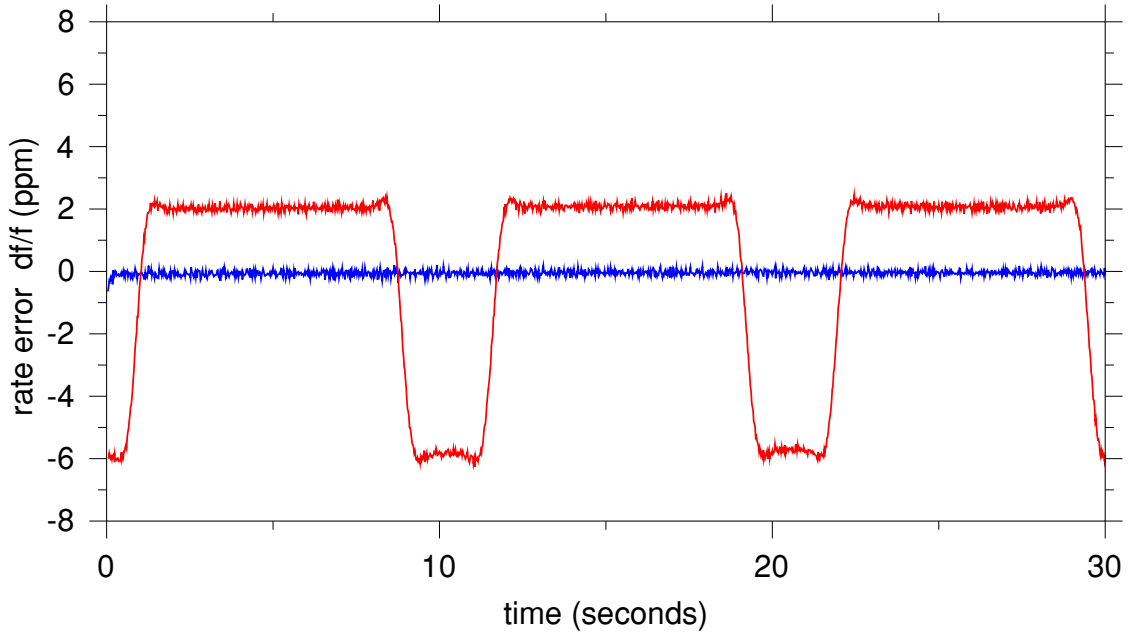
Going around in circles

When playing digital audio files and sources we have to supply a DAC (Digital to Analogue Convertor) with a series of digital sample values which then converts these into the required analogue waveforms. This process and the information carried are normally described in terms of a sequence of binary numbers. So we might expect that all that is needed is to supply the DAC with the correct series of numbers in the right order. Ideally, the samples being presented are converted in a perfectly uniform clock-like manner. The time interval between each sample and the next should be perfectly regular and uniform.

Alas, work on practical systems show there are often timing errors or irregularities which can upset the details of the analogue output. This is where the time between successive sample value conversions keeps varying in some way. The result is a subtle distortion of the output analogue waveforms. The departures in timing from perfectly uniformity are called 'jitter'. If you look at audio magazines you can often see measurements on CD (and DVD) players that show some kind of spectrum and give a measured 'jitter' value, usually in units like picoseconds (ps).

For domestic audio players like CD the standard way to measure this is to use what has come to be called the 'J-Test'. The 'J' in the name is for two reasons. One is the obvious – the initial of 'jitter'. The other is that the standard method was devised by an engineer called Julian Dunn. The technique he devised is aimed at probing the specific weaknesses of the way CD and DVD players tend to send the digital data to the DAC. Alas, computers work in a very different manner when they are used to play sound files or internet radio streams. So the J-Test may not be the most useful way to test using a computer to play music.

I've been puzzling over this for some time and I finally came up with an alternative approach which doesn't make the same presumptions as the standard 'J-Test' about the likely defects of what is being tested. For various reasons I couldn't resist calling it the 'IQ-Test'...



The above graph shows the results of applying the IQ-Test to my Shuttle home computer system when it plays a 48k sample/sec 16bit sound file using a good external DAC (Cambridge Audio DACMagic). The vertical scale (rate error) essentially shows the rate of replay of the file. The horizontal scale shows time (and the series of samples) passing. Note that the graph *isn't* displaying the output analogue waveforms being played. It shows how *quickly* the waveform is being played. In effect how 'sharp' or 'flat' it might sound, and how that varies with time.

There are two lines plotted on the graph. One is essentially a flat line at about 0. This shows the system playing the sounds at a uniform rate. The second line regularly jumps up and down from +2 to –6 ppm. This shows the rate of replay changing in a way that it shouldn't. The jumps occur when I use a direct USB connection. The steady uniform result is when I send the sound data via a Halide Design USB to SPDIF Bridge (convertor). The improvement is quite obvious. However the 'jumps' are on a timescale that the standard 'J-Test' would probably miss detecting.

