CHAPTER 11

DSP Post Processor

While most modern transceivers have extensive digital signal processing (DSP) functions for post-processing, older rigs do not. Also, there are many inexpensive transceivers that have no DSP capability. Regardless of their shortcomings, many amateurs still like to use their older (vintage) or less-expensive units on a regular basis. Indeed, sitting on a park bench operating *al fresco* with a $10,000 transceiver probably isn't going to happen.

Al had an older Yaesu that was not up to modern standards of signal processing, so he created an all-analog post processor unit with "tone controls" and low-pass filters. It works fine, but is still not what he wanted. He then discovered the Teensy microcontroller and its Audio Library, which has extensive digital signal processing functions in the audio frequency range. Now it is possible to readily put together digital filters, digital compressors, and lots of other useful functionality in one reasonably-priced package. It is the bundle of features presented in this chapter that has formed the basis of the post-processor Al really wanted.

So, what can a post processor do for you? Some of the possibilities are:

- Variable low-pass, high-pass, and band-pass filters, with up to 160 dB/decade filter skirts
- Notch filter with variable Q
- Automatic notch filter
- Automatic level controls and compressors
- Multi-band equalizers
- FFT display
- DSP noise reduction
- Any combination of the above
- Power final amplifier to drive less efficient speakers
- Multiple inputs
- Dual channel

There are several commercial units with some of the functions mentioned above, but typically at relatively high prices. The unit we are describing does all of the functions listed above and costs less than $100 if you build it yourself. In addition, the software, which is the heart of the unit, is open-source, so you can extend and tailor the processor to your needs. We also present some ideas of ways to augment the features of our DSP Post Processor (DPP), if you choose to do so.

User Interface Overview

The stuff under the hood of our DPP is quite interesting, but the every-day interaction with the user interface (UI) is what makes the high-power DSP technology accessible. We have spent a significant amount of time and countless prototypes streamlining the way our DPP interfaces with you and the functionality it brings to the table.

A UI has several parts, most important of which are: 1) the control/data input mechanism, 2) feedback to the user, and 3) the ease with which the user can make changes. We have elected to present data about the state of the processor using a high-resolution TFT color LCD screen. Data input/control uses the touch capability of the TFT screen as well as rotary encoders, push button switches, and even analog level controls and input switches. Our experience has shown that this combination of physical controls and TFT touch screen input is quite efficient and minimizes the number of steps necessary to make changes and set the state of the unit.

Figure 11.1 shows our final front panel iteration with the TFT display, two level controls, two rotary encoders, eight pushbutton switches to turn functions on and off, and a pair of six-position rotary switches for input selection. With the TFT display, the user gets graphical feedback as to the state of the unit's functions and can select functions to set parameters. A real-time FFT display shows the spectral content of the various signals, after DSP. Because of graphics involved and the use of a touch screen, the display necessarily has higher resolution (480×320) than most projects in this book. Still, a high-resolution touch screen TFT display costs less than $10.

Figure 11.1 — DSP Post

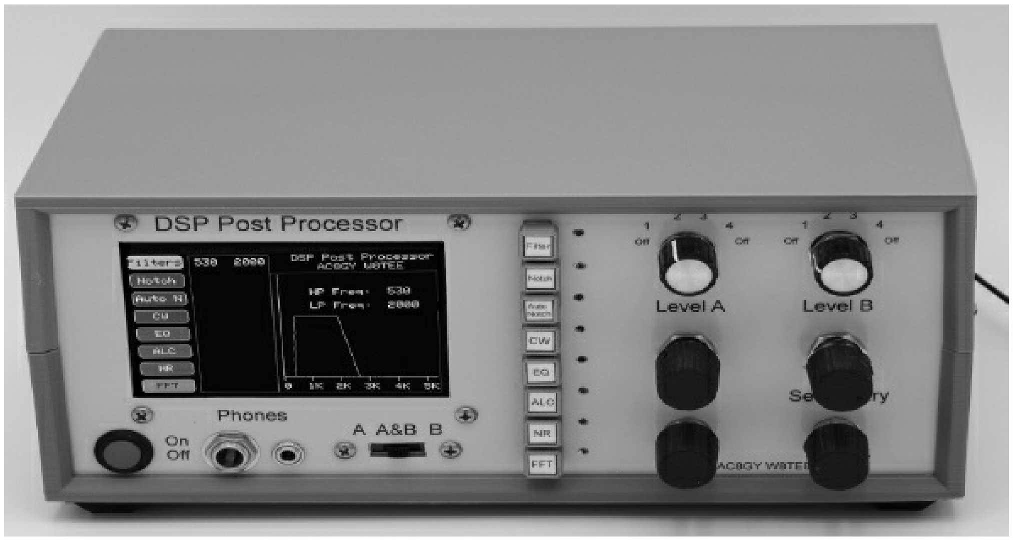
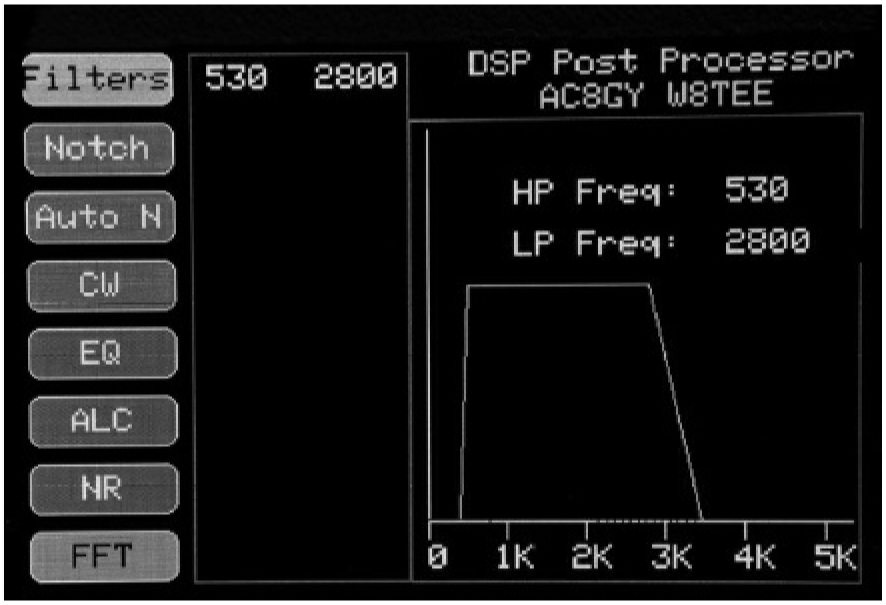


Figure 11.2 shows a close-up of the TFT screen with the graphical display of the low-pass (LP) and high-pass (HP) filter setting function. Here the user can select the LP and HP cutoff frequencies while listening to the results through the speaker or headphones. Real-time interaction makes fine tuning the unit a breeze. The continuously variable filters mean fine control of the signal quality is very easy. The UI is explained in detail later in the chapter.

Figure 11.2 — User interface.



530 2800

DSP Post Processor
AC8GY W8TEE

HP Freq: 530
LP Freq: 2800

Circuit Description

This project is about 25% electronic hardware and 75% software, so let's get the circuit out of the way first and then dive into the software.

Figure 11.3 shows a block diagram of the circuit. The heart of the unit is the Teensy microcontroller and the Teensy Audio Adapter. We are using the Teensy 4.0 (T4 from now on), a powerhouse clocked at 600 MHz with more than enough memory and enough GPIO pins to do everything we want to accomplish.

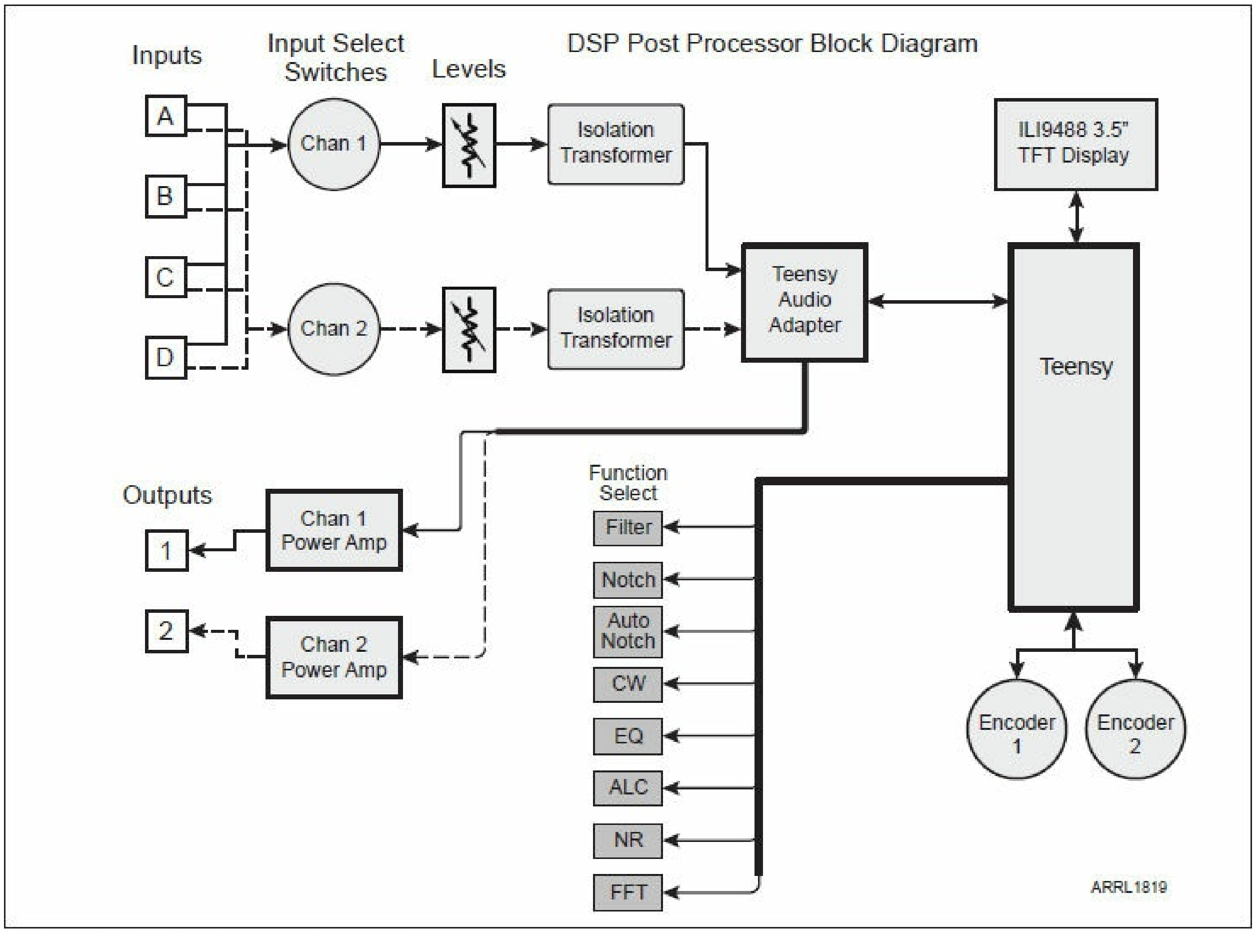


Figure 11.3 — DPP block diagram.

