Computers in Amateur Radio

by Steve White, G3ZVW

with contributions by other authors

Radio Society of Great Britain

Contents

2.
Datamodes

by Steve White, G3ZVW

Computers have revolutionised datamode operation for radio amateurs. With their advent we were no longer restricted to the use of huge, clunking, mechanical teleprinters, although there is nothing to prevent us from using such equipment if we choose. Neither were we restricted to the modes that these machines were capable of employing – invariably only one per type of machine.

The first advances in electronic datamode operation took place pretty much as soon as personal computers became available, usually by the use of an external modem. By about 1980 there were modems and software packages available for the computers of the day; machines such as the Tandy TRS-80. IBM introduced their Personal Computer in 1981, but it was too expensive for most people to consider having one at home. Besides, in the first half of the 1980s Sinclair Electronics established a strong presence in the UK computer market, first with the ZX80 and later the ZX Spectrum. Huge numbers were sold and a wide variety of communication (and other) programs written for them. There were numerous other brands and types of computer available for the home market, but software wasn't compatible across the platforms.

Code tables for the common datamodes described here can be found in the Code Tables chapter.

Morse

The original 'datamode', Morse code is attributed to Samuel Morse, but the Morse in use today – and which has been is use for over 150 years – is not the same as the code developed my Samuel Morse.

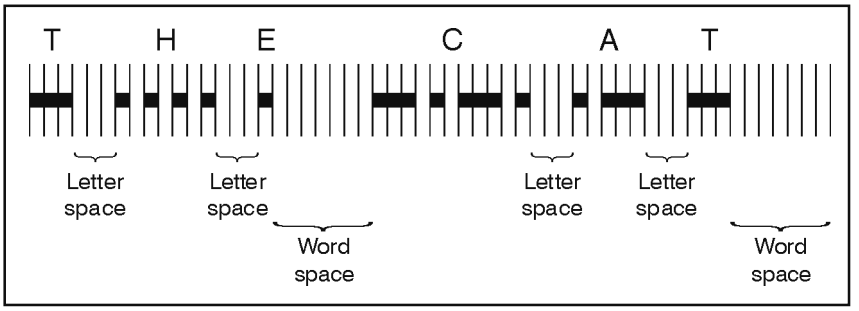

Fig 2.1: An example of two words sent in Morse, with vertical lines to show the official timing and spacing.

With Morse, the dots, dashes and spaces are all intended to vary in proportion to one another, as transmission speed varies. Depending on the speed of transmission, it may sound like a heterodyne (steady whistle) that is switched on and off. As **Fig 2.1** shows, a dash (usually pronounced 'dah') should be three times the length of a dot (pronounced 'dit'). The space between the letters of a word should be the same length as a 'dah', i.e. three times the length of a 'dit' and the space between words should be seven times the length of a 'dit'.

Depending on its use and the ability of individual operators, Morse may be sent at widely differing speeds. On VLF it may be sent *extremely* slowly. This is known as QRSs operation, where a 'dit' may be many seconds long. Such slow Morse would be sent by computer, because manual operation would be too tedious. At the receive end, it would also be detected and displayed on a computer. This is because at extremely slow speeds a computer running a Fast Fourier Transform program can detect and display a signal that is not audible to the human ear.

The speeds that are commonly sent and copied manually by an operator range from about 5-40 words per minute, although speeds in the 20s are the most common.

Prior to the development of modern datamodes, looped recordings of high-speed Morse (often at 80 words per minute) used to be employed by Meteor Scatter enthusiasts. Whilst incomprehensible to the human ear, repetitive high speed Morse messages sound slightly rhythmic. On receive, fragments of transmission (for that is all that are typically received in a meteor scatter contact) would be recorded on tape and then played back at slow speed, for the receiving operator to copy by ear.

Morse is normally transmitted by on/off keying (Amplitude Shift Keying), although many beacon stations use Frequency Shift Keying. Although not often referred to in such terms, Morse code is the International Telegraphic Alphabet No.1.

Baudot (RTTY)

The so-called International Telegraphic Alphabet No.2 was developed for numerous reasons, not the least of which is that Morse is not easy to

receive and decode with a mechanical machine. This is primarily because, irrespective of the sending speed, not all the characters take the same length of time to transmit. Emile Baudot overcame this problem in the 1840s, by developing a code in which all the characters were the same length. This permitted receiving equipment to be synchronised for each character sent.

The construction of RTTY data is shown in **Fig 2.2**. The standard Mark condition tone is 2125Hz (logic '1') and the Space condition tone is 2295Hz (logic '0'). The grey areas are 1's or 0's, depending on the characters being sent. Transmission of each character commences with a 'start' bit. The length of the start bit – indeed all the bits – depends upon the rate of transmission. For a 50 baud transmission (as used by the Telex network), the length is 20 milliseconds. For 45.45 baud transmission (as commonly used by radio amateurs), the length is 22 milliseconds. The start bit is followed by five data bits (10010 for a letter D), then 1.5 stop bits. The extra length of the stop element of the character is to give mechanical teleprinters the opportunity to come to a halt before the next character arrives.

Five data bits give a possible 32 combinations, which is not sufficient for transmission and reception of the 26 letters of the alphabet, the numbers 0-9, and common punctuation marks and control characters such as a bell, new line, carriage return, etc. This problem is overcome by dedicating two of the 32 possible character combinations for Letters Shift and Figures Shift. In Letters Shift, the 26 letters of the alphabet plus some common functions (carriage return, new line, space) are possible. In Figures Shift, numbers, punctuation, a few symbols and the same common functions are available.

RTTY is normally transmitted by a process known as Frequency Shift Keying (FSK) or Audio Frequency Shift Keying (AFSK). In a loudspeaker this results in a wobbling sound, because the transmitted signal is being switched between two frequencies. During periods when no data is being transmitted but the transmitter is still on, there is a steady tone. Practically any two audio tones can be used, the standard

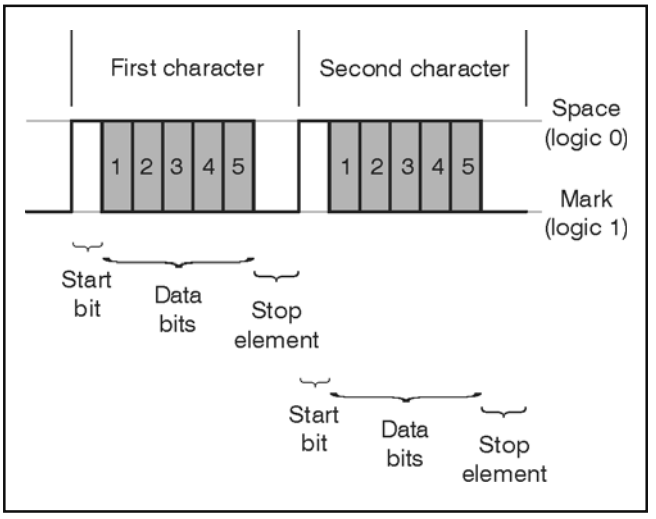

Fig 2.2: How RTTY characters are constructed and transmitted.

Mark-Space shifts being 85Hz, 170Hz, 425Hz or 850Hz. The most popular shift used by radio amateurs on HF is 170Hz. 85Hz tends to be used at VLF, because the bandwidth of antennas is extremely small, while sending RTTY at higher rates requires a bigger frequency shift and consequently greater bandwidth. When FM or SSB is being used, the two tones are usually 2125Hz and 2295Hz. These two frequencies ensure that any audio harmonics are outside the passband of amateur transmitters, because their audio cut-off is about 3kHz. Data throughput is about 60 words per minute at 45.45 bauds.

