63dB Attenuator Version 1.0
©John Price – WA2FZW - 2024

Table of Contents

Introduction

Recently, I've been playing with SDRs and various preamplifiers. In some cases, the signal levels coming out of the SDRs are much higher than needed for applications such as WSJT-X, so I decided I needed some attenuation. Not having an attenuator, I decided to build one!

This device uses a pair of PE4312 attenuator chips in a series arrangement. Each PE4312 is capable of providing zero to 31.5 dB of attenuation in 0.5 dB steps, thus the unit is capable of providing anywhere from zero to 62 dB of attenuation in half dB steps.

The attenuator is controlled by an Arduino Nano processor (clones are available and cheaper).

This is not intended to be a well calibrated laboratory type device; however it can be calibrated fairly accurately.

I never actually built this version. My SMD assembly skills leave a lot to be desired so I was never able to properly mount the PE4312 chips on the PCB. As a result, both the hardware and software are un-tested.

The Hardware

The major components of the hardware are a pair of PE4312 attenuator chips, an Arduino Nano processor, a rotary encoder and a 128 x 32 OLED display.

Theory of Operation

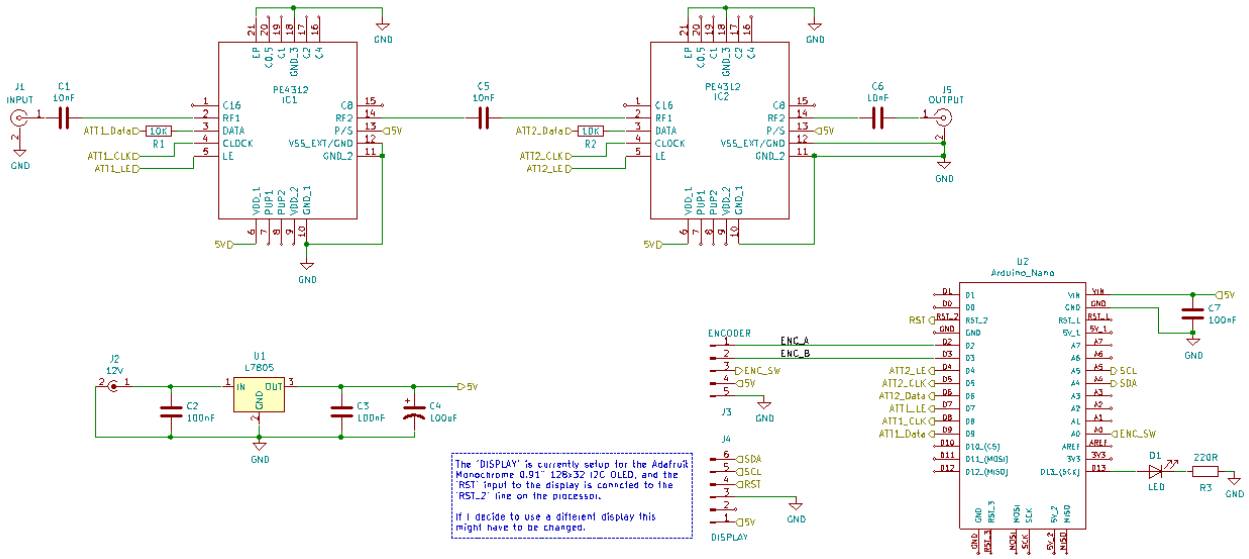
Each of the PE4312 chips is capable of providing zero to 31.5 dB of attenuation in 0.5 dB steps, thus the pair of them in a series configuration can provide a total of 63 dB of attenuation.

The amount of attenuation is controlled by the encoder; each click will add or subtract 0.5 dB of attenuation. The amount of attenuation is displayed on the front panel display.

As there is a slight amount of attenuation even when both attenuator chips are set for no attenuation, there is a provision in the software to account for the inherent loss in the attenuation value displayed on the screen (See the *User Customization* section). The amount of inherent attenuation might be frequency dependent; I'll figure that out once I build it!

