psychoacoustics to generate harmonics above the chosen crossover frequency. This can have the effect of increasing the perceived amount of bass.You will likely find this effect to be desirable for some genres and speakers, but less desirable for others, so make sure to test it thoroughly at different volumes and with different genres of music before finalizing the settings.
- This algorithm only comes in a stereo form.
- **Xover Freq** - Harmonics will be generated from the incoming signal below this crossover frequency. You will want to set this near the low end of where your speaker can play well, but experiment with this value to get the results you like.
- **Intensity –** Can be set from 0.1 to 3, and will affect the intensity of the generated harmonics.
- **Bass Gain –** Can be set from 0.1 to 3, and will apply a gain directly to the bass signal below your chosen 'Xover freq'

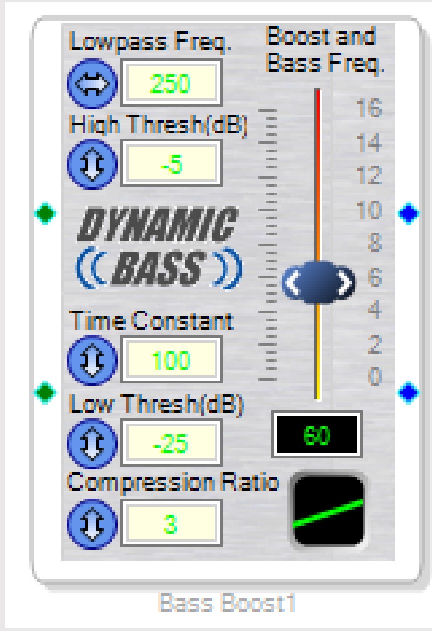

(IC 1) ADAU1701
ADI Algorithms
Surround & 3D Audio
Pitch Modification
Dynamic Bass
Mono
Stereo
Dynamic Bass Boost
Dynamic Enhancement
SW Slew
SuperBass


SB On/Off
Xover Freq
60
Intensity
1.3
Bass Gain
1.2
SuperBass1

Bass Processing Algorithms – Dynamic Bass Boost

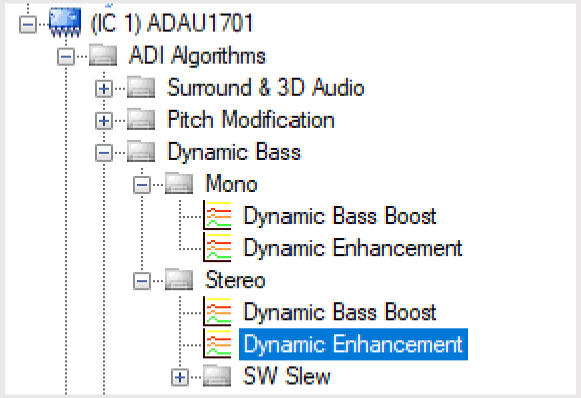
- Analog Device's Dynamic Bass Boost Algorithm will provide more bass boost at lower input levels than it will at higher volumes. This type of boost is incredibly common in portable speakers, creating the effect of the speaker sounding full and bass-y at low volumes, but prevents overdriving the speaker at higher volumes. If this block is too confusing or in depth for you, see the next page for a simpler version.
- **Low Pass Freq** – Frequencies below this point are used for determining the amount of boost.
- **High Threshold (dB)** – Signals higher than this threshold will not influence the boost calculation
- **Time Constant –** Changes attack and release rates when the algorithm is being applied
- **Low Threshold (dB) –** Any signal below this threshold will not influence the boost calculation, and will receive a fixed boost.
- **Compression Ratio –** Controls the rate at which the bass boost changes from the low to high threshold.
- **Boost –** Controls the maximum gain that is applied to the algorithm
- **Bass Freq –** Designates the center frequency for the boosting filter that's applied. So if you have a preferred bass boost frequency, choose it here.



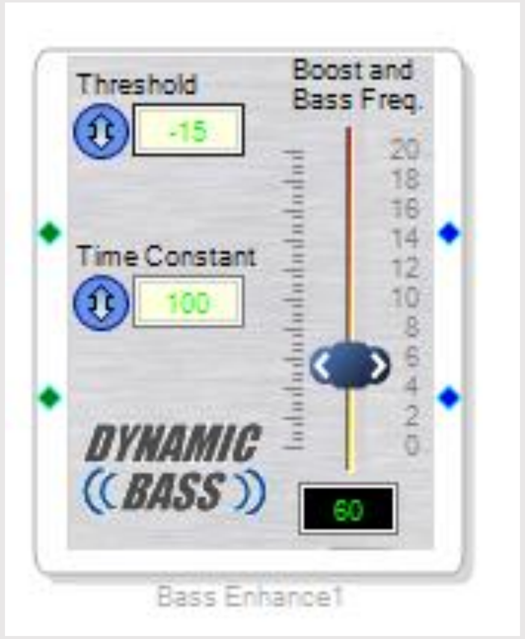


Lowpass Freq. 250
High Thresh(dB) -5
Time Constant 100
Low Thresh(dB) -25
Compression Ratio 3
Boost and Bass Freq. 60

Bass Processing Algorithms – Dynamic Enhancement



- (IC 1) ADAU1701
  - ADI Algorithms
    - Surround & 3D Audio
    - Pitch Modification
    - Dynamic Bass
      - Mono
        - Dynamic Bass Boost
        - Dynamic Enhancement
      - Stereo
        - Dynamic Bass Boost
        - Dynamic Enhancement
      - SW Slew



Threshold: -15

Time Constant: 100

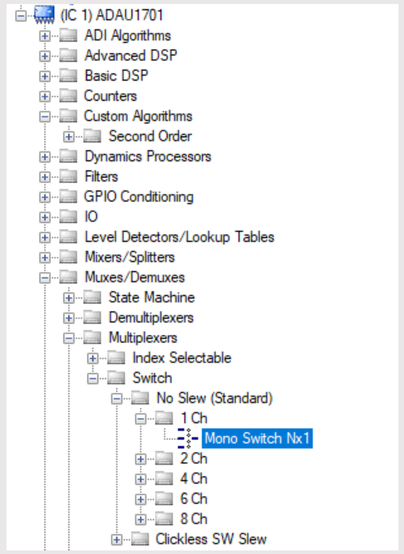
Boost and Bass Freq.: 60

- Very similar to the "Dynamic Bass Boost" algorithm on the previous page, this block will provide higher levels of bass at lower input levels vs. higher input levels, giving the effect of more bass at lower volumes, and less bass at higher volumes, which is very common in portable or otherwise small speakers to protect them when being played loudly.
- **Time Constant** – Attack and release times of the algorithm will become longer as this value is higher.
- **Threshold** – Any signal below this threshold will not influence the boost calculation and will receive a fixed boost.
- **Boost** – Controls the maximum gain that is applied to the algorithm
- **Bass Freq** – Designates the center frequency for the boosting filter that's applied. So if you have a preferred bass boost frequency, choose it here.