Although RTTY is limited by the fact that it contains no error detection or correction, it remains an extremely popular datamode amongst radio amateurs.

ASCII

The original ASCII (American Standard Code for Information Exchange) used electromechanical teletypes that worked at 110 bauds. This equated to an element length of 9ms.

ASCII was a significant improvement on Baudot, because it contained seven data bits. The 128 possible combinations meant that upper and lower case letters could be sent, plus a lot more symbols, punctuation and control characters.

As standard, ASCII does not incorporate error detection or correction, but it is possible to add a measure of error detection by adding a parity bit.

Although there is no reason why it should not be used in its basic form by radio amateurs, it tends not to be. Rather, it tends to be used as the core code for some other datamodes.

PSK31

PSK31 was invented by Peter Martinez, G3PLX in 1998. It employs Phase Shift Keying at 31.25 bauds and has the advantage of requiring a very narrow bandwidth (31Hz). It was designed primarily for real-time keyboard-to-keyboard QSOs, is capable to transmitting all the characters on a keyboard and has even been used to transmit small pictures.

Rather like Morse, where not all characters take the same time to transmit (with the most commonly used ones taking the least amount of time), PSK31 uses a ‘varicode’ system. More commonly used letters such as ‘e’, ‘n’ and ‘s’ employ less bits and take significantly less time to be transmitted. If transmitting a passage of text in lower case letters, a throughput of about 53 words per minute can be achieved, but if the same text is transmitted in upper case letters the throughput drops to about 39 words per minute.

Whilst not supporting error detection or correction, PSK31 is self-synchronising. It is very good at working through noise, indeed it is copyable by a computer when it is too weak to detect by ear. It has a high duty cycle though, which can easily lead to overheating if a transceiver is run at full power. Consequently it is advisable to run equipment at a significantly lower power level than its maximum. 30 watts is considered by many to be more than sufficient on the HF bands.

On air, PSK31 sounds like a pulsating high-pitched whistle. Tuning and frequency stability are critical for correct operation.

Olivia

This data mode was designed by Pawel Jalocha, SP9VRC, to work effectively over difficult radio paths, i.e. those suffering from fading, interference, auroral distortion, flutter, etc. It can work when the signal is 10-14dB below the noise level, which means that worldwide communication is possible using low power.

Olivia has 40 possible formats, this being brought about by the fact that eight possible numbers of tones can be transmitted in five possible bandwidths. There can be 2, 4, 8, 16, 32, 64, 128 and 256 tones, and bandwidths of 125, 250, 500, 1000 and 2000kHz. All these combinations result in differing characteristics and capabilities. The ‘standard’ formats (bandwidth/tones) are 125/4, 250/8, 500/16, 1000/32, and 2000/64, while the formats most commonly used (in order of use) are 500/16, 500/8, 1000/32, 250/8 and 1000/16. Data throughput is just under 60 words per minute at 1000/8.

The basic code used is 7-bit ASCII and Olivia works by sending characters in blocks of five over the course of two seconds. Multi Frequency Shift Keying (MFSK) is employed, with Forward Error Correction (FEC).

Packet Radio

Packet Radio is a datamode used for sending text messages. It is a time division multiplex system, so more than one contact can take place on one frequency at a time. It also includes full error correction. These two facets resulted in the creation of a worldwide network that supports real-time QSOs and non-real-time contacts via mailboxes. The former is akin to Instant Messaging used across the Internet, while the latter is akin to e-mail, but they were both in widespread use by radio amateurs before many members of the general public had Internet access. On HF, Packet is transmitted at 300 Bauds, while on VHF the rates used are 1200 and 9600 Bauds.

Data is transmitted in frames, which contain a start flag, an ad-

4.
Antenna Modelling

by Ian Birkenshaw, G4UWK

Introduction

The average computer-literate radio amateur might initially ask why he should bother with antenna modelling, not being an antenna designer or guru. However, every amateur is faced with limitations imposed by real estate, available supports, planning permission from the XYL, neighbours or local council and will want to maximise the radiated signal from his QTH within these restrictions.

Antenna modelling using a home computer can answer many questions, particularly in comparing one possible antenna against another. It can also answer questions about the real performance available from commercial antennas, often described in glowing terms by manufacturers or suppliers.

All antenna modelling software likely to be of interest to the typical amateur is based around a modelling system called 'Numerical Electromagnetic Code version 2' or NEC2. This was created in 1981 by the Livingstone Livermore Laboratories in California, the original client being the US Navy. Initially the system was classified, but over the years became available for general use. Originally written in FORTRAN, the code has been translated over the years for use by the Microsoft Windows operating system.

An intermediate version written in BASIC for early PC's called MININEC also exists, but has a number of problems due to it being a cut-down version of NEC2.

NEC2 works by breaking the radiating elements of the antenna to be modelled into small portions called 'segments' and summing the overall electromagnetic radiation from the current and phase on these

segments to produce the actual radiation pattern in a mathematical process called 'Method of Moments'.

Readers with a strong physics or maths background may like to peruse the original design methodology. It is available on the Internet at: www.nec2.org

A number of antenna modelling software packages are available based on NEC2. Some of these are freeware and some have to be purchased. It should be stated that the original NEC2 software is far from user friendly. It was written for professional antenna designers using mainframe computers and requires considerable additional add-ons to make it more intuitive and easier to use.

All versions define models on a 3-axis grid, X and Y orthogonally in the horizontal plane and along the Z axis vertically. Some knowledge of basic geometry (sine, cosine and tangent) is essential to model with NEC2.

All antenna elements are defined as wires made up as a number of segments.

A more powerful version NEC4 is available, with additional features beyond NEC2. This still has a restricted security status and requires the user to obtain a licence before purchasing the actual software as part of an antenna modelling package. Unfortunately this costs several hundred pounds and would only be of interest to the serious antenna designer.

A list of available NEC2 based antenna modelling software is detailed in **Table 4.1.**

Product Name	Supplier	Web Site	Free Demo Version?
4nec2	Arie Voos	http://home.ict.nl/~arivoors/	Freeware
Nec2Go	Nova Plus Software	http://www.nec2go.com/	Yes
EZNEC V5.0	Roy Lewallen W7EL	http://www.eznec.com/	Yes
NEC – Win Plus	Nittany Scientific	http://www.nittany-scientific.com	No

Table 4.1: NEC2-based antenna modelling software.

It is left to the reader to select the appropriate package for themselves. They all do basically the same job and any of them will require an investment in time to get familiar with the features. Obviously the software available as freeware or as free demo will be more attractive.

Practical Example

We will now model a real life antenna, the popular G5RV multi-band dipole, complete with transmission line feeder to show how it is done. We will use the NEC2-based EZNEC program for this.

EZNEC is de-facto the amateur standard antenna modelling package, and is relatively easy to use. EZNEC stands for Easy NEC! It is available as a download or CD and runs on Windows version 98. ME, 2000, XP, Vista and Windows 7.

The standard EZNEC V5.0 has a 500 segment capacity; the larger EZNEC+ V5.0 has 1500 segment capacity, allowing more complex or accurate models. These versions have to be purchased.

A free download demonstration version with a maximum of 20 segments is available from the EZNEC website. This is also supplied with the *ARRL Antenna Handbook* on its accompanying CD. The free demo version can be used for the example about to be described, but the number of segments will be restricted to 19 rather than the 51 used. This results in inaccurate feed impedance and SWR values above 10 MHz. However, if this limitation is accepted, the radiation plots are similar.

