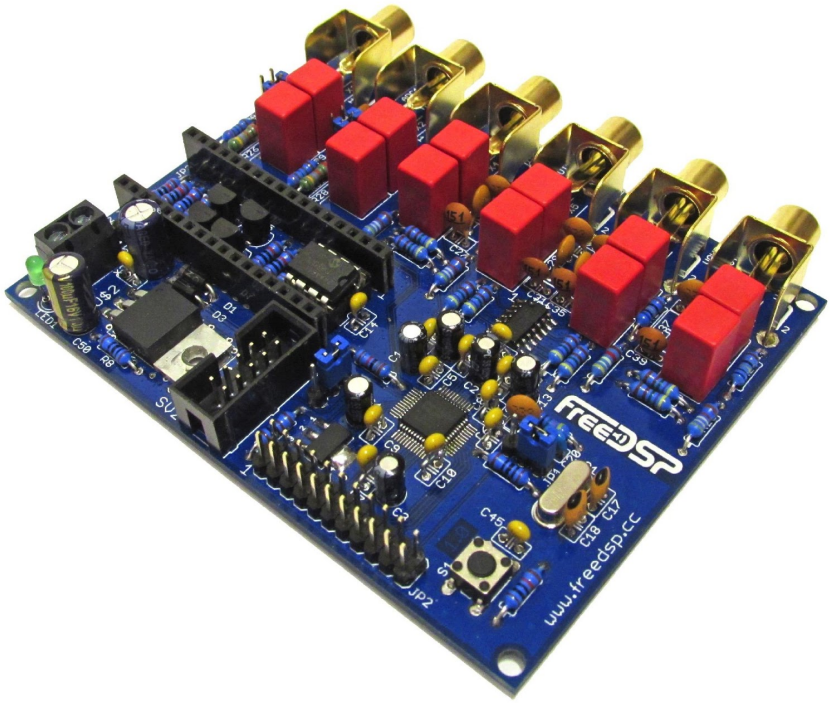
Getting started with the freeDSP CLASSIC



Revision history

Revision	Description	Date
V 7.1	Conversion to Google Docs + new revision history	06 Nov 2015
V 7.2	Added link to forum and note for new standard expansion connector	11 Feb 2016
V 7.3	Added comment how to download Git files on page 12	06 Sep 2016

Table of contents

About the freeDSP

The freeDSP is a cost-effective real-time audio signal processing solution for researchers and the do-it-yourself community. It is a bare circuit board that can be incorporated into your own projects. It comes with no housing. Easy assembling and simple programmability are the main focus. It is based on Analog Devices' ADAU1701 DSP chip together with the free graphical development environment SigmaStudio. The programming model is function-block based – comparable to other graphical programming languages like PureData or Max/MSP. Many prebuilt blocks (e.g., filters, compressors, effects, or logic) can be placed in the signal path via drag and drop. If the included libraries do not have the functions needed, low-level blocks, such as multipliers and delays, can be wired together to create custom algorithms. For more information please refer to the Analog Devices website.

The freeDSP board offers two analog inputs and four analog outputs. Compared to other DSP solutions with easy programmability, the freeDSP and SigmaStudio offer a much wider range of DSP processing options and interface controls. It can be used in various audio applications, e.g.:

- Room compensation / system equalization
- Digital crossovers in active loudspeaker concepts
- Multiband dynamics processing
- Delay compensation / phase shift
- Bass enhancement
- Subwoofer integration
- Advanced instrument audio effect units
- Stereo image widening
- …

Important information

The freeDSP board might generate signals that may damage your audio equipment. **Please read and understand this manual** before starting to work with your board**.** Adjust all hardware settings and configure your software before connecting any audio equipment to the freeDSP. Always start with low volume on your amplifier and slowly increase the level to reduce the risk of damaging your audio system.

The freeDSP is provided to you 'as is'. We make no express or implied warranties whatsoever with respect to its functionality, operability, or use, including, without limitation, any implied warranties of merchantability, fitness for a particular purpose, or infringement. We expressly disclaim any liability whatsoever for any direct, indirect, consequential, incidental or special damages, including, without limitation, lost revenues, lost profits, losses resulting from business interruption or loss of data regardless of the form of action or legal theory under which the liability may be asserted, even if advised of the possibility or likelihood of such damages. Features and specifications might change without prior notice.

Please keep in mind that freeDSP is an open-source spare-time project. Because the freeDSP is very flexible, many applications are possible. Questions and new ideas can be discussed online with other DIYers. Please use the 'Digital Line Level' subforum @ diyAudio.com to connect with other people working with the freeDSP. Please create individual threads for your topics. Link these threads in the **freeDSP main thread** - so that others can find them. Some questions can be answered by carefully reading this manual. **We cannot provide individual support via email.** Thank you for your understanding!

Overview

Figure 1 shows the key features and **default jumper settings** of the freeDSP board. E.g., the input sensitivity can be selected for each channel with jumpers. Try to get maximum output level, but avoid clipping of the audio signal. The output signal is always max. 0.9 V RMS. The following sections will guide you through the freeDSP workflow. For advanced information (e.g., pinout of the expansion connector, …) have a look in the Appendix.

