21 Computers in the Shack

Andy Talbot, G4JNT

Computers have been part of a well-equipped radio shack for some years, mostly for data communications and logging/contesting. More recently, cheap online connections have brought Internet resources into many amateur stations.

This chapter aims to show that there are many other uses for the shack computer, making it an essential tool for the constructor.

INSIDE A COMPUTER

Before discussing what a computer can do for you as a radio amateur, it is useful to take a brief look at how it works.

For the purposes of this book, it will be assumed that the computer is a 'PC' operating under Windows. Other computers such as the Apple Mac and older 'hobby' machines (eg Sinclair Spectrum, BBC 'B') could be pressed into service for some applications with the appropriate software. Other operating systems such as DOS or Unix can also be used.

The essential components of a computer (shown in **Fig 21.1**) are a processor to do the work, memory to store information, inputs and outputs for communication between the computer and the operator (eg keyboard and screen), software to give it instructions and a power supply.

Central Processing Unit

The CPU is the engine of a computer and is measured by the number of cycles of work it can carry out in one second. Thus a '800MHz' computer carries out 800 million instructions per second. Modern CPU chips get very hot and have heat sinks or even their own cooling fans.

Memory

Just like the human brain, a computer cannot work without being able to store information. There are several types of memory, divided into their function, the storage medium and the amount of time that information can be stored.

Bootstrap

This is a tiny, permanent, memory on a chip. It is known as Read Only Memory (ROM) as it cannot be over-written with new data. Its function is to give the CPU the very basic information it needs to start up and function as a computer.

RAM

In contrast to ROM, Random Access Memory (RAM) is designed to be continuously re-used. It is the temporary storage used to hold all of the data required during processing. RAM is located on chips so it can be 'written to' and 'read' very rapidly. It is commonly described in 'Megs', though this is Megabytes, not Megahertz.

Hard disk

Most of a computer's storage is done on a magnetic disk. Although reading and writing is nowhere near as fast as RAM, it has the advantage that it keeps its information indefinitely, even when the computer is not powered up.

Removable memory

This refers to the disks that can be taken out of a computer for future reading by the same, or another, computer at a later date. Older machines have 3.5in so called 'diskettes' which can store about 1 megabyte of data, whereas all modern computers use CDs capable of storing up to 800MB or DVDs which can store several gigabytes - that's thousands of megabytes. Additionally it is now possible to plug in an external memory device (flash stick) capable of storing 1 gigabyte or more in a small space and with rapid read and write. External hard disks can be added, and these can store hundreds of gigabytes.

Input / Output

Abbreviated to I/O, these devices are what is needed for human beings to interact with the computer. They include the keyboard, screen, mouse and sound card.

Operating System

Usually stored on the hard disk, this is the permanently installed software, that makes the CPU into a usable computer. The most commonly used operating system is Microsoft's Windows, although there are amateur radio programs that run under DOS or Unix. The operating system defines how the various parts of the computer work together and how it connects to real people.

Power Supply Unit

Like any piece of electronic equipment, the computer has a PSU, to run from the mains. Additionally lap-top computers have hefty batteries capable of running the unit for an hour or two.

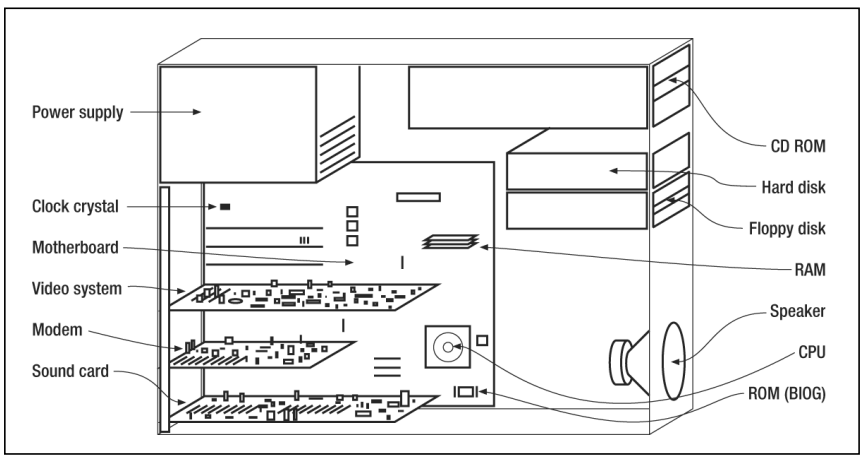


Fig 21.1: Hardware in a typical desk-top computer

SOFTWARE

Although the operating system is an essential piece of software, it cannot do anything other than make a computer. To perform any useful task, such as word processing or sending e-mail, additional software known as programs must be installed.

A new computer will usually come with some programs, usually an Internet browser and some office functions, but there is an almost infinite number of additional programs that can be added to perform specialist

functions. Although some programs, especially those for commercial applications such as producing this book, are very expensive, many are quite cheap or even free. Fortunately, many amateur radio programs are in the latter category.

Operating Aids

Computers are used to enhance the shacks of many keen DXers. Facilities available to the operator include:

- Logging
- Contest aids
- Rig control
- Maps
- Data communications
- DXCluster
- Propagation information
- Maps and locators

Most of these are outside the scope of this *Handbook*, but detailed descriptions of all of the above can be found in [1].

Drawing

If you are not good at drawing circuits, good quality illustrations, such as many of those in this book, may be drawn using 'computer aided design' (CAD) software. There are many generic drawing packages, from professional quality software such as CorelDraw to inexpensive or even free programs available for download on the Internet.

These save the work of producing neat straight lines, boxes and circles but components must be individually drawn. An alternative is to use a CAD program specially tailored for electronic circuit design.These have the facilities of a generic program, but also have a library of component symbols. Most of the simulator programs described below include schematic drawing facilities.

Basic drawing facilities quite suitable for producing block diagrams and simple circuit diagrams are also included within a number of office type programs such as wordprocessors and spreadsheets.

Circuit Drawing and PCB Layout Packages

A number of packages aimed at electronics design allow circuit diagrams to be drawn, with most components types stored in libraries. In many cases, schematic design is coupled with Printed Circuit Board layout, with the ability to autoroute a PCB directly from the original circuit diagram.

PCB layouts can be exported to files in standard formats that can be sent to PCB manufacturing companies or machinery. Other facilities like parts lists, design rule checking are often included in these software packages, as is the facility to export net lists to circuit simulators.

One package that is widely used by amateurs is EAGLE Light, by CadSoft Solutions [2]. This free program provides a limited subset of that provided by the full, professional, version. Restrictions include: The maximum board area is limited to 100 x 80 mm; Only two signal layers can be used (top and bottom); The schematic editor can only create one sheet; Use is limited to non-profit applications or evaluation purposes.

Circuit Simulators

These allow circuits to be drawn and then analysed, and most are based on the industry standard SPICE. Several are available to try out as free demo versions, usually with restrictions. A useful list can be found at [3].

Some simulators incorporate printed circuit board (PCB) design, whilst others can export data to a dedicated design program. The result can be printed and used in producing the PCB itself.

It is possible to simulate both analogue and digital circuits, or even a mixture of the two. Several of the projects in this book

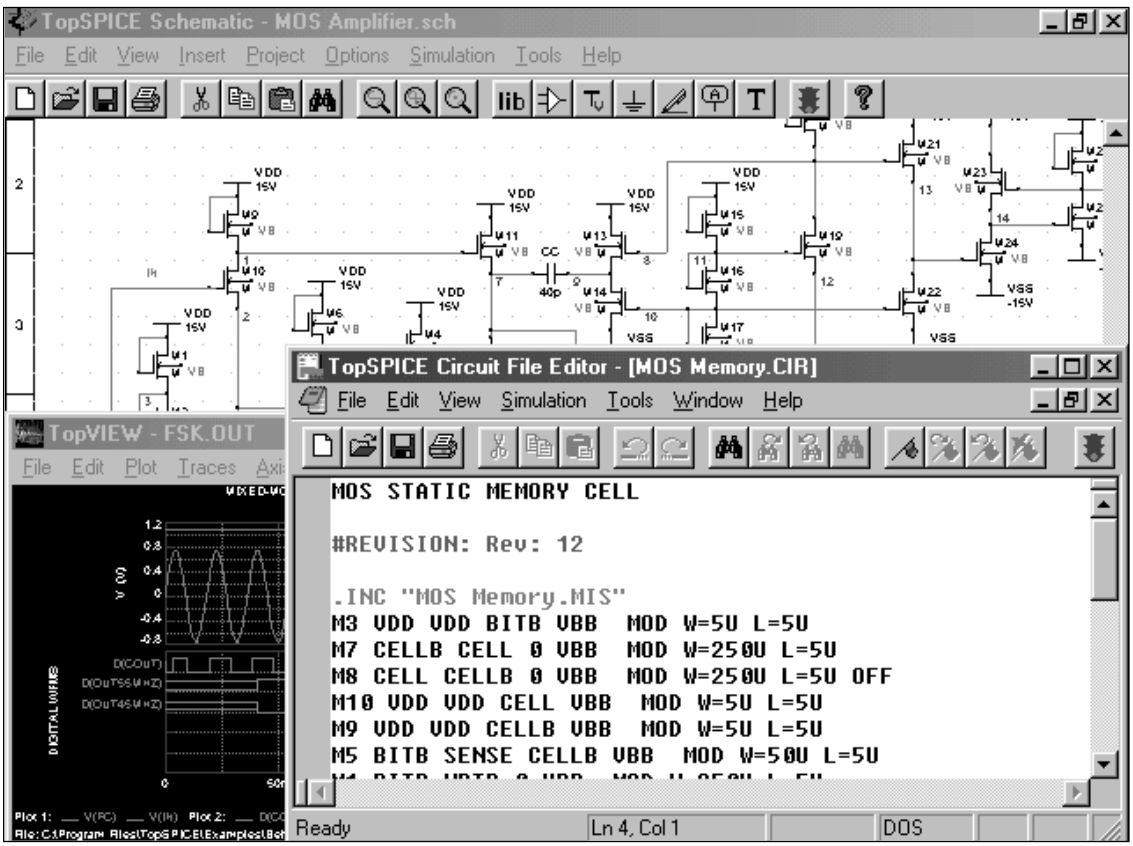


Fig 21.2: A typical circuit simulator / analysis screen. [Source http://penzar.com/topspice/topspice.htm]

have been initially designed by this type of program, most notably *PUFF*.

The user starts by drawing the circuit diagram, made from graphical elements provided with the software. The result can be analysed by the program's 'virtual' test equipment, to check how well (or whether) it works.

The information displayed can include amplitude vs frequency response, phase vs frequency, group delay vs frequency, gain or loss, and the impedance at input and output. A typical display is shown in **Fig 21.2**.

Changes can be made and the results tested, all before any real construction takes place. As with all software simulations, it is no substitute for knowledge and errors can be reduced by having a basic understanding of how real electronic circuits work.

