January 2015 • RadCom
Mike Richards, G4WNC • e-mail: mike@photobyte.org

Getting started in... Software Defined Radio

How can SDR enhance your radio reception?

STARTING OUT IN SDR. Software Defined Radio (SDR) is revolutionising radio equipment design and delivers a level of performance and flexibility that was previously unattainable. In this article I will attempt to explain some of the technology behind SDR and show you how you can use the technology to enhance your radio activities.

WHAT IS SDR? There are many definitions of SDR and the most universal is that produced by the IEEE, which states SDR is: 'Radio in which some or all of the physical layer functions are software defined'. From an amateur radio viewpoint, this can range from the use of software to handle the final demodulation and filtering, through to a complete digital transceiver. SDR delivers many benefits but some of the highlights are: steep sided continuously variable filters, panoramic displays, multiple receivers, point and click tuning, spectrum analysis, spectrum recording and more. Many of these features would only have been possible in top range professional receiving equipment just a few years ago. Now you can enjoy these facilities for the investment of just a few tens of pounds! The technology behind SDR has been around for many years but it's the availability of low cost hardware and open source software that has driven today's boom in SDR.

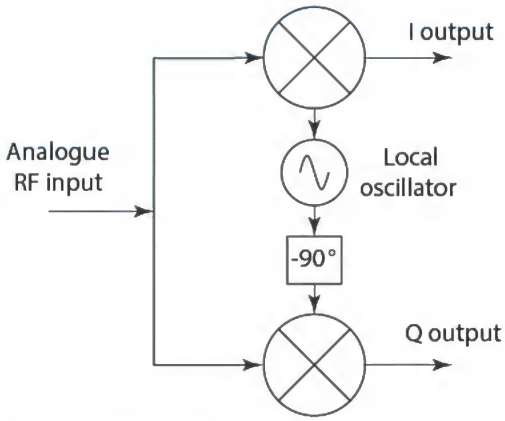

FIGURE 1: Generating I/Q signals using a pair of identical mixers.

I/Q SIGNALS. Before I move on to look at practical SDR systems, it's worth spending some time to understand a few of the key technologies that make SDR possible. One technique that is common to all SDR systems is the use of In-phase and Quadrature signals, commonly known as I/Q signals. The use of these signals is not new and was at the heart of the 'phasing' method of creating SSB developed by R V L Hartley in the 1950s. However, achieving precise phase shifts over a wide bandwidth using analogue circuitry was always a problem and the method was little used. The march of the microprocessor has revitalised that work and I/Q signals are now at the forefront of all things connected with digital radio.

The I/Q signals comprise two versions of the same signal but with one delayed by 90 degrees with respect to the other. 90 degrees is one quarter of a 360 degree cycle, hence the term quadrature, and you will often hear these signals described as quadrature signals. The simplest way to create the I/Q signals is to apply the original signal to two identical mixers with a common local oscillator but with one local oscillator feed delayed by 90 degrees, **Figure 1**. The result is I and a Q outputs.

So why are I/Q signals so important to SDR? A simple way to visualise the importance is to consider the way in which our brain/ear combination deals with sounds. You will be aware that we can instinctively locate the source of a sound without moving or using our eyes. That is possible because our brain/ear combination is able to sense small phase differences between the sound arriving at each ear and use those differences to determine the location. A similar effect occurs with I/Q signals but we use the difference between the I/Q signals to indicate the movement of the signal. This enables us to determine whether the frequency and amplitude of a sample is increasing or decreasing. Once we know the movement of the signal we can use that information to demodulate the signal.

To be able to realise the full potential of I/Q signals we need to convert them to a digital format and for that we need analogue to digital conversion.