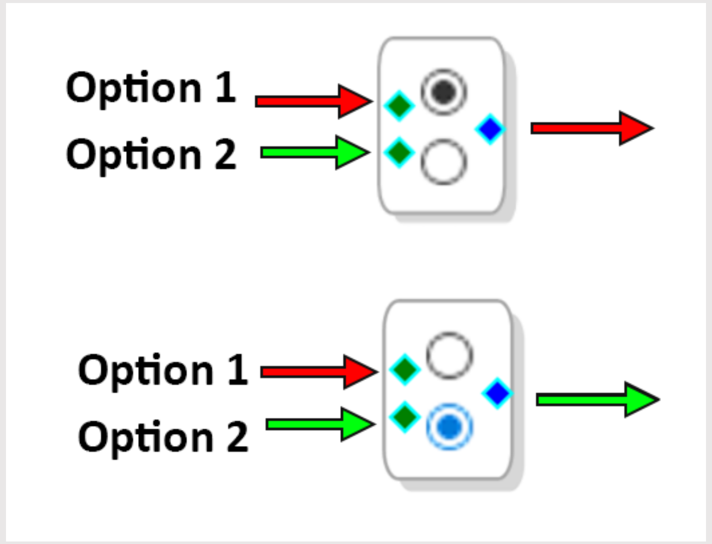
Using buttons and switches to create presets or modes (The multiplexer block)

- In order to use buttons to switch function, change presets, change inputs, etc, it's first important to understand multiplexers, which act as a simple switch in their most basic form.
- The multiplexer block works by supplying its input with multiple options, and selecting which option will come out of the output based on an index selection. We will start small with a basic, click controllable switch, and build up to a stereo, switch controlled multiplexer.

1. The first example of a multiplexer we will use is a simple mono switch, as seen on the right. If upper radial button is enabled, option 1 will pass through the block. If the bottom radial button is enabled, option 2 will pass through. Easy!

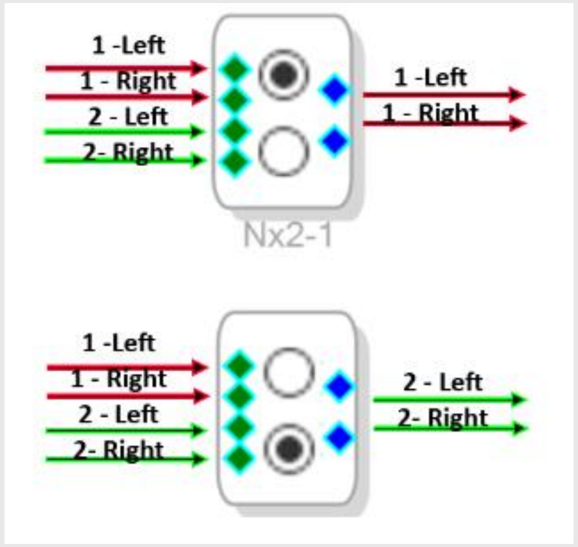


```
(IC 1) ADAU1701
  ADI Algorithms
  Advanced DSP
  Basic DSP
  Counters
  Custom Algorithms
    Second Order
  Dynamics Processors
  Filters
  GPIO Conditioning
  IO
  Level Detectors/Lookup Tables
  Mixers/Splitters
  Muxes/Demuxes
    State Machine
    Demultiplexers
    Multiplexers
      Index Selectable
      Switch
        No Slew (Standard)
          1 Ch
            Mono Switch Nx1
          2 Ch
          4 Ch
          6 Ch
          8 Ch
        Clickless SW Slew
```

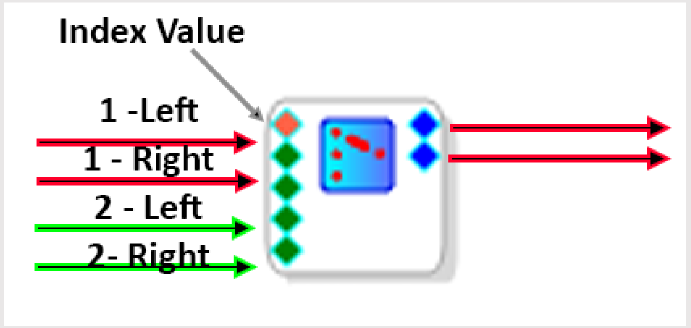


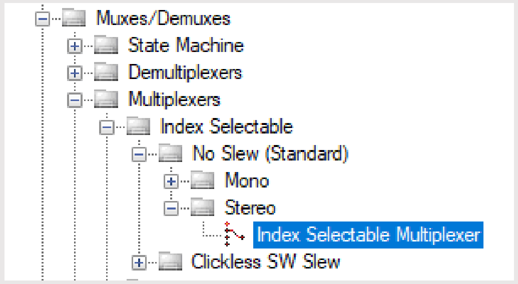
Using buttons and switches to create presets or modes (The multiplexer block)

2. The previous example is only useful for mono signals. We can instead select (or right click and 'grow' the original block) the stereo version of the switch, and pass a stereo signal through the block.
3. This simple type of switch can be useful for A/B'ing different options during programming. However, it loses its usefulness once you disconnect from the computer. To make on the fly switch adjustments or presets, we can control the switch externally using the 'index selectable multiplexer' block, which is shown below. This will let us connect an external switch for control.