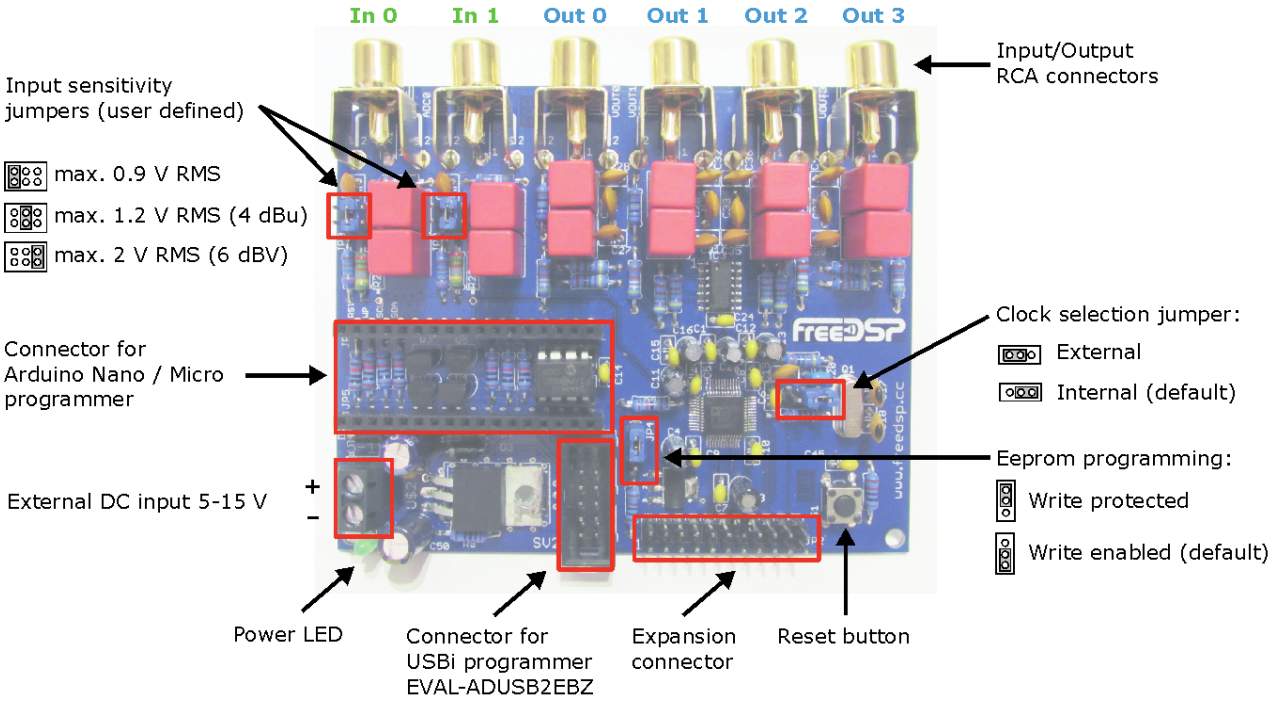


Figure 1 - Overview of the freeDSP board V1.0.

How to get the freeDSP up and running

The following steps will guide you through the workflow to get your freeDSP up and running. Everything starts with ordering all necessary components **and** a programmer. After soldering your board, the needed software is explained. Finally, your program needs to be transferred to the board, fine-tuned and enjoyed.

Get everything needed

We are trying to offer **a freeDSP DIY kit** of parts **including all components** (through hole) **and the printed circuit board with the DSP preassembled** via the website. You will need a soldering iron plus some soldering experience to assemble the remaining through hole components. However, we think it is feasible for everyone. We cannot offer a completely assembled version of the freeDSP. Note that the programmer [freeUSBi or Arduino or the standard USBi adapter from Analog Devices (EVAL-ADUSB2EBZ)] is not included in the kit. You can find a separate documentation with instructions for building the **freeUSBi programmer** on our website.

a. **Order the freeDSP kit.** Unfortunately, no kits are available at the moment due to lack of time. Sorry. ~~To reduce the costs, we offer *centralized buying* of all parts. We collect orders until the magic number of 20 is reached. Then, the boards and parts will be ordered, SMD-parts will be soldered onto the boards (professional reflow-oven) and the kits will be put together and prepared for shipping. Expect your kit to arrive some weeks after ordering. Please be patient and keep in mind that freeDSP is a spare-time, non-commercial project. Payment has to be done in advance.~~

~~*Alternatively,*~~ You can manufacture your own circuit board or modify and extend the original design. This might be the faster option, but needs more expertise (and money) to get up and running.

(a.) **Alternatively,** manufacture the printed circuit board and order all parts yourself.
You might want to locally organize centralized buying and board production together with some friends. You can find the necessary EAGLE files of the board on the freeDSP website www.freeDSP.cc. You will also need to order all electronic parts. Most through hole parts are available from Reichelt using this shopping basket. The DSP (ADAU1701JSTZ-ND), the OP AMP (AD8608ARZ_ND) and one ceramic capacitor (56nF, 100V) can be ordered via DigiKey. Alternatively, Eric send us his almost complete Digikey BOM he used, which *"doesn't include RCA connectors (I couldn't find compatible RCA connectors from a US supplier) and it also isn't very well cost-optimized -- the total BOM comes to ~$80 including the ADAU1701, opamp, and memory. It might still save someone time for someone looking to self-buy."*
The trickiest part is soldering the DSP, which comes in an SMD packaging. We made a video to show the soldering process (currently only in German language). However, this is definitely not recommended for soldering beginners! If you are in doubt of your skills better go for the freeDSP kit with preassembled SMD parts.

Every freeDSP user will need a programmer, an external power supply (5 – 15 V DC) and a Windows PC plus the programming software. All of these are not included in the kit.

b. Install **SigmaStudio** (free) on your PC as described below and order or build a **programmer**. We highly recommend using our freeUSBi programmer or the original USBi programmer EVAL-ADUSB2EBZ. This enables interactive modification of your program on the DSP from within SigmaStudio. The original USBi programmer can be bought directly from Analog Devices for 80 $. You will need to buy the programmer only once. However, it is also possible to do the programming using an Arduino Nano or Micro (see below).
c. Finally, you will need a **power supply** (DC 5-15 V) for your board and some **cables** to connect your equipment.