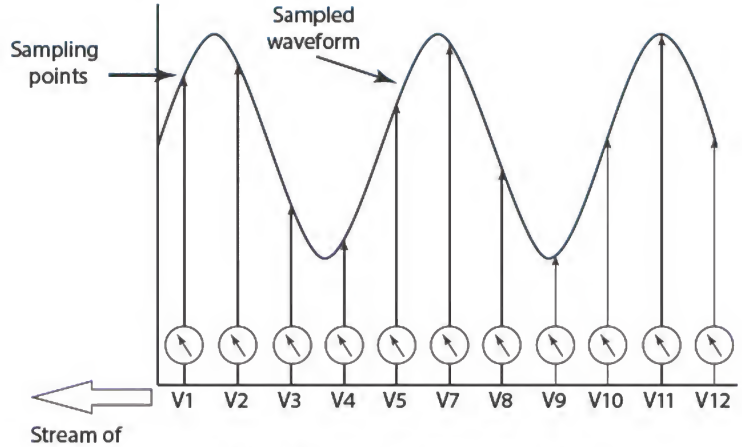

FIGURE 2: Sampling an analogue signal by taking regular measurements.

ANALOGUE TO DIGITAL CONVERSION. The basic process of converting a signal from the analogue world to the digital domain is extremely simple. Computers can only deal with numbers so the process of analogue to digital conversion (ADC) entails converting the incoming signal into a stream of numbers that accurately reflect the content of the original signal. The

January 2015 • RadCom
Mike Richards, G4WNC • e-mail: mike@photobyte.org

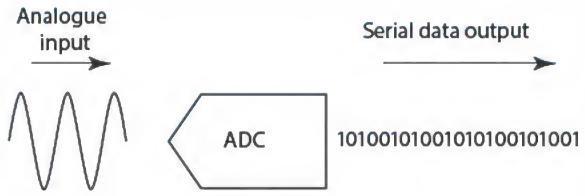

FIGURE 3: Using an analogue to digital converter to produce a stream of data from an analogue signal.

technique for achieving this is simply to take voltage measurements of the incoming signal at regular intervals, see **Figure 2**. This measurement technique is called sampling because we are taking samples of the signal as it arrives. I'm sure you can see that, if we want to record an accurate representation of the signal, we will need to take regular samples at frequent intervals. The question is, how frequent? The definitive work on sample rates comes from a paper published in 1928 and known as the Nyquist-Shannon Sampling Theorem and more commonly called the Nyquist theorem. This shows that the sampling rate of a signal needs to be more than twice the highest frequency contained within the signal. Let's look at a common example. If we wanted to digitise an audio signal containing frequencies up to 20kHz we would have to sample at a rate of more than 40,000 samples per second or 40ksps. In practical systems, rates of 44.4ksps or 48ksps are usually used.

Now we need to consider the measurement resolution of each sample. We know that computers can only handle numbers but it's more prescriptive than that as the numbers need to be in binary format. Let's just briefly look at binary numbering as it's really very straightforward. In binary numbering there are only two possible digits, ie 1 or 0. The reason this works well for computers is that all the data lines have just two possible states, either on or off, which represent binary 1 or 0. Looking at **Table 1**, you can see that larger numbers are made simply by adding more digits, with each added digit having a different weighting. Starting from the right of Table 1, the first digit signifies 1 or 0 whilst the next shows the number of 2s and the next the number of 4s, etc. I've populated the table with a few examples so you can see how 8 binary digits can be used to represent any number from 0 to 255. At this point, I need to introduce a couple more computing terms. Each digit in a binary number is called a 'bit', which is a simple shortening of BInary digiT. Most computing systems also work with multiples of 8 bits and you will no doubt have seen computing systems specified as 8-bit, 16-bit, 32-bit or more recently 64-bit. In computing terms a group of 8 bits is called a 'byte'.

Getting back to our sampling, you can see that if we use a sampling system with 8-bit resolution we will have 256 measurement steps available, ranging from 0 to 255. However, if we increase the resolution to 16 bits we have a much finer resolution of 65,536 steps. If we convert those steps to dB, then 8 bits gives a range of 48dB from the weakest to the strongest signal whilst 16 bits give a much more useful 96dB range. The output from most ADCs comprises a string of binary numbers that's passed over a very high speed serial connection, see **Figure 3**.

TABLE 1: Binary numbering.