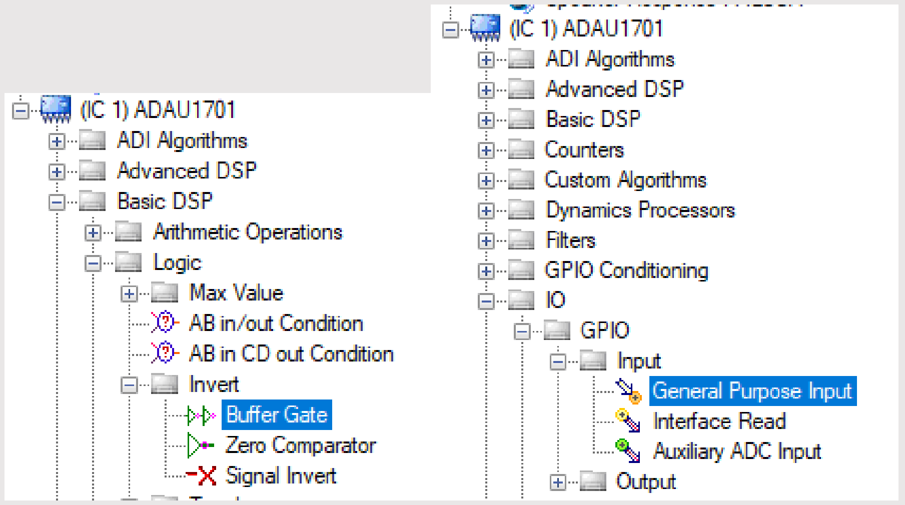
4. On the next slide, a technique to generate an index value based on the position of a switch will be shown, which will allow us to create different presets or modes for our speaker by flicking a switch.

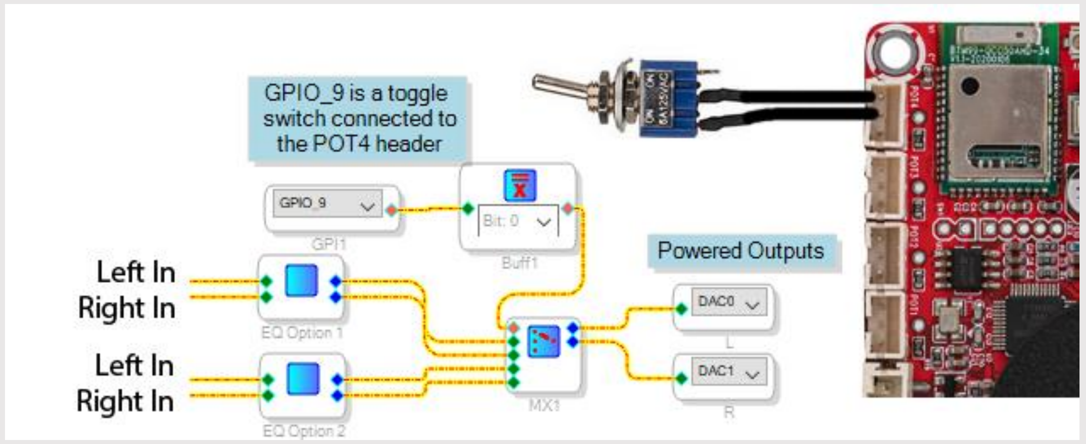




Using buttons and switches to create presets or modes (The multiplexer block)

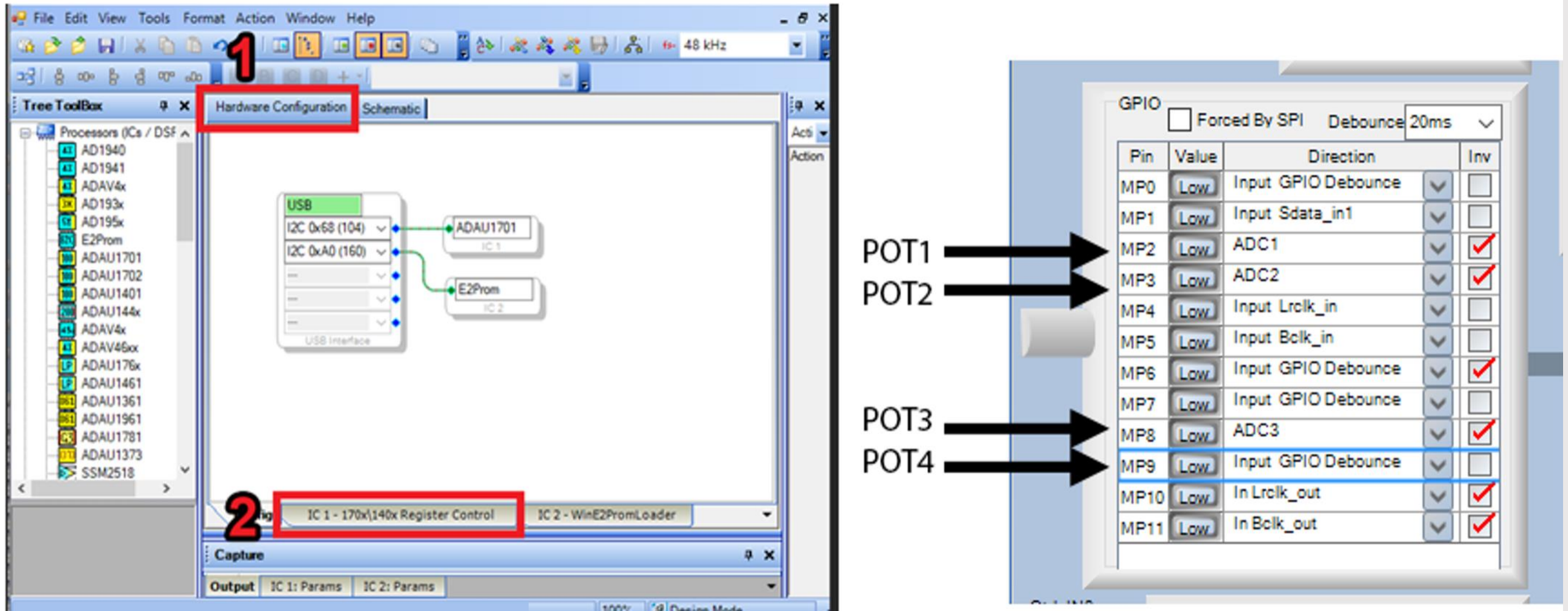
5. In this example, we would like to generate a '0' or a '1' to send to our multiplexer (MX1) block to switch EQ presets.
6. We first need to connect a switch to a GPIO pin of the ADAU1701. GPIO pins are exposed in each POT port (the middle pin). We want to connect the switch between GND and GPIO9 (active low) like the image to the right. Check the specific user manual for the KABD version you are using.
7. We then need to make sure this pin is configured as a general purpose input with debounce, instead of being configured for a potentiometer. See the next page for instruction on this config.
8. Now that we have a switch connected physically, we can use it in our project. First, we drag a "General Purpose Input" block into our project, so we can access the switch.
9. We then connect that input block to a 'buffer gate', which gives us a 0 or a 1 to send to our index pin.
10. This example switches two different EQ blocks options, one that is 'normal', one that is a 'night mode'.