Solder the board

All unsoldered components are through hole, so they can be soldered with basic soldering knowledge. You can find the part list and the assembly print in the appendix.

Download and install SigmaStudio

To configure the freeDSP you will need a Windows computer running SigmaStudio. It can be downloaded from the Analog Devices website with no charge. However, you will have to create an account.

Create a new project

After installing, you need to create a new project in SigmaStudio as shown in Figure 2.

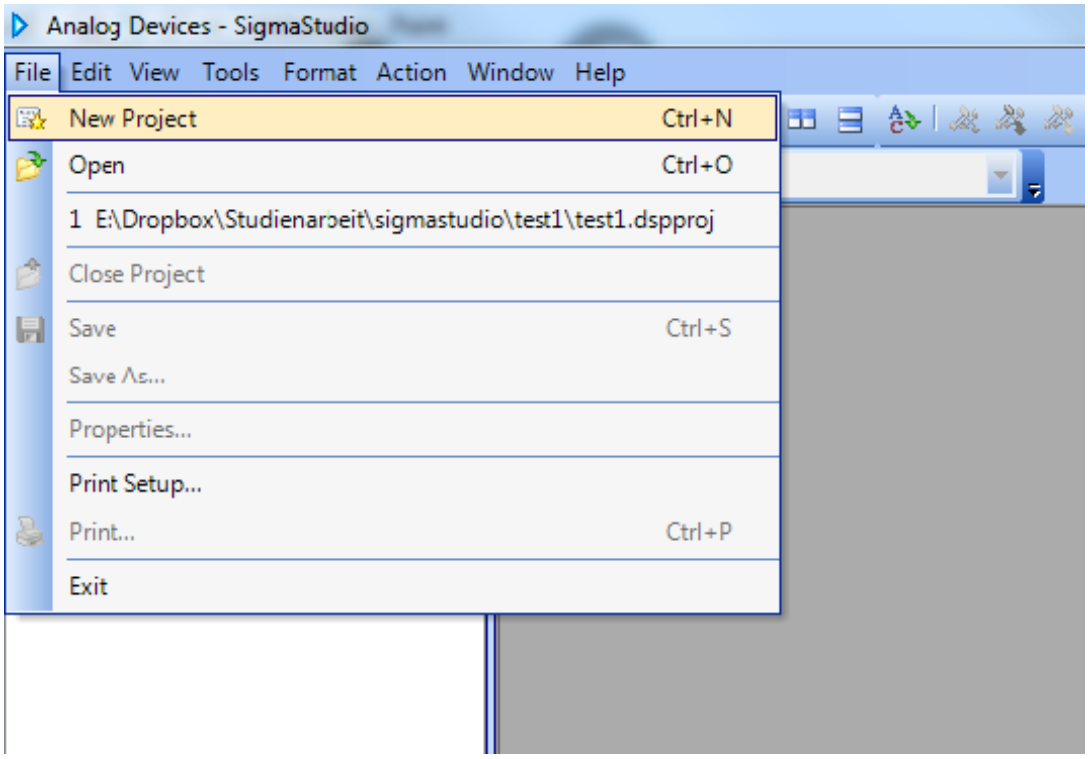


Figure 2 - Create a new project in SigmaStudio.

Now drag and drop three components from the left column into the hardware configuration window at the right (see Figure 3 and Figure 4). You will need the DSP called ADAU1701, the EEPROM (electrically erasable programmable read-only memory) E2PROM and the USBi programming adapter.

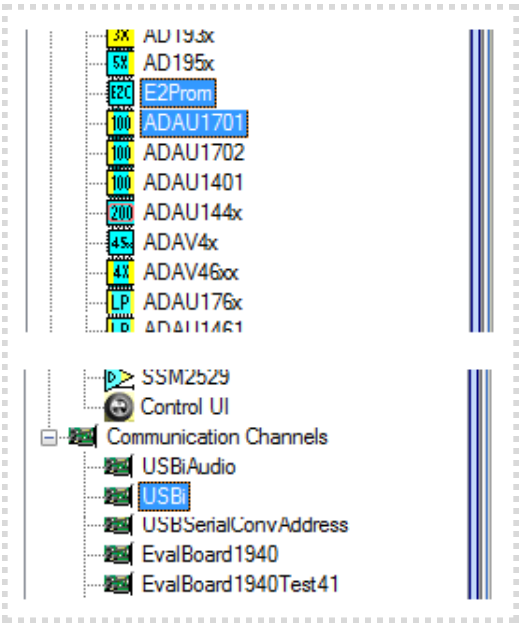


Figure 3 - Necessary hardware components.

Then **wire** the blocks as shown in Figure 4.