The second most important component is the Teensy Audio Adapter (TAA), which contains all of the audio I/O, such as ADCs and DACs necessary to capture the audio signals, send the digital data to the T4, and convert and output the processed signals back to analog audio again. The TAA has 16-bit conversion and operates at a 96-kHz conversion rate, enough for full 16-bit, 44-kHz CD-quality audio. The TAA can directly piggyback onto the T4, too, significantly simplifying the circuit wiring. Finally, the T4 has enough horsepower to run a real-time FFT process at the same time as doing the DSP filtering on the audio channels.

Starting at the input, we see that provision is made for four inputs that can be routed to either of the two computational channels. At this time, only Channel A has DSP implemented. However, it would be very easy to add the DSP functions to Channel B. We did not need two independent DSP channels, so this is left as an exercise for the reader. Channel B is a straight pass-through, with no DSP. However, each channel has its own Level control (Level A and Level B), so two separate inputs with different levels can be accommodated. Note that each switch has an off position at each end of its range, so either channel can be easily muted, without changing any other setting. Outputs from the Level controls go to isolation transformers and then to the Line Inputs of the Audio Adapter. The isolation transformers eliminate the possibility of ground loop noise and isolate the grounds of the transceiver and Post Processor.

Line Outputs from the Audio Adapter go to dual 10 W power amplifiers, which boost the signal to be able to drive a modest-sized speaker. For best efficiency, we have employed inexpensive Class D amps that are 90%+ efficient and run cool, even at full power. Class D amps have a reputation of being quite noisy because of the switching technology employed. To minimize any interference with receiver performance, the power supplies have been bypassed, with extra filtering throughout.

The remainder of the circuit is comprised of the ILI9488 SPI interfaced display and UI inputs and controls. Two rotary encoders allow easy manipulation of parameters such as the corner frequency of the filters, CW filter width, and other characteristics. The eight latching, on-off switches are used to turn on any of the eight primary functions.

The detailed circuit diagram, including pin assignments, is shown in **Figure 11.4**. Note that some Teensy pins, such as USB, are not available because of system use. Two voltage regulators provide power for the Teensy, display, and LED indicators. The power amplifier is operated straight from the 12 V input. A 5 V fixed regulator drops the 12 V to the T4 Vin pin. A 3.3 V regulator is also used to power the display and indicator LEDs. This regulator needs to have a heat sink attached. Note the enhanced power supply bypass, using ceramic capacitors at several locations. **Table 11.1** presents all of the interconnections between the Teensy and other devices.

Table 11.1
Teensy T4 Pin Assignments

Teensy 4.0 Pin	*Attachment*	*Teensy 4.0 Pin*	*Attachment*
2	Touch IRQ	17	Encoder2 A
3	Encoder1 A	18	Audio Adapter
4	Encoder1 B	19	Audio Adapter
5	Encoder1 SW	20	Audio
6	N/C	21	Audio
7	Audio	22	N/C
8	Audio	23	Audio Adapter
9	LCD D/C	24	Filter PB
10	LCD CS	25	Notch PB
11	SPI MOSI	26	Auto N PB
12	LCD, Touch MISO	27	CW PB
13	LCD, Touch SCK	28	EQ PB
14	TOUCH CS	29	ALC PB
15	Encoder2 SW	30	NR PB
16	Encoder2 B	31	FFT PB

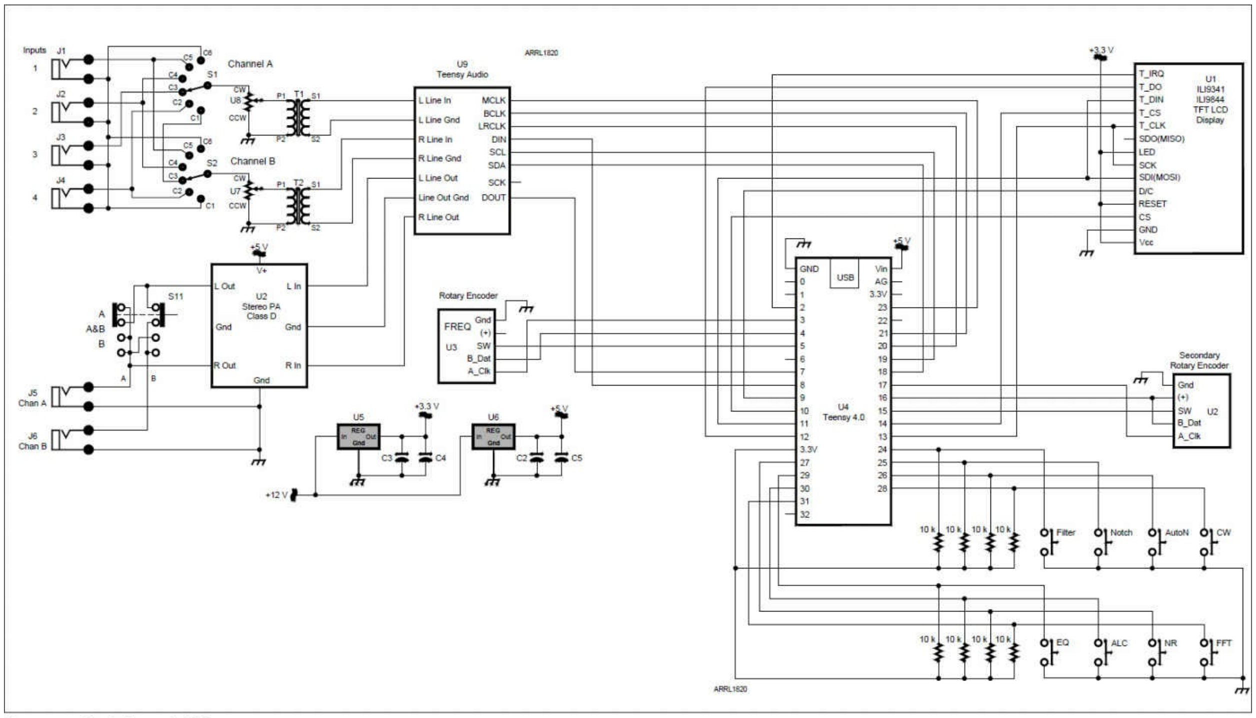


Figure 11.4 — Circuit diagram for DPP.

Building the DPP

One advantage of the audio frequency range is the relaxed requirements on layout. The DPP can be constructed using any of the popular methods. We built our prototype on a modified breakout board. The Audio Adapter plugs directly onto the T4, so control wiring is minimal. We always use sockets for the microprocessors for ease of insertion and removal, if necessary. The other components are connected using IDC connectors and ribbon cable for convenience. No special precautions are required. We like to use #30 AWG solid hookup wire because of the close 0.1-inch hole spacing in the proto-board (same for most perf boards.) The main power connections use #24 AWG wire (or larger) to minimize resistive losses. When mounting the board to the chassis, keep the T4 USB connector as clear as possible in case you want to modify the software.

Eight of the T4 connections, which go to the pushbutton switches, are located on the bottom of the T4. We connected to these by soldering a small wire to each and attaching the wires to an 8-pin male header, which can be plugged into the circuit board sockets as shown in **Figure 11.5**. Alternatively the user could use spring-loaded connector pins on the circuit board, but we felt that soldered connections would be more reliable.