The capability to model with more segments allows more complex models to be analysed but be aware that processing time increases with the number of segments used.

It is impossible in this short chapter to show all the bells and whistles available or all the caveats inherent in NEC2, but the hope is it will wet the appetite of the reader. A full 185-page manual is available from the EZNEC website.

Fig 4.1: EZNEC Desktop Logo.

Fig 4.1 shows the Desktop Icon for EZNEC. (Version EZNEC+ V5.0) Double clicking on this produces the main EZNEC Control Centre screen – **Fig 4.2**.

This has a top menu, a left hand toolbar with buttons and to the right of this another toolbar with buttons.

EZNEC does not allow you to start with a blank canvas, rather one of the pre-loaded example models needs to be opened and modified to suit the antenna to be modelled. After that, EZNEC always starts up with last antenna modelled. This is automatically saved under filename 'LAST.EZ' when the program is closed. The pre-loaded models enable the first time user to find what each menu item does. Simple models of the basic antenna types, dipoles, yagis, and full wave loops are included in the preloaded library.

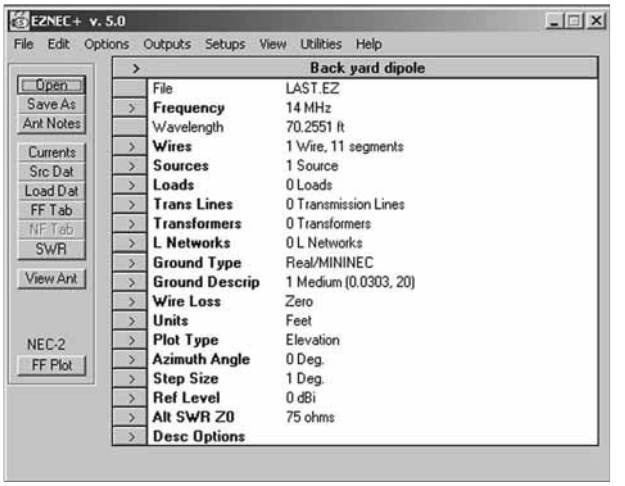

Fig 4.2: The EZNEC Control Centre Window with the sample model BYDipole opened.

Click 'File' and then 'Open' and look for 'BYDipole' in the list of supplied

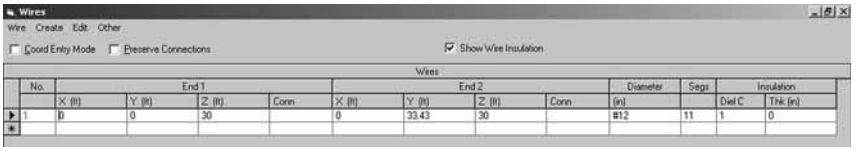

Fig 4.3: The Wires Window for BYDipole.

example models. Left click on this, then click 'Open'. The Control Centre window reveals we have opened the model file BYDipole.EZ which is a 'Back yard Dipole' for 20 metres at a height of 30ft.

Click 'Wires', which opens the Wires definition window – **Fig 4.3.** The single wire dipole is defined as a wire with End 1 and End 2 xyz co-ordinates. The wire is 12 gauge bare copper and is 33.43 feet long at a height of 30ft with 11 segments. The Wires window is where the antenna to be modelled is defined as a series of wires. We will be using only one wire, but more complex models will need many wires and it can be a tedious process to correctly define all of them. Certain tools are provided under the Create function to more easily define structures like radials, loops and helixes.

Note the last two columns, dealing with insulation. This allows antennas with plastic coated wires to be modelled. The effect of the coating is to slightly shorten the required wire lengths compared to bare copper.

We will now turn this 20m dipole into a 102ft G5RV dipole, made of copper wire, 0.1-inch diameter, complete with a 20m $\lambda/2$ feed stub made of 450-ohm open wire feeder. We will then examine the radiation plots and feed-point impedances at the bottom of the feeder.

Fig 4.4 shows the Wires window, modified to create the model of the G5RV.

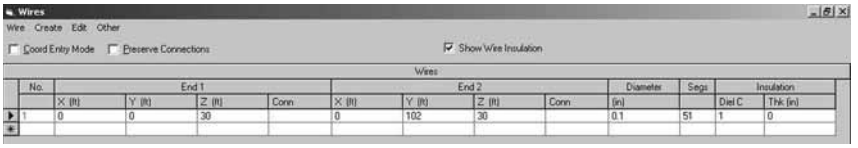

Fig 4.4: The Wires Window edited for the G5RV. If the Free Demo version is used, the number of segments should be 19.

We have changed the length to 102 ft, the number of segments to 51 (19 if using the free demo version)and the wire diameter to 0.1-inch. To change these parameters, simply highlight each value to be changed and type in the new value. Close the window when finished. The new values are automatically saved.

Note we have used an odd number of segments. This is necessary with NEC2, to allow connection of feed points, or in this case transmission lines to the exact centre of the dipole.

We will now add a 20m $\lambda/2$ 450-ohm ladder-line in the Transmission Lines Window. We need a half wave long line at 14.175MHz made from 450-ohm ladder line with a Velocity Factor of 0.9. This is 32.23 feet long. Real life transmission line losses are 0.082dB per 100ft at 10MHz.

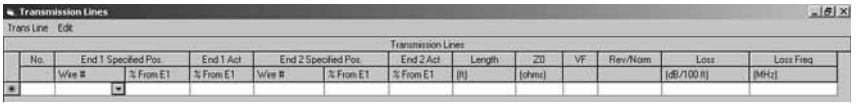

Fig 4.5: Blank Transmission Line window.

Click on the **Trans Lines** button in the Control Centre window to open the blank Transmission Line window – **Fig 4.5**.

We now enter the required transmission line parameters. End 1 of the line is in the centre of Wire 1 at 50%. End 2 would normally be another wire defined in the Wires window. However, to make life simpler, we use an EZNEC specific shortcut and use a 'Virtual wire' (v1), which does not require a physical location to be defined.

Fig 4.6 shows the completed Transmission Line Window. Close the window.

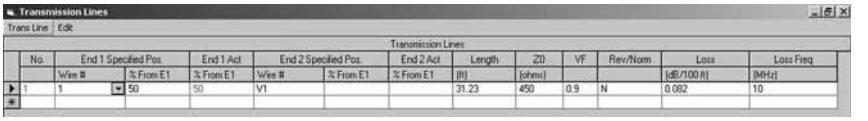

Fig 4.6: The Transmission Line Window completed for the G5RV.

Note that EZNEC V5.0 has a transmission line calculating engine built in, which will calculate the transmission losses for any given frequency and subtract these when calculating the antenna model gain. Impedance transformations down the line are also calculated.

We now move the source from the centre of the wire to the virtual wire at the far end of the transmission line.

Open the **Sources** window. Change the Specified pos from wire 1 to v1 – **Fig 4.7**.

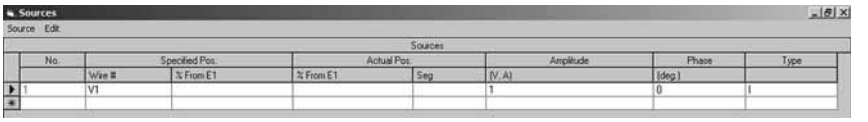

Fig 4.7: The Sources Window completed for the G5RV.

Note we are using a 'Current' type source with amplitude of 1-amp. Click on OK to close the window.

We will now change a number of other model parameters to better reflect real life conditions.