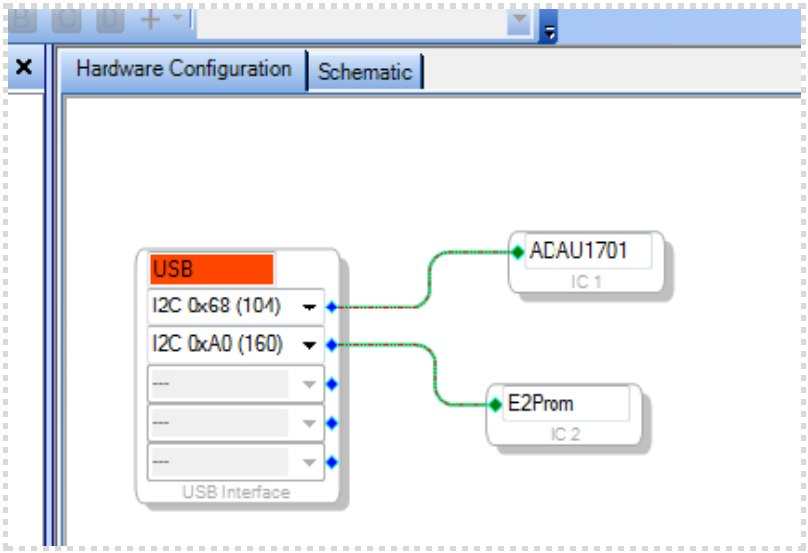


Figure 4 - Wiring of the hardware blocks in SigmaStudio.

Create a schematic

To create the signal processing schematic, which is supposed to run on the DSP, change from the 'Hardware Configuration' tab to the 'Schematic' tab. The menu on the left will now show

all available processing blocks, structured into various categories. First, drag and drop the audio input block and two audio output blocks into the schematic. The inputs can be found via ‘IO/Input/Input’ and outputs via ‘IO/Output/Output’.

To get started, we want to implement a simple 2-way crossover filter. Therefore, drag the corresponding block into your schematic: ’Filters/Crossover/Single Precision/2-Way/Crossover’. Configure the modules as shown in Figure 5.

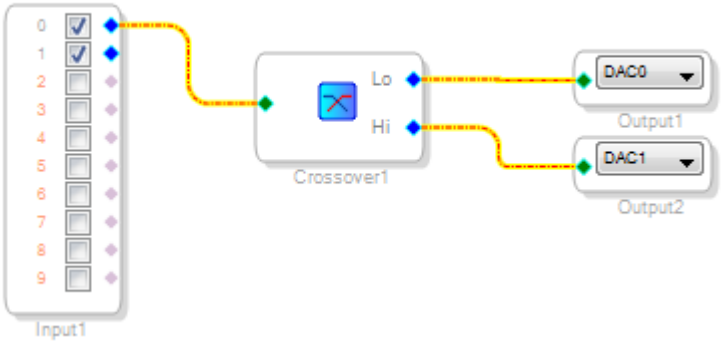


Figure 5 - Simple 2-way crossover filter with one audio input and two outputs.

By clicking the blue crossover symbol, the settings menu for the filter appears as shown in Figure 6. If you are using the USBi programmer, you can change all parameters like filter types or corner frequencies in real-time. This way, you can hear the resulting modifications immediately.

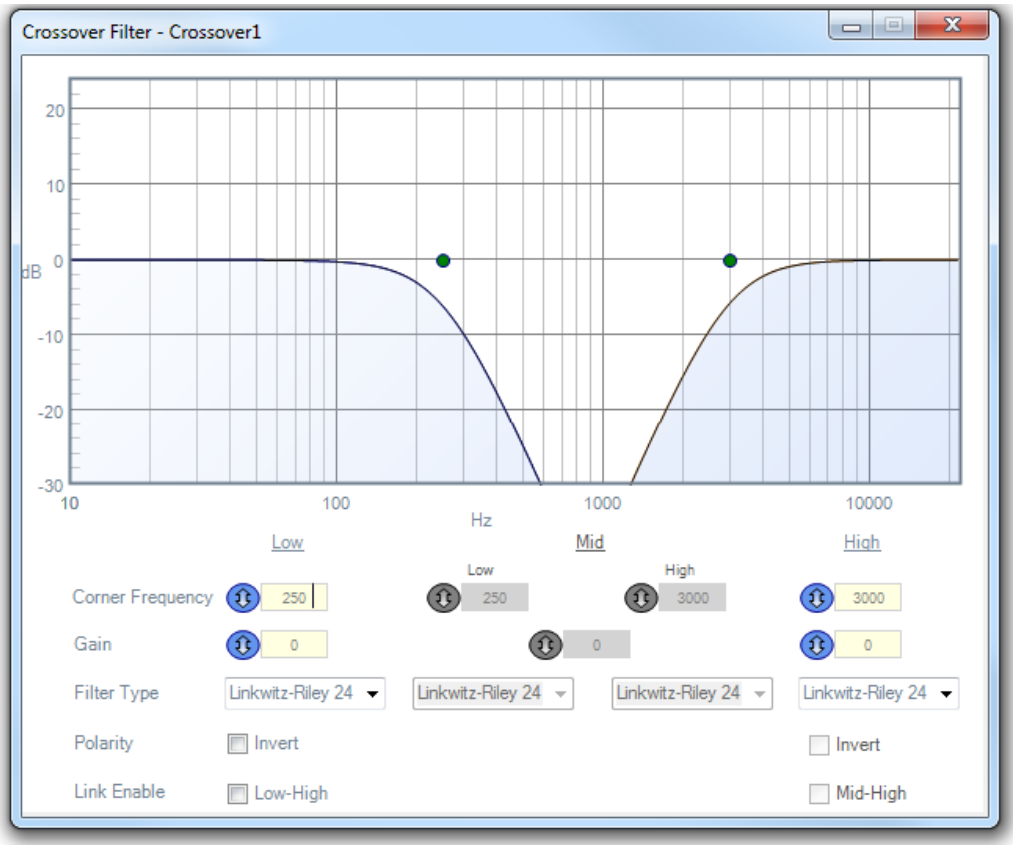


Figure 6 - Settings window for a 2-way crossover filter.