Configuring a POT header port for switches instead of potentiometers

- Configure the GPIO section of the register control menu to ensure correct operation of switches or potentiometers, depending on use case.
- We configured MP9, GPIO9 to be "input GPIO debounce" type to be used in our example from the previous page. We connected a switch to the POT4 port on the KABD.

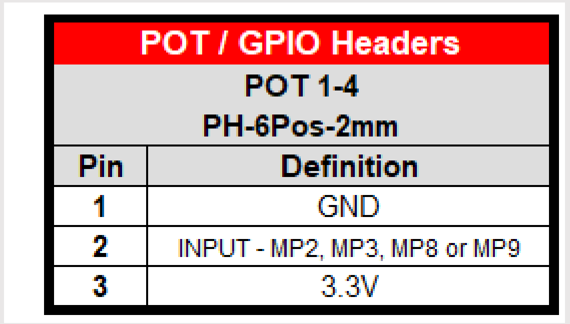


GPIO

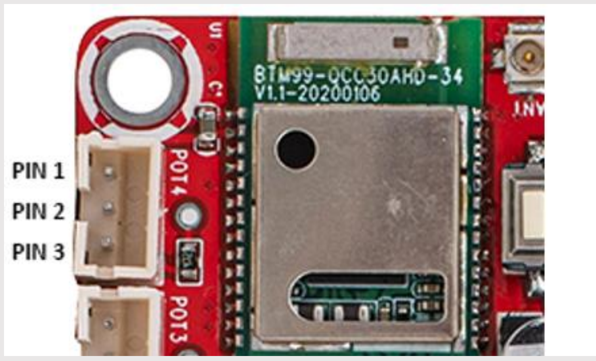
Forced By SPI ☐ Debounce 20ms

Pin	Value	Direction	Inv
MP0	Low	Input GPIO Debounce	☐
MP1	Low	Input Sdata_in1	☐
MP2	Low	ADC1	✓
MP3	Low	ADC2	✓
MP4	Low	Input Lrclk_in	☐
MP5	Low	Input Bclk_in	☐
MP6	Low	Input GPIO Debounce	✓
MP7	Low	Input GPIO Debounce	☐
MP8	Low	ADC3	✓
MP9	Low	Input GPIO Debounce	☐
MP10	Low	In Lrclk_out	✓
MP11	Low	In Bclk_out	✓

"ADC" is for Potentiometers
"Input GPIO Debounce" is for switches

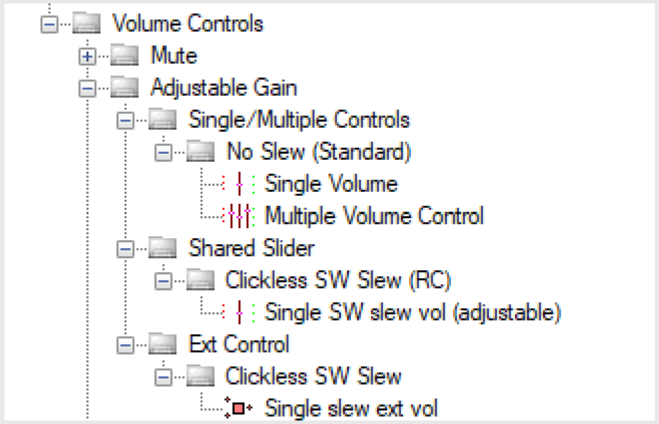


POT / GPIO Headers	
POT 1-4 PH-6Pos-2mm	
Pin	Definition
1	GND
2	INPUT - MP2, MP3, MP8 or MP9
3	3.3V

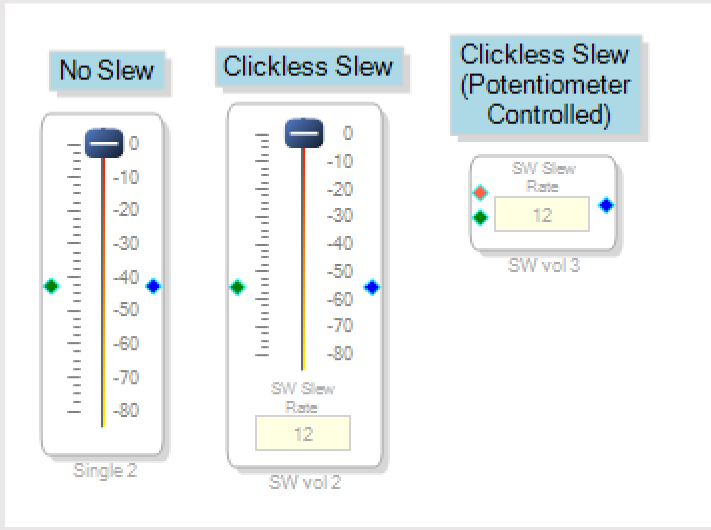


What is ‘slew’?