Decimal weight	128	64	32	16	8	4	2	1
0	0	0	0	0	0	0	0	0
17	0	0	0	1	0	0	0	1
29	0	0	0	1	1	1	0	1
230	1	1	1	0	0	1	1	0
255	1	1	1	1	1	1	1	1

FAST FOURIER TRANSFORMS. Don't panic, I'm not going to start delving into complex maths, but an overview of Fast Fourier Transforms (FFTs) will help you better understand SDRs. Let's start by assuming we have a signal that's been digitised. In that case, the output from the ADC will comprise a stream of binary numbers, as in Figure 3. How can we find out what frequencies are contained within that data stream? This is where FFTs come to the rescue. The Fourier series is a mathematical solution that allows any waveform to be decoded into a combination of sine waves of different frequencies. As you can imagine, Fourier analysis is a complex process that takes a lot of computing power. However, the Fast Fourier Transform (FFT), as the name suggests, is a quick way to carry out Fourier analysis with minimal processing power. FFTs operate by first of all taking a segment or block of the incoming digital data. The FFT then creates a number of bins (data stores), each of which is used to measure the power of a narrow frequency band from the incoming data segment, see **Figure 4**. As the data stream is processed, the bins are populated with the appropriate energy level for their narrow band. In an audio system with a sample rate of 44.4ksps and a FFT size of 4096 bins, we could expect to see audio frequencies from 0Hz to 22kHz. In this example each FFT bin would have a bandwidth of 5.37Hz (22,000/4096). Once the incoming data stream is split into sub-bands or bins we can use that information to drive a spectrum analyser display. Alternatively, we could use a similar principle to pass only signals from a range of bins, thus creating a very steep sided filter.

One of the problems with using FFTs occurs at the beginning and end of each data segment. For the FFT to work accurately all the signals in the sampled segment have to be periodic, ie contain an exact number of complete cycles. That is obviously not practical in a real system as some signals will have been abruptly cut off at the start and end of the segment, see **Figure 5**. The result is rather like a key-click, with spurious readings being shown in the FFT bins. The spurious readings are called spectral leakage and spoil the accuracy of the FFT. The simple solution is to artificially reduce all the signals to zero at the beginning and end of each data segment. This process is called windowing as it's rather like shaping the segment to fit through a window. Not surprisingly there are a number of different windowing shapes, each with its own set of benefits and drawbacks. One popular window is

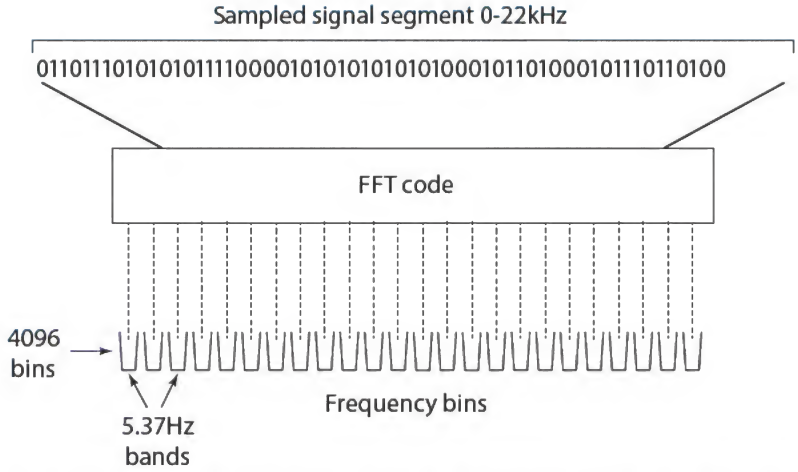

FIGURE 4: Illustration of the use of FFTs to analyse the frequency content of a digital signal.

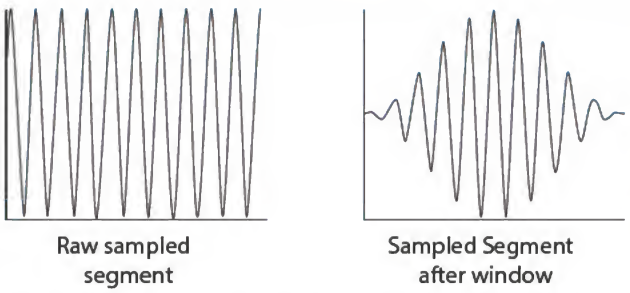

FIGURE 5: Comparison of a raw signal (left) and after a window is applied (right).

the Hanning function, which is particularly effective for measuring continuous signals. An alternative for dealing with transient signals is the rectangular window.