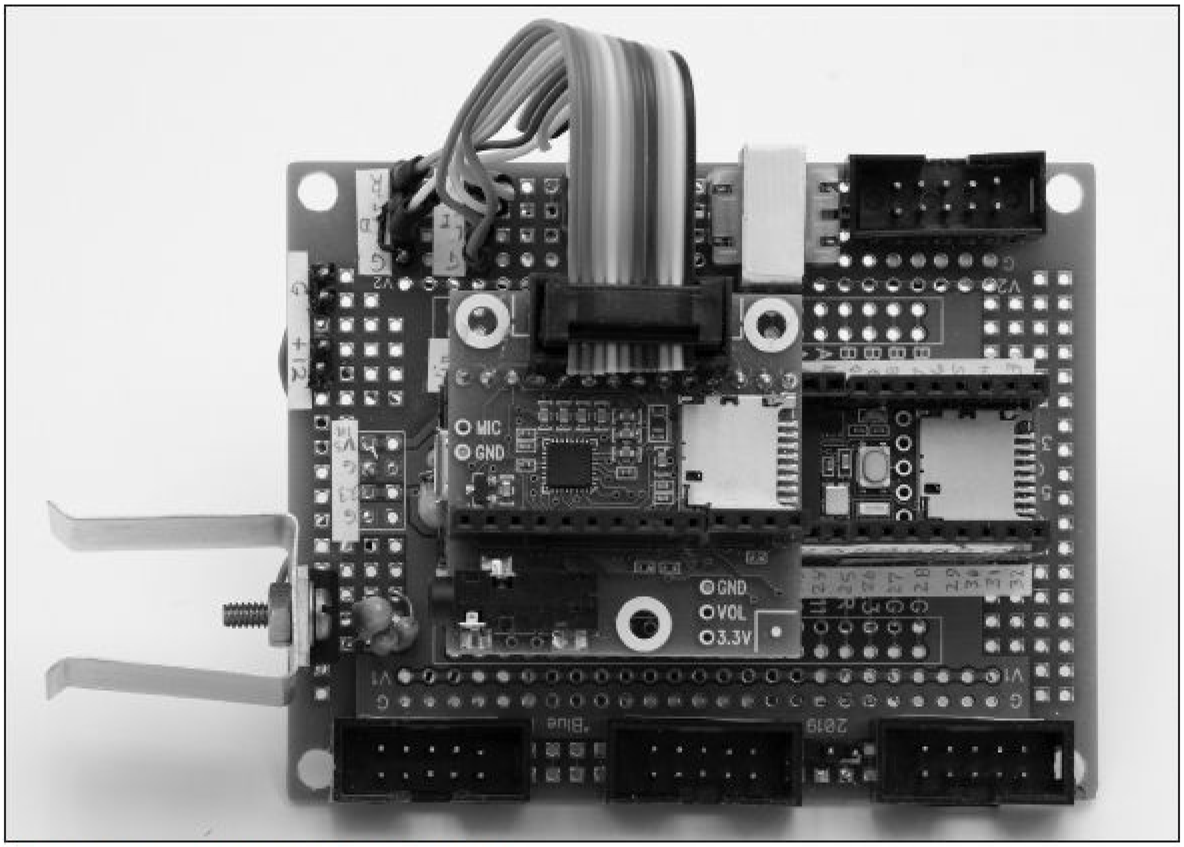


Figure 11.5 — DPP circuit board (top).

Figure 11.5 shows the top of the main circuit board, with the Teensy and Audio Adapter installed. Note the use of multi-pin IDC connectors for display, front panel connections, and encoders. **Figure 11.6** shows a view of the bottom of the circuit board. **Figure 11.7** shows the front panel wiring, with the ribbon cables and female IDC connectors. The back panel is shown in **Figure 11.8**.

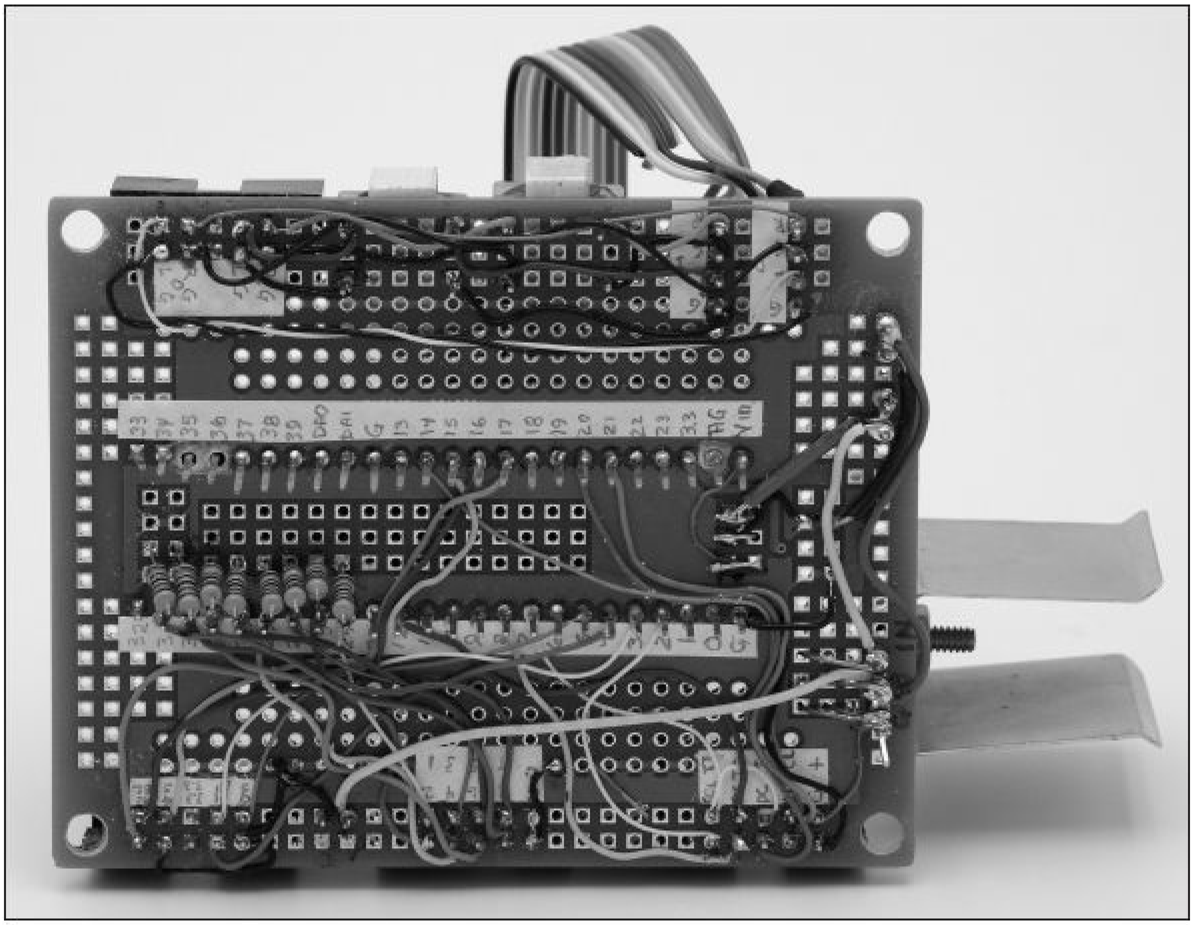


Figure 11.6 — Bottom view of circuit board.

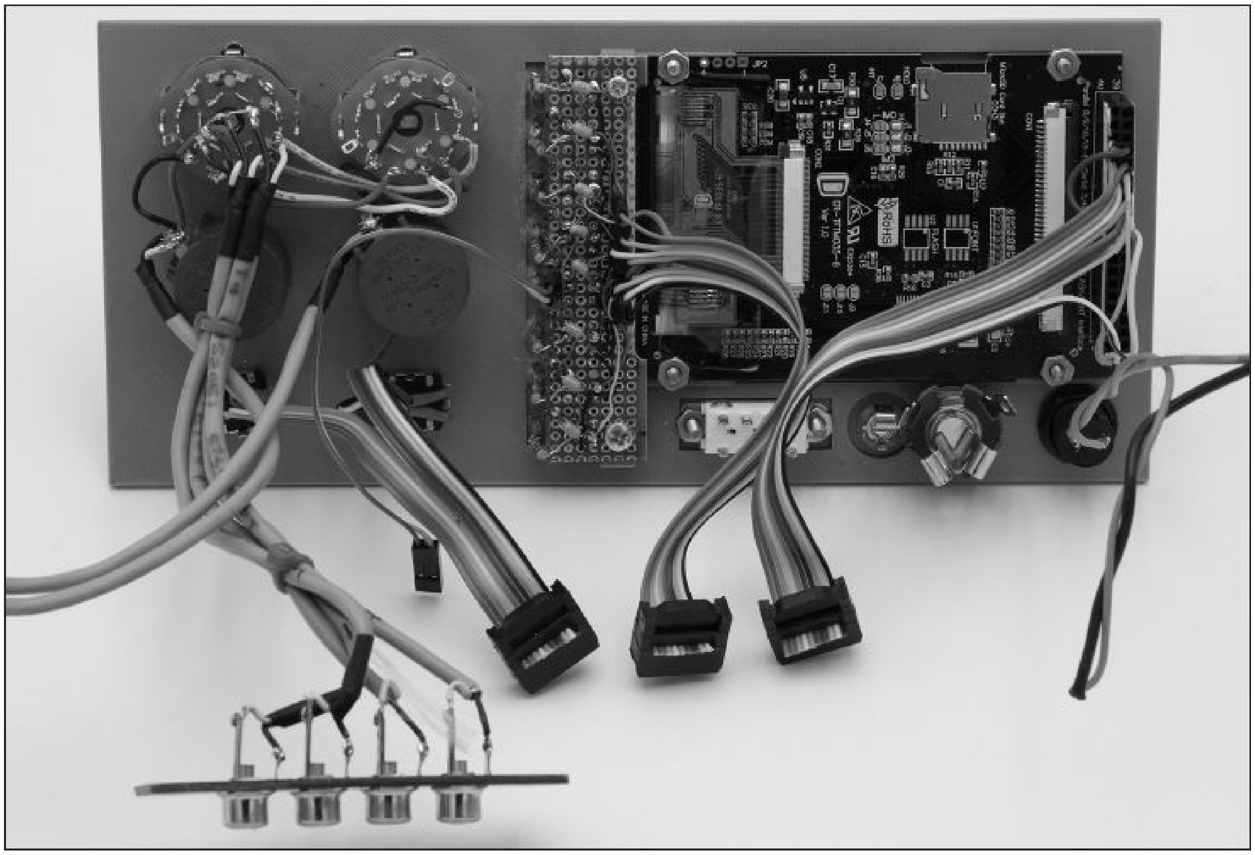


Figure 11.7 — Front panel wiring.