Simulators can usually predict performance quite accurately (mostly for analogue circuits), although they do require substantial effort to learn to use effectively.

A circuit simulator won't do the designing for you, but it does help a lot in finding out what is wrong with a design, and predicting what will happen before you actually build the circuit. Note that the cheaper programs will not operate with time-varying inputs, eg intermodulation analysis.

Spreadsheets

Many aspects of circuit design can be calculated, so a spreadsheet program is useful for making the most complex and interactive calculations. Examples are the optimisation of inductors and filters, Yagi dimensions and linear amplifiers. See also the chapter on Software Defined Radio for a spreadsheet-based tutorial.

The 'number crunching' abilities of modern spreadsheet software can remove the need to write small custom programs for solving many design problems. In days gone by, special software written in a high level language was often needed to solve long equations or to optimise solutions. Now, spreadsheets such as Excel have optimisation and data analysis tools available for uses as well as all the basic mathematical functions. Spreadsheets are the ideal solution to quickly plotting and displaying results.

Word Processors

As well as their obvious use for producing written text and documents, modern word processing software permits seamless integration of graphics and pictures such as circuit diagrams exported from circuit design software and digital photographs, allowing a one-stop solution for producing complete, professional looking, documents.

In the past, constructors would use rub-down lettering or adhesive characters to produce professional-looking legends on a front-panel. Nowadays a word-processor can generate text in many fonts, and with a little practice it can also be used to produce arrows, meter scales, etc.

Other Software

A useful tool is AADE Filter Design which can be downloaded free from [4]. It will calculate gains, impedances, group delay, phase, return loss and more. Filter types handled include Butterworth, Chebyshev, elliptic (Cauer), Bessel, Legendre and linear phase low-pass, high-pass, band-pass, and band-reject filters, as well as coupled resonator band-pass and crystal Ladder band-pass filters using identical crystals. A tutorial is included for those wishing to know more about filters.

A large range of programs can also be downloaded from the website originally produced by Reg Edwards, G4FGQ [5]. These include calculators for the design of antennas, transmission lines, inductors, filters and amplifiers. Also included are propagation path-loss calculators.

Another useful source is Tonne Software [6] which has programs for filter design, meter scales, customised maps, antennas and matching.

Most are totally free, or available free in a cut-down version. The filter design program, *ELSIE* (**Fig 21.3**), was used in the PIC-A-STAR project which features on the CD accompanying this book.

Test and Measurement

A computer can be used to make all sorts of measurements with the appropriate software, and either a sound card or additional hardware (eg an analogue to digital converter).

Audio analysis software such as Spectrum Lab [7] uses the computer's sound card and can be used for such things as measuring distortion, intermodulation, noise, oscillator drift and modulation. It has a built-in tone generator.

Also using the sound card are programs that can measure and display audio signals (in fact up to several tens of kHz), including those at the output of a radio, and may also be used as a data logger for monitoring beacons. One of these is shown in **Fig 21.4**.

Antenna Simulation

It is no longer essential to dismantle your antenna in order to experiment with ideas for a new one. Thanks to software originally developed for the US military, and made available in a usable form by amateurs, it is possible to make a 'virtual' antenna on the computer. You can then measure its performance, alter it and measure it again until something practical is found -

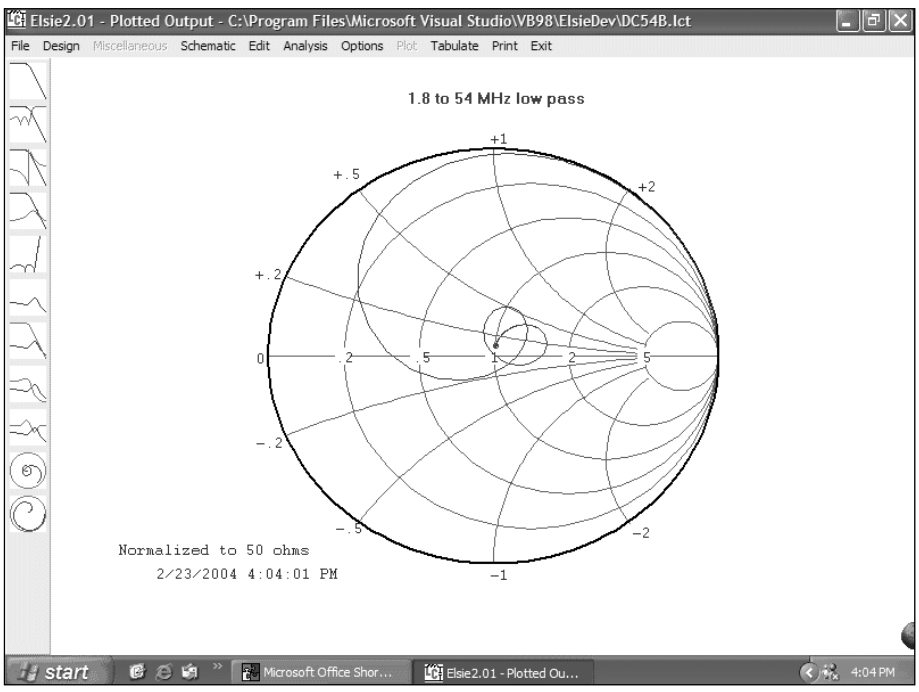


Fig 21.3: A Smith Chart produced by *ELSIE*, the filter design program from Tonne Software

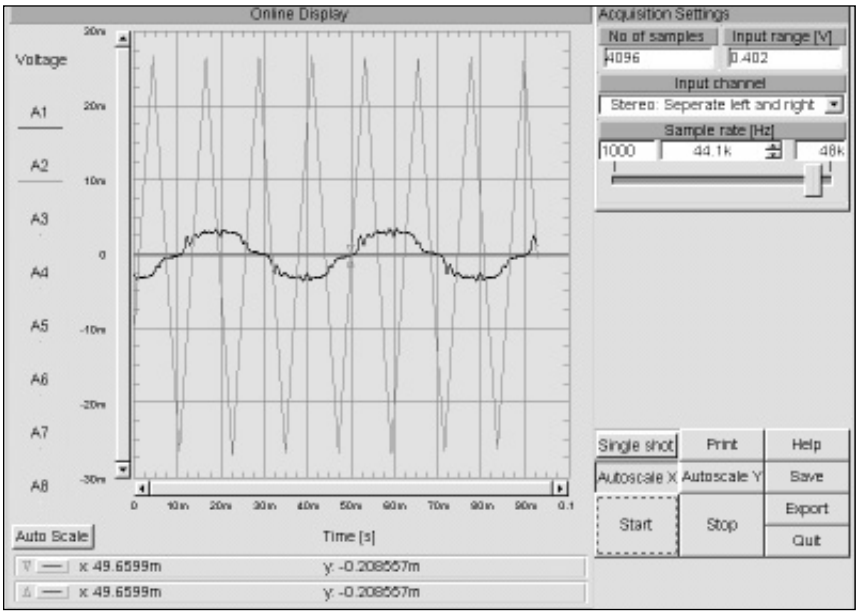


Fig 21.4: A sound-card-based signal analyser with data logging facilities [From http://www.hacker-technology.com]

or the project is abandoned as a failure. Work on a practical antenna need only take place when the 'virtual' one has been optimised.

As with all computer programs, it is possible to get the wrong answer, but this is usually a case of 'garbage in, garbage out'. A good knowledge of antenna theory and practice will help you avoid the major pitfalls. Information on how to obtain antenna simulation software can be found at [8].

Much more about the use of antenna simulation programs can be found in the chapter on Antenna Basics.

The Internet

For very little money, you can have facilities at your fingertips that only a few years ago would have been the envy of the best equipped library in the world, together with communications facilities that have almost totally eclipsed the postal service.

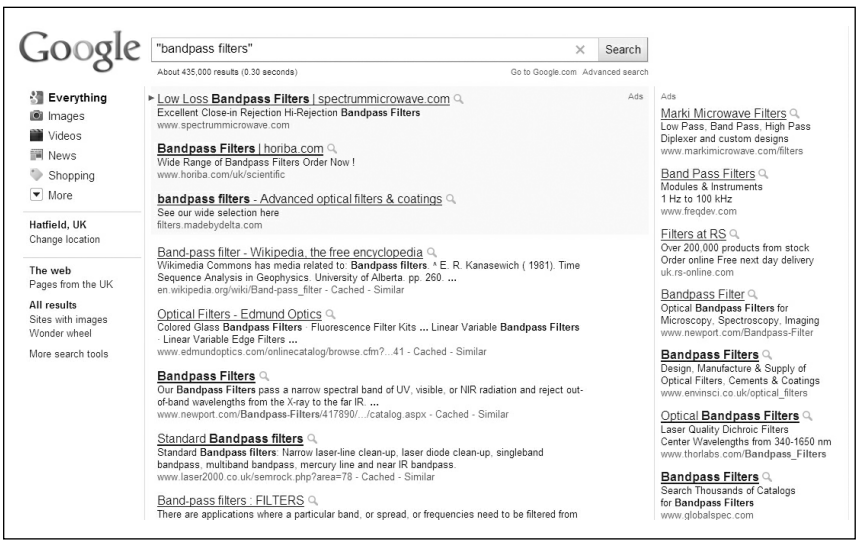


Fig 21.5: An Internet search for the phrase "bandpass filters" produces nearly half a million results, including advertisements for commercial filters, technical papers, descriptions and advice by radio amateurs, and lists of where to buy components

Although the Internet is often used to describe the vast array of web sites, it is actually the network on which sits the web, e-mail and many other facilities. The most common are:

World Wide Web

The 'web' is a repository for many millions of documents, including text, pictures, sound and video. Amateur radio is well represented with information on just about every aspect of the hobby.

There are two main ways to find the information you want. You could start with a large site that covers most aspects of amateur radio, such as those run by the Radio Society of Great Britain [9] or the American Radio Relay League [10], and follow the links to sites carrying additional information.

Alternatively, use a 'search engine' such as Google [11] or Yahoo [12], type in one or more keywords (eg circuit simulation) to get a large list of sites that may be useful (see Fig 21.5).

The sort of information that can be found on the web includes:

- News and events diaries
- Calculators for component values
- Equipment modifications
- Component catalogues
- Advice from experienced amateurs
- DXpedition information and logs
- Circuits and construction projects
- Propagation and solar data

E-commerce

Most shops have a presence on the world wide web, and some are available only via the 'web'. This includes major component suppliers as well as much smaller specialist outlets. All display their wares, and most encourage electronic sales by credit card. This has two advantages, you can browse without leaving your home and you can buy from overseas shops - note, though that VAT, import duty and other charges may be payable on entry to the UK.

The on-line catalogues of Farnell [13], Maplin Electronics [14}, and RS Components [15] and many others are not only places where components can be purchased, but also a really good source of information such as data sheets.

Auction sites such as eBay [16] are where second-hand (and new) bargains may be found. Simply enter a search term (eg "ATU") and choose from an array of items. As with all 'blind' purchases, such as a newspaper advertisement, the buyer should take precautions to prevent fraud. If possible visit the seller and check before you buy.