- In many of the blocks discussed in this guide, many have ‘clickless slew’ or ‘no slew’ versions. The slew versions of a block will additionally have a slew rate. What does that mean? Essentially, it means smoother transitions as your settings change in real time.
- Slew is used to prevent clicks or other transition sounds from occurring when a control is used in real time. A ‘slew’ version of a block will make smaller increments as it transitions, for example, when you turn up the volume. This is at the cost of using more DSP resources, but it is worth it for real time controls. For example, if you want to subtract a fixed 3dB from your signal before it enters a filter block where it gains 3dB, you would use a no-slew block because it will not change in real time. If you are using the volume block as a volume control that will be changed in real time, you should use slew so it sounds correct as you turn up the volume.
- It is for these reasons that you will find that a no-slew version of the external volume control block does not even exist in the toolbox, because it would not sound good as you move the dial!



- Volume Controls
  - Mute
  - Adjustable Gain
    - Single/Multiple Controls
      - No Slew (Standard)
        - Single Volume
        - Multiple Volume Control
    - Shared Slider
      - Clickless SW Slew (RC)
        - Single SW slew vol (adjustable)
    - Ext Control
      - Clickless SW Slew
        - Single slew ext vol

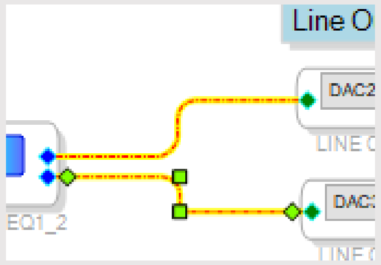


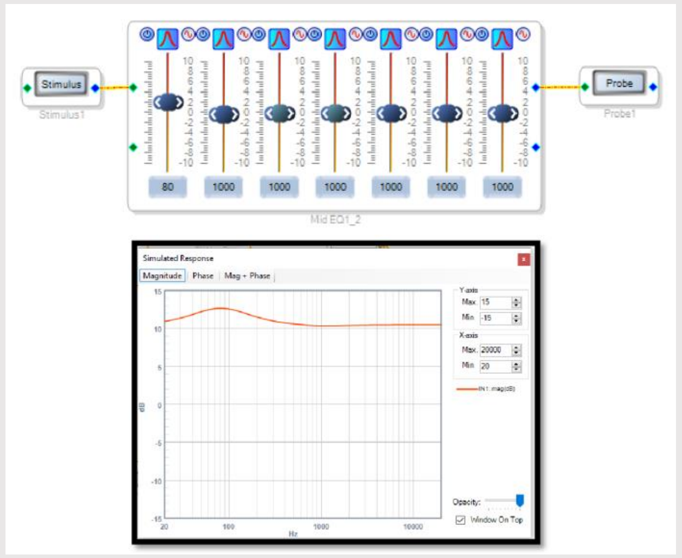
Tips

- If you have never touched anything like the visual block based programming that Sigma Studio uses, it might look daunting at first, but truly it is not too hard to use if you are patient, give it a chance and start with project examples. The learning curve is worthwhile to use once you realize how much flexibility and power SigmaStudio programming can give you.
- Experiment and have fun with it! The possibilities of a KABD amplifier programmed with SigmaStudio are practically endless. The Dayton Audio team has crafted numerous project examples, however mixing parts of them together might be desired as well. This guide a covers many advanced topics, but information about advanced topics even beyond the scope of this guide can be found online, particularly in the SigmaDSP section of Analog Device's Engineer Zone.
- Regularly hover over DSP block's pins for a better understanding of how to use them.
- Carefully read the manual for your specific KABD amplifier, as it will answer questions specific to hardware that is out of the scope of this guide.
- Save often, and save different versions of your projects as your project evolves. Sometimes you might end up trying to over process your signal, and decide you want to go back to an earlier version. Make sure you have something to revert back to!

SigmaStudio Tips

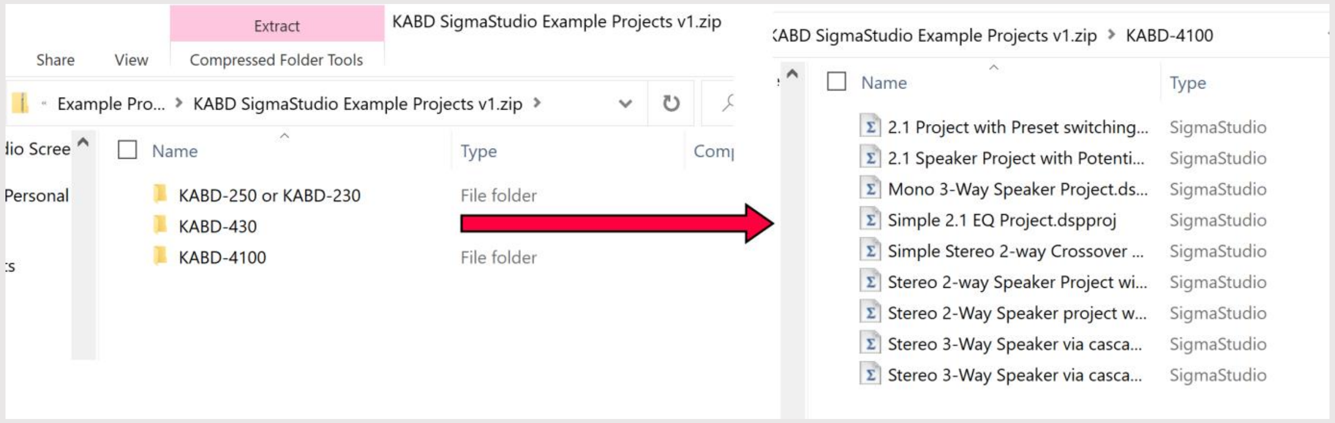
- Look through all of the blocks in the SigmaStudio tool box. This will give you ideas, and show the incredible potential of the software and the ADAU1701.
- Click and drag in the white space of SigmaStudio to create a selection box to select multiple items at once for rearranging projects, deleting or duplicating sections.
- Hold control on your keyboard and drag any block to quickly duplicate it. This also works if you have multiple blocks selected using the method above.
- When you click a wire that is connecting two blocks, you'll see green squares appear. Dragging these squares will allow you to control the wire's route.
- Utilizing stimulus and probe functions allow you to see the response and phase of your project without needing to take measurements! Add stimulus to the input, and probe to the output anywhere in your project, press the "stimulus" button, and then open the "probe" button to view this response.