WHERE TO DIGITISE? Now that we've covered some of the key technologies, let's start by looking at practical SDR systems. One of the first questions has to be: where do we digitise the signal? There are essentially two main options that are driven by cost. The ideal is to digitise as close as possible to the antenna as that puts all the processing under software control and so gives the most flexibility. This is usually called Direct Digital Sampling (DDS). However, this is also the most expensive option, so let's see why. Let's assume we want to create a high quality 160m to 6m SDR system with digitisation of the full spectrum soon after the antenna. From the Nyquist theorem I mentioned earlier, the ADC needs to sample at more than twice the highest frequency in the band. In our example, the highest frequency is 52MHz so we need to sample at more than 104Msps (so let's choose 110Msps). The next design choice is the resolution of each sample. For a high quality design we ought to go for 16-bit resolution. This is where it gets scary! With these design parameters, the ADC will be sending a stream of 16-bit data samples 110 million times per second. That works out to be a data rate of 1,760,000,000 bits (1.76 gigabits) per second! The processor receiving that data needs to be fast enough to carry out a number of FFT calculations whilst also managing the user controls, display, etc. This is way beyond the capabilities of the home PC. The solution is to use a Field Programmable Gate Array (FPGA) chip. These devices contain large numbers of logic and digital signal processing (DSP) sub-systems that can be interconnected in software. In addition to being very fast, FPGAs can also run many tasks in parallel and it's this combination of high speed and parallel processing that makes them ideal for use in top-end SDR systems. The prime role of the FPGA in most SDR systems is to extract a more manageable chunk of spectrum from the incoming data and send it to the host PC as I/Q samples for final processing in software. This process is called 'decimation' and is the digital equivalent of a local oscillator/mixer/IF filter combination in an analogue radio, as shown in **Figure 6**. Although the block diagram of a modern DDS SDR receiver may look quite simple, the development of specialist FPGA software (and the ADC and FPGA chips themselves) are very expensive, hence the higher cost of this type of SDR receiver.

A SIMPLER WAY? The simplest and cheapest way to experience SDR is to use direct conversion or 'zero IF'-based system. Early examples of this can be found in the Flex Radio and SoftRock designs. The technique is still very much in use and the popular FUNcube Dongle series are based on a direct conversion design. The low cost of these systems is made possible thanks to the good quality soundcards that are found in most modern computing systems. With a direct conversion SDR receiver, the filtered and amplified RF signal is applied to a pair of identical mixers that are fed from a common local oscillator at the radio frequency. The local oscillator signal output to one of the mixers is delayed by 90° to produce the I/Q signal, see **Figure 7**. The output from the mixers is an analogue I/Q signal that is sent to the left and right channels of the host PC. The soundcard then digitises the I/Q signals, thus providing the digital I/Q stream than can be demodulated and processed by the SDR software. One of the many benefits of SDR receivers is the spectral display of a band segment along with the facility to click the mouse to tune to a signal. The tuning span available from a direct conversion SDR depends on the sample rate used by the soundcard. Many of the early systems employed sample rates of 48kHz or 96kHz, which provided a tuning range of just under 48 or 96kHz respectively. If you've been paying attention, you may just be wondering how you can have a tuning spectrum that's the same as the sample rate: surely a 48kHz sample rate can only support 24kHz of bandwidth? Don't worry, this is a common point of confusion for those new to SDR. The answer comes from some nifty programming in the SDR software. If you think back to mixer theory, you will recall that the output of a mixer contains both the sum and difference frequencies. The SDR software is able to use the I/Q signals to separate these two bands of frequencies. As a result, we can access 0 to +24kHz and 0 to -24kHz, which makes around 48kHz total. One of the common drawbacks of zero IF SDR systems is the central spur in the display that occurs at the changeover between the two bands. There are two factors in play here. In order to have a smooth progression from the +24kHz range to the -24kHz range the sound card has to have a completely flat response down to 0Hz plus there can be no DC offset between the I and Q channels at the central point. In practice this is not achievable and the result