One of the first examples of e-commerce was book selling, and the Internet makes it possible to search for what you want from millions of books, including many specialist publications that you will never see in a high

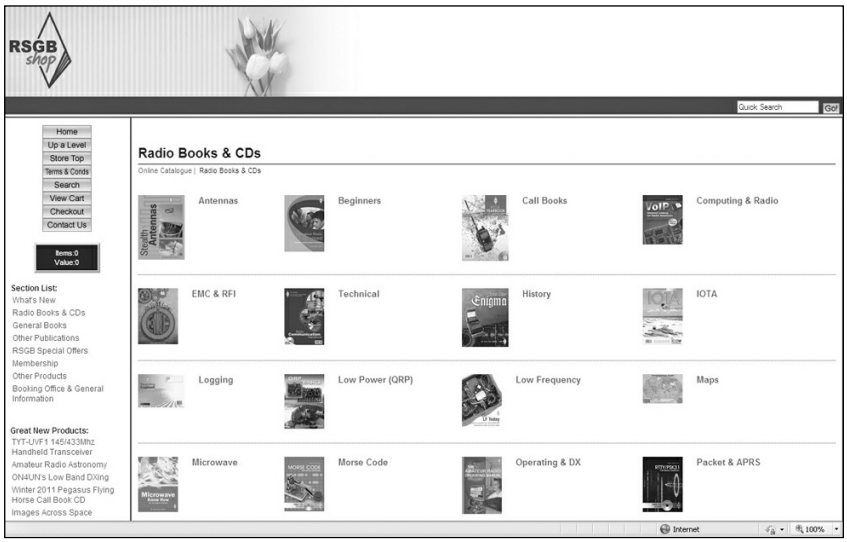


Fig 21.6: Part of the RSGB's e-commerce site where books, CD-ROMs, maps etc can be bought by credit/debit card. It includes a search facility and detailed descriptions for those who like to browse before they buy

street shop. Amateur radio books can be bought direct from the ARRL [10] and the RSGB [9] (**Fig 21.6**).

All sorts of radio and electronics books - even technical papers -can be searched for (by keyword, author or title) on big sites such as Amazon [17] which often has second-hand books listed alongside the new ones. Again, auction sites are a good source of second-hand and antique books.

E-mail

Although your main contact with other amateurs may well be on the air, or at the local radio club, e-mail can still be useful. It can be used to maintain a dialogue with like-minded amateurs all over the world, and allows the exchange not only of text but also pictures (including circuit diagrams), sound files and programs.

Groups

Similar to e-mail are newsgroups and reflectors. These allow groups of amateurs who have something in common, for instance an interest in VHF contesting or the city they live in, to share news and information with all of the other members. if you are new to a particular aspect of amateur radio, this is often the place to get advice from those with much more experience than you.

Some groups, such as those hosted by Yahoo or Google require you to enter a password (available free) before gaining access. A good example of a special interest group is http://uk.groups.yahoo.com/group/picastar/ which is used by those building the PIC-A-STAR project to compare notes and get help.

CDs and DVDs

An alternative to the Internet is the compact disk. Many computer programs are available, both free and paid-for, on CD. Advantages over downloading include no phone costs and the ability to re-install the software on a replacement computer with ease.

Publications such as the ARRL's *QST*, RSGB;s *RadCom* (including archives going back to pre-WWII days) and books such as this one are also available in full on disk. In addition to saving shelf space, digital books and magazines are usually fully-searchable (ie any word or phrase can be searched for - much more useful than a conventional Index). Visually impaired people can considerably magnify each page on screen if required, or even use a program that reads the publication out loud.

Developing External Hardware

The PC can be used to develop software that will eventually be copied to a chip in order to drive stand-alone equipment. This is often called embedded software, or firmware. An example of this is development of software for running on microcontrollers such as the family of PIC devices, described later on in this chapter.

Other external programmable hardware includes gate arrays and Complex Programmable Logic Devices (CPLDs) which are the modern replacement for the old TTL and CMOS chips.

A wide range of these devices from those equivalent to a few logic gates, to chips with millions of programmable cells, made by Xilinx and Altera, is available. Both of these companies provide free development software, which after a bit of practice will allow both simple and complex logic (and even some analogue) functions to be built using just the low cost chip and a few external components. There is even a choice of ways CPLD circuitry can be designed - from use of a pseudo high level language such as VHDL or by thinking in terms of logic gates and drawing out the functions schematically as if they were going to be made from separate logic gates.

Programming and development using CPLDs is outside the scope of this Handbook, but full details of two companies that provide devices and support hardware and software can be found at [18, 19].

The chapter on Software Defined radio also deals with the use of Complex Programmable Logic Devices

CONNECTING TO THE REAL WORLD

Sound and Audio Interfacing

To operate digital modes, a modern computer with sound card is required (computer specification depends on software, but even a 233MHz Pentium will provide a lot of pleasure). Some simple cables, easy to make, are also required, or a commercial interface such as the RIGBLASTER™ can be purchased. A conventional HF SSB transceiver is used, connected via the receiver audio output and microphone audio input. It is best if this is a modern solid-state unit, with good filters and low drift, but many operators use older rigs for RTTY, MT63, Hellschreiber and SSTV with no particular problems. These are the modes least affected by drift and poor frequency netting.

Some of the newer modes require very high stability, and very low frequency offset is necessary between transmit and receive. Most synthesised transceivers will suffice. A transceiver that drifts less than 5Hz per over will operate the newer modes very successfully. Unfortunately offset cannot be accurately corrected by using the Receiver Incremental Tuning (RIT).

Older VFO-controlled transceivers with poorer stability can still be used with wider bandwidth modes such as RTTY and MT63. DominoEX, while a relatively narrow-band mode, is very effective with older rigs, as it was specifically designed to be immune to drift and frequency offset.

The connections between computer and transceiver are quite straightforward. Most amateurs should be able to build the required cables. **See Fig 21.7**.

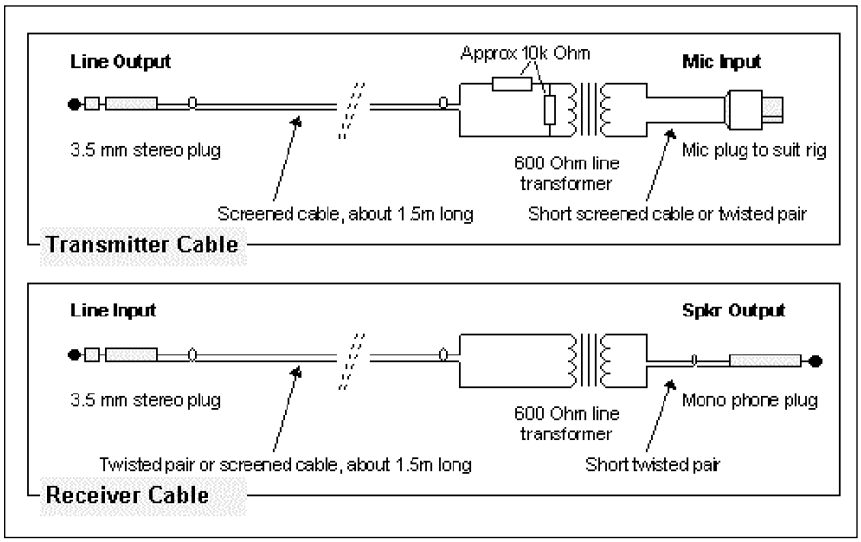


Fig 21.7: The simple cables used to connect PC to radio

The resistors in the transmit cable are used to attenuate the sound card signal so that it does not overload the transceiver. If the microphone socket is used, a lower value of resistor may be necessary across the transformer to further attenuate the audio. If an accessory socket is used, the values shown may suffice.

While the receiver cable is shown with a connector to be directly plugged into the external speaker socket on the receiver, with many rigs this will disconnect the speaker, which isn't helpful. It is best in this case to use an adaptor allowing both the PC cable and an external speaker to be connected.

There is a very good reason for not using the computer speakers instead of an external speaker on the transceiver. Computer speakers receive their audio from the sound card output, and by connecting the LINE IN signal from the radio to the LINE OUT or SPEAKER OUT and the speakers, the receiver output signal will also be sent to the microphone input of the transceiver. This causes feedback problems, especially if VOX is used.

The Importance of isolation

The transformers shown in Fig 21.7 provide complete DC isolation between the computer and the radio transceiver. The most compelling reason to do this is to prevent serious damage to the radio and computer.

Most power supplies are grounded for safety reasons. If the power supply cable to the transmitter becomes loose, the full 20A transmitter current can pass through the microphone circuit, down the cable and through the computer sound card to ground via the PC power cable. Even if the transmitter power cable is considered reliable, significant current could still flow through the sound card cable, causing instability, hum and RF feedback. The simple expedient of isolating the connections also reduces the risk of RF in the computer, and computer noises in the radio.

External soundcards

Although the computer's internal soundcard will fulfil the majority of requirements for operating most digital or data modes (see the chapter on Digital Communications) there are circumstances where an additional external audio interface would be useful.

For instance, a second soundcard module can be useful if a Software Defined Radio normally requires the computer's own soundcard for its audio, but another input / output is still required for running data modes (see the chapter on SDR).

While there are ways of transferring audio in software such as the Virtual Audio Cable software package [20] it may be easier to use a second soundcard and cross-connect audio cables between the two. Using a second, or even multiple soundcards can also make it easier to connect multiple receivers, modems and test equipment without having to continuously unplug and rearrange connectors. Modern operating systems from Windows 2000 onwards will accept multiple soundcards without even blinking.

A second soundcard can be installed into one of the computers extension slots, but these days the USB interface offers a more versatile route.

There is a wide range of external USB sound adapters on the market; these range from low cost headphone / microphone adapters such as the top unit shown in **Fig 21.8** up to the high quality top end external line interface such as that shown below it.

The high-end units offer proper line inputs and outputs and can usually be connected in exactly the same way as described above. Being designed for high end audio and music processing, some include audio isolation and proper decoupling.

Low cost headphone adapters, often referred to as 'headphone dongles' usually have to be treated differently as they only have a single mono microphone input and a stereo output jack for directly feeding headphones. To avoid the need for large value electrolytic capacitors inside the module, the headphone output is usually delivered as a directly coupled signal from the audio amplifier with all connections, including the common - the barrel of the 3.5mm jack socket - sitting above ground at typically 2.5V. The headphone signal lines swing

Fig 21. 8: (top) USB headphone dongle; (bottom) high-end external USB soundcard

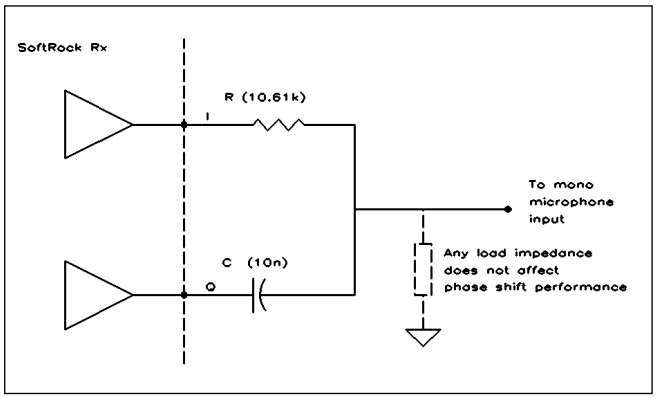


Fig 21.9: Combining I and Q channels to achieve sideband rejection over a narrow frequency range

either side of this to give a bi-directional drive across the headphone. The presence of this mean DC level gives us a problem if we want to use the audio output for wrapping round to another audio input, or to go off to a modem. If the common point is connected directly to ground, this DC will be shorted, probably resulting in damage to the headphone adapter unit.