Setting-up operation

The final program needs to be compiled and transferred to the DSP. Therefore, a suitable programmer is needed. To offer real-time programmability the USBi adapter from Analog Devices (EVAL-ADUSB2EBZ) is recommended, which is currently sold for approximately 80 $. This is the most comfortable way of working with the freeDSP. If the application is intended to run the board only in self-boot mode, a cheaper programmer (e.g., an Arduino) can be used.

Transfer your program using the freeUSBi or USBi programmer

Connect the freeUSBi programmer or USBi programmer to your freeDSP. Now, connect the adapter to a free USB port on your computer. Then, click ‘Link Compile Download’ as illustrated in Figure 7.

Figure 7 - The 'Link Compile Download' button in SigmaStudio loads the schematic into the freeDSP.

That’s it. Your freeDSP is now powered over USB and ready to be wired up with your audio equipment. For operation with a different power supply, see the section Operate without computer connection below.

Transfer your program using an Arduino programmer

Alternatively, the program can be written to the onboard EEPROM and booted in stand-alone mode. This offline programming can be done using a microcontroller, e.g. an Arduino Nano or Micro. The Arduino should be aligned with the outer edge of the freeDSP board. With the Nano, some of the plugs in the middle of the freeDSP remain unused - see Figure 8.

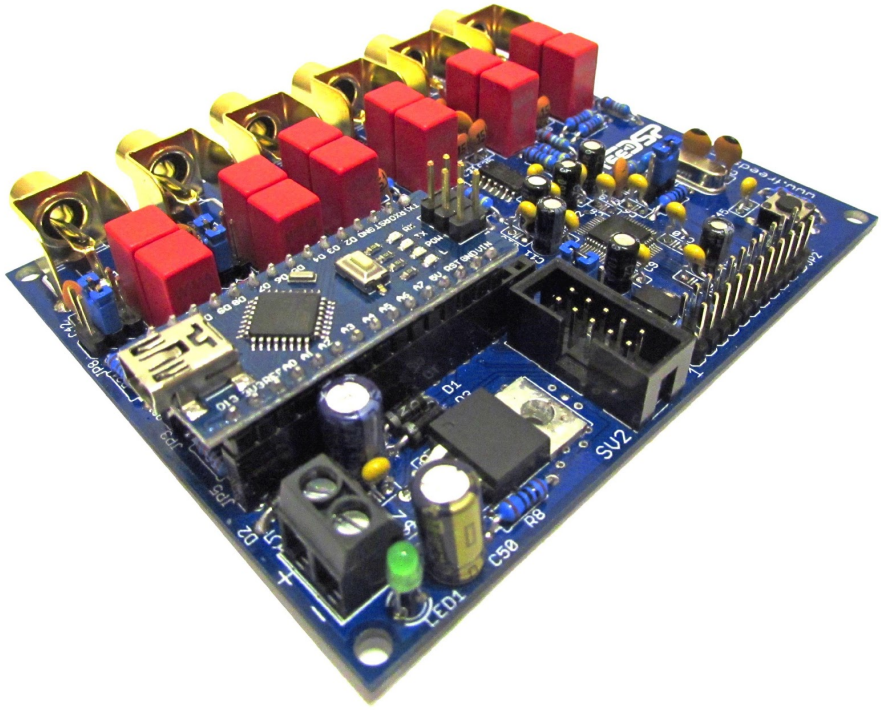


Figure 8 - The freeDSP board V1.0 with mounted Arduino Nano for programming.

The following description will guide you through the necessary steps to setup the software:

1. Download and install the Arduino software including the Arduino drivers.
2. Connect your Arduino. After Windows has automatically updated and installed the drivers, you will find your Arduino here: Start/Control Panel/System/ Hardware/Device Manager/Ports (COM & LPT). Please note the number of the COM port. (If Windows does not install the drivers automatically, please have a look in the "Install the drivers" section of the Arduino guide. If you own an Arduino with a CH340 chip, you might need to install the CH340 drivers manually. If you are running Windows 8.1 the CH340 driver might install automatically.) The following steps require that bootloader is installed on your Arduino, which is the default case when you buy a new board.
3. Download and unzip the I2C libraries from GitHub (To download all files go to the parent folder in GitHub and download or clone all files including the I2C libraries!). Copy the folder 'freeDSP_I2C_libraries' to the Arduino library: C:\Program Files (x86)\Arduino\libraries
4. Open the Arduino software and select your Arduino version (Tools/Board/…) and COM port (Tools/Serial Port/…).

5. Download and unzip this Arduino sketch. Open ‘freeDSP.ino’ and upload it. Close the Arduino software.
6. Download and install Tera Term.

The previous steps need to be completed only once. In the following, you will now need to locate the program text file generated by SigmaStudio and burn it to the EEPROM.

7. Be sure that the EEPROM programming jumper is in the ‘write enabled’ position as labeled in bottom right corner of Figure 1. Please note that the jumper in the photo is in the wrong position for programming!
8. Please check if your Arduino is still connected to the same USB port as during installation. If not, please identify the active COM port following step 2.
9. Open Tera Term, select ‘serial’ in the pop-up window and choose the active COM port connected with your Arduino as shown in Figure 9. (~~Set the baud rate to 19200 in the settings of your serial port.~~ The default baud rate in Tera Term is 9600, which is also the default baud rate in the latest Arduino sketch. If you are an advanced user, you can manually change the baud rate in the Arduino sketch and Tera Term. We have successfully tested 19200.)
10. You should get a welcome message from your Arduino ‘… eeprom found … waiting for E2Prom.Hex file…’
11. Open your SigmaStudio project file (.dspprog) from your project folder and click ‘Link Compile Download’ as illustrated in Figure 7. Ignore the possible ‘Communication Failure’ message.
12. Change to the ‘Hardware Configuration’ tab. Right click on the ADAU1701 block and select ‘Write Latest Compilation to E2PROM’ as shown in Figure 11.
13. Now switch back to Tera Term and click ‘send file’. Navigate to the subfolder IC2 in your project folder. You will find a file called E2Prom.hex. Select it and confirm. A loading bar will show you the progress of the file transfer.
14. You are done. If you only see zeros in the terminal, please check if the programming jumper was in the ‘write enabled’ position.