Example Projects!

- The section that follows shows numerous project examples possible with the KABD amplifier series. When you download the project pack, it will be a zip file. Within that zip file, it contains folders that separate files for the 2 channel KABD-250/230 and the 4 channel KABD-430/4100. There are similar projects in each section, however the charts and labelsin the project match the specific board.
- If parts of the example projects are not wanted (like a superbass block), simply right click and disable it, or delete it and re-route the wires.


Extract
KABD SigmaStudio Example Projects v1.zip
Share
View
Compressed Folder Tools
Example Pro... › KABD SigmaStudio Example Projects v1.zip ›
dio Scree
Personal
s
Com

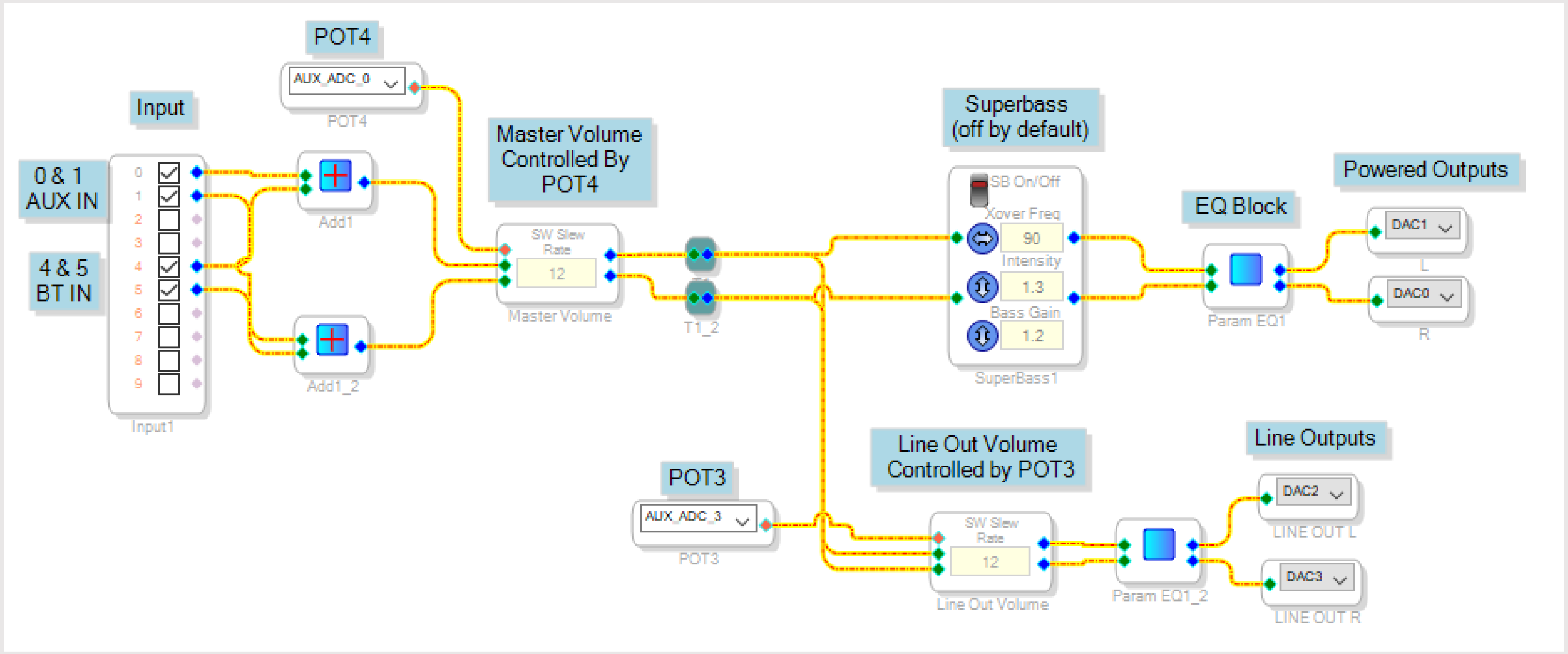
Name	Type
KABD-250 or KABD-230	File folder
KABD-430	
KABD-4100	File folder

KABD SigmaStudio Example Projects v1.zip › KABD-4100

Name	Type
2.1 Project with Preset switching...	SigmaStudio
2.1 Speaker Project with Potenti...	SigmaStudio
Mono 3-Way Speaker Project.ds...	SigmaStudio
Simple 2.1 EQ Project.dspproj	SigmaStudio
Simple Stereo 2-way Crossover ...	SigmaStudio
Stereo 2-way Speaker Project wi...	SigmaStudio
Stereo 2-Way Speaker project w...	SigmaStudio
Stereo 3-Way Speaker via casca...	SigmaStudio
Stereo 3-Way Speaker via casca...	SigmaStudio