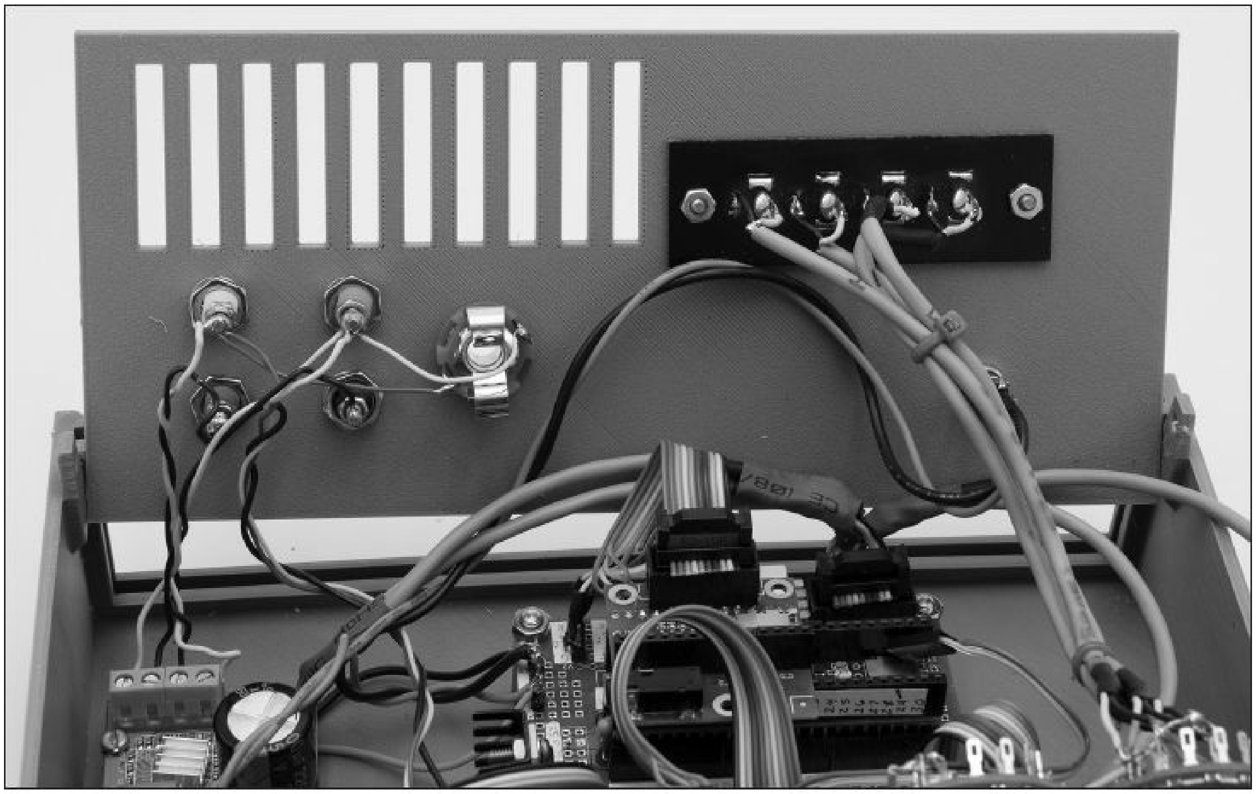


Figure 11.8 — Back panel wiring.

Finally, a top view of the assembled unit is shown in **Figure 11.9**. Shielded cables are used to carry the low-level audio signals from the front panel to the Audio Adapter and from the inputs to the switch assemblies.

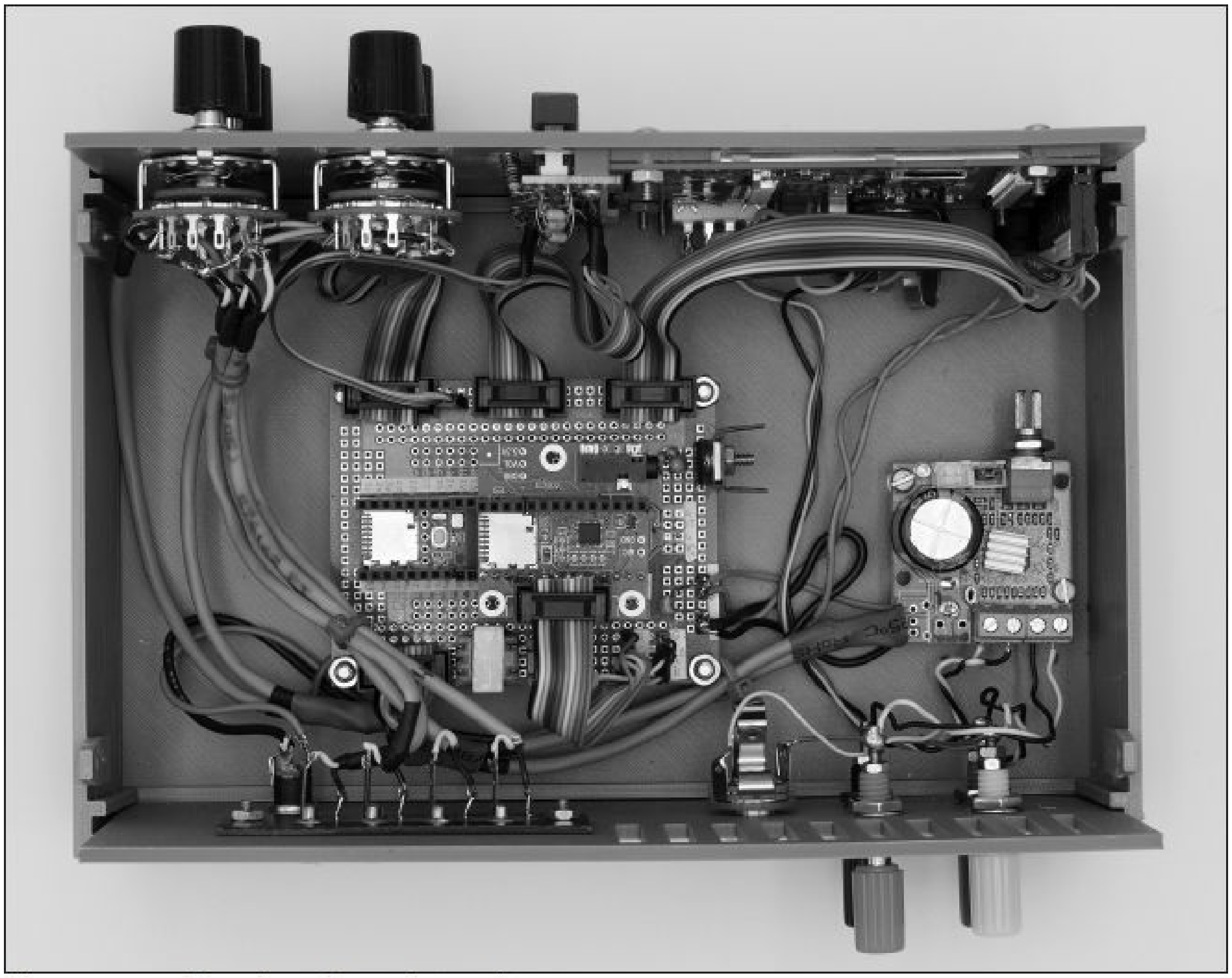


Figure 11.9 — Top view of complete unit.

For most of the projects in this book, we assume the user has a 12 V or 13.8 V power supply available, similar to the one discussed in Chapter 5. The current project doesn't require much power, except for driving the speakers at high levels. Even then, the power demands are modest.

If an adequate supply is not available, a small switchmode 12 V supply capable of a couple of amps works just fine. Be sure to add adequate filtering of the incoming 12 V supply to minimize noise from getting into the audio path.

We made a simple and inexpensive 3D printed case with a custom faceplate as shown in **Figure 11.10**. Once again, Chapter 16 has ideas on how to finish your projects and give them a professional look.

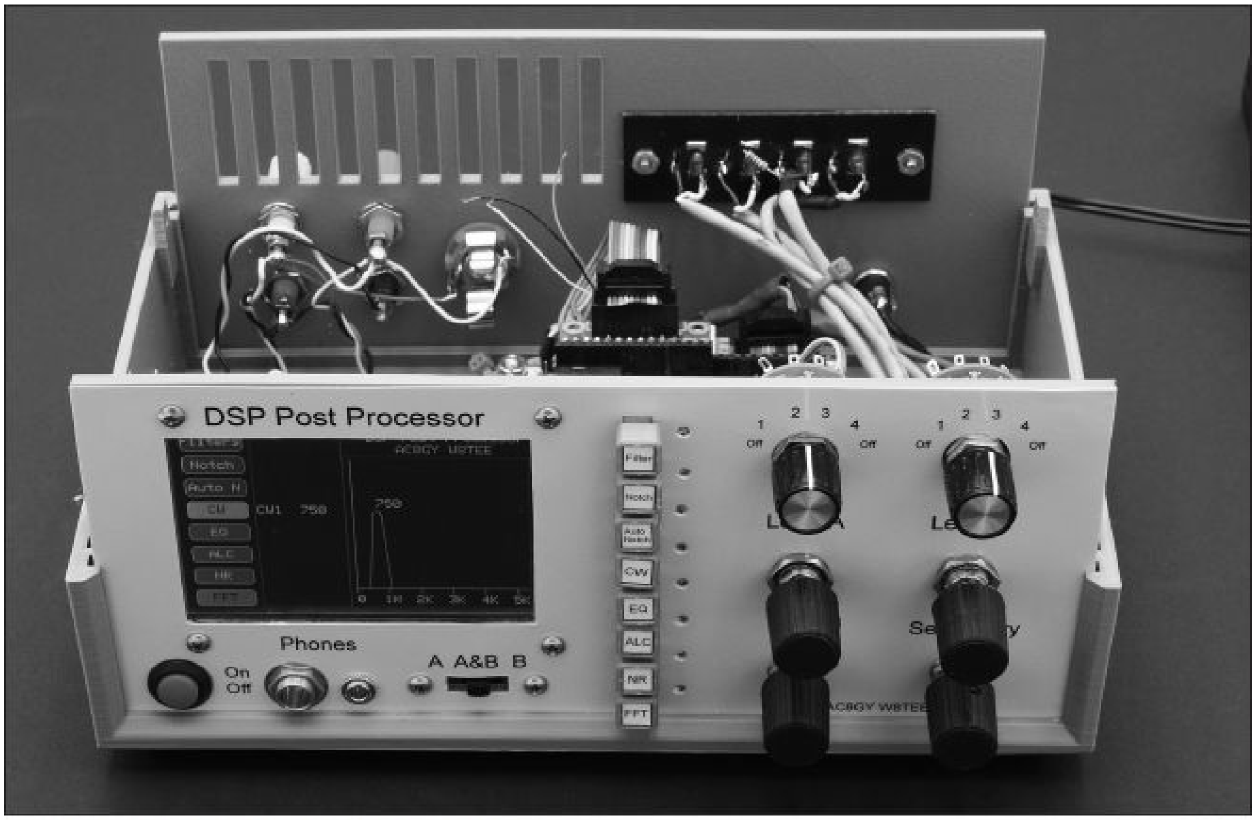


Figure 11.10 — The DDP at home in its case.