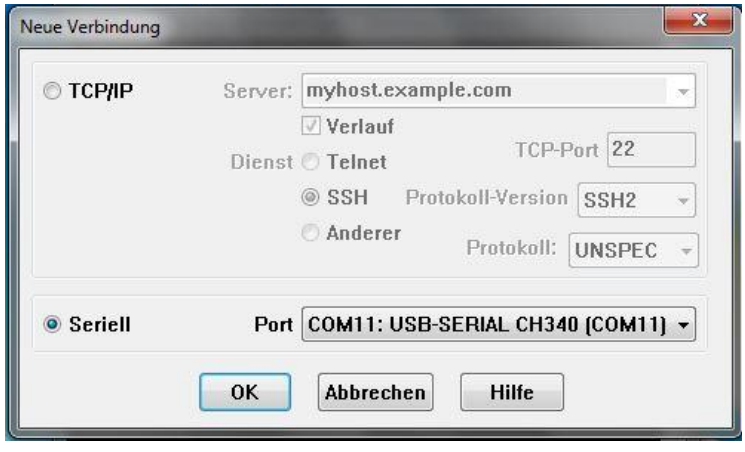


Figure 9 – Choose COM port in Tera Term.

Transfer your program using an EEPROM programmer

As a third option, a dedicated I^2C EEPROM programmer can be used, which supports the EEPROM mounted on the board. However, this option is not recommended.

Connect your audio equipment and adjust your program

After transferring your program, connect your audio equipment to the analog inputs and outputs via the RCA jacks. Be sure that your **audio equipment is powered off** when making these connections. Please avoid twisting the RCA connectors – just **push and pull** the plugs. Now switch on your connected audio equipment with the volume down low (both source and amplifier). Check if your program (e.g., the crossover) is working as intended, while gradually increasing the volume.

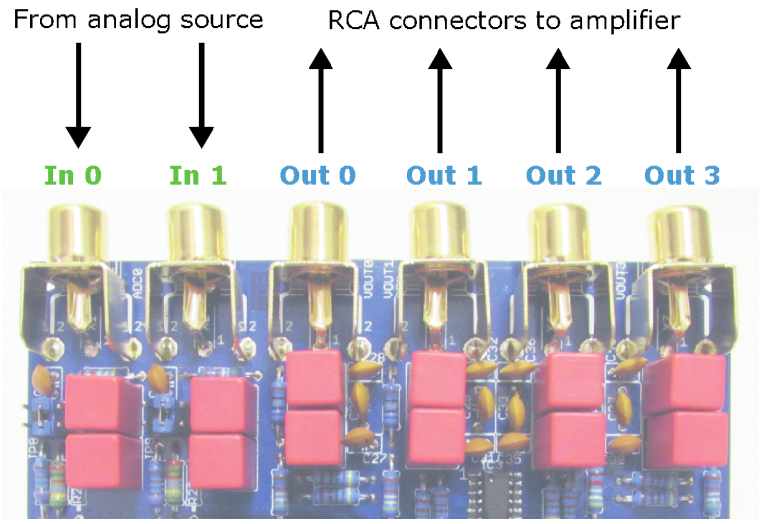


Figure 10 - Analog audio connections

It is highly recommended to use a measurement microphone to tune your program. Various mics exist on the market, e.g. miniDSPs UMIK-1. You will also need measurement software like the Room EQ Wizard (free, Windows), ARTA (Windows) or FuzzMeasure (Mac) or any other software of your choice.

For our digital crossover example above, you would measure each driver individually and in combination. Using these measurements, you can adjust crossover frequency and slopes or add additional equalizers for each driver. **Please save your modifications in SigmaStudio on a regular basis! Note that it is NOT possible to transfer a configuration stored on the freeDSP hardware back to your computer.**

Operate without computer connection

If you are happy with your adjustments, you can continue to operate the freeDSP without a computer. Therefore, you need to write your program permanently in the hardware memory and connect an external power source.

Store your program permanently in the freeDSP memory

To run the freeDSP without a computer connection, the program needs to be stored in the internal memory of the board – the EEPROM. This step is only necessary for USBi programming, because the program will always be written to the EEPROM if you are using an Arduino. Be sure that the EEPROM programming jumper is in the 'write enabled' position as labeled in bottom right corner of Figure 1. Please note that the jumper in the photo is in the wrong position for programming! In SigmaStudio change back to the 'Hardware Configuration' tab. Right click on the ADAU1701 block and select 'Write Latest Compilation to E2PROM' as shown in Figure 11 and confirm the next window. Than right click on the ADAU1701 block again and 'Check Last Compilation vs. E2PROM'. If you now switch to an external power supply or push the reset button on the freeDSP, the board will load the program from the EEPROM. (If you unplug the USBi programmer from your computer, it also needs to be disconnected from the freeDSP board. Otherwise your freeDSP will not start.)