A simple solution is just to insert DC isolation in the form of capacitors in series with both ground and audio paths. However, transformer isolation as shown in Fig 21.7 is the preferred solution as it offers additional noise immunity by separating audio and computer ground connections.

Also, beware that some very low cost units (such as 'eBay specials') provide an unfiltered Pulse-Width Modulated digital drive direct to the headphones. These will require extra filtering to remove the PWM switching components at a few tens of kHz and are best avoided, but the PWM signal may be useful as a drive to an EER type transmitter (see the LF chapter).

The mono microphone input is ground-referenced so no harm will come with a direct connection, although transformer coupling will help in reducing RF and digital induced noise into the audio path. Having a mono input prevents a full stereo I/Q input from SDR front ends such as the Softrock from being able to make use of such adapters.

If only a narrow bandwidth is wanted, for example of no more than a couple of hundred Hertz wide to use with a narrowband mode like WSPR, a simple phase shift circuit can be used to combine I/Q channels into one for sideband rejection. The circuit of **Fig 21.9** will give sideband rejection over a narrow range around 1500Hz when driven by the I/Q audio outputs from an SoftRock or similar SDR.

Another use of low cost headphone adapters is to build them into the actual equipment, such as an SDR or modem, for which audio interfacing is required so that a single USB interface cable is all that is needed to connect to a PC.

The five volt signal present on the USB can be tapped off to power external hardware by opening up the adaptor casing and connecting to the two outer pins of the USB socket. A continuity check will show which of these is connected to ground and which must be the 5V supply.

A low cost pair of computer speakers - especially battery operated ones, will usually function with a 5V supply, and a headphone 'dongle' built into a pair of scrap speakers can make a low cost additional audio output for an SDR, while keeping the computer's own soundcard free for other purposes.

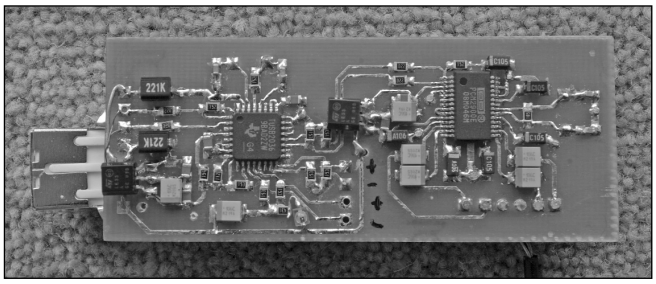


Fig 21.10: USB Codec and Hub for a stand alone soundcard module suitable for building into homebrew projects. An FTDI232 chip on the reverse side of the PCB gives a serial COM port

Single chip USB soundcard

Another solution is to build a dedicated USB soundcard into any new receiver or transmitter project. The PCM2900 Stereo Audio Codec with USB interface is a single chip 28 Pin SSOP package audio input / output facility. With the addition of little more than a 12MHz crystal, a few coupling/decoupling capacitors and a 3.3V regulator, a high specification dual line input / line output becomes available as an additional soundcard. As the USB port supplies 5V at up to 100mA a USB powered plug-and-play project is feasible.

The PCM2900 is available from several major component suppliers such as Farnell [13]. The photograph **(Fig 21.10)** shows one of these chips mounted on a PCB along with a TUSB2036 three port USB hub chip. Not shown, on the underside of the board is an FTDI FT232R chip providing a USB COM port.

Soundcard controls

Recording and playback levels, as well as input / output selection are made using the computer's own operating system, and invariably accessed on Windows machines by clicking on the loudspeaker icon in the lower right taskbar. This usually brings up a 'Sound Mixer' or 'Volume Control' window similar to that in **Fig 21.11**. The menu options vary between manufacturers, but it will always be possible to select between all the Input and output connections, to set recording and playback (volume) levels and to select which soundcard is the default. Hidden settings, or those not on immediate display, can usually be found in 'Advanced' or the Record menu options.

VOX and PTT Control

Most operators find VOX operation of digital modes quite appropriate and reliable, although the delay may need to be set longer than for Morse or SSB. If for some reason direct control

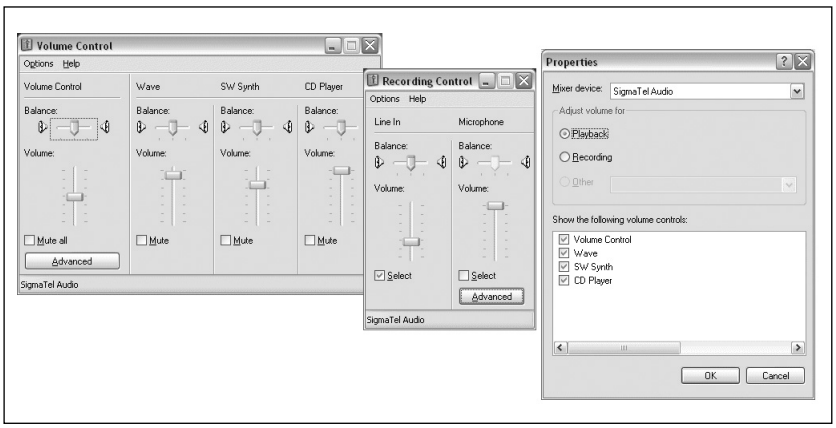
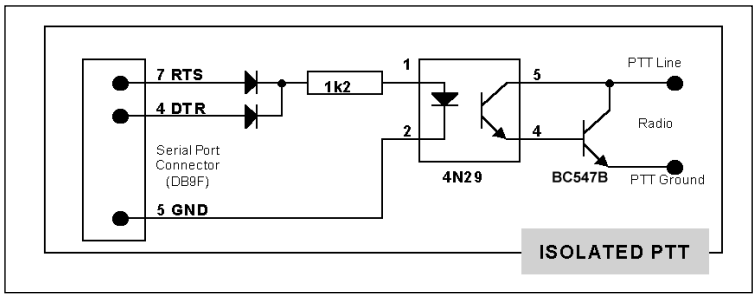


Fig 21.11: Some of the controls available in the Windows 'volume control' software

Fig 21.12: An opto-isolated PTT circuit



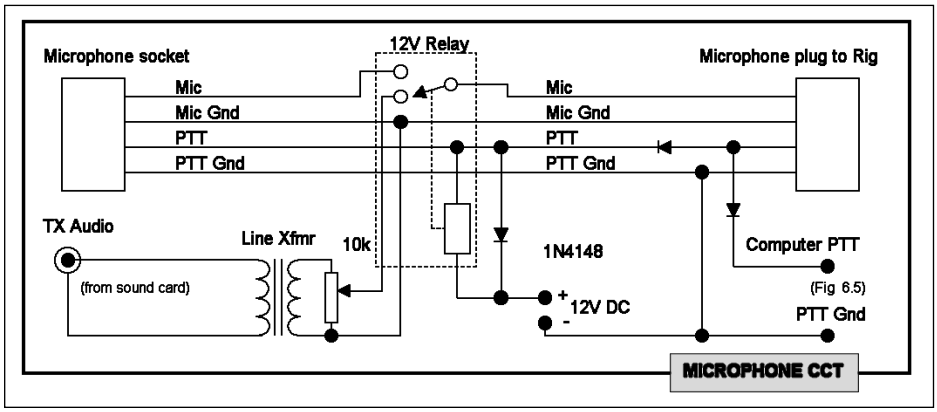


Fig 21.13: A simple way to connect microphone and computer

A simple home-made adaptor (see **Fig 21.13**) provides a way to operate voice and data modes interchangeably without disconnecting anything. Using this design, the data transmit cable is connected by default, but when the microphone PTT switch is depressed, the relay switches over the audio input and normal microphone use occurs. Operation is simple, and feels natural. The isolated PTT circuit of Fig 21.12 can be built into the same box, and the whole assembly replaces the transmit cable in Fig 21.7. There are several similar designs offered as kits [21].

There are several commercial interface designs available for users not disposed to building a home-made or a kitset interface, but not all provide full isolation. There are also USB interfaces suitable for laptop computers and others with no serial port or sound card.

An area that causes confusion among beginners is the business of setting up and adjusting the sound card. The adjustments are all performed in software, mostly using an application provided with the operating system, and once set for one mode or program, the settings should be correct for all the rest. There are two main software adjustments, for transmit and for receive, and it is not very obvious where to find these, especially the receiver adjustments. The better applications provide direct access to the adjustments. In addition to the gain settings, you need to select the correct inputs and outputs, and disable those not being used. The procedure and these adjustments are described in detail in the RSGB publication *Digital Modes for All Occasions* [22], a reference work recommended for both novice and experienced operators.

of the rig is necessary, the transmit control must also use an isolated circuit. An opto-coupler does this nicely, driving the Press-to-Talk (PTT) directly without requiring a relay or any further power supplies.

The digital mode software usually controls the transceiver via a serial port, by driving RTS or DTR (often both) positive on transmit, with an appropriate delay before sending tones out from the sound card. The design in **Fig 21.12** is an appropriate PTT circuit for a transceiver with positive voltage on the PTT line and a current when PTT is closed of up to 100mA or less.

Many transceivers include an 'accessory socket', offering line-level audio inputs and outputs for transmit and receive. Using these instead of the microphone socket and speaker socket can be really convenient, but can lead to a range of unexpected problems.

Sometimes PTT is not available from the accessory socket, and sometimes the VOX does not operate from this socket. The signal levels can also be quite different to the speaker and microphone connections. Even more troublesome, some transceivers leave the microphone operating while sending data through the accessory socket, so coughs, mutterings and keyboard clatter go out over the air!

Sometimes it is not desirable, or feasible, to use direct PTT control. Many modern PCs do not have a serial port, and those that do sometimes pulse the control signals on it which can cause unexpected transmit switching. The audio VOX circuit shown in **Fig 21.14** can be used to overcome this problem. The audio from the soundcard, which must be activated only during transmit, is rectified in a precision rectifier and fed to a comparator. When its level exceeds a threshold it triggers a timer to holdover short breaks in the audio. The output from the timer interfaces to the PTT control. Adjust delay and threshold settings to best suit your preferences and audio drive level.