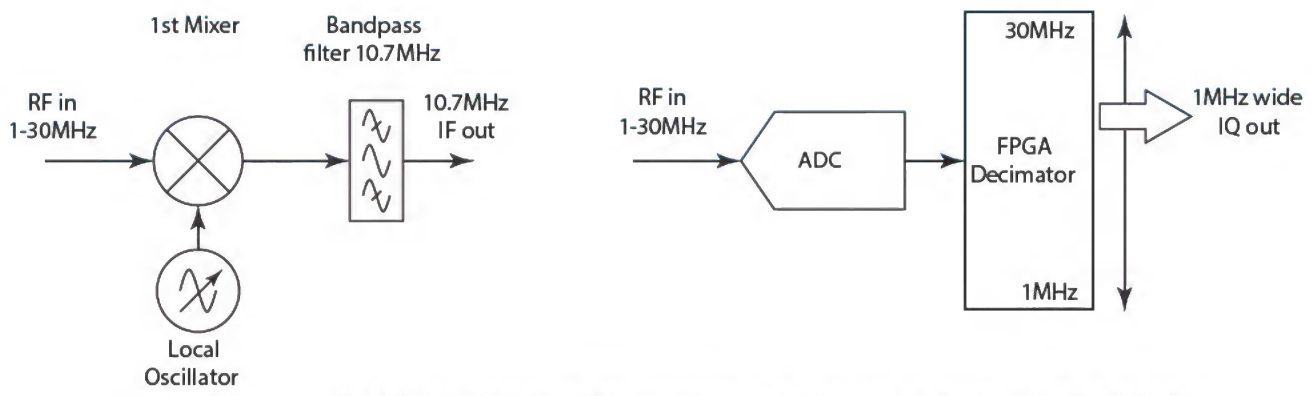

FIGURE 6: Comparison between an analogue first mixer and decimation in a digital receiver.

January 2015 • RadCom
Mike Richards, G4WNC • e-mail: mike@photobyte.org

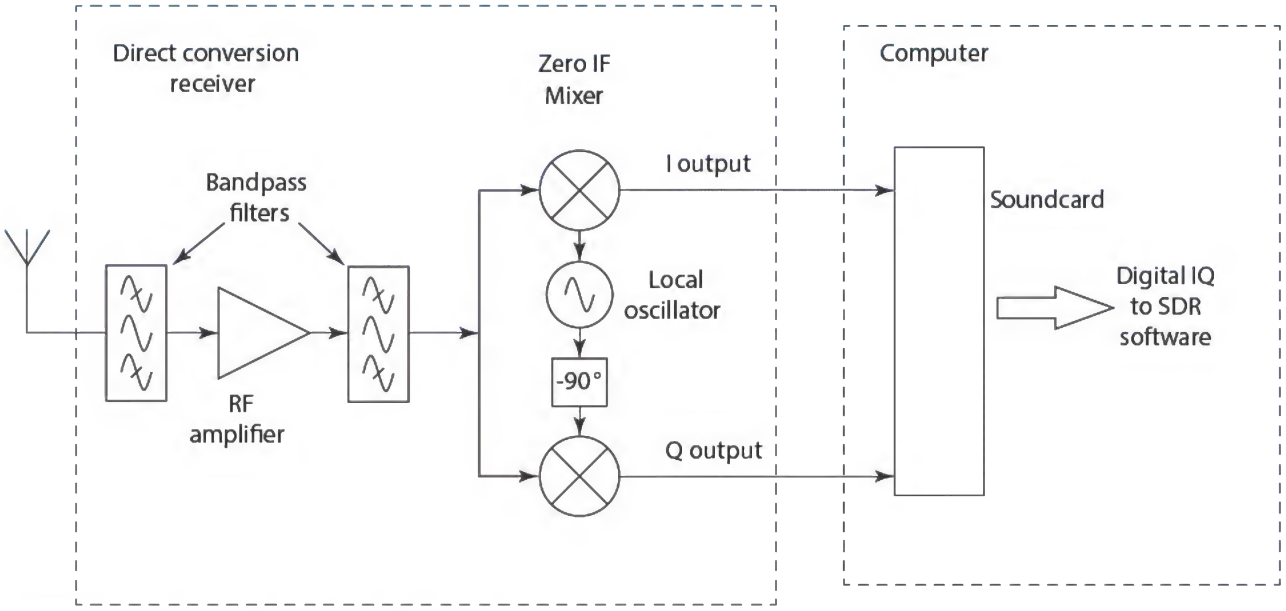

FIGURE 7: Illustration of a direct conversion SDR receiver.