Operating the Post Processor

Most of the difficulty of using the processor is in the setup and, as we indicated previously, we spent a lot of time trying to make the user interaction as easy and intuitive as possible. The operational steps are a simple 1, 2, 3 step process:

1) After hooking up the DPP to your rig and some speakers, you must set the parameters for the functions you want to use. To enter the "Set" mode for any function, touch the on-screen function button (such as Filters or Notch — see Figure 11.2) to bring up the setup routine for that function. The current status is displayed next to the screen function button on the left side (see **Figure 11.11** for an example). Using the two encoders, set the parameters for that function. A detailed description of each function's setup is given below.

2) Turn each function on or off using the pushbuttons on the right side of the display. The status of the function is then displayed by the color of the button for that function and by means of the indicator LED next to the respective pushbutton. The pushbutton is depressed when the function is on

and extended when it is off.

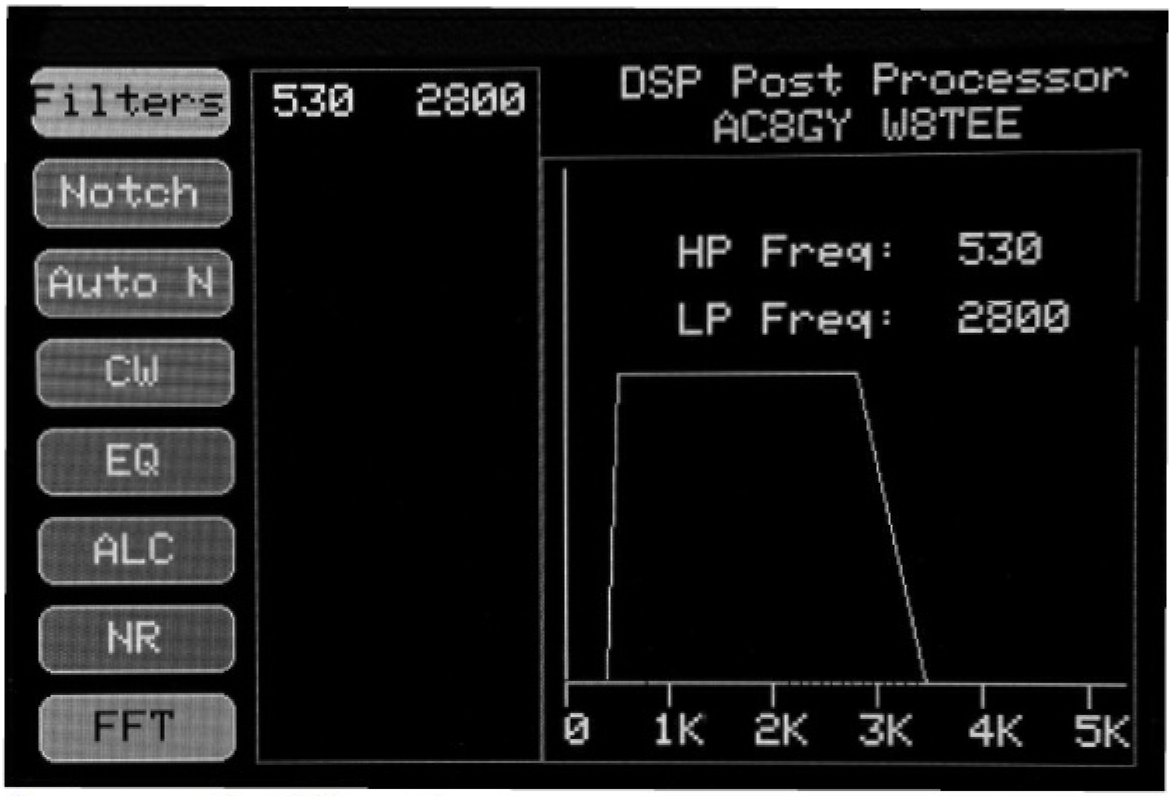
a. Note that some of the functions are mutually exclusive, particularly the CW Filter, which cannot be used at the same time as the LP or Notch functions.

b. All of the others functions are available with LP, Notch, CW, or each other.

3) Select the inputs for each channel and set the levels. You are now ready to use the DPP!

Function Detail

Each function is set up in a slightly different way, but the general process is similar for all options. In each case, the desired function is selected by enabling its "Set" mode button as seen on the left side of Figure 11.11.



HP Freq: 530
LP Freq: 2800

Figure 11.11 — Cutoff filter setting.

1) Cutoff Filters (LP and HP)

Low-pass (LP) and high-pass (HP) filters limit the audio passband to improve speech recognition. High-frequency noise, above 2000 Hz, may mask speech, reducing clarity. Low frequencies, under 400 to 500 Hz, generally add little to speech recognition and may reduce intelligibility. The cutoff filters allow the operator to tailor the spectrum to reduce noise.

Both the HP and LP filters are characterized by their –3 dB cutoff frequencies. The encoders allow real-time adjustment of the cutoff frequencies while listening to the result. The HP filter is set with the primary (left) encoder and the LP cutoff is set by the second encoder. Figure 11.11 shows the filter setting screen (note the 530 and 2800 numbers in the setup area of the display, representing 530 Hz cutoff for the HP filter cutoff and 2800 Hz for the LP filter). Simply rotate the encoders while listening to the signal through the speaker or headphones until the desired result is achieved. The –3 dB cutoff frequency is shown both on the graphical display and in the status display.

The HP filter is adjustable from 150 Hz to 1000 Hz, and the LP filter goes from 1000 Hz to 5 kHz. The HP filter has a final slope of 24 dB/octave and the LP final slope is 48 dB/octave. Given that each operator has their own audio "signature," the DPP can make it easier to listen to someone comfortably.

Figure 11.12 shows a frequency response plot of the filters at several cutoff frequency settings. Note the extremely sharp falloff above and below the cutoff points. Several LP frequencies are shown for one HP filter setting. Also note that the signal "tops" are relatively flat.

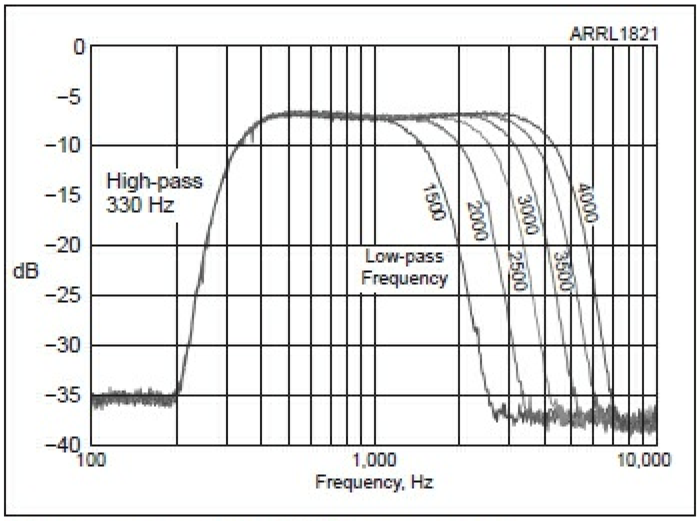


Figure 11.12 — Cutoff filter response.

2) Notch Filter

The purpose of the Notch filter is to remove a single pure tone, while preserving the ability to recognize the desired signal. The Notch setting screen is shown in **Figure 11.13**, with a typical notch filter setting at 1 kHz.

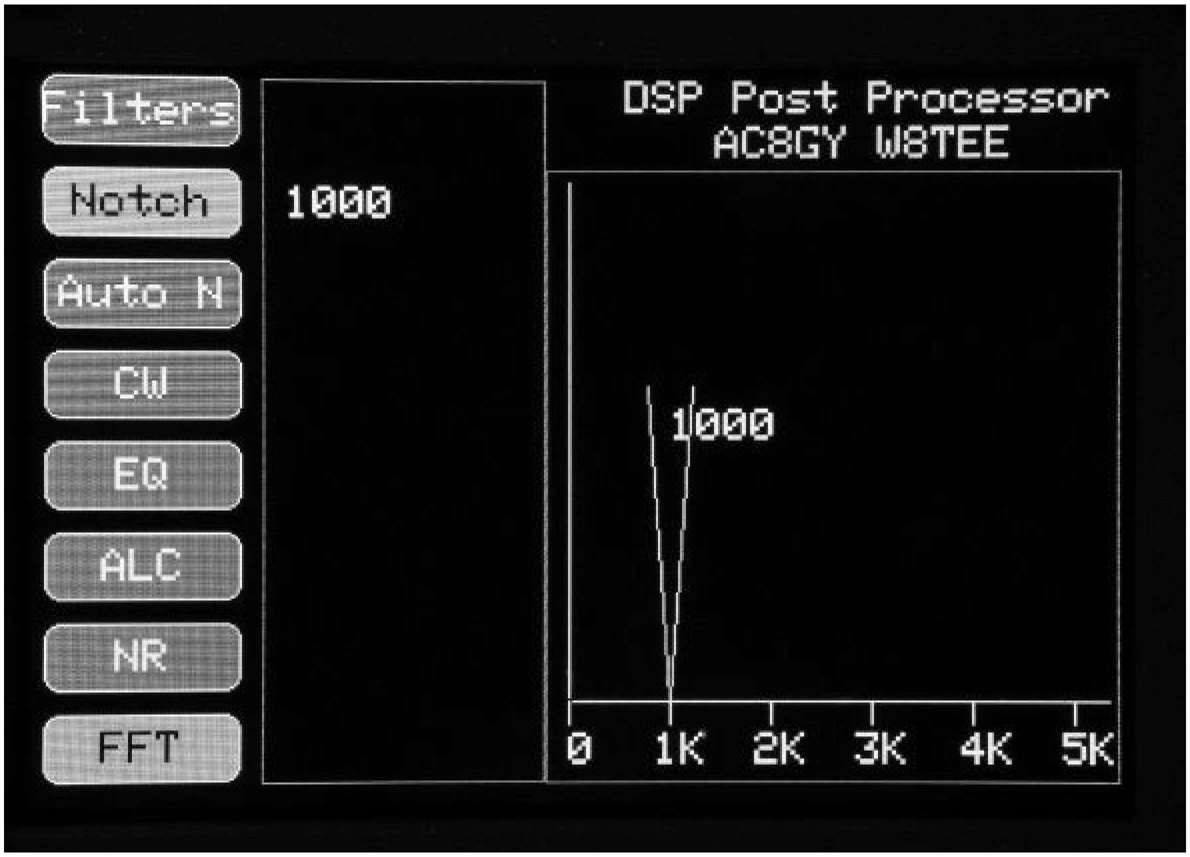


Figure 11.13 — Notch filter setting screen.