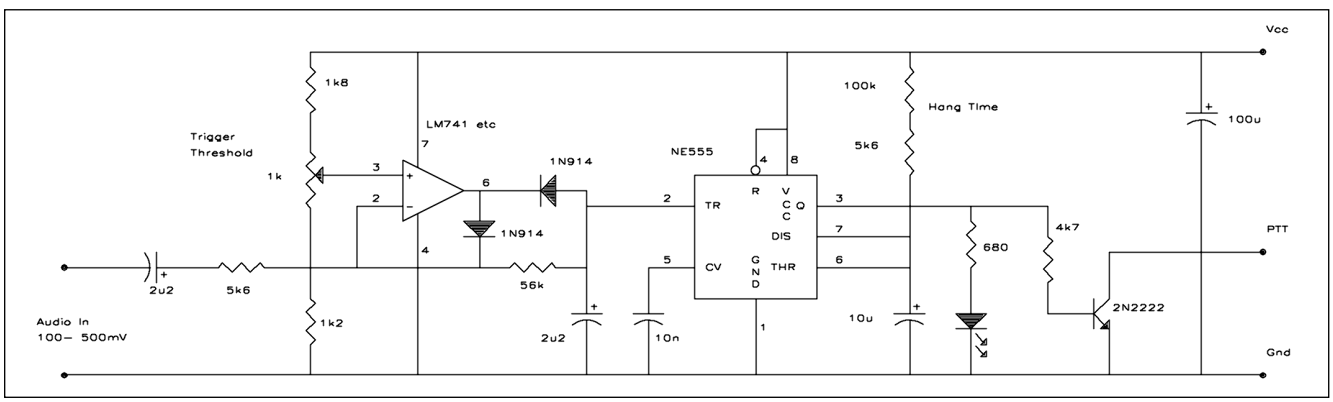


Fig 21.14: An audio-operated VOX switch is useful for when direct PTT control is not possible

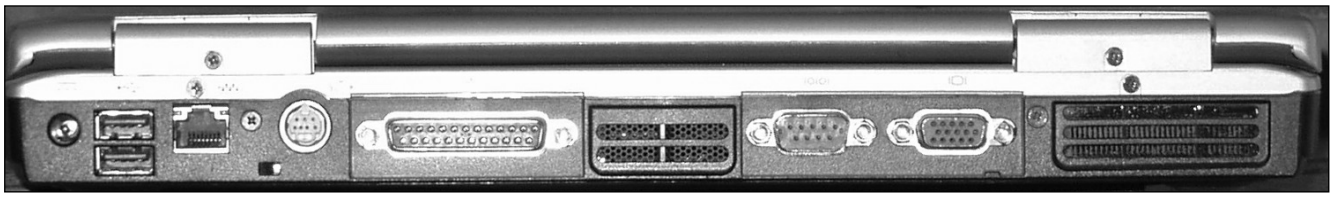


Fig 21.15: Input and output connectors on a laptop computer. From left: Two USB ports, network cable, video out, parallel port (25-way), serial port (9-way) and external monitor.

I/O Sockets

The range of I/O sockets on a standard computer has grown in recent years. Previously they used to offer one or two serial ports for connections to a modem or plotter for example, and a parallel port for a printers.

Serial means that the eight bits in a byte of data are sent one at a time down a single wire, whereas a Parallel port sends the bits simultaneously but needs eight wires for each byte, plus other lines for control. Both of these were slow, and could not provide an easy 'plug-and-play' solution for the growing range of computer accessories becoming available. To cope with this, the more modern USB interface was developed to overcome the limitations, and USB has now taken over just about all aspects of computer interconnectivity

More recently still, the high speed Firewire interface has emerged for streaming digital video and faster downloading from digital cameras. These later ports have in many cases, and particularly on laptops, displaced the older serial and parallel ports. **Fig 21.15** shows the range of input and output ports available on a typical laptop computer.

The serial ports

These are often called 'COM' or RS-232 ports, and were traditionally used for external modems (including packet radio TNCs), connecting to other computers, plotters and some early printers and other media.

Table 21.1 gives the pin connections for 9-pin and 25-pin serial connectors. Data is transmitted serially using stop and start bits to synchronise each end of the link, as shown in **Fig 21.16**.

The speed of transmission can be set over a wide range of values to suit the equipment and data transfer requirements, but 9600 bits per second (otherwise known as 9600 baud) is a useful mid-range value for many activities. Higher speeds up to 115200 baud (and in some cases depending on hardware, even higher rates), can be set. The number of bits sent can also be adjusted over the range 5, 6, 7 or 8, although these days 8 bits is the only real practically useful setting. Finally, simple error checking is provided for by the use of an optional parity bit.

This huge range of speeds and data formats, with its limited speed made the serial port a nightmare to set up in some cases, and contributed to its fall from favour for modern usage. However, there is one very big advantage, that makes the serial port concept live on. It is absolutely universal, everyone understands it and can implement the standard whatever computer language, operating system or hardware is in use!

Every computer made has always, in some manner or other, incorporated a serial port. This makes it the most widely supported format, ever, for interfacing between external hardware and in spite of its complexities in setting up, continues to live on. In many cases the inelegance of the multitude of serial formats is hidden from users, who just see a connection, but the serial RS232 format is there, hidden beneath a layer of software that has done the configuring automatically. A prime example of this is seen in GPS modules that communicate with their host over a Bluetooth radio interface. With suitable Bluetooth hardware the serial data can be seen and decoded to a form for connection to a true serial port. Serial interfaces, or UARTS, are also built into just about every microcontroller there is.

The other advantage of the serial port, is that, unlike the parallel port (see below) it is still fully supported in all programming languages for the latest operating systems. So writing your own software for communicating with the outside world does not require special drivers specific to the operating system.

Serial port support can be integrated seamlessly, usually from a pull down menu or by a line or two of high level programming code.

Pin No. 25-Pin	Pin No. 9-Pin	Function	Signal Name	Direction of signal flow
1	-	Protective Ground		
2	3	Transmit Data	TXD	DTE > DCE
3	2	Receive Data	RXD	DTE < DCE
4	7	Request To Send	RTS	DTE > DCE
5	8	Clear To Send	CTS	DTE < DCE
6	6	Data Set Ready	DSR	DTE < DCE
7	5	Signal Ground	GND	
8	1	Data Carrier Detect		
20	4	Data Terminal Ready	DTR	DTE > DCE
22	9	Ring Indicator		

Table 21.1:RS232 Port connections and signals

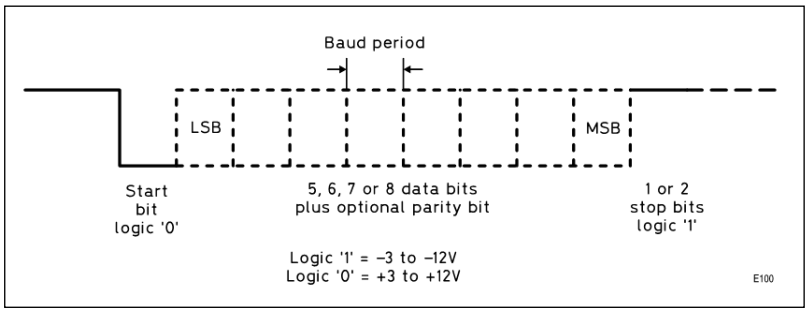


Fig 21.16: RS-232 Serial bit format

Parallel port

Parallel connectors, or printer ports, were commonly used for connecting printers before USB ports became universal. They are characterised by having parallel 0/5V TTL level outputs (eight data and four handshaking) and five status inputs on a 23-pin connector. The use of logic levels made the parallel port a favourite for interfacing to external logic hardware, and a number of the older chip device programmers, such as for PICs and CPLDs, used the printer port for their interface.

Fig 21.17: FDTI USB module

Since the advent of Windows 95 and later operating systems for PCs, direct access to the printer port became more difficult, requiring special software drivers, so its use as an optional I/O port has gradually diminished.

The Universal Serial Bus (USB)

Nowadays, the USB port reigns supreme on modern computers. It has solved many of the compatibility problems experienced with the older I/O systems, and offers a plug-and-play solution, having the ability to interrogate peripherals as they are connected and prompt for (named) drivers if new equipment is introduced. Furthermore, the USB port carries its own 5 volt power, allowing most low power accessories to be self-powered and removing the need for separate power supplies.

But the USB bus is not easy for amateurs to use directly - the USB specification runs to several hundred pages. However, the situation has improved recently as a range of chips have become available that allow users to interface to PCs via USB, and to make the connection completely seamless and transparent.

FTDI Chip [23] supply two integrated circuits, the FT232 and FT245 devices with very useful properties. The FT232 (as its type number might suggest) allows a virtual serial port to be established. The chip connects directly to a USB cable on one side of the interface and can extract the USB port's own power to use for accessories. The user side of the chip supplies the serial lines that would be seen on a COM port, receive and transmit data and two handshaking lines.

Drivers supplied with the device (in practice these can be downloaded from the manufacturer's web site) configure the interface as a new COM port which can be opened from any PC software capable of communicating with a serial port - which can be your own software written in any language of choice.

The FT245 chip adds another level of interfacing capability. As well as implementing the same Virtual Com Port mode, this device adds a high speed parallel interface with FIFO (first in, first out) suitable for connecting to custom logic and microprocessors; sending and receiving data at megabit per second rates. Full details, datasheets and programming details can be found in [23].

Complete USB-RS232 leads and modules are available that plug into a USB port on a PC and provide a 9-pin serial interface with correct signalling voltages, all powered from the USB port itself. These can provide a quick and easy solution for adding a serial port to a laptop or modern desktop. One such module can be seen in (**Fig 21.17**).

USB Parallel port interfaces are also produced, but unlike the COM port there is no universal way of addressing these. In practice they are only useful for adding an older style printer without USB capability. However, it may be possible to make some external devices or hardware 'look' like a printer, in which case such an interface could prove useful.

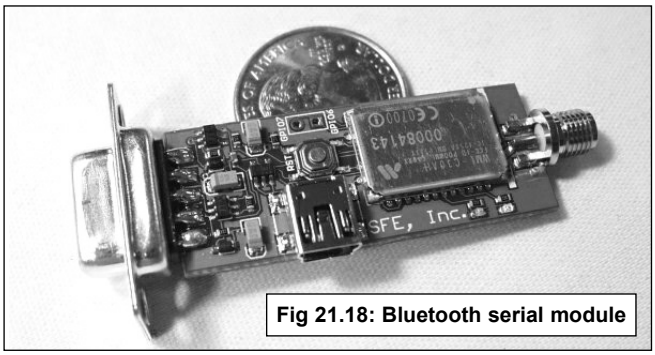


Fig 21.18: Bluetooth serial module

Bluetooth

Another option is to use the short range radio interfacing offered by Bluetooth. Working at 2.4GHz with a few milliwatts of power, this bi-directional radio link gives wireless communications over ranges of a few tens of metres. The protocols permit networking, printers, serial communications and audio to be carried between suitable peripherals. A Bluetooth serial module allows pseudo COM ports to be added in the same way as USB ones.