Click on the **Ground Type** button. In Real Ground Types, click on the High Accuracy button to change from MININEC ground - **Fig 4.8**. The High Accuracy gound model is more accurate than the MININEC ground. Click OK to close the window.

The High Accuracy gound model takes into account losses through the ground close to the antenna, the MIMINEC ground does not. The downside is wires can not be directly connected to ground with the High Accuracy ground making accurate modelling of vertical antennas more difficult.

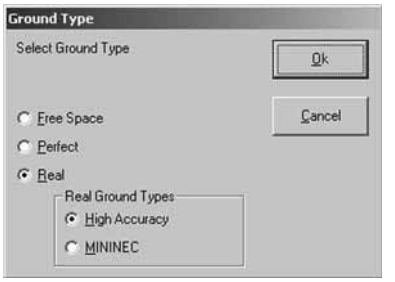

Fig 4.8: The Ground Type Window for the G5RV.

Click on the **Ground Descrip** (Ground Description) button.

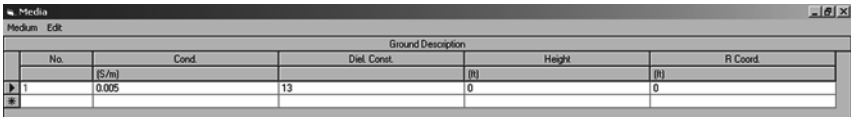

Fig 4.9: The Ground Description Window for the G5RV.

The **Cond** (Ground Conductivity) and **Diel Constant** in the sample model are set to those for very good ground. To make the model more realistic we will change these parameters to those for average ground.

Change the **Cond** (Conductivity) value from 0.0303 to 0.005 Siemens per metre. Change the **Diel Const** (Dielectric Constant) value from 20 to 13 – **Fig 4.9**. Close the window.

This changes these values from those for good ground to average ground. These may not be those pertaining at any particular amateur QTH, the only certain way being to measure them. However, an idea of the likely ground conductivity anywhere in the UK can be obtained from **Fig 4.10.** Also, http://andycowley.net/ant/vmox/gm.html has a useful colour map, showing ground conductivity in the UK and a list of permittivity values for various soil types.

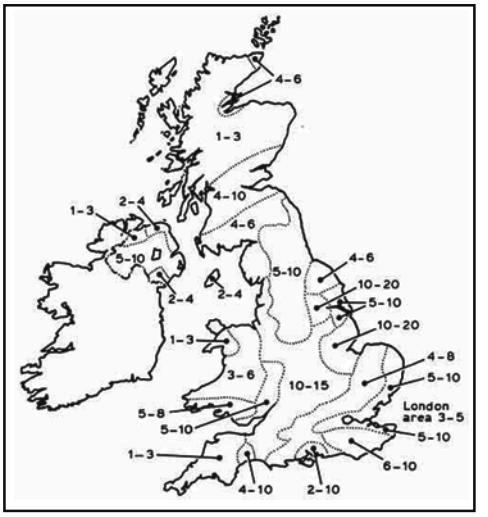

Fig 4.10: Ground Conductivity Map for the UK. The values are in milli-Siemens per metre.

Commonly accepted values for conductivity and permittivity values for good, average and poor ground are shown in **Table 4.2**.

Ground Type	Conductivity (Siemens/m)	Conductivity (Milli-Siemens/m)	Permittivity
Poor	0.001	1	5
Average	0.005	5	13
Good	0.0303	30.3	20

Table 4.2: Typical ground parameters.

Click on the Wire Loss button, click on the copper button to change from zero loss – **Fig 4.11**. Click OK to close the window. We are using copper wire and the resistive loss needs to be taken into account in calculating the actual antenna gain. Click OK to close the window.

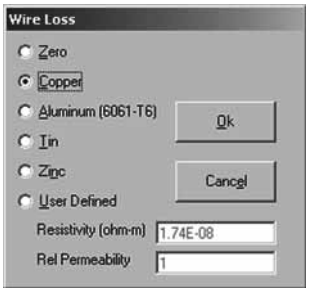

Fig 4.11: Wire Loss window for the G5RV.

Click on the Plot Type button and change from elevation to 3D by clicking on the 3D button – **Fig 4.12** (overleaf). Click OK to close the window.

7.
Software Defined Radio

by Mike Richards, G4WNC

Microprocessors have had a massive impact on just about all modern electronics and radio is no exception. Whilst this began with the digital control of local oscillators and noise reduction units, the technology has now spread into the main core of the transceiver and is progressively creeping ever closer to the antenna. In this chapter we will take a look at SDR to see how it works and how we can make best use of it.

What is SDR?

SDR is very broad term that describes any system where software performs one or more of the core functions of a receiver. The first examples to make a widespread impact on amateur radio were the excellent SoftRock receivers and transceivers that were designed and produced by Tony Parks, KB9YIG. These comprised a relatively simple hardware direct conversion receiver/exciter with an analogue IQ output (more on this later). The IQ signals were fed to a standard PC soundcard where tuning and demodulation took place. Dealing with filtering, demodulation and final tuning in software brought about great flexibility, as the performance and features could be completely transformed with new software. It is this ability to introduce changes and upgrades with ease that makes SDR technology so attractive. In addition to the great flexibility offered by SDR, the use of Digital Signal Processing (DSP) techniques facilitates the inclusion of filters and advanced demodulation systems that would be virtually impossible with conventional hardware circuitry.

There are three key technologies that are at the heart of SDR and

we ought to take a look at these before moving-on to real-world systems. They are:

Analogue to Digital Conversion (ADC),
In-phase/Quadrature Data (IQ data), and
Fast Fourier Transforms (FFT).

Analogue to Digital Conversion

As you are no doubt aware, all computer systems talk in numbers, so before a microprocessor can do anything useful in radio, the analogue signals need to be digitised. The method of achieving this has remained stable for some time and involves measuring and storing the instantaneous level of the analogue signal at a very high speed. You can visualise this as taking regular measurements with a digital voltmeter (see **Fig 7.1**). So how frequently do you think we need to take measurements (samples) in order to create a realistic digital equivalent of the original signal? The fundamental work to determine this was first published in 1928 as the Nyquist-Shannon sampling theorem. The theorem states that you must sample at a rate that is at least twice as fast as the highest frequency component you want to digitise. Therefore, to digitise an audio signal that contains frequencies from 20Hz to 20kHz you need to take measurements at 40,000 times per second, i.e. twice 20kHz.

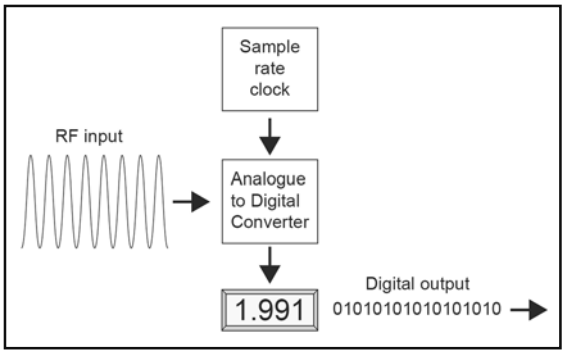

Fig 7,1: Illustration of the analogue to digital conversion process.