Any communications engineers who read this might already have also spotted that 'IQ' may be more than a pun on 'J' or my thinking my new test is clever. IQ modulation and demodulation is a common technique in modern digital communications.

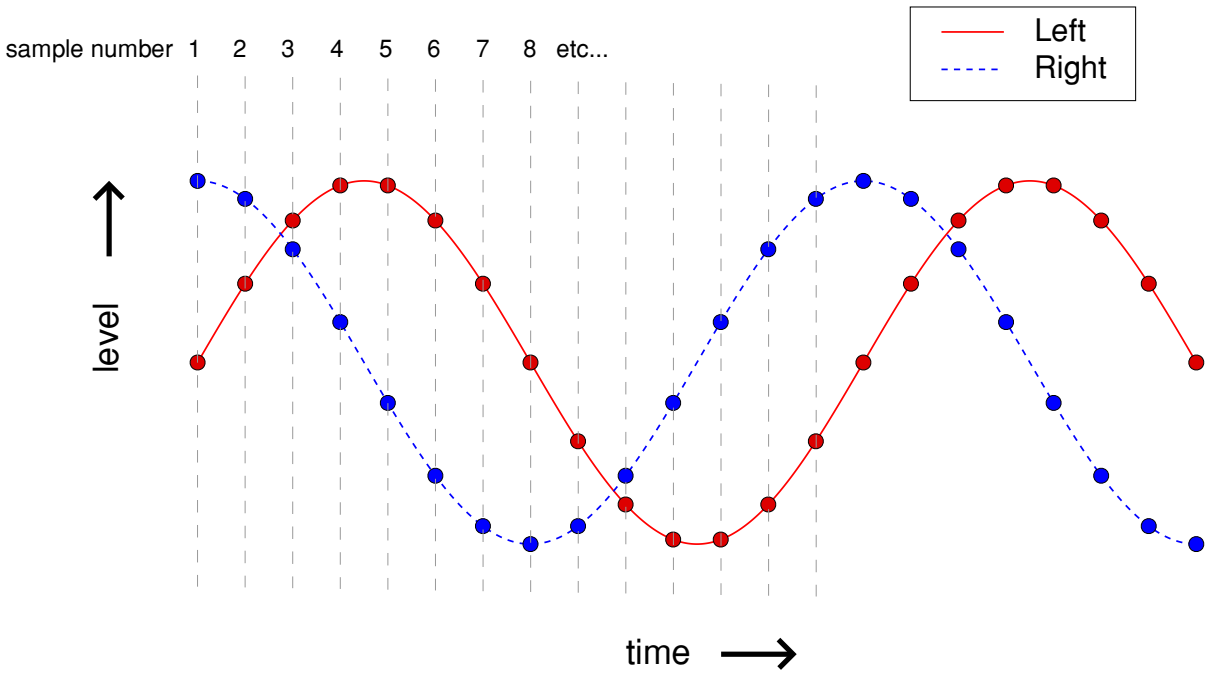


Figure 2 Left (I) and Right (Q) waveforms and sequence of samples

The trick is to use the left and right hand channels of the stereo signal to carry a related pair of waveforms – a sine and cosine wave of the same frequency and amplitude. This produces an IQ waveform like the example shown in Figure 2. You can think of the left and right pair of values at any instant as being the real and imaginary parts of a complex number. Communications engineers tend to think of them as the 'In phase' (I) and 'Quadrature' (Q) parts of the signal. The advantage of this approach is illustrated in Figure 3.