A Bluetooth serial interface is shown in **Fig 21.18**. Modern laptops often come with Bluetooth capability already built in, and interfacing to such a module is merely a matter of 'pairing' the devices, then opening the COM port that is then allocated. For older computers and desktops without Bluetooth capability, a Bluetooth 'dongle' can be inserted into a spare USB port, or a Bluetooth card can be installed. Again, the serial protocol offers wide versatility when it comes to programming and using your link with the outside world.

Network or Ethernet

Normally associated with inter-computer linking, the Ethernet port is increasingly being used as a high speed interface in its own right. With a data capability up to 100 MBits/s (with 1GB/s systems now appearing), it exceeds the capacity possible with USB and is the interface of choice for several Software Defined Radios.

A stand alone module such as the XPort from Lantronix shown in **Fig 21.19** permits a pseudo COM port (again!) to be set up, operating at speed up to a few MB/s. The module is set to a unique IP address on the network, and data is communicated to and from it as if it were any other network node, such as another computer or a modem.

In fact, the module contains a web page in its own right, and can be accessed using browser software such as Internet Explorer. Being a networked accessory, by suitably setting up IP addresses in your router or modem, it is possible to directly access this module from anywhere in the world via the Internet. This opens up the possibility for very long distance remote control.

Fig 21.19: XPort Ethernet interface module

Controlling your Radio

Many modern base-station transceivers incorporate sockets so that they can be controlled by a computer using special software. Station control programs, such as those used by DXpeditions and contesters, have facilities to control at least transmit and receive frequency. This can be extended to being able to control your station remotely over a modem (with the appropriate approval, if necessary), as described in [24].

Most rigs use a serial RS232 protocol, so the connection methods mentioned above can all be used. There are considerable differences between the control protocols from different manufacturers, but all are well documented in their respective manuals, and most transceiver control software can cope with all types, which can be changed in the setup menu.

Communications

Amateurs have been using home computers for making contacts ever since they became available - RTTY programs were available for the 'BBC-B' computer, for example. The chapter on Data Communications gives much more information.

Towards the end of the 20th century, ways were developed to link amateur radio and the Internet, so that radio amateurs can communicate even though one of them has no radio, or to interlink repeaters via the Internet. Three systems are in use at the time of writing: Echolink [25], eQSO [26] and IRLP [27].

MICROCONTROLLERS

Small microcontroller chips such as the PIC appear in several projects throughout this Handbook. Here a short overview of these is described, together with how you can program these for your own personalised tasks. Microcontrollers are single chips that can probably be considered the simplest to run stored programmes and execute actions based on the results.

A microcontroller chip is usually a small processor that includes its own programme and data memory, a range of peripherals such as a serial interface and timers, and drivers for input and output

Mnemonic.	Operands	Description	Cycles	Status Bits Affected	Notes
Byte Oriented File Register Commands					
ADDWF	*f* , d	Add W to *f*, place result in d (*f* or W)	1	C,DC,Z	1,2
ANDWF	*f* , d	AND W with *f*, place result in d (f or W)	1	Z	1,2
CLRF	*f*	Clear *f*	1	Z	2
CLRW		Clear W	1	Z	
COMF	*f* , d	Complement *f* place result in d	1	Z	1,2
DECF	*f* , d	Decrement *f*	1	Z	1,2
DECFSZ	*f* , d	Decrement *f*, skip next instruction if zero	1 or 2		1,2,3
INCF	*f* , d	Increment *f*	1	Z	1,2
INCFSZ	*f* , d	Increment *f*, skip next instruction if zero	1 or 2		1,2,3
IORWF	*f* , d	Inclusive OR W with f	1	Z	1,2
MOVF	*f* , d	Move *f* to destination	1	Z	1,2
MOVWF	*f* ,d	Move W to *f*	1		
NOP		No Operation	1		
RLF	*f* , d	Rotate Left *f* through carry	1	C	1,2
RRF	*f* , d	Rotate Right *f* through carry	1	C	1,2
SUBWF	*f* , d	Subtract W from *f*	1	C,DC,Z	1,2
SWAPF	*f* , d	Swap nibbles in *f*	1		1,2
XORWF	*f* , d	Exclusive OR W with *f*	1	Z	1,2
Bit Oriented File Register Commands					
BCF	*f* , b	Clear bit B of register *f*	1		1,2
BSF	*f* , b	Bit Set *f*	1		1,2
BTFSC	*f* , b	Bit Test *f*, skip next instruction if clear	1 or 2		3
BTFSS	*f* , b	Bit test *f*, skip next instruction if set	1 or 2		3
Literal and Control Operations					
ADDLW	k	Add literal to W	1	C,DC,Z	
ANDLW	k	AND literal with W	1	Z	
CALL	k	Call Subroutine	2		
CLRWDT		Clear Watchdog timer	1		
GOTO	k	Go to address	2		
IORLW	k	Inclusive OR literal with W	1	Z	
MOVLW	k	Move literal to W	1		
RETFIE		Return from Interrupt	2		
RETLW	k	Return with literal in W	2		
RETURN		Return from Subroutine	2		
SLEEP		GO into standby / sleep mode	1		
SUBLW	k	Subtract W from Literal	1	C,DC,Z	
XORLW	k	Exclusive OR literal with W	1	Z	

NOTES:

1) *When an I/O register is modified as a function of itself (eg MOVF PORTB, f) the value used will be that value present on the pins themselves rather than that stored in the output registers.*
2) *Clears prescalar when executed on the TMR0 register*
3) *If the Programme Counter (PC) is modified or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a NOP.*

Table 21.2: PIC 16Fxxx instruction set

pins; they are often designed to be used in completely stand-alone applications.

Programming a PIC for any task is similar to writing in any other programming language, although much more knowledge of the target processor is needed. Many programmers use a high level language such as C or Basic for microcontrollers, and many users prefer this route for the ease of programming it offers. The best compilers can 'hide' the complexities of the chip, but these versions can be quite expensive - low cost, and even free or shareware C language compilers do exist, but using these often gives little advantages over programming in assembler.

There is no doubt that for optimum processing speed and keeping resultant code size small, as well as understanding all the device's idiosyncrasies, the native assembler is more efficient than a high level language such as C. For some tasks however, such as string handling, a high level language can prove more suitable. **Table 21.2** shows the assembler commands available for the mid range PIC family

To develop PIC software (more correctly called firmware as it is embedded into a final application) a Personal Computer of some sort is invariably used to develop the code via various software tools supplied, usually free of charge, from the PIC manufacturers.

A programmer driven from the PC is then used to load the developed code into the chip. Programmers of different sorts and levels of sophistication can either be purchased from most electronic component suppliers, or simpler ones capable of programming many, but not necessarily all, of the PIC processor types can be quite easily built. A modern USB controlled programmer is shown in **Fig 21.20**. Most programmers run off either the parallel port or the serial port, and now most of the commercially manufactured units employ a USB interface.

There are four stages to producing a fully functional PIC design:

1. **Design the circuit and hardware**, making use of the PIC data sheets to determine which pins are to be used for each function - many of the special peripheral functions requires their interfaces to be on specific pins.
2. **Write the assembler code** - the list of instructions to perform the function required. This in the form of an assembler, or source, file in text format and usually generates a file with a name like xxx.ASM where xxx is the file name. There is usually a section that defines memory locations and register names and sets up peripheral functions, followed by the operating code itself. A range of 'include' files are provided by Microchip for each processor type to make naming programming easier by naming all the registers and individual control bits. On a PC, a text editor such as *Notepad* or the DOS 'EDIT' command can be used to write the source code.
3. **Assemble the source code into machine code**. The assembler generates a file xxx.HEX which contains the machine code in a form useable by the programmer. It also generates a full listing, xxx.LST which contains a repeat of the source code with all associated memory maps, variables and labels shown. Any errors in the source code such as illegal calls, mistyped labels or variable names or illegal values appear in an error file, xxx.ERR; this will be empty for a 'perfect' assembly. Warnings are also produced when certain operations that may cause potential problems in some circumstances are made. An excellent assembler is that produced by Microchip, called 'MPASM', which can be downloaded from [28] .
   MPASM is called from the command line by invoking it along with the filename of the .ASM file eg 'MPASM TESTPROG' After assembly is complete, it reports back on the number of errors encountered in the process along with any messages or warnings about the code. Details of these are stored in the. ERR file.
4. **Program the chip**. Most modern devices are programmed serially by making use of three of the pins plus ground. The master reset pin is raised to +13V while the chip is powered normally from its 5V rail, and the data is clocked in serially using two pins, one for clock and the other for data. Some programmers have a Zero Insertion Force socket for programming raw chips; others just provide the four connections needed for in-circuit programming. Microchip supply a range of programmers and debugging tools and details of a several programmers suitable for home constructors are included in **Table 21.3**. This list is by no means exclusive and many more programmer designs can be found by searching the web.

An intermediate stage between steps 3 and 4 is available - that of simulation. Host computer software takes the .ASM or .HEX files and simulates the functioning of the PIC hardware with the user code which can be single stepped if required, allowing all intermediate values, the state of all registers and I/O lines to be examined at any time for debugging. Alternatively, break points can be set and registers checked at this point. It is only a simulation, and the functioning of peripherals such as A/D converters has to be assumed. Timing and clock cycle count is shown directly, a function which can be very tedious if undertaken manually on the source code.

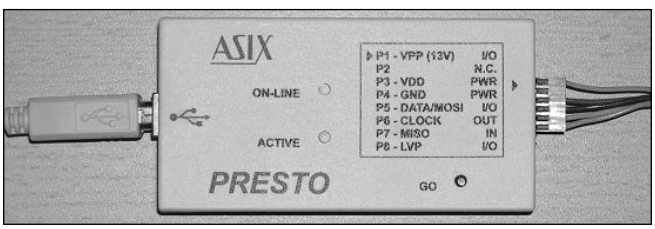


Fig 21.20: The Presto PIC programmer

Microchip PICkit 2 & PICkit 3
A small versatile USB programmer produced by Microchip. It will handle all the modern devices although the obsolete 16F84 and a few other older devices cannot be programmed with it. As well as PIC programming, it can also be used with serial EEPROMS and offers in-circuit debugging and several other useful utilities. See the microchip website [28] for more details. The PICkit is stocked by most of the major component suppliers, and often comes bundled with a PIC development board - sometimes as part of a special offer.

EPIC Programmer
Supplied by MELabs [29]. A simple programmer supplied as a PCB with ZIF socket making use of the parallel port. Can be used with any operating system.

The WISP648 Programmer
A design for home construction that requires a programmed PIC as part of the design. A chicken and egg situation arises here if this is to be the first programmer, but ready programmed PICs are available from the designer in [30].

The Presto Programmer [31]
A modern USB powered device. It can handle all microcontrollers and CPLDS as well as most serial EEPROM and Flash memory chips.