is the response spur you see at the central crossing point. Most current SDR software systems include very effective techniques to automatically reduce the central spur. Another option for dealing with the spur is to include a dedicated soundcard chip in the SDR hardware. Using this technique, the designer can configure the chip to keep the LF response as flat as possible and minimise the DC offset. This has been done in the FUNcube range; the FUNcube Dongle Pro+ includes a high quality audio chip that can sample at 198kHz, thus providing just under 200kHz of SDR tuning spectrum. If you have an SDR receiver that uses your computer's soundcard, be very wary of laptops and cheap USB soundcard dongles. Many of these devices only provide a mono input so they are unsuitable for SDR use. The problem can be solved by adding a good quality USB soundcard that has stereo inputs.

SDR TRANSMITTERS. The generation of an SDR transmit signal is surprisingly straightforward, as shown in **Figure 8**. In a top-end direct digital synthesis system, the microphone signal is first digitised and then subjected to digital filtering and processing to shape the audio. This signal is then up-sampled to the required transmit frequency in the FPGA and passed to a high speed digital to analogue converter (DAC). This produces the RF signal that is then amplified and filtered using conventional analogue techniques. One of the newer features that's being built into the top end systems is pre-distortion. Efficient high power PAs suffer from a variety of amplitude and phase distortions that result in unwanted output. In a pre-distortion system, the digital transmit waveform is modified to follow the inverse of the PA distortion characteristics, which has the effect of reducing the unwanted output.

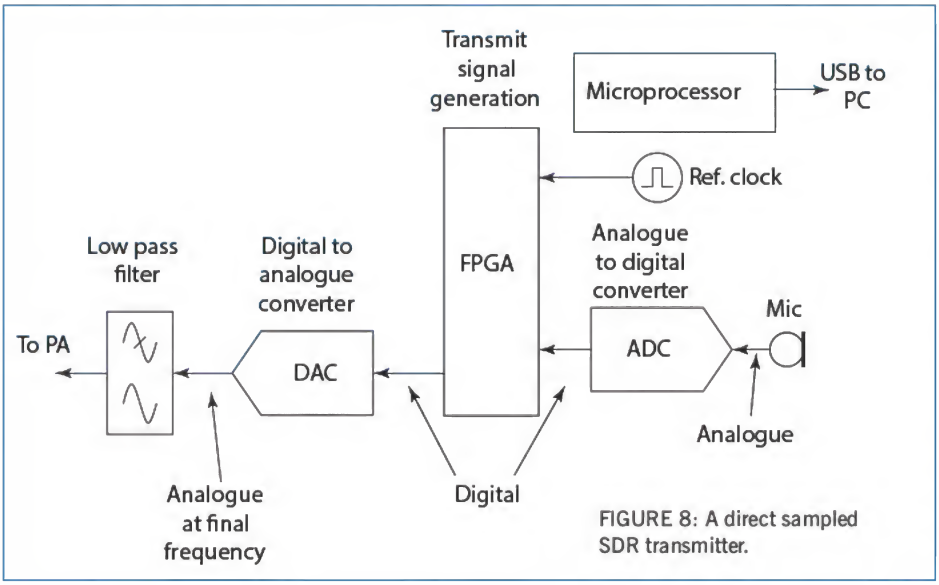

FIGURE 8: A direct sampled SDR transmitter.

Pre-distortion is very effective and has enabled systems such as the FLEX-6000 and ANAN series of SDR rigs to produce extremely clean transmit signals.

As with the receive techniques described earlier, there is a simpler way to generate a transmit signal. The SoftRock SDR transmitter operates by creating the transmit signal in software on the computer and presenting the output as I/Q analogue audio signals on the left and right computer soundcard. These signals are then applied to a quadrature mixer which is effectively the same as our two mixers with a 90° lag applied to one of the local oscillator sources, as shown in **Figure 9**. The result is the desired transmit signal that is ready for final amplification and filtering.