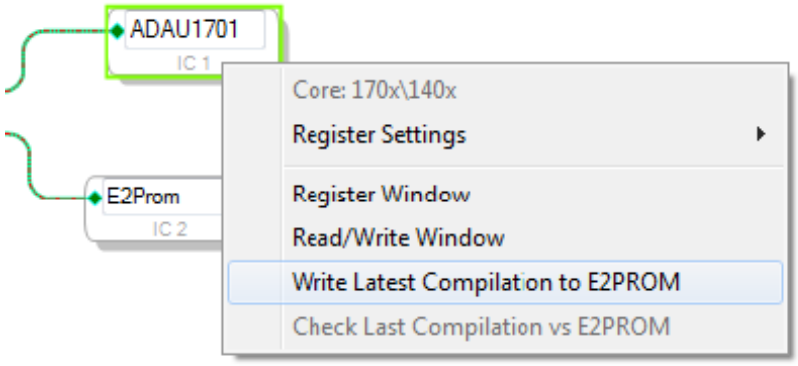


Figure 11 - Store the latest compiled version of your schematic to the freeDSP.

Supply external power

If the freeDSP is no longer connected to a computer, it needs an alternate power source. You can use an external supply (5-15 V DC) to power the freeDSP via the screw terminal (recommended). Alternatively you can use a 5 V DC USB charger. Please note that there is no USB port available directly on the board. However, a USB charger can be connected with the help of an Arduino Nano or Arduino Micro, which is placed in the corresponding freeDSP Arduino connector.

You can connect a DC power supply, the USBi programmer and an Arduino at the same time without any problems.

You might have noticed that there is no power switch on the freeDSP. We recommend keeping it powered on. If you want to power the freeDSP board on, please switch on connected audio equipment (amps, …) **after switching on the freeDSP**. Please ensure that you always power off connected equipment **before disconnecting power from the freeDSP**!

Troubleshooting

1. Please take the time to carefully read this getting started guide and have a look at the online application examples. **Please keep in mind that freeDSP is an open-source spare-time project.** Thank you for your understanding!
2. Maybe your issue has already been discussed in the **freeDSP forum**. There might already be a solution for the problem you are facing. Please use the 'Digital Line Level' subforum @ diyAudio.com to connect with other people working with the freeDSP. Please create individual threads for your topics. Link these threads in the **freeDSP main thread** - so that others can find them.
3. **We cannot provide support via email or the contact form on the website.** We hope that you understand that freeDSP is not a commercial product. It is an **open-source spare-time project**. We hope that questions can be answered together with the freeDSP community in the freeDSP forum. Please be patient and remember that forums work on a voluntary basis. If you post to a **forum**, please provide:
    a. The version of the freeDSP board you are working with. A good resolution photo of your soldered board from top and bottom. If you made some changes to the original design, please describe them in detail.
    b. A detailed and clear explanation of the symptoms you are seeing.
    c. A description of the troubleshooting steps you already performed and the results obtained.

Appendix

Specifications

Table 1 - Specifications.

Digital Signal Processor	ADAU1701 audio DSP with 2 ADCs and 4 DACs integrated
Sampling rate	44.1 kHz / 48 kHz (max. 96 kHz)
Bit depth	24 bit (internal 28 bit single precision/56 bit double precision)
Programming	Graphic oriented programming with SigmaStudio (Windows only) and a USBi programming adapter or an Arduino
Analog audio connections	2 unbalanced RCA inputs / 4 unbalanced RCA outputs
Input range (RMS)	0.9 V – 2 V or more (configurable via jumpers)
Input impedance	7.5 kΩ (max. 0.9 V), 9.9 kΩ (max. 1.2 V), 13.9 kΩ (max. 2 V)
Output range (RMS)	0.9 V (2.5 V p-p)
Output impedance	580 Ω
Dynamic range	98 dB analog input to analog output
Digital audio connections	8 inputs + 8 outputs via I2S through expansion connector (expansion boards necessary, currently not implemented)
General purpose input/output interface	4 auxiliary 8 bit ADC inputs (e.g. for potentiometers) 8 GPIO pins (e.g. for push-buttons or as control outputs)
Dimensions	100 mm x 80 mm x 15 mm
Power supply	5 V – 15 V DC (screw terminals) or USB-powered via Arduino
Power consumption	0.6 W @ 5 V / 1.44 W @ 12 V / 1.8 W @ 15 V

Part list

All parts in the following list are included in the freeDSP kit. However, some additional ordering information is given in case you want to go on your own. There is also a Reichelt shopping basket, which might help to order parts individually. Unfortunately, as far as we know, not all parts are currently available from one supplier (see the last column of Table 2). The DSP (ADAU1701JSTZ-ND), the OP AMP (AD8608ARZ_ND) and one ceramic capacitor (56nF, 100V) can be ordered via DigiKey.

Alternatively, Eric send us his almost complete Digikey BOM he used, which *"doesn't include RCA connectors (I couldn't find compatible RCA connectors from a US supplier) and it also isn't very well cost-optimized -- the total BOM comes to ~$80 including the ADAU1701, opamp, and memory. It might still save someone time for someone looking to self-buy."*

Table 2 - Part list REICHELT.

