!USBScopePlus Ver 1·01 (7th Apr 2021)

This application allows you to simultaneously use a suitable USB Audio ADC and a USB DAC to perform a “probe and response” measurement on a stereo audio device. It can be used as a 2-channel oscilloscope for other purposes. It can also save data and screen displays, but note that this requires you to have sufficient RAM Disc allocated since the data is saved to RAM disc.

If you have already used !USBScope that side of the program's operation should be familiar to you. However in addition to the scope/spectrum analysis facility of !USBScope this 'Plus' version can also generate output test waveforms which can be used as input for items being examined. You can then collect the resulting output from that equipment and assess its performance.

This version has been tested and can be used either with:

- A combined USB ADC + DAC unit like the Focusrite 2i2 or similar
- Two distinct USB units - e.g. a 2i2 for capture (ADC) and a USB DAC - e.g. a Cambridge Audio DAC Magic Plus for the test tone output.
- The Behringer UCA202 ADC+DAC. This also works and is an example of a USB Class 1 device that employed 2-byte/value transfers.

The application will only work on hardware that supports USB Audio. e.g. the ARMX6 or ArminiX using a recent version of the OS supplied by R-Comp. It assumes you are using a 1920 x 1080 pixel screen mode and operates in single-tasking mode. i.e. it pauses all desktop WIMP applications until quit.

The application should work with Stereo (i.e. 2 channel) USB Audio devices that adhere to the USB Audio ‘Class 1’ or ‘Class 2’ standards provided they employ either:

- 4byte/value transfers (Class 2)
- 2byte/value transfers (Class 1)

To use the application, connect the input, etc, and check you are in a suitable screen mode. Also ensure you have some vacant RAM Disc space. Ensure you allow at least 10MB of RAM disc per screen dump. Then run the application in the usual way.

Tone section info

Inside the application directory you will file a text file called ‘Tones’. This lists the details of the test tones currently available. As provided the file contains a similar to the following:

```
8
S   750.0 0.50
Q  1500.0 0.25
S  1500.0 1.00
T 16000.0 0.25
T    50.0 0.25
I  750.0 0.25
B  750.0 0.25
U 1234.0 0.25
```

The first line specifies how many test tones are included. Note that the number of these must be

at least 1, and no more than 9.

Each of the following lines specifies a test tone.

```
<type>  <tone frequency> <tone amplitude>
```

At present the following types of waveform are provided

'S' = sinewave : 'Q' = squarewave : 'T' = Two-tone : 'I ' = Impulse : 'B' = Bipulse and, 'G' = Gaussian Noise : 'U' = User Waveform.

The squarewave is computed from a series of its sinewave components, but is limited to no more than the 25th harmonic. So acts as a band-limited squarewave. The details of the two-tone waveform depend upon the specified tone frequency. Given a high frequency the wave will be two tones, one at that frequency, and another differing from it by 1 kHz. Given a very low frequency it the second will be much higher. This mimics two different types of intermodulation test that are often used.

The Impulse and Bipulse tests are waveforms often used to test digital items like ADCs and DAC. In this case the frequency of the actual waveforms may not be exactly the value you specify as the pulses will occur a regular integer number of samples apart. The waveform data consists of a series of zero sample values in between one (impulse) or two equal-but-opposite (bipulse) samples. So the chosen frequency will have $2n$ samples/cycle where n is an integer.

The Gaussian Noise waveform is wideband, so in this specific case the specified frequency is ignored. Other types of waveform may be added at a later date, as may the ability to read in 'user generated' waveforms.

The 'User' waveform is one which – as the name implies – can be defined by the user. As supplied the defined waveform is one I developed as a sort of 'torture' waveform for digital devices like DACs. But, as explained later, you can alter this as you prefer.

NOTE that the program makes no 'sanity checks' on the values given. So take care to ensure you keep the amplitudes specified to be no more than (and preferably less than) unity. Also ensure frequencies are less than half the tone sample rate..

Settings File info

The overall behaviour of the program is controlled via a "Settings" file inside the application directory. This contains the values

```
<scope (ADC) sample rate>  <tone (DAC) sample rate>
<time delay in centiseconds>
<signal level scaling factor>
```

Note that not all ADCs/DACs can support all possible sample rates. Also some devices which combine an ADC with a DAC may not be able to run them at different rates symultaneously. Hence unless you know which rates your USB device/devices support I suggest you initially ensure the sample rates are '`48000 48000`' as this rate is generally available. If you try other rates which the devices can't provide the program will try to degrade gracefully to 48k (48000).