The other vital point to consider is the accuracy or resolution of each sample. You can think of this in terms of how many digits you have available on your digital voltmeter. If you were to use an 8-bit measurement you would have just 256 possible values for each measurement, which may be a bit too coarse for many purposes. As a result, it is common practice to use 16-bits or greater for radio applications – a 16-bit sample contains 65,536 possibilities for each measurement. When the ADC process is complete, the result is a data stream of 16-bit numbers that are generated at the sampling frequency. Returning to our simple audio signal, that would produce a data stream running at 16 (bits) x 40,000 (sample rate) which is 640,000 bits per second. As you can see, one of the problems with the ADC process is the relatively high-speed data streams that have to be processed.

So where in the receiver chain do you put the ADC? In many systems the ADC in the computer's soundcard is employed and whilst this works very well, there is so much more you can do if you can move the digitisation closer to the antenna. At the time of writing, there have been a number of systems introduced that digitise the

entire spectrum from VLF through to 30MHz or more! Most of these employ some digitally switched band-pass filtering close to the antenna, followed by some modest amplification, before sending the entire spectrum to the ADC.

From my earlier explanation you will know that the sample rate would need to be twice the highest frequency, i.e. at least 60MHz, with 66.666MHz being a popular choice. When combined with 16-bit samples, that gives a data rate from the ADC of 16 (bits) x 66,666,000 (sample rate) = 1.067Gb/s. Now that's a lot of data to process and is way too fast for most home computers to handle. To tame this data rate a process known as decimation is employed. This is the digital equivalent of a mixer and local oscillator in a conventional receiver. The decimator reduces the bit rate and extracts a segment of the sampled 30MHz spectrum that can then be passed to the PC for processing (see **Fig 7.2**). In most practical designs the size of the extracted segment can be controlled by the software running on the PC and ranges from a few tens of kHz to 10MHz or more. Rather than passing a single data stream to the PC for processing, two data streams are presented, one that's in-phase and the other that's delayed by 90 degrees and is known as quadrature. These two streams are the In-phase (I) and Quadrature (Q) data mentioned earlier. The hardware used for this decimation or down conversion has to be extremely fast and many of the current designs use a Field Programmable Gate Array (FPGA). This is a large array of high-speed gates and logic devices on a single chip that can be programmed and interconnected by software to create highly customised functions.

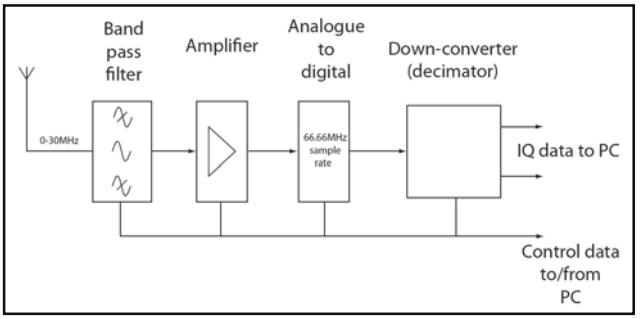

Fig 7.2: Block diagram of a digital HF receiver.

IQ Data

So, if sampling a signal at the Nyquist sample rate produces a good representation of the signal, why do we need to bother with In-phase (I) and Quadrature (Q) signals? Put very crudely, you can imagine IQ-data as stereo for digital signals. In the same way as stereo audio allows your ear/brain combination to determine the location of instruments and provides a sense of space and increased detail, so IQ data facilitates the extraction of more information from our digitised radio signal. Let's just look at IQ data in a little more detail to see how that works.

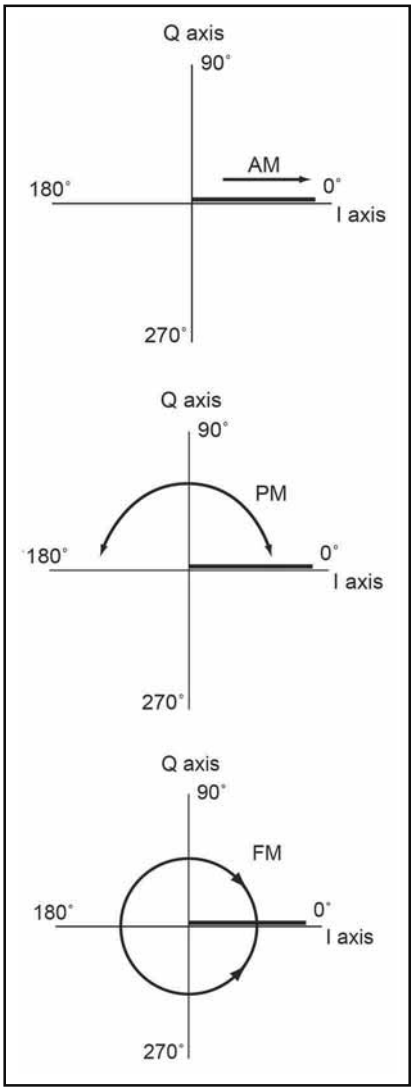

Fig 7.3:
(a) IQ data for an amplitude modulated signal,
(b) IQ data for a phase modulated signal,
(c) IQ data for a frequency modulated signal.

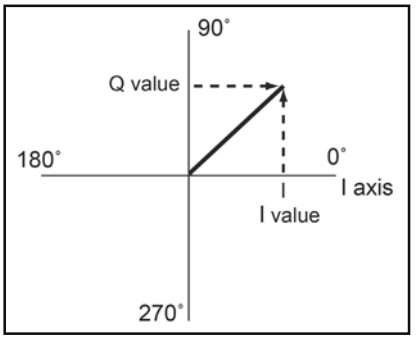

Fig 7.4: Using IQ values to control vector position.

I've shown an IQ polar diagram in **Fig 7.3a-c**. Here the I axis represents a change in amplitude of the carrier wave, whilst the Q axis shows any change in the carrier's phase. If we start with amplitude modulation, the only part of the vector that changes is the magnitude along the I axis and this changes in response to the modulating signal. Let's now move on to look at what happens with simple phase modulation, where a 180 degree shift is used to convey a digital signal. In this case the vector amplitude remains constant but the vector will flip between 0 and 180 degrees in response to the modulating signal. Frequency modulation can also be demonstrated and the vector will remain the same length but will rotate clockwise to show an LF shift and anti-clockwise for an HF shift.

If we take control of the I and Q values we can adjust the values to generate amplitude, frequency or phase modulation with comparative ease (see **Fig 7.4**). In addition to these relatively simple modulation systems, by manipulating the IQ data we can generate a range of complex modulation systems that employ both amplitude and phase modulation that would be very difficult to implement with traditional hardware. Using IQ data the generation of simple or complex modulation schemes can be completed entirely in software, which brings tremendous flexibility. The resultant baseband IQ signal can then be applied to an IQ up-converter to produce the final operating frequency. The up-converter is a relatively simple device that mixes the IQ data with a pair of local

oscillator carriers at 0 and -90 degrees and then combines the result, as shown in **Fig 7.5**. This makes for a very elegant modulation system that can handle a huge range of operating modes at any carrier frequency.

When it comes to reception with SDR systems, IQ data is the vital ingredient, as monitoring the change in IQ values over time will reveal the modulating message, regardless of whether it's a form of AM, FM, PM or a combination of all three, thus providing a simple multi-mode demodulator.

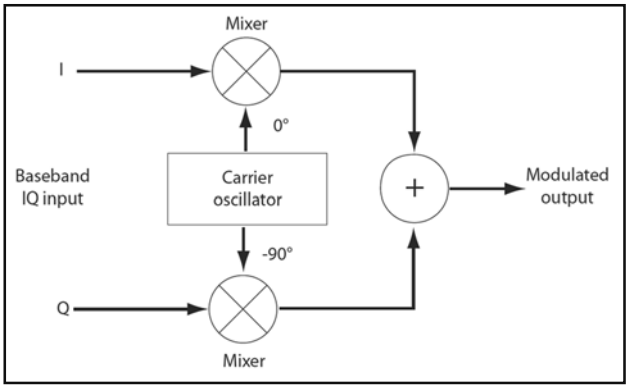

Fig 7.5: Transmit up-converter for baseband IQ data.