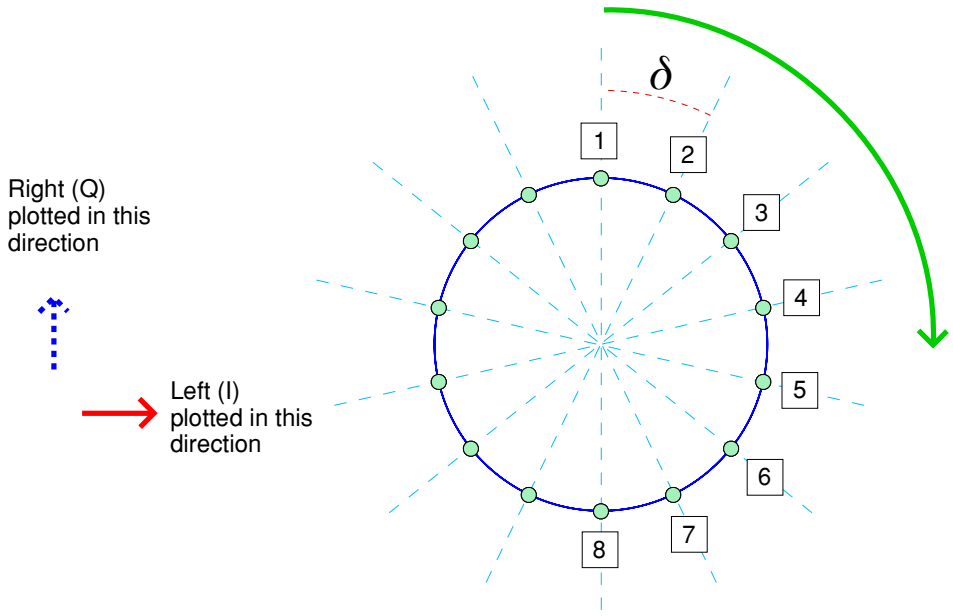


Figure 3 Left/Right sample pairs step around a circle

This shows the sequence of left-right values plotted on an XY-graph. The effect is to step around a circle because the left and right values are essentially the sine and cosine of a value which we can think of as a steadily changing angle. The result is like a hand moving around a clock-face. It allows us to represent the passage of time. When a DAC converts the sequence of left-right value pairs into a pair of output analogue waveforms we can then use them to 'tell the time' at any instant. If the replay process is converting the series of left-right pairs with precisely correct and uniform spacings then the clock we measure will run accurately and smoothly. Any delays or variations in the times at which each successive left-right pair are converted to analogue will mean the 'hand on the clock' won't be rotating smoothly at the correct rate.

In terms of 'C' programming the way this works can be outlined as follows.

Start by defining a test IQ waveform frequency and amplitude. Choose the sampling rate of the test waveform to be played by the system via the DAC and from that choose the time interval between successive sample pairs. Then use this to create an LPCM Wave file containing the required sequence of stereo sample pairs. Play the file and record the results using some type of digital recorder. Note that the sample rate, etc, of the recorder don't have to match exactly the rate of the test signal since we are recording the analogue output of the DAC. In principle you can use sound inputs to another computer to make the recording and collect the data. However to do this you would need to be happy that this second computer doesn't have its own timing problems!

This gives us a recorded set of data values we can call `left[i]` and `right[i]` taken at a series of recorded instants `t[i] = i*dt`, where `dt` is one over the sampling rate of the recording we made from the DAC output. For each sampled instant we can now calculate a phase (angle of the hand on the clock face) `phi[i]=arctan2(-left[i],right[i])`. To determine how swiftly the clock hand is moving (i.e. the actual rate of replay) we can then calculate the change in phase between each successive pair of recorded values. `dphi[i]=phi[i]-phi[i-1]`. By default in 'C' these values will be in radians.

You can also argue that the frequency of a sinusoidal wave is related to the rate of change of its phase. So we can then work out the effective frequency during each interval between successive

pair of samples using **`f[i]=dphi[i]/(2.0*pi)`**. This can then be used to determine the actual rate at which samples were converted and played out as an analogue stereo waveform. Thus discovering if the overall rate was incorrect, and find any fluctuations. In ye olde analogue terms these variations might have been called “wow and flutter”.

Note that **`arctan2()`** always gives a value for the phase within the range +/- pi. So every now and then the values we obtain for **`dphi[]`** will ‘jump’. This will happen once per rotation of the ‘hand around the clock’. The analysis program therefore has to keep an eye on this and whenever the value differs from its neighbours by more than pi we need to adjust it by +/- 2*pi to undo this jump. This correction ‘stitches together’ the series of phase change values and removes the effect of movements across the boundary of the arctan function. In practice it is useful to make recordings with a high sample rate and with 24 bit resolution to maximise the accuracy of the results. That said, recordings at, say, the CD standard should be quite OK for basic measurements and checks on a decent audio system. However the results in Figure 1 were based on recording the DAC output with a recorder running at 192k samples/sec and using 24bit values. This makes it easy to see any variations in replay rate down to the level of a part per million (ppm) or less.

As usual I’ve provided a copy of the two RISC OS applications I used. One, **`!WAV_IQGen`**, will generate an LPCM Wave file containing an IQ sin/cos stereo pair which can then be played on a system you wish to test. The other, **`!WAV_IQTest`**, lets you analyse a recording you have made of the analogue output from the system and DAC being tested. Each application includes a !Help file that gives more details on how to use the applications. At present the Test examines variations over the medium and long term – i.e. timescales from a fraction of a second up to over a hundred seconds. But I also plan to write another analysis program that examines timing variations on a sample-by-sample timescale and give results more directly comparable with the traditional ‘J-Test’.

1550 Words
Jim Lesurf
30th Mar 2011