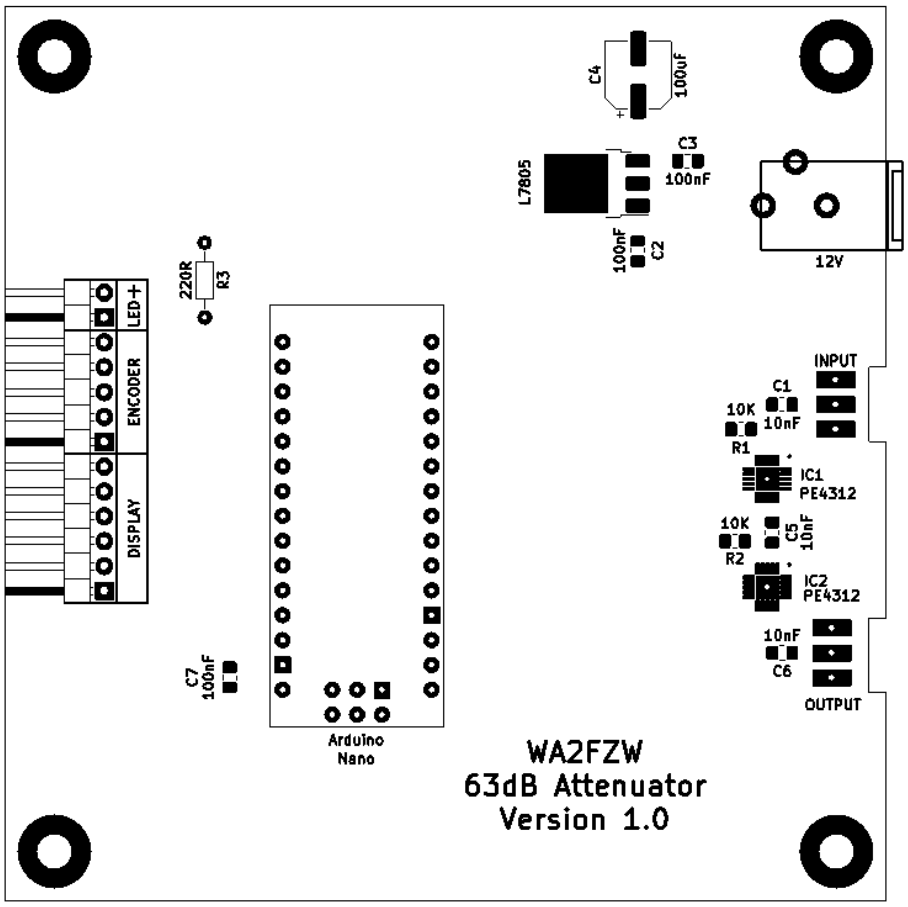
The unit can be powered via a 9 to 12VDC source or via the USB connector on the processor.

Here's the schematic:



The Printed Circuit Board

Here's the layout of the PCB:



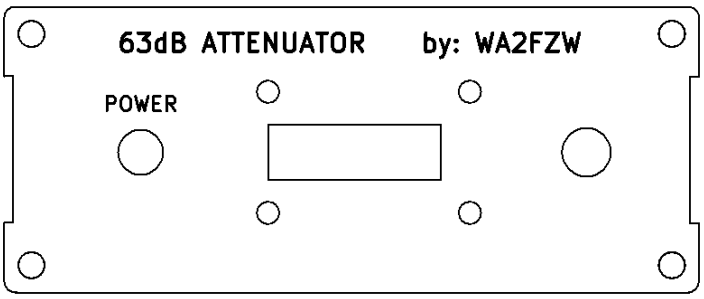
Note that I used a lot of SMD parts, particularly in the RF section to keep the track lengths as short as possible. The SMD resistors and capacitors are all size 805 (that's what I had on hand).

The PCB is designed such that input and output connecters are SMA connectors mounted directly on the board. One could use different connectors such as BNCs wired to the PCB as well.

Note, the position of the '+' next to LED pin header connection is correct here. It is on the wrong pin on the PCBs. It is also corrected in the Gerbers. See the section on *Some Construction Notes*.

Front Panel Controls and Indicators

Here's what the front panel looks like:



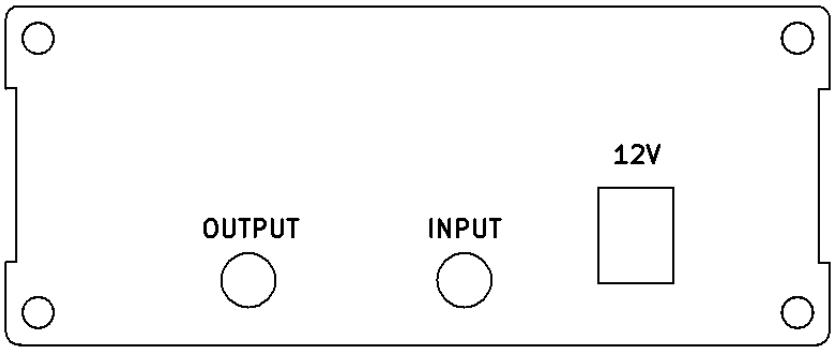
When on, the 'POWER' LED indicates that the processor is running normally and the current attenuation setting will be displayed on the screen. Turning the encoder clockwise will increase the attenuation; moving it counter-clockwise will decrease the attenuation.

Operating the push-button switch built into the encoder will reset the attenuation to the default value (see the *User Customization* section).