But the program may not always be able to tell if the device has correctly implimented the specified settings. You can also use the separate *!USBAudioProbe2* program to scan your system to find out the capabilities of any class compliant USB Audio devices connected to your machine. This can then tell you what sample rates, etc, any device can handle.

The time delay is to deal with a problem called 'latency'. This is the tendency for a audio device to take a short time to respond to a data stream or instruction. i.e. The output from your DAC may start after a delay. It is also advisable to make an audio capture a short time *after* a test tone has commenced in order to avoid any 'starting transient' effects. By default with my own system a value of '3' seems to work OK. But you may need to alter this for your devices.

The scaling value allows you to calibrate the signal levels in Volts. As above you may need to adjust this value to suit your setup, and the adjustment of any input level controls on your ADC.

When started the program scans to find and then choose the *first* USB Audio device it detects that can act as an ADC or a DAC. If you only have a combined ADC+DAC like the Focusrite 2i2 it will then use that for both purposes. However if you *also* have a DAC like the DAC Magic Plus connected and the program finds that listed *before* the 2i2, it will use *that* as the DAC. Hence when there is more than one USB Audio device, you may need to choose which USB ports to connect then via to determine which device(s) will be used.

Screen Display and user control

The screen display is virtually identical to that provided by *!USBScope*. However at the top-right hand of the screen an additional text window shows the current tone output status so you can see which test waveform is being generated. By default this will be tone '0', which means *no* tone has been selected or will be playing. However you can use the top row of 'number' the keys on your keyboard to change this. e.g. pressing '2' will change the tone being generated to tone '2'. Up to 8 tones can be listed and used, and the relevant key will then create them and cause them to begin to play. Note that there will be a delay when you choose a tone. This is because the program calculates the waveshape and required sample values before resuming operation. (Note in particular that the calculation of the 'Q' squarewave takes some time as this generates the correctly band limited waveform.)

For tone generation sample rates higher than 48k the tones generated consist of a series of 0·2 second bursts, once per probe-measurement scan. For 48k and lower rates this is increased to 0·4 sec bursts.

The two main areas displayed represent a 2-channel oscilloscope and a spectrum analyser. You can alter the scan rate of the oscilloscope display using the +/- keys on the top row of the keyboard. By default this has a horizontal scan rate/resolution of 1 millisecond per small division. The vertical resolution autoscales depending on the amplitude of the input signal.

The spectrum analysis employs a FFT. The horizontal scale shows the frequency range with divisions at 1 kHz intervals (linear scale). The frequency span covered will be up to half the samplerate. e.g. when capturing samples at 96k rate the span of the spectrum displayed will be 0 - 48kHz. The vertical scale is 10 dB per division and shows a range of just over 80dB in total. Note that this vertical scale also autoranges. The spectra are obtained by FFT using BH windowing. The FFT uses 16384 data points per channel.

The vertical ranges of each display area are indicated by values shown to the right of the areas.

Along the bottom of the scope area the rms and peak values for the Left and Right channel are reported.
Along the bottom of the FFT spectrum area two frequency values are shown. These show the frequency of the spectral component with the largest amplitude, and a frequency value obtained via a 'counting' method. The values for the channel with the largest signal are shown. A value for THD (Total Harmonic Distortion) may also be shown. This is normally only meaningful when the input waveforms being captured are are (approximately) sinewaves.

A pair of 'PPMs'. or Peak Programme Metres at the top left of the display. These show the peak signal levels in a specific way which can help monitor brief peak levels. One graphical PPM 'bar' per channel. Note that although the other displays provided employ Voltage units, these PPMs scale with the input sample values *without* being scaled into voltages. Because of this, the PPM display is in units of '**dBs**' i.e. dB wrt to the 0dBs level which corresponds to the largest sample values the ADC can output. Any larger input to the ADC would be clipped, so these PPMs can be useful when adjusting an ADC for best results whilst avoiding clipping losses.