The notch filter frequency is continuously variable using the left encoder. To use it, set the Notch filter frequency (for example, 1000 Hz) with the Frequency encoder to eliminate the tonal noise while listening to the signal.

The Notch filter has two parameters: The reject frequency and the Q value, which determines the "sharpness" of the filter. The frequency is displayed next to the graphical display of the Notch. Q values are set in the program source code file, which the user may change to suit. A *#define* value in the .h file defines the Q constant, which may be altered, and the code uploaded again. Q values less than 2 result in a broad filter shape, whereas values above 5 result in quite sharp filters.

If you wish to alter the Q setting, pick a value that eliminates the noise but doesn’t reduce the usable audio. The default is set to Q=20. In practice this value eliminates the pure tone but does not interfere with speech recognition. **Figure 11.14** shows plots of the actual Notch response with no other filters and in conjunction with the cutoff filters active.

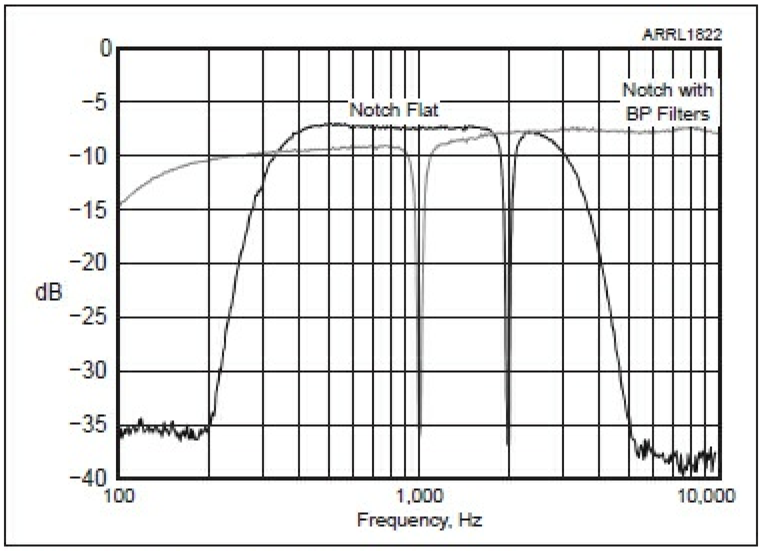


Figure 11.14 — Notch filter response.

3) Automatic Notch Filter

When the Auto Notch function is started, the routine searches for any stationary tones with significant amplitude. The Notch filter is then “tuned” to the tone frequency, removing the noise tone from the output. If the tone frequency changes the notch filter tracks the tone.

The Auto Notch function requires a certain level of tone amplitude to be effective. If the Auto Notch “lock” is lost periodically during a QSO, the “hold” function may be used. (The “Hold” button is only available when the Auto Notch option is selected.) Once the tone frequency is located, the “hold” function can be used to lock the frequency. This reduces the chance that the lock will be lost. Pressing the “Hold” button again turns off the lock. **Figure 11.15** shows the Auto Notch setting screen.

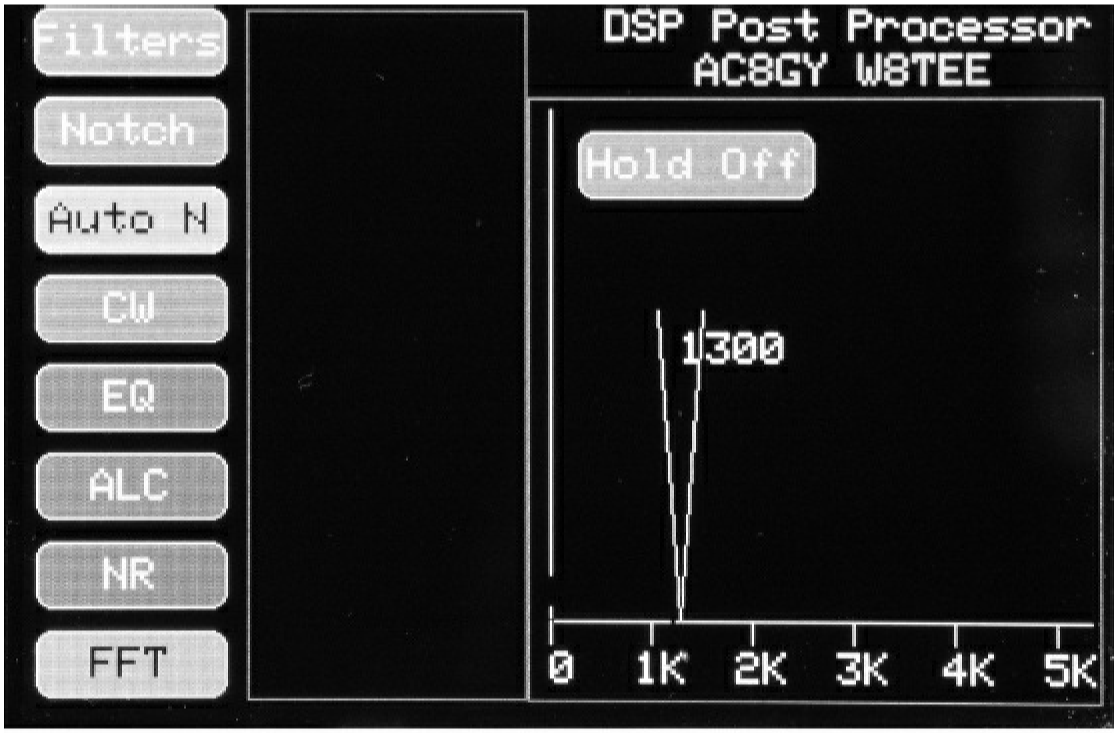


Figure 11.15 — Auto notch setting screen.

4) CW Filters (CW)

There are three preset CW filters, with –3 dB bandwidths of approximately 60 Hz (CW1), 100 Hz (CW2) and 230 Hz (CW3).

The filter center frequency is selected with the *Frequency* encoder and the filter width is selected with the *Secondary* encoder. **Figure 11.16** shows the CW filter setting screen and **Figure 11.17** shows the actual responses of the three filters.

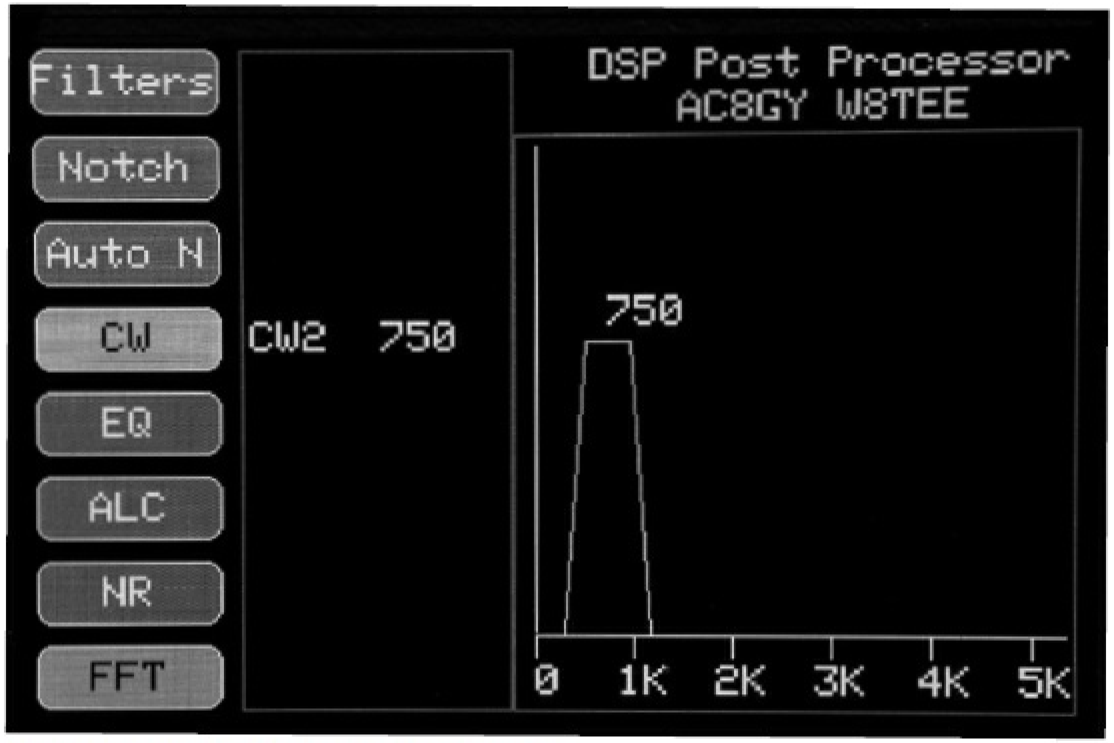


Figure 11.16 — CW2 filter setting screen.