Part	Description	Label	Qty.	Order number www.reichelt.de
C1, C5, C6, C7, C8, C9, C10, C12, C14, C15, C24, C45, C51	Ceramic capacitor, 100nF	104	13	KERKO 100N
C2, C3, C4, C11, C13, C16	Electrolytic capacitor, radial, 10µF, 16V/50V	10µF	6	SM 10/16 RAD
C17, C18	Ceramic capacitor, 22pF, 63V/200V	22	2	KERKO 22P
C19, C28, C32, C36, C40	Ceramic capacitor, 3.3nF, 63V/50V	332	5	KERKO 3,3N
C20	Ceramic capacitor, 56nF, 100V	563	1	Order via DigiKey / or replace with two parallel capacitors
C21, C22, C23, C26, C30, C34, C38, C41, C43, C46, C47, C48	Foil capacitor, 4.7µF	4,7µF	12	MKS-2 4,7µ
C25, C29, C33, C37	Ceramic capacitor, 470pF, 500V	470	4	KERKO-500 470P
C27, C31, C35, C39	Ceramic capacitor, 150pF, 63V	151	4	KERKO 150P
C42, C44	Ceramic capacitor, 100pF, 50V	101	2	KERKO 100P
C49	Electrolytic capacitor, radial, 47µF, 35V	47µF	1	RAD 47/35
C50	Electrolytic capacitor, radial, 100µF, 6.3V	100µF	1	RAD FC 100/16
D1, D2, D3	Shottky diode, 1N5818, DO41, 30V, 1A	1N5818	3	1N 5818

IC1a	EEPROM, serial, 256kBit, DIP-8	24AA256	1	24AA256-I/P
IC1b	IC socket, DIP8 (for EEPROM)	-	1	GS 8
IC3	AD8608	AD8608	1	Order via DigiKey
JP1, JP4	Multi-pin connector, 1x3, RM 2.54mm	-	2	MPE 087-1-003
JP2	Multi-pin connector, 2x13, RM 2.54mm, needs to be shortened to 2x11	-	1	SL 2x13G 2,54
JP3, JP5	Plug connector, 1x20, RM 2.54mm, needs to be shortened to 1x17	-	2	BL 1x20G8 2,54
JP8, JP9	Multi-pin connector, 2x3, RM 2.54mm	-	2	MPE 087-2-006
LED1	LED, 3mm, green	-	1	LED 3MM 2MA GN
Q1	Quartz crystal, 12.288Mhz, HC49/U-S	-	1	12,2880-HC49U-S
Q2, Q3, Q4, Q5	n-MOSFET 2N7000	-	4	2N 7000
R1, (R34)	Resistor, 1kΩ	brown black black brown brown	2	METALL 1,00K
R2, R26, R28	Resistor, 18kΩ	brown grey black red brown	3	METALL 18,0K
R3, R4, R5, R11, R35, R36, R37, R38	Resistor, 10kΩ	brown black black red brown	8	METALL 10,0K
R6, R8	Resistor, 100Ω	brown black black black brown	2	METALL 100
R7	Resistor, 475Ω	yellow violet green black brown	1	METALL 475
R9, R10, R14, R15, R18, R19, R22, R23	Resistor, 4.75kΩ	yellow violet green brown brown	8	METALL 4,75k
R12, R16, R20, R24	Resistor, 604Ω	blue black yellow black brown	4	METALL 604
R13, R17, R21, R25, R27, R29	Resistor, 49.9kΩ	yellow white white red brown	6	METALL 49,9K
R31, R33	Resistor, 7.15kΩ	violet brown green brown brown	2	METALL 7,15K
R30, R32	Resistor, 10.5kΩ	brown black green red brown	2	METALL 10,5K
S1	Switch, 6x6mm, H: 4.3mm; 12V	-	1	TASTER 3301

SV2	Boxed header, 10 pins	-	1	WSL 10G
T1	Zetex PNP transistor	-	1	FZT 751TA
U1	ADAU1701	-	1	Order via DigiKey
U2	LM1117 +3V3 voltage regulator	-	1	LM 1117 T3,3
X1, X2, X4, X5, X6, X7	Cinch connector	-	6	CBP-G
JP1, JP4, JP8, JP9	Jumper	-	4	JUMPER 2,54 BL
X3	Terminal, 2x, 5.08mm grid	-	1	AKL 101-02

Assembly print

Figure 12 - Board silkscreen and parts placement for freeDSP 1.0.

Schematic

Figure 13 - Schematic of the freeDSP 1.0 in vector format. Please zoom in to see more details.

Pinout of the expansion connector [1]

The expansion connector can be used to connect external potentiometers, buttons, sensors, or similar. In addition, it can be used to connect the freeDSP with external audio devices via I2S (Inter IC Sound). The pinout of this header is shown in the following.

Figure 14 – Pinout of the expansion header.

[1] Note that this connector will be replaced by a new standard connector, which will be used for all future freeDSP boards. The corresponding freeDSP development guidelines can be found here: https://github.com/freeDSP/WIKI-AND-GENERAL-TOPICS/wiki

Pinout of the USBi connector

The pinout of the USBi connector is shown below.

Figure 15 - Pinout of the USBi connector.

Pinout of the Arduino connector

The pinout of the Arduino connector is shown below. An Arduino Nano or Micro can be connected. The Arduino should be aligned with the outer edge of the freeDSP board. With the Nano, some of the plugs in the middle of the freeDSP remain unused.

Figure 16 - Arduino connector pinout.

Input sensitivity options

The input sensitivity of the freeDSP is configurable via jumpers. You can customize the input sensitivity in even finer steps with user-defined resistors at the input channels. Table 3 shows some resistor values and the corresponding maximum RMS input voltage. Note, that the output signal is always max. 0.9 V RMS.

Table 3 – Resistor values for various input sensitivities.

Resistance	Maximum RMS input voltage
1.20 kΩ	0.3 V (-10 dBV)
7.15 kΩ	0.9 V (default)
8.06 kΩ	1.0 V
10.5 kΩ	1.2 V (+ 4 dBu)
13.7 kΩ	1.5 V (+ 6 dBu)
18 kΩ	2.0 V