Fast Fourier Transforms

Another important process in the development of SDR and digital signal processing in general, has been the use of Fast Fourier Transforms (FFT) to analyse sampled data streams from the ADC. One of the great operational benefits of practical SDRs is the presentation of band segments in spectrum analyser style. This enables the operator to quickly locate active stations within a band and to use the click of the mouse to tune to that station. Converting the sampled data stream into a spectrum analyser format is handled by the FFT, so let's take a quick, non-mathematical look at how this works.

The FFT is an ingenious mathematical algorithm that can examine a stream of sampled data from an ADC and separate the stream into its component frequencies. It does this by dividing a section of the sampled data into a number of bins (containers) each of which contains a narrow band of frequencies – see **Fig 7.6**. It then measures the energy in each bin and it is this data that's used to feed the spectrum display or drive other facilities. The size of each bin is dependent on the sample rate of the digital signal and the number of bins used by the FFT. When using FFT to analyse an audio signal we might typically have a sample rate of, say, 44kHz – so that it can accurately capture frequencies as

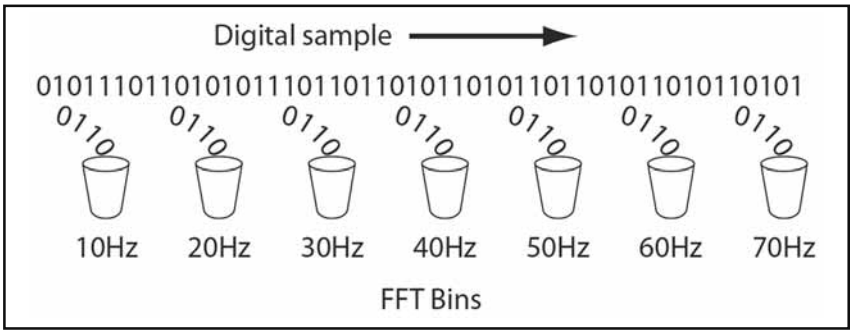

Fig 7.6: Illustration of using FFTs to convert data samples to FFT frequency bins.

9. Internet Remote Operation

by Wojtek (Berni) Bernasinski, G0IDA, SP5GU

The technological advances in computing and microcontrollers that have taken place in the last few years have resulted in it being easier than it has ever been to set up and operate a remote station. This practice would appeal mainly to those who have limited real estate at home to put up a good antenna, or indeed for those who live in a flat where a garden is not available.

For people without the ability to erect antennas, a remote station can literally transform their enjoyment of amateur radio. It can be installed far away in the countryside, or at a friend's property where ample antenna space is available.

Basic Overview

This chapter is based around experiences with the RRC-1258 MkII from Microbit. It comes in two parts; one for the remote end and one for the home end. The remote unit controls Voice Over IP and serial data to the rig and is connected to a router via a Local Area Network hub, which in turn is connected to the Internet and a Web Switch for switching relays (which will switch the power supply on and off as well as the remote controller and the rig). The home end comprises a PC with Windows XP running Ham Radio Deluxe, a Virtual Comm Port Emulator (for the Microham unit) and the RRC-1258 MkII controller, which is connected directly to the router at home via an Ethernet cable. Headphones, iambic key, foot switch and microphone all connect to the control box.

When complete, the Internet will carry signals between home and the remote station, and here lies the essence of using a remote station which could be located anywhere in the world.

Advantages

A clear advantage of operating remotely is that the station can be connected to a large antenna, perhaps a full sized dipole on 160m, a beverage or a tower with a beam on it. Other advantages may include a reduced level of noise, a problem which most of us who live in a town or city would undoubtedly suffer from.

For those who like to experiment, gain knowledge, enjoy 'fiddling and tweaking' and having the satisfaction of a remote station working reasonably well (and I'll get back to why it doesn't work well all the time) then it is all worth the time and expense.

Disadvantages

Cost

For a start the Remote Control units I will be describing are not cheap and are priced at around £400. The Web Switch, described later, costs around £160. Assuming you already have a transceiver, a power supply, antenna system and a free place with free Internet and mains, then your basic cost at the time of writing has already mounted up to £560, the price of a small but adequate HF rig!

Latency

Delays in the Internet may cause problems and there will certainly be a delay between talking at the home end and what is actually transmitted. I measured mine using 'traceroute' command and found 15 hops (routers) between my home and the remote site and a latency of 60-90ms. This changes a bit every time I connect to the remote station, as different routes are used through different routers on the Internet.

Legality

A remote station needs to conform to current radio licensing regulations, the most important one being for the radio to go back to receive should the transmitter get stuck. Some modern radios include in their menu a setup for timeout. I have set mine to five minutes, in the hope that none of my transmissions will be any longer than that.

Our licence states that our messages 'must be adequately secure' from one end to the other, but do not mention what is meant by the word 'secure'. I am happy with the idea that my signal going into the Internet and out into my radio is secure, as 'Joe Public' is unable to intecept the data easily. There isn't much we can do about data between the time that it leaves our router and arrives at the remote end anyway.

Security

Security of equipment can be of concern if it is hosted on a remote site, for example in a shed in the middle of a field. These are just the basic

Fig 9.1: Front view of the RRC-1258 MkII.

disadvantages but they could be enough to put you off such a project.

I have been very lucky at the remote end, as I have free Internet access, free mains and good security for the equipment, hence this project has been worthwhile.

Hardware Requirements

The project revolves around the remote control units RRC-1258 MKII, as seen in **Fig 9.1**, produced by Microbit of Sweden.

On the left hand side you can see the controller which connects to the radio at the remote end, while the controller on the right connects to the PC and router at the home end. They look almost identical, except for a CW speed control on the right hand unit.

The small USB format sockets on the far left are used to program the initial IP address of the units via a utility program which can be downloaded from the SM20 website www.remoterig.com Alternatively, a serial connection can be used on Com 1 for the same purpose. The next socket, AUX/MIC, is where the microphone and PTT are connected via a network type plug. Here I just made patch leads to plug into my head set and PTT footswitch. My speaker plugs directly onto the socket marked 'SP'. At the remote end, microphone and PTT leads are connected from this socket to the transceiver. A lead (grey) is connected from the speaker socket of the transceiver to the back of the controller, into the microphone input socket, as seen in **Fig 9.2**, as well as a serial cable which controls the rig, the 12V DC cable and the network cable.

Fig 9.2: RRC-1258 MKII back view.

The lowest band-

width which the audio channel uses is approximately 85kbps using mode '0' and the codec G711 at a sampling rate of 8KHz. Other rates are available and used to suit your needs.

The control channel uses far less bandwidth and is not specified at the time of writing.

Setting up the Controllers

Setting up the controllers is relatively straightforward. Detailed information can be found on the manufacturers web site.

First of all there are five jumper leads inside to configure in each controller. Four positions will depend on the make of your radio and the fifth is to provide power to your microphone. After this you need to set up the IP addresses for each unit. I used 192.168.2.200 and 201 for each controller.