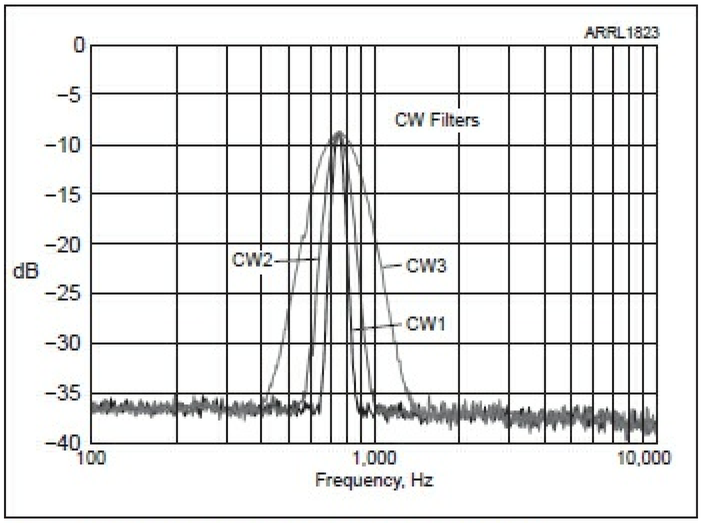


Figure 11.17 — CW filter response plot.

5) Equalizer (EQ)

The Equalizer (EQ) has eight bands: 150, 240, 370, 590, 900, 1300, 2000, and 3300 Hz. **Figure 11.18** shows the individual band responses.

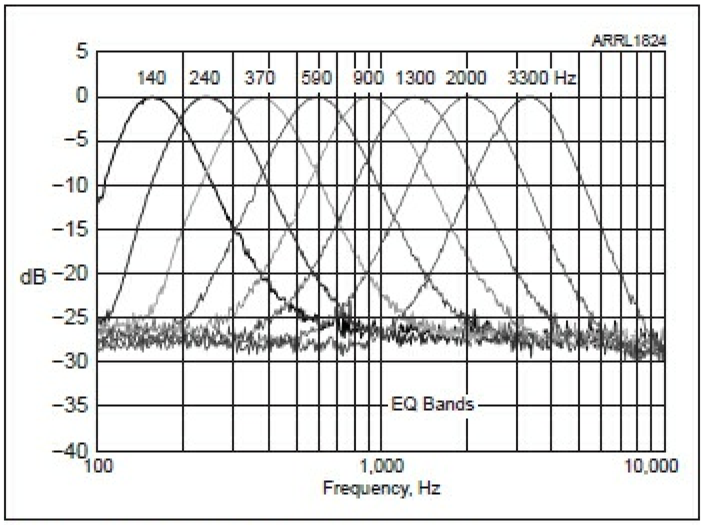
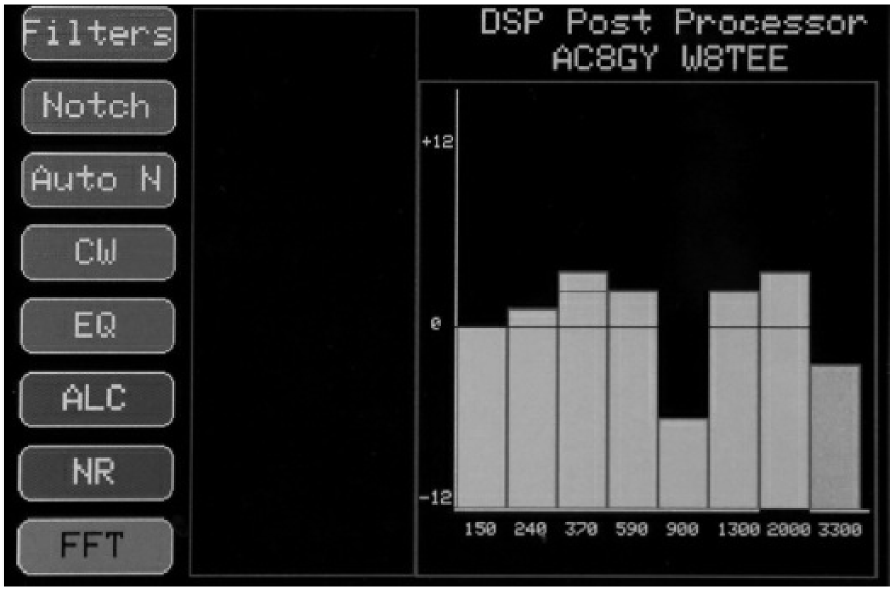


Figure 11.18 — EQ band responses.

In the EQ Setting screen, the bands are graphically depicted on the screen, with two parameter selections, as shown in **Figure 11.19**. The *Frequency* encoder selects the band and the *Secondary* encoder sets the level for that band. Each band has a range of +12 dB to –12 dB. Note that there is interaction between adjacent bands, since the individual bands have finite skirt slopes and the filters overlap. In practice this does not affect the performance, because the amount of adjustment to achieve “natural” sound is typically not large.

Figure 11.19 — EQ setting screen.



The best way to set the EQ profile is to listen to a typical SSB signal while adjusting the individual bands. Alter the band levels to achieve the most “natural” sound. Don’t rely on just one QSO, but try several receptions to get the best result. The EQ may be used in conjunction with any of the other Filter functions.

A response plot of the EQ is shown in **Figure 11.20** with the Notch filter, along with a second response without the Notch. If the SSB signal still sounds like Donald Duck after band adjustments, you’re doing something wrong! For instance, perhaps you did not tune the SSB signal exactly.

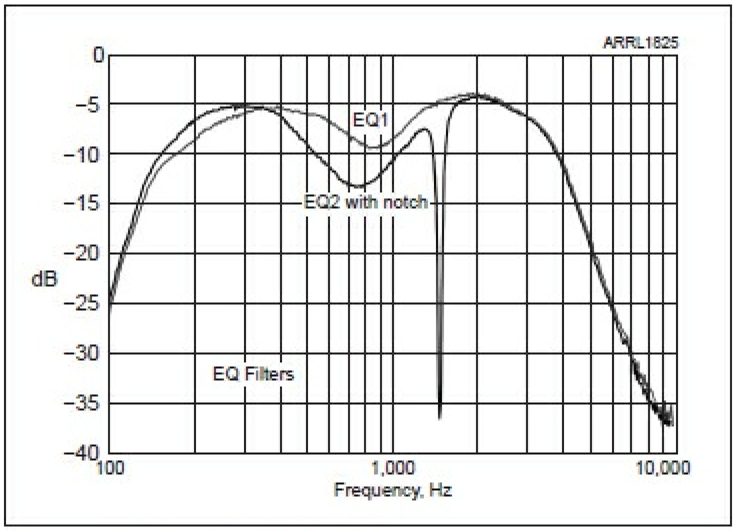


Figure 11.20 — Typical EQ frequency response.

6) Automatic Level Control (ALC)

Audio level varies not only from QSO to QSO, but also because of fading and band variations. Most transceivers have automatic level controls that may be effective or leave something to be desired. The ALC function here augments the built-in automatic level controls by boosting low volume levels and limiting the maximum volume levels. The Threshold level, at which the Auto Level control starts to act, is adjustable using the first encoder. The Threshold is variable from 0 dB, relative to full scale, to about –50 dB. A typical level for speech is about –18 dB.

7) Noise Reduction (NR)

A simple DSP Noise Reduction algorithm is activated by pressing the NR on/off button. There are no parameters to be set for this function.

8) FFT

A Fast Fourier Transform (FFT) display is available during operation, showing the frequency spectrum from 100 Hz to 5 kHz. Its 128 frequency bands clearly show the frequency content of the audio. You can use this information to make informed changes in the equalizer bands. The display is refreshed approximately every 500 ms.

To activate the FFT, press the FFT button. FFT is only available during operation, not while making settings. All other functions may be used with FFT enabled, so the effect of Filters and other noise reduction functions can be observed. **Figure 11.21** shows a typical FFT spectrum of white noise.

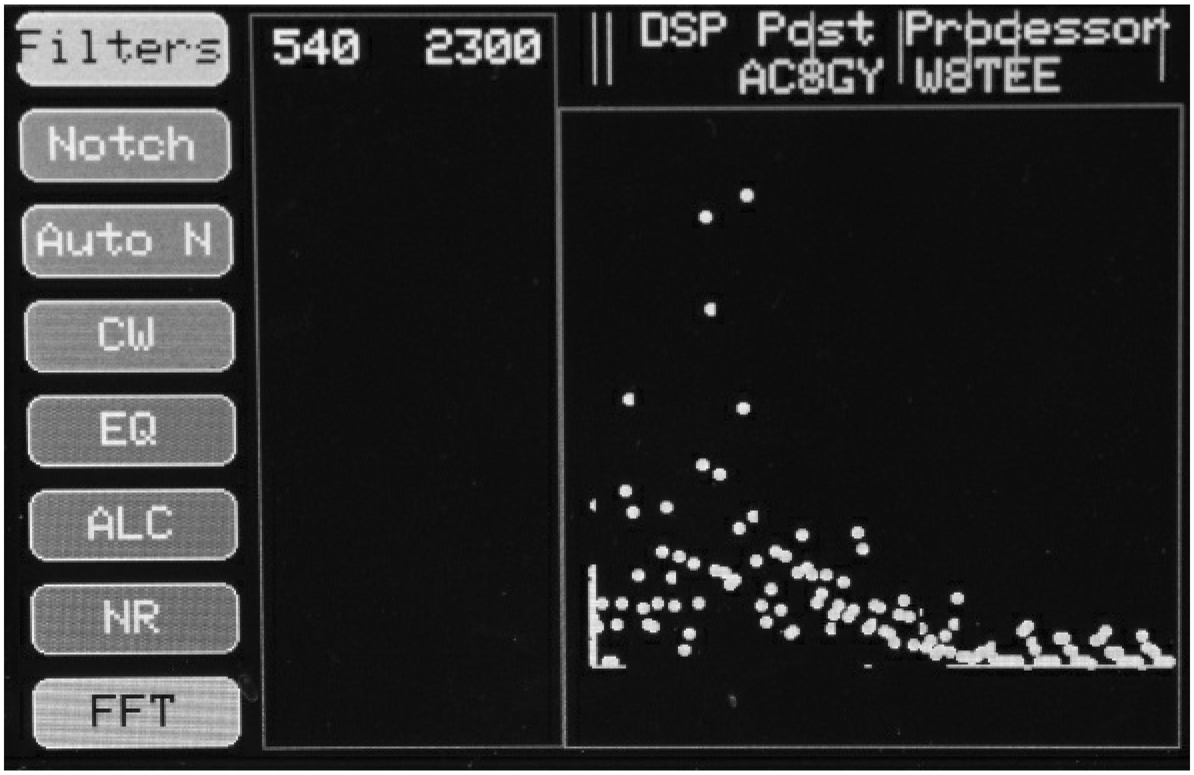


Figure 11.21 — FFT display.