Table 21.3: PIC programmers

As an alternative, stages 1 to 3 can be replaced by a high level programming language. Special versions of Basic and C can be used to write code for the PICs (as well as most other types of microcontroller). These produce the.HEX file directly by the compiler and users rarely see the assembler code produced.

Programming in a high level language can be considerably easier for many people as it removes most of the need for a detailed understanding of the device's architecture. However, the amount of object code produced by most high level languages, particularly low cost or free ones, is generally larger than when writing the same functions in assembler. Therefore a high specification, more expensive, chip with more memory has to be employed.

Also, writing time-critical code, such as that to be used in some simple DSP functions is impossible as the high level languages do not take account of processing time and clock cycles.

DSPic

The latest addition to the Microchip stable is the new DSPic range which has a more complex instruction set aimed at digital signal processing as well as microcontroller functions. While being aimed mostly at the automotive and power supply / conditioning market, the range of instructions makes them suitable for many audio and SDR related tasks. [32] has more details.

Other Microcontrollers

The Microchip PIC is not the only microcontroller available for experimenters, although it is probably the most well known. It is not the fastest and for significantly higher operation speed, Atmel's AVR microcontrollers come in a range of sizes and capabilities, and are the microcontroller of choice for many. More details can be found in [33].

Project: PIC Based Audio Source

The circuit of **Fig 21.21** is for a PIC based audio source that implements a low rate Direct Digital Synthesizer (DDS) running on a standard 'workhorse' PIC 16F628. This device contains a counter-timer that can be configured to generate an interrupt when it overflows. An interrupt, as its name suggests, is a routine that pauses whatever the processor is currently doing and sends it away to do something else; when complete, the processor returns to its original task - which for the PIC can just be sleeping. Interrupts are ideal for timing and Digital Signal Processing tasks as their timing can be accurately controlled.

On the Microchip PIC range [28], the counter timer can be run from the processor's internal clock signal which in turn is derived from the crystal or externally supplied oscillator. On the 16F family, the specification states this can be up to 20MHz, which is divided by four internally for a maximum processor clock of 5MHz. For LF and audio generation we need to be able to generate at least the range of frequencies that a PC soundcard can manage, which these days means a sampling rate of at least 48kHz although a sampling rate closer to 100kHz would be desirable. To be able to generate audio tones of virtually any arbitrary frequency, a 32 bit DDS architecture is needed.

A DDS is no more than a counter that is incremented by a fixed value for each clock input, rolls-over when it reaches maximum and starts counting again. The counter value at each instant is used as the input to a sine lookup table. The resulting value looked-up from the table is sent to a Digital to Analogue (D/A) converter whose output, after filtering to remove clock and alias products is the wanted audio waveform. The value the counter is incremented each time is related to the wanted output frequency as follows

A counter of length 32 bits rolls over every 232 counts, or each 4294967296 clocks. This defines the underlying resolution or step size for the DDS. For a 100kHz input clock it is therefore 100kHz / 232 = 0.00002328Hz (23.3µHz) The value by which the counter has to be incremented each time is then given by N = Fout / Fclock * 232. For an output of 8.98kHz with a 100kHz clock N is 8980/100000 * 232 = 385688063 or expressed in hexadecimal 0x16FD21FF.

The PIC's internal clock can run at up to 5MHz which is the rate at which instructions can be executed. If we want to generate a waveform by sampling at 100kHz this means an absolute maximum of just 50 instructions can take place during each sampling instant. Several instructions are used-up just processing the interrupt and forcing the timer-counter to divide by 50 without actually contributing to waveform generation. Fortunately is it just possible using a PIC's internal 8 bit architecture to implement a 32 bit addition, table lookup and output to a D/A converter in less than 40 clock cycles.

The algorithm

The registers in the PIC are eight bits wide so four are concatenated into a 32-bit accumulator - we'll call these D3 through to D0. Four more registers, F3/2/1/0, hold the value N determining the frequency, and for a single wanted tone these remain unaltered. Every interrupt cycle (at 100kHz for a 20MHz oscillator) the value in the F registers is added to the accumulated D3 - D0, which just overflows back to zero when its maximum count is reached.

The most significant byte stored in D3 alone is then used as the address for the sine look-up table. The resulting 8-bit value is placed on the PICs 8-bit parallel output Port-B.

The 32 bit addition, table lookup and output can be managed in about 35 clock cycles within the interrupt service routine,

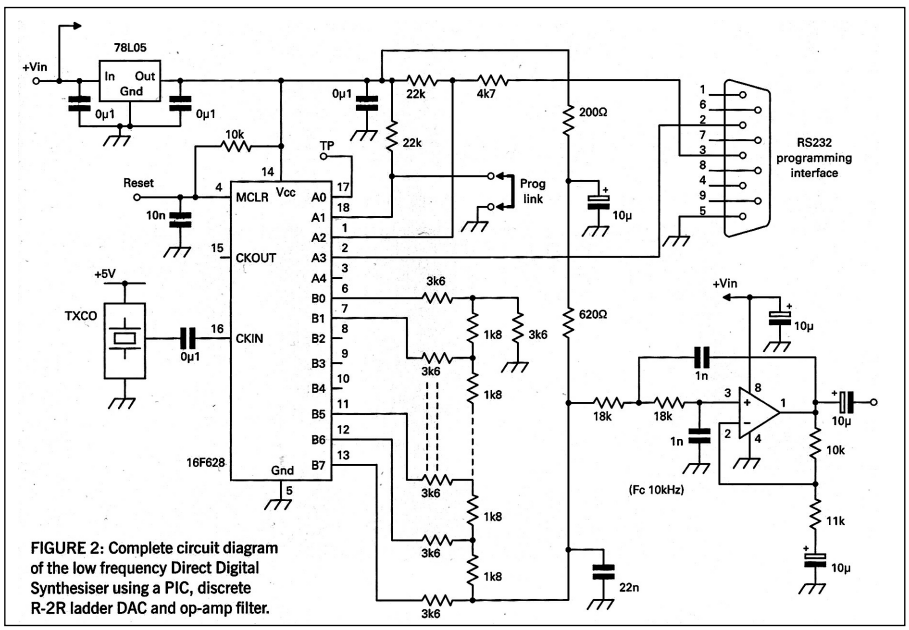


FIGURE 2: Complete circuit diagram of the low frequency Direct Digital Synthesiser using a PIC, discrete R-2R ladder DAC and op-amp filter.

Fig 21.21: Complete circuit diagram of the low frequency Direct Digital Synthesizer using a PIC, discrete R-2R ladder D/A converter and op-amp filter

leaving a small amount of spare capacity. The complete listing can be found at [34],

D/A converter

A ladder made up of resistor pairs in the ratio of 1:2 driven directly from PORTB. CMOS outputs, especially those on PIC devices switch between 0V and Vcc (usually 5 volts) with about 10 ohms of internal resistance so the 8-bit PORTB output from the PIC can directly drive the sixteen 1.8k and 3.6k resistors making up the chain at even lower cost than a custom D/A chip.

Amplitude resolution and spurii

The restricted table-lookup capabilities of this baseline PIC family mean that tables with more than 256 entries and more than an 8-bit output word are complicated to arrange, and would not fit into the limited amount of clock cycles allowed, but for simple audio and VLF purposes the 8-bit DDS table is adequate. The rule of thumb for all DSP that says spurious and quantising noise levels will lie at roughly 6.N dB below full scale, where N is the number of bits. suggests an 8-bit lookup table ought to give about -48dBc spurious.

The DDS concept works best where output frequency is kept lower than a quarter of the clock. It cannot, in any case ever go above half the clock rate due to the Nyquist sampling criteria. For a simple 8-bit table like this it is probably better to stay below one-sixth of the clock

Output filter

The D/A output now has to be filtered to remove alias products and spurii above 10kHz. A single stage, third order opamp filter can do the job admirably. The R-2R ladder maintains a constant output impedance of 1.8kΩ whatever the output voltage.

Biasing the output opamp filter and output levels ideally needs the output from the ladder to have its DC level raised and its 0-5V amplitude range attenuated. This is done by adding a 620Ω pull up resistor to 5V, through another 200 ohms decoupled to prevent noise of the 5V rail appearing directly in the audio path.

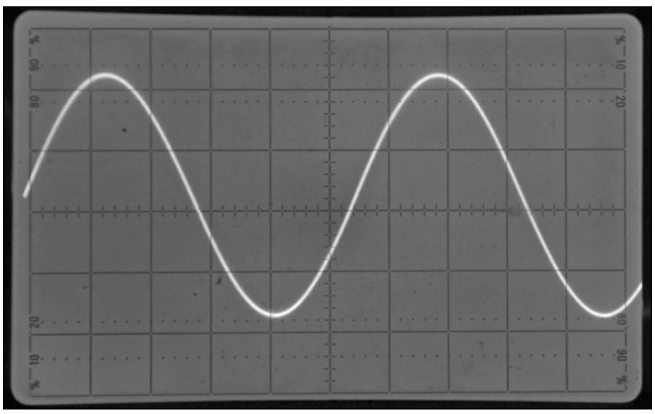


Fig 21.22: Oscillogram of the PIC DDS sinewave output

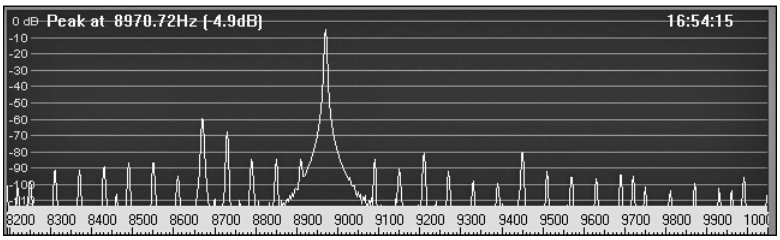


Fig 21.23: Output spectrum of the simple PIC-based audio frequency DDS

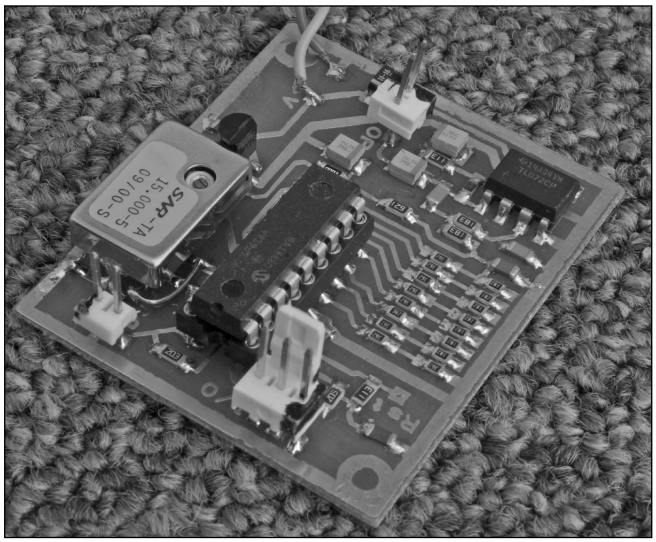


Fig 21.24: The finished PIC based Low Frequency DDS Source

The result is an output impedance of 461 ohms and a peak - peak waveform amplitude of 1.28V superimposed on a mean DC level of 3.44V. The first stage of filtering comes from the 22nF capacitor from between the output of the resistor ladder to ground for a first stage cutoff at 15.6kHz. The opamp filter, whose values are taken from standard tables has an underlying gain of 2.1 times, resulting in an output of 2.7V peak-peak, or very close to 1V RMS.