PRACTICAL SDR. By far the cheapest entry into SDR is to use one of the DVB-T digital TV dongles. The only ones that work are those that use the Realtek RTL 2832U device. This is because the RTL 2832U has an undocumented feature that enables it to send I/Q signals via the USB port. Once this mode has been activated, the I/Q signals can be routed to common SDR software to produce a receiver with a tuning range of around 64MHz to as much as 1400MHz. The simplest software to use with one of these dongles is *SDR Sharp* (*SDR#*). The team at Adafruit in the US have put together an excellent tutorial on how to use a dongle and install the *SDR Sharp* software, which be found at http://goo.gl/qhGyF4

SDR SOFTWARE INTRODUCTION. For this section I'll use *SDR Sharp* as it is very popular and has the simplest interface. I've shown a screen shot of *SDR Sharp* in **Figure 10**. Let's start at the top left where you'll

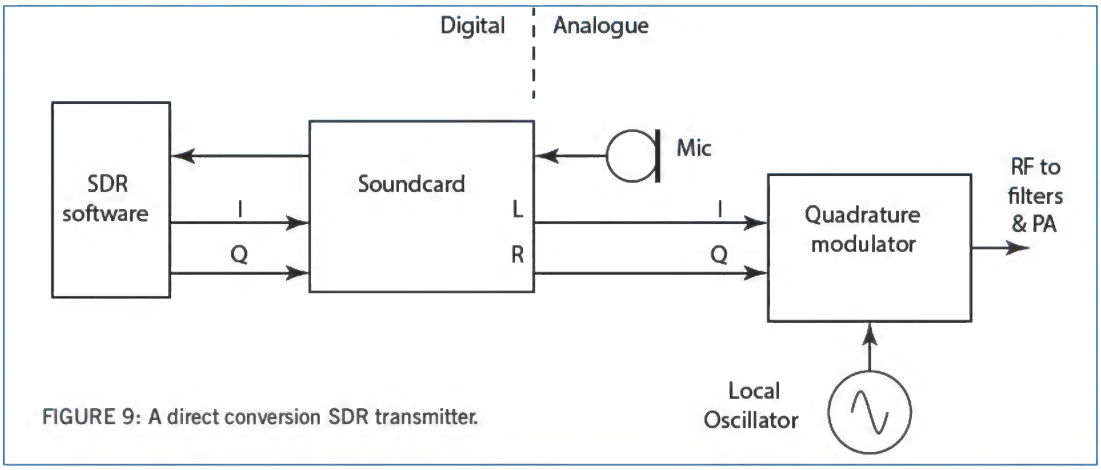

FIGURE 9: A direct conversion SDR transmitter.

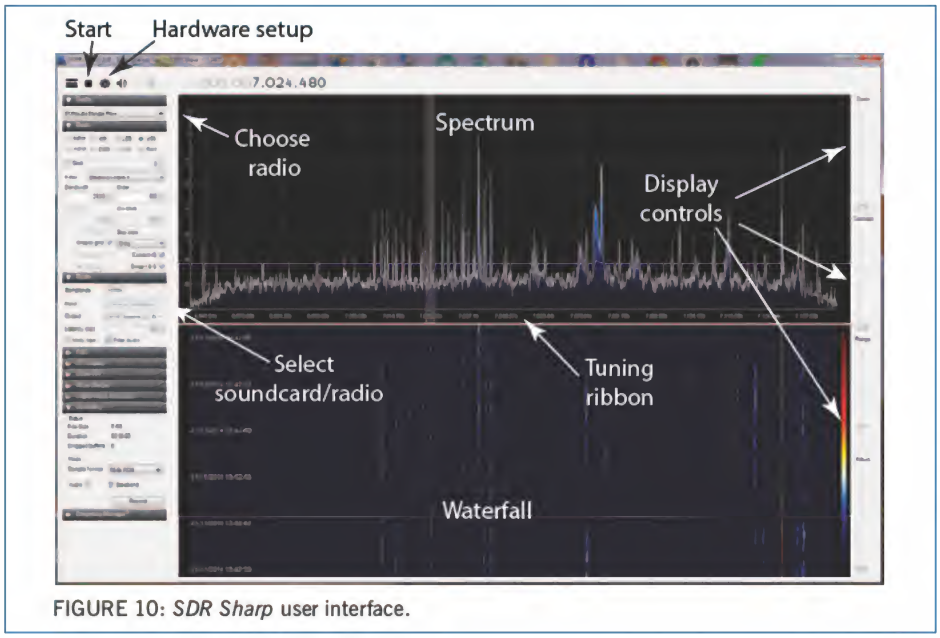

FIGURE 10: *SDR Sharp* user interface.