Software Code Description

The DSP Post Processor performs a number of functions to modify and improve the audio signals from your receiver. Some of these functions are similar to circuit functions in the analog world, some are not. In any case, all of our processing is done digitally by performing mathematical operations on the audio signals to alter frequency response, change levels and reduce noise. The speed of the T4 is sufficient to do all of this seamlessly.

Much of the DPP code is like that of previous projects. What is new is the use of the Teensy Audio Library to implement DSP functions. The library functions consist of a number of modules that are connected to form a signal path. Each module also has a set of individual functions with parameters. The software design consists of using the online graphical design tool to lay out the data flow for the audio DSP and then copying the automatically generated code to the Arduino IDE. The graphical design tools is at www.pjrc.com/teensy/gui/. (The design software is very slick, but takes some time to master. It's worth the effort if you want to extend our DPP.) Function parameters are set with the code to tailor the DSP responses to suit. The flow diagram for Channel A is shown in **Figure 11.22**.

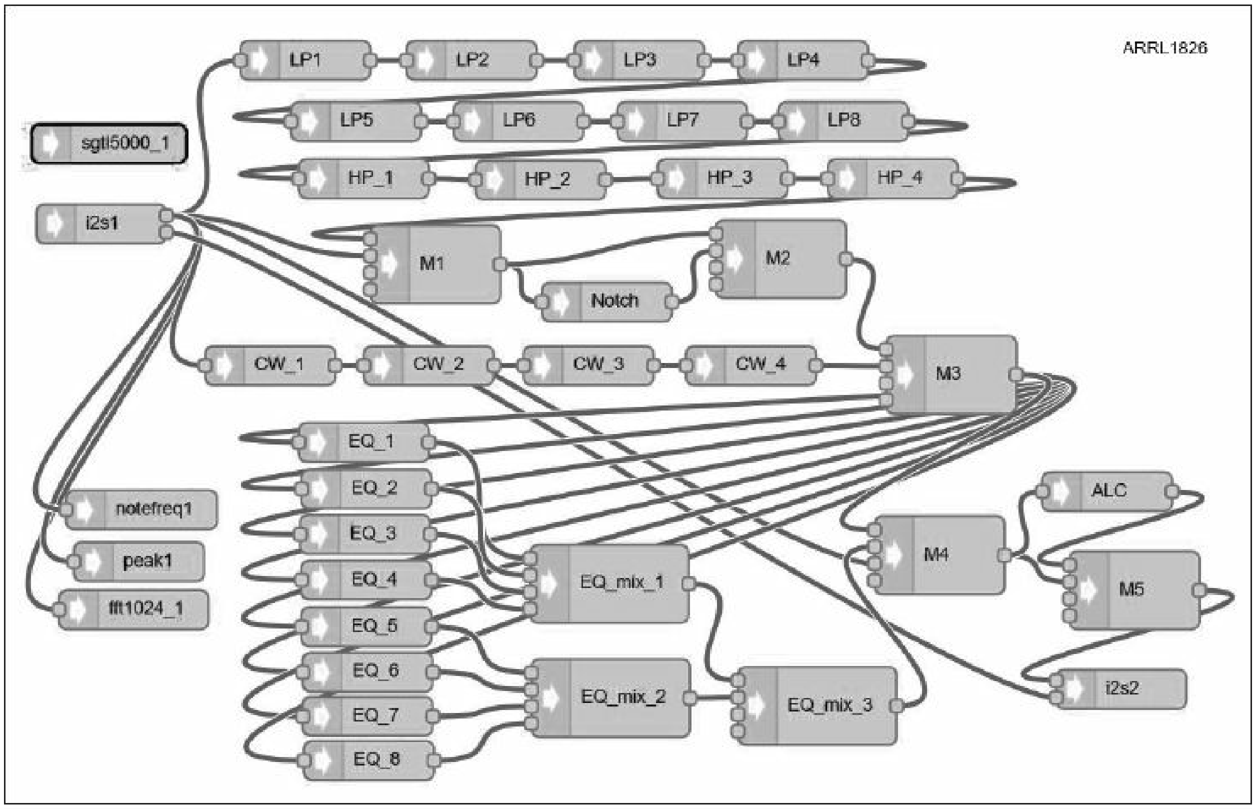


Figure 11.22 — Single channel DSP signal flow diagram.

In the case of the various frequency domain filters we use, there are analog circuit equivalents. For instance, the low-pass filters are equivalent to a cascade of RC (resistor-capacitor) filters with gain. Each single analog RC stage yields a frequency roll-off of –3 dB/octave. Our eight cascaded digital filters have a total roll-off of 48 dB/octave or 160 dB/decade, which is the equivalent of 16 stages of RC filters! Imagine wiring up such an array, with appropriate gain stages. We do it all in software.

The compressor, or Automatic Level Control, is the equivalent of a variable gain stage, controlled by the signal level — again a complex circuit. We do all of that with a few lines of code and a very nice audio library that works with the Teensy Audio Adapter.

The filters we use are called biquad or biquadradic cascaded filters, which are a variety of recursive linear filters based on a mathematical formula that has feedback incorporated. The interested the reader can find out more at **peabody.sapp.org/class/350.838/lab/biquad**. In essence, the filters we use have a grouping of four filters in a module that can be cascaded with additional modules to obtain better performance. The BiQuad filters can be configured as low-pass (LP), high-pass (HP), notch, or band-pass (BP) filters. We use all of those configurations in our DPP.

In addition to digital filters, we also use mathematical operations to calculate Fast Fourier Transforms (FFT) to give us real-time spectrum plots of our audio signals. Once again, the T4 is fast enough to calculate all of the filter function operations and the FFT at the same time. A variant of the FFT is used to find single tones in the audio signs for an Auto-Notch function.

Signal Flow

At the input in Figure 11.22, an audio capture module labeled "I2S1" is the starting place for the audio signal. This connects the analog audio input to the DSP functions. Each module is interconnected using a (graphical) digital "connector" wire.

Starting at the top, we see the LP1 – LP8 blocks which are DSP low-pass filters cascaded to achieve steep roll-off after the cutoff frequency. The eight LP filters attach to the four HP low-frequency cutoff filters for the high-pass function. Each of the major filter sections is interconnected using a "Mixer" function (labeled M1, M2, EQ_mix1, and so on) that allows setting gains and turning individual functions on or off, as well as bypassing that section.

After the HP/LP filters there is a Notch filter in between two mixers that allows for bypassing the notch function. Auto Notch uses the same filter, but uses a routine with "notefreq1" function that finds the frequency of the tone to be suppressed and sets the notch frequency automatically.

In a parallel circuit, the CW band-pass filters are used instead of the LP/HP combination to give very narrow filtration for CW work. Three CW bandwidths are available, and the center frequency of the CW filter can be altered using the Setting function.

Next in line we have the eight-band equalizer. Signals are sent to eight different biquad filters each with a different frequency. Human hearing is approximately logarithmic with frequency, so the spacing is approximately set to be 1.5× the previous band frequency, which gives an equal spacing on a log scale. The eight bands cover the range of 150 Hz to 3300 Hz in eight steps. The output of each filter is sent to a series of three mixers, in which the individual band gains are set. The eight outputs are then combined into one composite signal. The spectrum is shaped by setting the individual band gains from +12 dB to –12 dB. Because the filter skirts drop off at a finite value, there is interaction among the filters, limiting the sharpness of peaks and dips. The purpose of the EQ section is to gently shape the spectrum to eliminate humps or dips in the ultimate frequency spectrum, compensating for microphone variations on the other end. Note that the filters create phase changes, so the gains alternate + and – indicating the appropriate phase reversals.

Automatic Level Control (ALC) is computed in a special section using an augmented Audio library. As with the other functions, ALC is turned on or off using mixer gains.

In the software, there are other code routines that perform various functions. For instance, there is an ongoing routine that reads the on-screen touch-activated buttons to determine when one was pressed. Then each Setting function has three major routines: 1) Draw the graphical representation on the screen, 2) Read the encoders and set the appropriate biquad filter parameters and/or the mixer gains, and then, 3) Turn the functions on or off using the front panel pushbuttons.

In the code there are also the usual menu functions, encoder interrupt functions, screen management, and so on, all of which has been adequately described elsewhere in the previous chapters. The code is broken up into 10 different files for convenience of both coding and debugging. Adequate comments throughout should help anyone wishing to make extensions or improvements.

Finally, a simple Noise Reduction routine has been included to reduce random high-frequency noise using a FIR filtering technique. There are many other more comprehensive mathematical routines such as convolution filtering that could be implemented but were beyond the scope of the current project function set. This is left to the reader to add later.

Possible extensions for the adventurous include:

- Independent filter functions for Channel A and Channel B. This is primarily a UI coding exercise, adding setting and selection routines for each channel.
- A more rigorous implementation of the NR routines.
- Additional filtering and/or averaging of the Auto Notch signals to improve signal-to-noise ratio.
- Adding real-time spectrum during setting in order to observe the spectral response variations while making changes.

Conclusion

In summary, the DPP presented here can bring an older transceiver up to modern signal processing standards and could even be a valuable adjunct to lower-end modern rigs that are perfectly adequate in all other ways. There are a lot of good, inexpensive, QRP rigs that could benefit from our DPP, and the improved audio can make your listening experience that much more enjoyable.