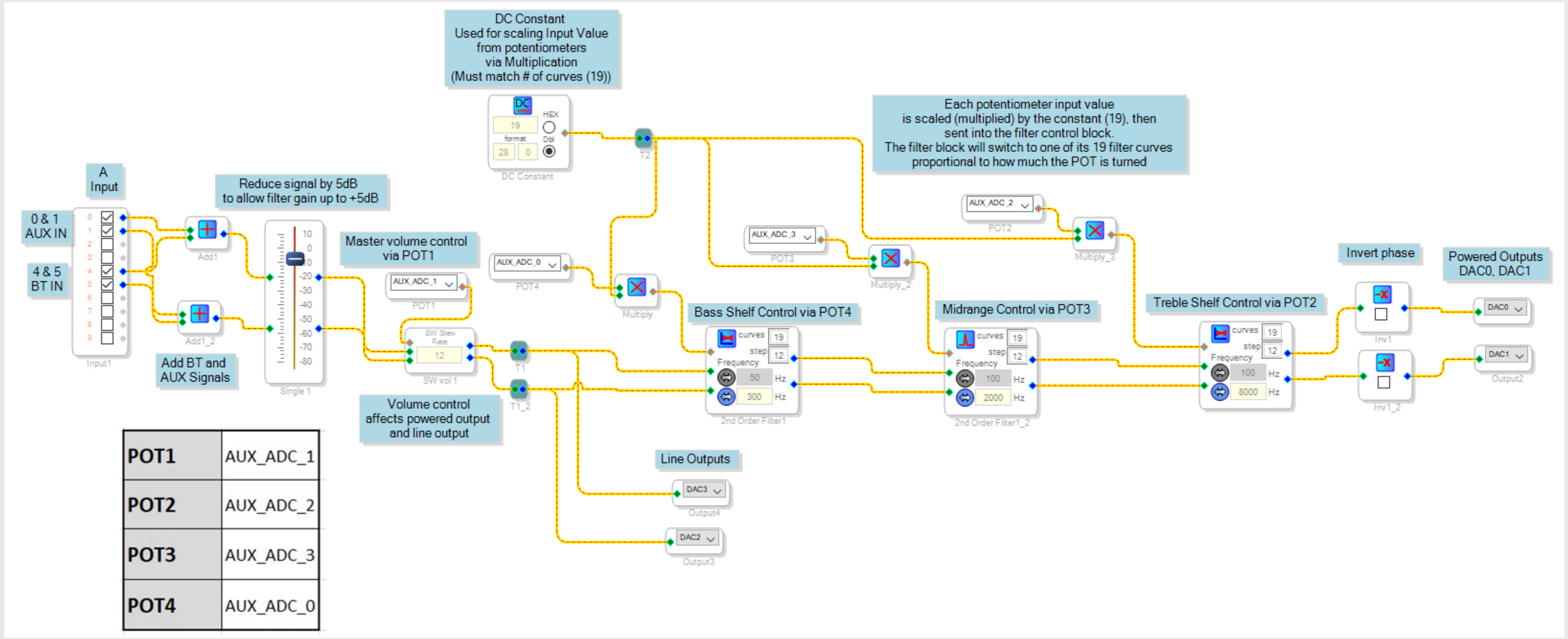
Example – KABD-250/230 - 2 Channel Project with SuperBass Processing

- This simple project features master volume control with a potentiometer connected to POT4, separate line out volume control via, Superbass processing, and full parametric EQ on the powered outputs and line outputs.
- Project Examples –
    - Stereo Boombox w/ DSP controlled full range speakers
    - Converting Passive Bookshelf speakers with their own crossovers into active, DSP controlled speakers with line out for powered subwoofer
    - Hidden sound system or DML speakers with stereo exciters optimized with DSP



Example – KABD-250/230- 2 Channel Project with Potentiometer Controls

- This project is more advanced, but it utilizes all 4 potentiometers available with a KABD amplifier. Ultimately, it allows you to control a bass shelf, a midrange filter, and a treble shelf with individual potentiometers. Even if it looks complicated, it can be easily configured to your liking by clicking the blue boxes in the filter blocks.
- Project Example – Stereo boombox with on the fly adjustment to optimize full range drivers during prototyping phase



DC Constant
Used for scaling Input Value from potentiometers via Multiplication
(Must match # of curves (19))

Each potentiometer input value is scaled (multiplied) by the constant (19), then sent into the filter control block.
The filter block will switch to one of its 19 filter curves proportional to how much the POT is turned

Reduce signal by 5dB to allow filter gain up to +5dB

Master volume control via POT1

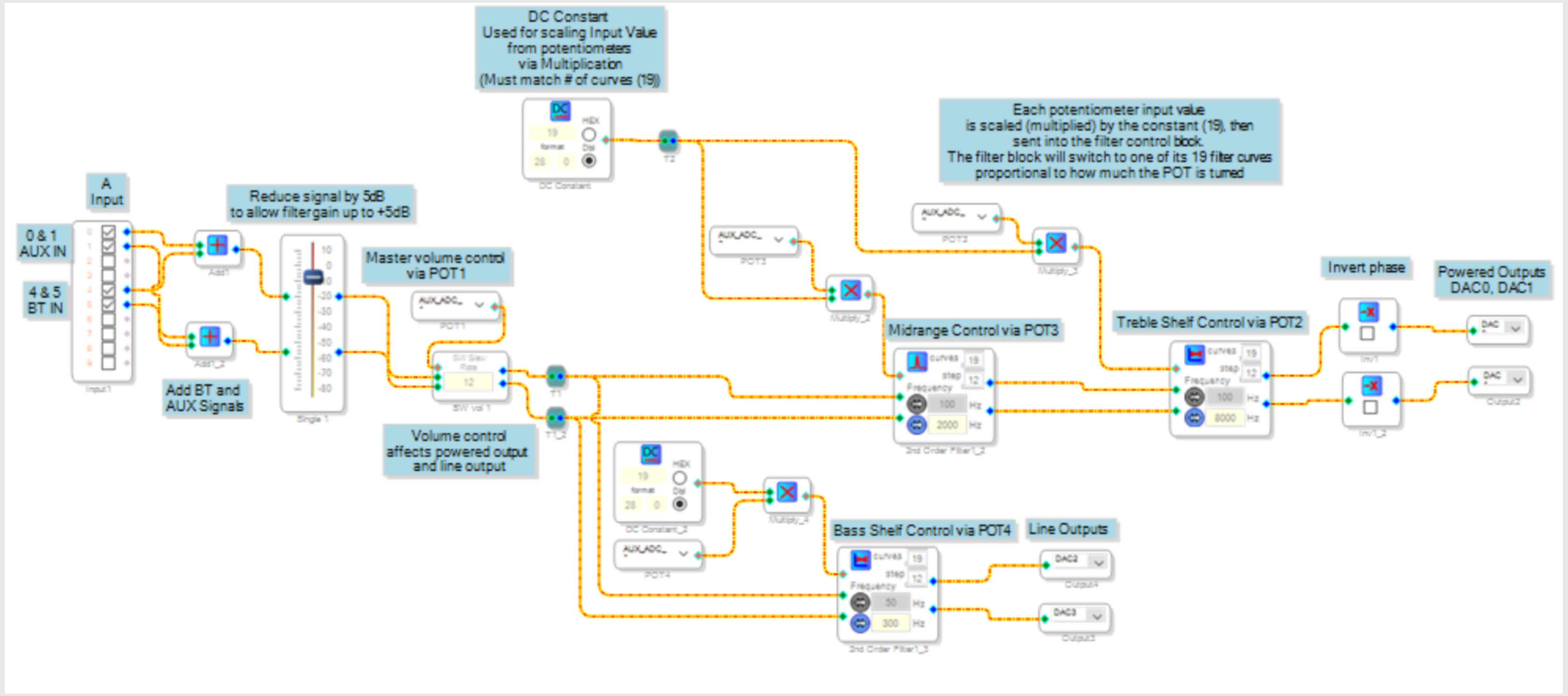
Volume control affects powered output and line output