find a number of important controls. The first symbol with three horizontal lines is used to show or hide the main control panel that normally sits to the left of the display. Next is the Play/Stop button that starts and stops the receiver. In addition to its obvious use, it can be helpful to stop the receiver when you want to take some measurements using the main spectrum displays. The cogs symbol is used to denote the SDR hardware configuration and opens up a panel that allows you to adjust the hardware settings of the attached receiver. Continuing along the top, we find the speaker symbol that mutes the audio, which is followed by the volume slider. Finally, on the top row, we have the frequency display. In addition to providing a readout of the current tuned frequency, this display can be used to change frequency quickly. By placing your mouse pointer over any digit you can change its value by scrolling the mouse wheel or by clicking on the top or bottom of the digit. Not only is this useful for making large frequency changes, it makes precise frequency setting very easy.

Before I move on to the control panel, let's have a closer look at the main display. The top section is a conventional spectrum display with a vertical scale indicating amplitude and the horizontal scale showing frequency. In its default state it displays the full tuning bandwidth available from your SDR hardware. However, you can zoom-in for a closer look at a signal by sliding the Zoom control to the right of the screen. If when doing this the display becomes too block-like you can increase the resolution using the FFT Display panel on the left. A useful setting for most amateur signals is 65536. You can also control the scaling of the display using the Range and Offset sliders to the right of the display. In addition to providing a useful view of the signals, the spectrum display can be used to take measurements. As you move your cursor over the screen, you will see that the software displays both the frequency and the signal level at the cursor. All you have to do is place your cursor over the point you want to measure and note the result. In the waterfall display the horizontal scale is still frequency but the vertical scale is time and the brightness of the trace is controlled by the amplitude of the signal. This gives you access to three parameters in a single display. The second slider on the right gives access to the Contrast control that you will need to adjust to get the best display on the waterfall. One of the main uses of the waterfall display is as a tuning aid. If you want to tune in to a signal where the transmission has just stopped, you can use the trace on the waterfall display to find the correct tuning point and double-click to complete the tuning. The remaining controls on the right hand side are the Range and Offset and you can use these sliders to adjust the position of the trace on the screen.

In between the spectrum and waterfall displays is a very handy tuning ribbon that is great for tuning around within a band. If you left-click and hold you can drag the frequency display from side to side.

The control panel has a number of sections that can be expanded or contracted using the white arrow at the side of each entry. At the top of this section is the main receiver controls where you set the receive mode, filter bandwidth and step size. The bandwidth is adjustable in 10Hz steps and can also be adjusted by dragging the highlighted area next to the tuning point on the spectrum display. You will also see a couple of tick boxes for Correct IQ that will reduce the central spur on direct conversion hardware and Swap I & Q for hardware that has the I/Q signals reversed.

The Audio panel gives access to the soundcard sample rate and this is usually left at the default for your receiver hardware. The Input section of this panel is where you have to select the appropriate soundcard if your using a soundcard based receiver such as the FUNcube Dongles.

The final panel that I'll cover in here is the Recording panel. This is a very powerful option that allows you to record the entire received spectrum to your hard disk. Once you've made a recording you can continuously replay it, which can be very useful for signals analysis work. Thanks to the advent of SDR receivers and I/Q recording, Signals Intelligence services are now able to record huge swathes of the RF spectrum for automated analysis by powerful computers. For the radio amateur it's perfectly feasible to record an entire band to validate contest entries.

SUMMARY. In this brief introduction to SDR I hope you have understood a little more about the underlying technology and have seen enough of *SDR Sharp* to encourage you to have a go.