For some reason, the 'POWER' LED will flash rapidly when the unit is first powered up then it should come on solid. If it continues to flash, it is an indication that the processor can't connect to the display for some reason.

Rear Panel Connections

Here's what the rear panel looks like:



The 'INPUT' and 'OUTPUT' connectors are SMA females and the '12V' connector accepts a 2.1mm coaxial plug.

Some Construction Notes

Front & Rear Panels:

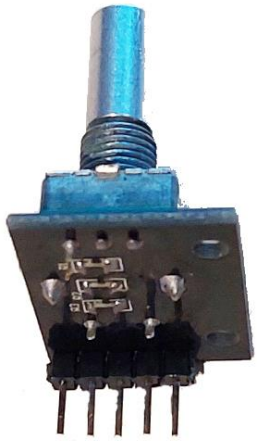
Both panels and the main PCB are designed to fit this specific extruded aluminum enclosure from Amazon (you might also find them on eBay). The front and rear panels are actually printed circuit boards; the Gerber files to fabricate them are included in the distribution package. Of course you can use a different enclosure.

You may notice that the edges of the front and back panels are more gray than black. I found that going over the edges with a black Sharpie makes for a much nicer look.

The screws that come with the enclosure are the countersunk type. In order to use those neatly, you'll need to modify the mounting holes in the panels with a countersink bit or use round head 3mm screws.

Encoder:

The KY-40 encoders that I have come with angled header pins attached to the front side of the PCB. They won't fit in the enclosure. You need to remove the angled header pins and install a straight header on the rear side of the encoder PCB as shown here:



Display:

The mounting holes in the Adafruit display that I used accept either an American #2 screq or a 2mm screw. I used one with countersink type heads and modified the mounting holes accordingly.

The display needs to be mounted with spacers. I found that the #2 nuts were exactly the correct thickness, so here's what the mounting arrangement looks like:

LED Pin Header:

The '+' is on the wrong pin! Not the first time I've done this! It's corrected in the Gerbers.

Software

Required Libraries

The software is fairly simple thanks to the availability of Arduino libraries for the display, encoder and PE4312.

The non-standard libraries required are:

- Adafruit_GFX
- Adafruit_SSD1306
- PE43xx
- Rotary (from https://github.com/brianlow/Rotary)

The first three can be installed using the 'Manage Libraries…' selection under the 'Tools' menu in the Arduino IDE. Note, I'm still using version 1.18.xx of the IDE; I don't expect there would be any issues using the latest version though.

User Customization

There are two definitions in the Attenuator_V1.0.ino file that you might want to change:

- The value assigned to 'ATT_INIT' specifies the initial attenuation setting when the unit is first powered up or when the encoder push-button switch is operated. But note, until the program actually starts running, the amount of attenuation is undefined.
- The value assigned to 'ATT_OFFSET' can be used to compensate for the small attenuation that will exist even when the unit is set for zero attenuation. The amount may vary with frequency. Refer to the *Specifications* section which shows the amount of inherent attenuation for my unit across a wide frequency range.

If you change anything else, you're on your own!

Serial Monitor Messages

If you have the processor connected to your computer and have the Arduino IDE's Serial Monitor running, there are 2 messages that will be displayed:

- 'WA2FZW 63dB Attenuator - Version 1.0' – Program is up and running
- 'SSD1306 allocation failed' – The processor could not to connect to the display for some reason. The 'POWER' LED will also be flashing rapidly.

Specifications

Since I never actually made the unit work, I am unable to provide any measurements.

Suggestion Box

I welcome any suggestions for further improvements. Please feel free to email me at WA2FZW@ARRL.net.

Bill of Materials

Here is a list of the parts you will need and in many cases, links to where you can get the less common parts.

All SMD resistors and capacitors are size 805 unless otherwise noted.

The PCB		Gerber files are available on Github.
Front & Back Panels		Gerber files are available on Github.
Enclosure		Extruded aluminum from Amazon. Gerbers for the front and back panels assume this specific enclosure.
Encoder	KY-40	
R1 & R2	10K SMD	
R3	220 Ohm 1/4W Through Hole	
C1, C5 & C6	10nF SMD	
C2, C3 & C7	100nF SMD	
C4	100uF SMD (6.3x7.7)	
D1	General purpose 20mA LED	
IC1 & IC2	PE4313C-Z	Available from Mouser
J1 & J5	PCB Edge mount SMA female connectors	Available from Amazon

J3	2.1mm Coaxial power receptacle	Available from Amazon
J3, J4 & LED Connector	1 x 13 Male 2.54mm horizontal pin header	
U1	L7805 Regulator	Available from Mouser
U2	Arduino Nano (or clone)	I've been using ones from Amazon