Add BT and AUX Signals

POT1	AUX_ADC_1
POT2	AUX_ADC_2
POT3	AUX_ADC_3
POT4	AUX_ADC_0

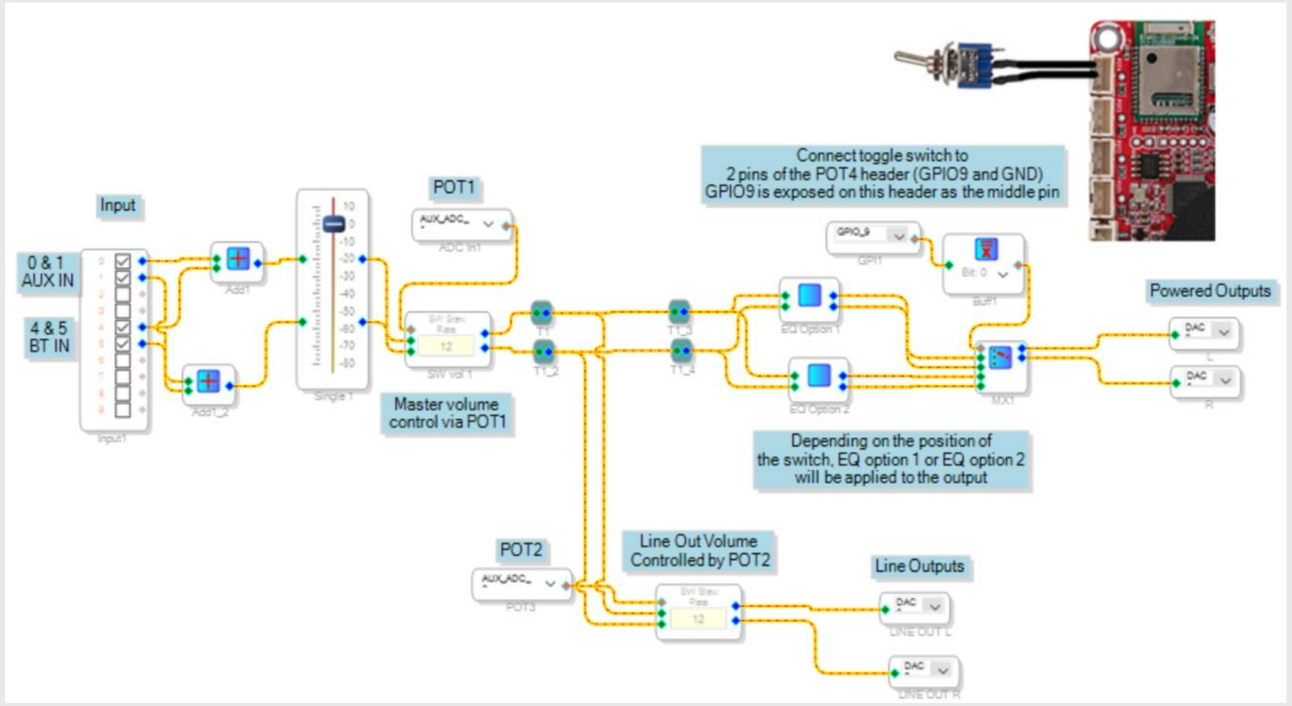
Example – KABD-250/230- 2 Channel Project with Potentiometer Controls and bass control for a connected powered subwoofer via line out

- Very similar to the last example, this program instead takes the bass shelf filter and instead applies it to the line output for controlling the bass of a powered subwoofer. Instead of the midrange filter, we can also can individually control the bass of our connected speakers with an adjustable high shelf filter.
- Project Examples –
    - Stereo boombox with line output for connecting to an external, powered sub and controlling a bass boost.
    - Full range powered desk speakers that connect to an external subwoofer via line out


DC Constant
Used for scaling Input Value
from potentiometers
via Multiplication
(Must match # of curves (19))
Each potentiometer input value
is scaled (multiplied) by the constant (19), then
sent into the filter control block.
The filter block will switch to one of its 19 filter curves
proportional to how much the POT is turned
A
Input
0 & 1
AUX IN
4 & 5
BT IN
Reduce signal by 5dB
to allow filter gain up to +5dB
Master volume control
via POT1
Add BT and
AUX Signals
Volume control
affects powered output
and line output
Midrange Control via POT3
Treble Shelf Control via POT2
Invert phase
Powered Outputs
DAC0, DAC1
Bass Shelf Control via POT4
Line Outputs
DAC2
DAC3

Example – KABD-250/230- 2 Channel Project EQ Preset Switching via a toggle switch

- This example utilizes an attached toggle switch to switch between different EQ presets on the fly. It is configured to have a ‘normal’ mode and a ‘night mode’ (night mode reduces the bass, increases the vocal range), but it could easily be configured for any other purpose.
- Project Examples –
  - Stereo boombox which is sometimes played loud with maximum bass during the day, but needs less bass at night to avoid disturbing neighbors or sleeping family.
  - Desktop speakers built with full range drivers with a toggleable ‘normal’ mode or an ‘enhanced’ mode which could boost vocals, footstep sounds in competitive first person shooters, etc.


Input
0 & 1 AUX IN
4 & 5 BT IN
POT1
Master volume control via POT1
Connect toggle switch to 2 pins of the POT4 header (GPIO9 and GND) GPIO9 is exposed on this header as the middle pin
Powered Outputs
EQ Option 1
EQ Option 2
Depending on the position of the switch, EQ option 1 or EQ option 2 will be applied to the output
POT2
Line Out Volume Controlled by POT2
Line Outputs