The values shown give a cut-off at 10kHz, but a higher frequency would be possible by proportionally reducing the value of the two 1nF capacitors and/or the 18k resistors if some degradation in spurii at the upper frequency band is accepted.

Results

Fig 21.22 shows the generated and filtered waveform at 8.9kHz with its close in spectrum in Fig 21.23. Spurii are typically -50 to -70dB individually, but as with any DDS, their actual frequency and spread is unpredictable.

All we can really give is an upper limit to their magnitude, and here they are comfortably below the -48dBc predicted from the 6.N dB rule-of-thumb. The relative power of all the spurii when combined over the full audio band probably do amount to about the expected value.

Software

The PIC code for the DDS to generate a single frequency can be found at [34]. The archive also contains a more complex version of the software with two stored frequencies that can be selected externally for frequency shift keying, These can be updated under RS232 serial control with auto baud rate determination allowing for arbitrary choice of PIC clock. Full details are supplied in the accompanying documentation, including the PCB layout as shown in Fig 21.24.

DIGITAL SIGNAL PROCESSING

DSP is the latest aspect of computing that makes all the soundcard based data modes and Software Defined Radio possible (see also the chapter on SDR).

Essentially, DSP requires an analogue signal to be converted to a succession of samples whose value represents the instantaneous value at the instant of sampling. The rate at which the waveform is sampled is called the sampling rate.

Sampling and the Nyquist Limit

An audio signal has to be sampled at a sufficient rate, the Nyquist limit, to minimise alias products and distortions, as described in the chapter on Principles. The other aspect to be considered is the number of discrete levels at the output of the A/D converter. Too few, and the errors introduced by the quantising process contribute an unacceptable level of quantising noise; too many, and the A/D converter becomes complex and expensive. The number of levels is related to the width of the output data in bits. An 8-bit A/D has 256 quantising levels, a 16-bit A/D (as in the majority of soundcards) offer 2^{16} levels = 65536.

Fig 21.25 shows the quantising process graphically. As a rule of thumb, the maximum dynamic range (or Signal to quantising Noise) that an A/D converter offers is given approximately by:

S/N (dB)= 6 x N where N is the number of bits

So a 16-bit soundcard gives approximately 96dB of dynamic range, a simple 8-bit converter only a maximum of 48 dB dynamic range.

Signal Processing

Once the waveform has been turned into a series of numbers, the real part of DSP can start. Several functions available in the analogue domain have direct equivalents in DSP. Frequency mixing is performed by multiplication of each waveform sample with a sample of a numerically generated Local Oscillator waveform (another series of numbers).

In practice, nearly all DSP is performed using two parallel channels consisting of 0 and 90 degree phase shifted signals, as this allows cancellation of unwanted mixer sidebands similar to the phasing method of SSB generation. In analogue processing terms, this can be thought of as separating the two image frequencies output from a mixer. This quadrature sampling of two parallel streams is usually referred to as In-phase and Quadrature, or I/Q processing. **Fig 21.26** shows this graphically.

I/Q data is referred to as a Complex signal, and consists of both negative and positive frequency components. Complex signals and negative frequencies are an aspect of DSP that many amateurs fail to visualise, but so long as it is thought of as only a mathematical short cut, and that negative frequencies are every bit as real as positive ones in the final result, you can't go far wrong.

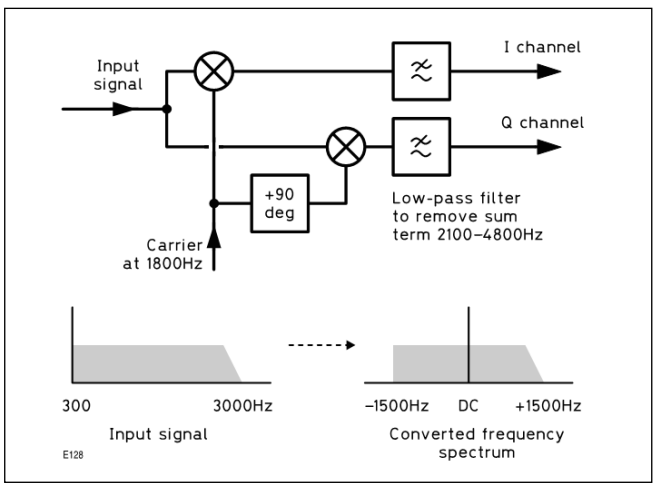


Fig 21.25: Frequency spectrum folding in I/Q mixing

A local oscillator in DSP consists of Sine and Cosine samples of a Numerically Controlled Oscillator, generated from a sine look-up table stored in memory..

Filtering is another task that is common with the analogue world, and consists of multiplying a block of successive samples with a set of filter coefficients, then summing all the results into a single, filtered, sample. The block of input samples is then shifted one sample to include the latest one and rejecting the earliest, and the whole multiplication / summation performed again to give the next filtered sample. When the DSP process is complete, the samples can be passed to a Digital to Analogue converter to generate real audio, for example.

The processes just described are all that is needed for a Software Defined radio for SSB or CW reception. A simple soundcard-only SDR doing just these processes is that designed for listening to VLF transmissions such as SAQ. SAQRx is a soundcard receiver written by SM6KLM.

It uses the soundcard sampling at 44kHz to receive radio signal up to 20kHz, which are then filtered to 3000, 1000, or 300Hz bandwidth and downconverted to audio for replaying. The LO can be tuned with the mouse or keyboard, and the IF bandwidth set the same way. SAQRx can be obtained from [35]

The FFT and Visual Displays

But DSP can do a lot more than replace analogue processing. The Fast Fourier Transform is a widely used example of a technique only available in DSP. The FFT takes a block of samples of any waveform, and performs a mathematical process on them to give a set of samples representing the spectrum of that waveform in the Frequency Domain.

The resulting data can be used to show the spectrum of the signal (up to half the sampling rate), or for other more complex tasks such as data demodulation.

For the process to work, the block has to be a binary power of two, so block lengths of 256, 512 all the way up to 262144 (and sometimes higher) are used. The resulting spectrum is generated to a resolution of the sampling rate divided by the block length. So, for example, an 8kHz sampled waveform passed through an 8192 point FFT will output a series of values consisting of the spectrum of the waveform in units of 8000/8192 = 0.977Hz multiples.

These successive frequency samples are usually referred to as bins (as in waste bin). Bin zero contains the DC component of the waveform, bin 1 the amplitude of the frequency components at 0.977Hz, all the way up to bin 4095 that contains the amplitude of the frequencies between 3998 and 3999Hz approximately.

Above this Nyquist limit, the values are repeated, mirrored. If complex, or I/Q values are put into the FFT process, this mirrored

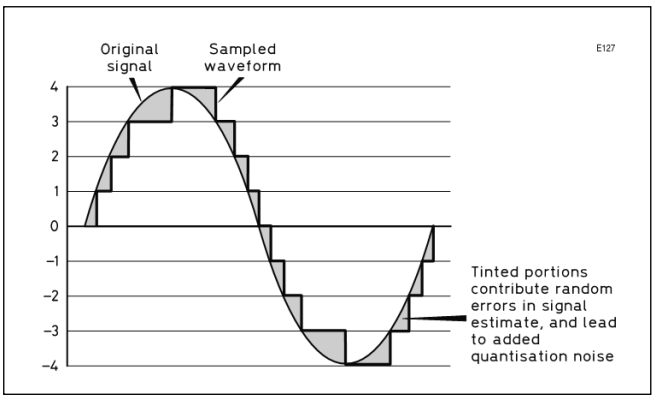


Fig 21.26: Generation of quantisation noise

spectral output is modified to show the complex spectrum above and below the zero term.

There are many spectral analysis programs around now, and several are described in the Data Communications chapter.

Other aspects of DSP are too numerous to mention, and beyond the scope of this Handbook. More details of DSP and some practical examples can be found in [36]

REFERENCES

[1] *The Amateur Radio Operating Manual*, 7th ed, Don Field, G3XTT, and Steve Telenius-Lowe, 9M6DXX, RSGB
[2] www.cadsoftusa.com/freeware.htm
[3] www.terrypin.dial.pipex.com/ECADList.html
[4] www.aade.com/filter.htm
[5] www.zerobeat.net/G4FGQ/index.html
[6] www.tonnesoftware.com/
[7] www.qsl.net/dl4yhf/spectra1.html
[8] www.cebik.com/model/nec.html
[9] RSGB web site, www.rsgb.org
[10] ARRL web site, www.arrl.org
[11] Google Internet search engine, www.google.co.uk, or www.google.com
[12] Yahoo Internet search engine, www.yahoo.co.uk, or www.yahoo.com
[13] http://uk.farnell.com
[14] www.maplin.co.uk/
[15] http://rswww.com/
[16] www.ebay.co.uk/
[17] www.amazon.co.uk, or www.amazon.com
[18] www.xilinx.com/
[19] www.altera.com/
[20] VAC (Virtual Audio Cable). www.screenvirtuoso.com/vac.html
[21] The BARTG publish designs from time to time. See www.bartg.org.uk/
[22] Appendix D, *Digital Modes for All Occasions*, Murray Greenman ZL1BPU, RSGB
[23] www.ftdichip.com/
[24] 'There's a remote possibility . . .', David Gould, G3UEG, *RadCom*, Aug/Sep 2005
[25] www.echolink.org/
[26] www.eqso.org
[27] www.irlp.net/
[28] www.microchip.com
[29] www.melabs.com
[30] www.voti.nl/wisp648/
[31] http://www.asix.net/tools/prg_presto.htm
[32] Microchip PIC Microcontrollers and DSPic. www.microchip.com
[33] www.atmel.com/
[34} www.g4jnt.com/PIC_DDS.zip
[35] http://sites.google.com/site/sm6lkm/saqrx
[36] *"Command" - Computer, Microcontrollers and DSP for the Radio Amateur*. Andy Talbot, G4JNT, RSGB 2003

About the Author

Andy Talbot *BSc, G4JNT, is a professional electronics engineer currently working in the field of radio communications and signal processing. He has been a radio amateur for 29 years, with his main interests being at the two opposite ends of the spectrum - microwaves and the LF bands. He has always been a home constructor, designing and building equipment from scratch right from the early days. Working on the new 73 and 137kHz bands started off his interest in DSP, and in conjunction with G3PLX pioneered the first use of ultra- narrowband techniques and low data-rate signalling on the amateur bands. Later, in conjunction with G4GUO, he took part in the first amateur use of digital voice at HF. He has written many articles for various journals, and writes the 'Data' column for the RSGB's* RadCom *magazine.*