The controllers each have a built-in web server, so having assigned them their own IP addresses and using port 80, log into them and configure a few basic options like the FQDN (Fully Qualified Domain Name) and the audio quality. There are more advanced options, but they do not need to be set to get you going. Configuring the controller at the remote end to have port 8091 for web operation is ideal as the Web Switch will be configured for port 8090. As you will see later, it too has a web server running for remote set up. The Web Servers in both units have username and password control, so the units are relatively safe from an outside attack.

Having set up the basics (by referring to the web site), the router at the remote end needs to have its ports opened to enable the two units to communicate and for you to be able to administer the controller and the Web Switch.

The ports which need to be opened and the protocol used are:

8090 - Web administration of the Web Switch (TCP)
8091 - Web administration of the remote controller (TCP)
5060 - SIP in/out port (UDP)
11000 - Audio in/out port (UDP)
12000 - Command in/out port (UDP)

There is no need to open any ports on your router at home, as all traffic is initiated by the controller in the shack, which then opens outbound ports automatically in your router; it's just inbound ports which need to be opened at the remote end. You can now type an FQDN like http://remote.dyndyns-net.org:8091 in your web browser and be presented with the controller's administration page.

Web Switch

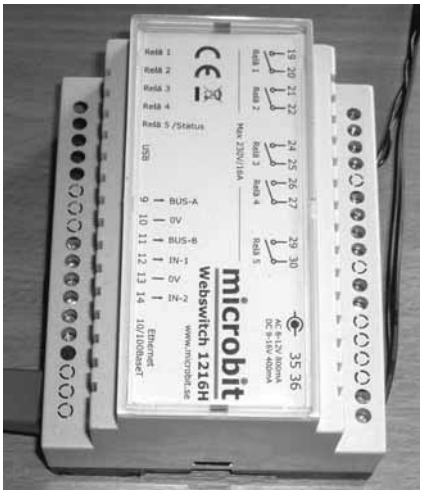

Fig 9.3: The Microbit 1216H Web Switch.

The clever little Web Switch 1216H from the same manufacturer can be seen in **Fig 9.3**. I say 'clever', as it can not only switch five relays at 240VAC 5A, it can also read temperature (rig or ambient) on two of its inputs if Dallas DS18B20 sensors are attached.

The Web Switch is an item of equipment that needs to be powered on all the time, as it provides the FQDN to the Internet where you can log in and switch relays on and off or read the current temperature.

On the left side of Fig 9.3 is the network cable which is connected to the hub, and then to the router. On the bottom right side is a 9V DC supply and the twisted wire is connected to Relay 2, the cable which when shorted switches the K3 on. Relay 1 is used to switch the mains side of the PSU, but is not shown in the photo. In **Fig 9.4** you can see that the PSU and the radio are on.

Setting up the Web Switch

The Web Switch is set up in the same way as the controllers, via the mini USB port at the side of the unit. All that is required is an IP address to get you started. I used 192.168.2.202.

Once connected, using your web browser on port 80 through your Local Area Network, you then need to change the port number to 8090 for remote administration. At the same time a username and password should be added. The menus are found on the left hand side.

The home end needs to know the IP address of the remote end, so a FQDN needs to be inserted into one of the setup fields. You can make one up from the web site www.dyndns.com It will look something like 'remote.dyndyns-net.org' where, when you type this into your browser, this address will find the remote station on the Internet.

The Web Switch will then send every 30 minutes (this is configurable) to the web site www.dyndns.com a request, and the web site will then do a reverse lookup on the remote end IP address and resolve the FQDN 'remote.dyndyns-net.org' to numbers

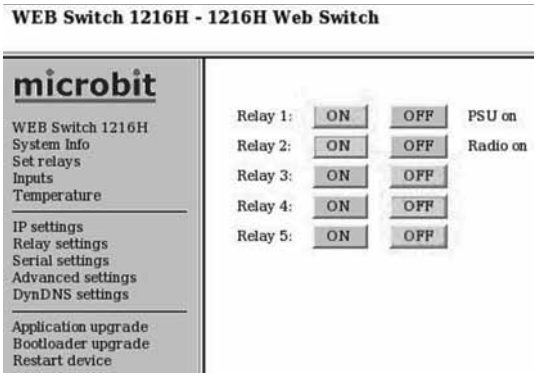

Fig 9.4: Web Switch relay status page.

10.
D-Star

by Dave Thomas, 2W0RUH

Contrary to widespread belief, Digital Smart Technologies for Amateur Radio (D-Star) is a communications standard, not a brand name. Moreover, it is not limited to one manufacturer. It was released in 2001, having been developed and funded by the Japanese Ministry of Post and Telecommunications, who were tasked to investigate digital technologies for amateur radio. The committee included representatives of the Japanese amateur radio manufacturers, including Icom, plus other observers.

The standard is published by JARL, but it is an open system, which means that any equipment complying with the standard can use it. It is a two part communications system, the first being formed by radio-to-radio transmissions - either direct or via a standalone repeater - and the second deemed to be the 'spine' (or backbone) of the system is by integrating with the Internet via 'gateways' to the wider amateur community. The D-Star standard also controls the way in which the signal is relayed, by converting voice to and from digital format. This digital exchange takes place by the use of the AMBE (Advanced Multiple Band Encoding) codec ('codec' being short for **co**ding/**dec**oding).

The early days

Initially you were only able to operate on D-Star (simplex range excepted) by purchasing Icom factory-built repeaters, but since then many new ways of entering and using the system have surfaced.

To encourage people to adopt D-Star, Icom offered discounted

deals on the purchase of a repeater if a number of radios were bought, which allowed clubs and individuals to enter into negotiation with those in a locality to take up the offer. Today, however, there are many more ways of joining the network. These will be discussed further on in the chapter, but now include the DV Node Adaptor, DV Dongle and DV Access Point to name but a few.

D-Star modes

D-Star carries digitized voice and digital data, but it does the job in two different ways, there being a combined voice and data mode (DV) and a high speed data only stream (DD).

Although data and voice are carried at different rates and are managed in different ways, they are transmitted as packets. The D-Star protocol is therefore similar in some ways to the Packet Radio (AX.25) protocol that allows the exchange of data between Terminal Node Controllers (TNCs) or the Ethernet protocol used by home and business computers.

The AMBE codec was mentioned earlier. It can digitise voice at several different rates. D-Star uses 2.4kbps (bits per second). In addition, AMBE adds information to the voice data that allows the codec at the receiving end to correct errors in the transmitted stream. The result of the overhead is that the digitized voice stream carries data at a rate of 3.6kbps.

Simultaneously to the digitised voice, DV Mode (low speed data mode) can also carry 8-bit digital data at 1200bps. This data is un-modified when transmitted, so it is up to the operator's software to manage the flow of data whilst it is being exchanged.

When operating in DD Mode (high speed data mode), the voice signal is unused and all packets are dedicated to the use of digital data. Transmitted data is sent as raw data at a rate of 128kbps. Like DV mode, this is transmitted with no modification, the flow control being undertaken by the software package chosen by the user. In DD mode the net data rate is comparable to or better than a high-speed dial-up Internet connection.

How D-Star works

Being a packet based protocol, D-Star data is processed and packaged using the required data and additional information. Packets are sent in their entirety and are processed as a group by the receiving station.