At the left-hand end of each PPM there are two values. The one on the left shows the value for the largest sample in the current data-grab displayed. The one to its right is a '*capture, hold, and decay*' value. It shows preserves for a short time the highest recent peak values during the last few scans. This makes it easier for the user to take note of a brief peak level that isn't repeated again in the following few scans. After '*dwelling*' for a while, this value starts to decay - until another peak arrives which pushes it up to the new '*most recent peak*' value. This behaviour is duplicated by the black and red cursors on the PPM display bars.

There are also orange and ref '*warning*' areas to let the user see more easily when the levels are getting close to clipping the samplings range of the ADC. The '*red*' range is for values above –2dBs. The 'orange' range is for values between –6dBs and –2dBs. When the metered level enter these ranges an additional rectangular 'LED' will light up to warn the user. This state will also 'dwell' to make the warning easier to notice.

Near the top-right part of the display a pair of text windows summarise the status of the use of the ADC and DAC. The upper text shows the status of the tone generation. The lower shows that of the capture.

To quit = press '**q'** or '**Q'** and then click the mouse or press the spacebar when invited.

Pressing '**d**' or '**D**' will save data from the most recent data grab to ramdisc. This will include the raw sampled values as 'Samples', the Voltage/time values as 'Voltages' and the Spectra as 'Spectra'. More details of this below.

Pressing '**s**' or '**S**' will dump a sprite of the display to ramdisc. (Note this requires just over 8MB per sprite if you are using a 16million colour screen mode!)

The saved results are given filenames which include an incrementing number. So you can choose to do more than one Data save or Screensave and each one will provide files with a changed name during a session.

Saved file details

Each time you press 'd' or 'D' the saved data will be in four files: *ToneWaveNNN*, *ValuesNNN*,

VoltagesNNN, and *SpectraNNN* where '*NNN*' is an integer that will increment for each save.

SamplesNNN consists of three values per line
`<sample number>, <Left sample>, <Right sample>`

all as integers. These represent the raw values obtained from analogue to digital conversion by the USB Audio ADC.

VoltagesNNN consists of three values per line
`<time>, <Left voltage>, <Right voltage>`

These values are now all floating point and have been calculated from the raw input using the scaling factor in your **Settings** file.

SpectraNNN consists of three values per line
`<frequency>, <Left power>, <Right power>`

The frequency values are in Hz. The powers are the levels wrt 1Vrms expressed in decibels. (i.e. in dBV) *provided* you have set the appropriate scaling value in the 'Settings' file.

ToneWaveMM_NNN consists of three values per line
`<sample number> <Left amplitude> <Right amplitude>`

This shows the pattern of the selected test waveform (*MM*). The amplitudes are scales such that a value of unity would mean the maximum possible unclipped amplitude.

When started the program also saves out a file, *Reporttones*, to RAMDisc. This lists the set of tones read in for use.

User Waveforms

Inside the application directory you will find a text file *UserWave*. This contains a series of floating-point values, one value per line. The first number specifies the scaling amplitude of the waveform. The following values represent the series of individual sample sizes. These are scaled such that +/- 1·0 corresponds to the maximum possible level when the amplitude is also 1·0.

When a ‘U’ wave is requested the program reads in these values to define the resulting wave shape for both channels. Note that in the example given there are are only 161 lines. i.e. one line to specify the actual scaling, and then 160 sample values. The program assumes these define one cycle of the required waveform, so having read them, it repeats them for the duration of the resulting test tone burst. This means that the nominal cyclic frequency of the User waveform is actually given by the ratio of the output (tone) sample rate chosen to the number of samples in the UserWave file. e.g. with a 48k sample rate this would be 300 Hz for the given UserWave. However you can use a longer or shorter series of values as you wish, provided it is has at least one value and doesn't not exceed the buffer limit of 38400 samples/channel per tone burst.

As with the other waveforms, the program outputs the same waveform on both channels

General

This program was developed by myself and Colin Grenville working together. In effect, I worked on the screen display, signal processing, etc. Colin developed the code that allows the application to be able to connect to the chosen ADC, set the sample rate, etc, and read the data. Frankly, the USB side of this was far too complex for my brain. Put simply, Colin did the hard bits, and I did the easy bits. :-)

As a result the code for this program is of two sorts:

So far as I am concerned, the code I write is basically rubbish that people can use at their own risk and are free to alter, reuse, whatever, however they like. However Colin's code is released under a specific license, which we expect users to honour.

J. Lesurf
7th Apr 2021