An important difference between AX.25 Packet Radio and D-Star is that AX.25 requires an acknowledgement and the TNC at the

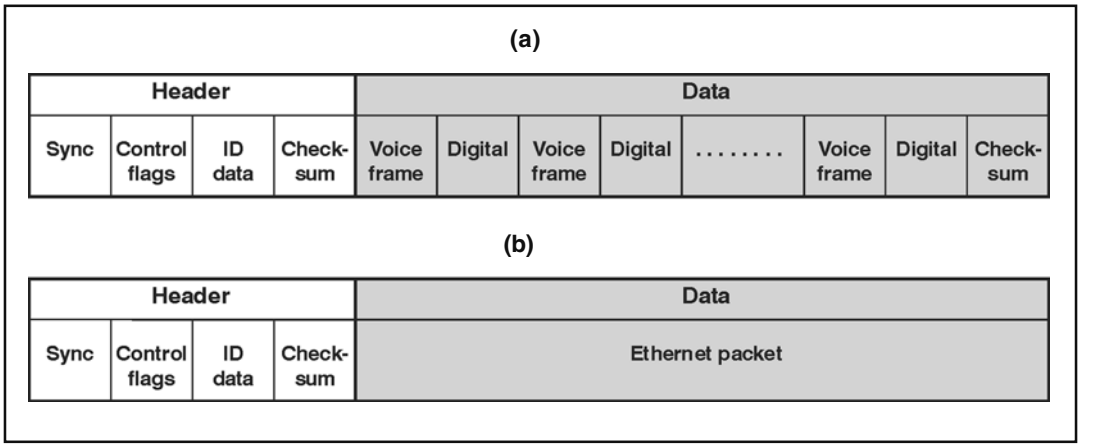

Fig 10.1: (a) Structure of DV mode, and (b) structure of DD mode.

receive end can request retransmission if a packet is received with errors, whereas D-Star is a one-way protocol, so no response is required from the receiver to acknowledge that a packet has been received. The reason D-Star does not require acknowledgements is because, as previously stated, it has error detection and correction built into the datastream.

The structure of the DV and DD modes are illustrated in **Fig 10.1**. Each consists of a header and data segment.

D-Star utilises a common method of using one protocol to send data formatted according to another protocol. In the DV packet, voice data is contained in short segments (frames) which are formatted according to the AMBE protocol. In the DD packet, the data is formatted using the Ethernet protocol. This process of putting data from one protocol 'inside' another is called encapsulation.

The illustration of the packet structure is broken down in **Fig 10.2** as follows:

Sync frame
Bit Sync is a standard pattern for GMSK 1010 modulation used by D-Star.

Frame Sync is '111011001010000' - a unique bit pattern in D-Star packets.

Control flags
Control flags are used to direct the processing of the packet.

Flag 1 Indicates whether the data is control data or user data, whether communication is simplex, repeater, set priority, etc.

Flag 2 Reserved for future use as identification data

Flag 3 Used to identify the version of D-Star protocol being used, so that as new functions are added the receiver can apply them

Sync		Control			Identification					Error
Bit sync	Frame sync	Flag 1	Flag 2	Flag 3	Received Repeater Callsign	Sent Repeater Callsign	Counterpart Callsign	Own Callsign 1	Own Callsign 2	P-FCS

Fig 10.2: D-Star packet structure.

Identification Data

Received Repeater Callsign	Callsign of the repeater that is to receive the packet
Send Repeater Callsign	Callsign of the repeater sending the packet
Counterpart Callsign	Callsign of station that is to receive the data
Own Callsign 1	Callsign of the station that created the data
Own Callsign 2	Callsign suffix information

P-FCS Checksum
A checksum is used to detect errors. The P-FCS checksum ID is computed from the flag and ID data.

How D-Star corrects errors in digital voices

D-Star uses two methods of combating transmission errors:

1. **Error Detection codes** are used to detect errors. These codes only tell the receiver that the data is damaged or corrupted, not how corruption came about. D-Star checksums follow the CRC-CCITT Standard.

2. **Error correcting codes** contain information about the data. Because the codes are sent with the data (to enable correction at the receive end), they are called Forward Error Correcting or FEC codes. FEC codes contain enough information for the receiver to repair most damage.

Both the DV and DD data packets in Fig 10.1 use the P-FCS checksum in the header, but the DD packet also contains the Ethernet data packet checksum at the very end.

With the DV packet data segment, each AMBE digitised voice frame contains its own FEC code to allow the receiver to repair errors. DV Digital data frames are not protected, relying on the applications to detect and correct errors.

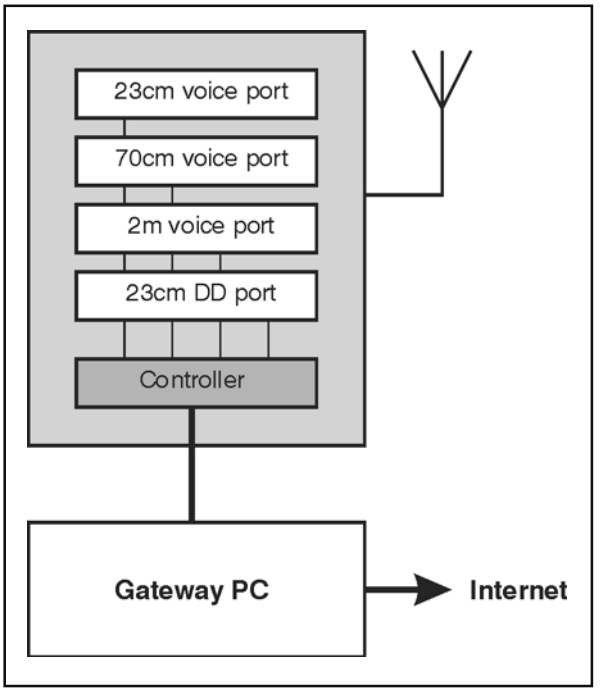

Fig 10.3: Icom D-Star Repeater.

D-Star system layout

We will begin with the D-Star repeater. Unlike an analogue repeater which operates on a particular band with an individual callsign such as GB3QQ, a D-Star repeater can be built according to the keeper's requirements and be active on several bands with the same callsign. GB7CD will be used as an example. At present it only has the 70cm (B port), but **Fig 10.3** illustrates a full repeater stack setup.

As you can see from the illustration, a fully loaded D-Star repeater can be constructed with four ports. It can also be built with any combination of the four ports. Funding, the frequencies available to the builder and any licencing restrictions in his or her particular area will determine how complete a D-Star repeater is.

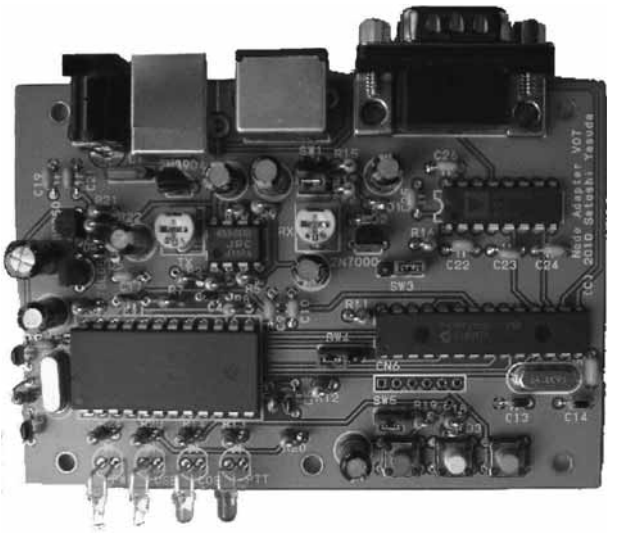

Fig 10.4: Satoshi node adaptor board.

Assuming a complete setup, a user would be free to access the system on any of the bands available within the repeater. Across the world it has been decided that wherever possible the 'A' port will carry 23cm (1.2GHz) voice traffic, the 'B' port 70cm (430-440MHz)

Index