Contents

Chapter 16 — CD-ROM Content

Supplemental Files

- Table of digital mode characteristics (section 16.6)
- ASCII and ITA2 code tables
- Varicode tables for PSK31, MFSK16 and DominoEX
- Tips for using *FreeDV* HF digital voice software by Mel Whitten, KØPFX

Chapter 16

Digital Modes

The first Amateur Radio transmissions were digital using Morse code. Computers have now become so common, households often have several and there are a large number of other digital information appliances such as tablets, netbooks and smart phones. E-mail and texting are displacing traditional telephone communications and Amateur Radio operators are creating new digital modes to carry this traffic over the air or to solve a particular need. Many new modes can be entirely implemented in software for use with a computer sound card, making tools for experimentation and implementation available worldwide via the Internet.

This chapter will focus on the protocols for transferring various data types. The material was written or updated by Scott Honaker, N7SS, with support from Alan Bloom, N1AL, and Pete Loveall, AE5PL. Kok Chen W7AY contributed the sections on RTTY, PSK, and MFSK. Mel Whitten, KØPFX, and David Rowe, VK5DGR, contributed a new section on *FreeDV* digital voice. Modulation methods are covered in the **Modulation** chapter and the **Digital Communications** supplement on the *Handbook* CD discusses the practical considerations of operating using these modes. This chapter addresses the process by which data is encoded, compressed and error checked, packaged and exchanged.

There is a broad array of digital modes to service various needs with more coming. The most basic modes simply encode text data for transmission in a keyboard-to-keyboard chat-type environment. These modes may or may not include any mechanism for error detection or correction. The second class of modes are generally more robust and support more sophisticated data types by structuring the data sent and including additional error-correction information to properly reconstruct the data at the receiving end. The third class of modes discussed will be networking modes with protocols often the same or similar to versions used on the Internet and computer networks.

16.1 Digital "Modes"

The ITU uses *Emission Designators* to define a "mode" as demonstrated in the **Modulation** chapter. These designators include the bandwidth, modulation type and information being sent. This system works well to describe the physical characteristics of the modulation, but digital modes create some ambiguity because the type of information sent could be text, image or even the audio of a CW session. As an example, an FM data transmission of 20K0F3D could transmit spoken audio (like FM 20K0F3E or 2K5J3E) or a CW signal (like 150H0A1A). These designators don't help identify the type of data supported by a particular mode, the speed that data can be sent, if it's error-corrected, or how well it might perform in hostile band conditions. Digital modes have more characteristics that define them and there are often many variations on a single mode that are optimized for different conditions. We'll need to look at the specifics of these unique characteristics to be able to determine which digital modes offer the best combination of features for any given application.

16.1.1 Symbols, Baud, Bits and Bandwidth

The basic performance measure of a digital mode is the *data rate*. This can be measured a number of ways and is often confused. Each change of state on a transmission medium defines a *symbol* and the *symbol rate* is also known as *baud*. (While commonly used, "baud rate" is redundant because "baud" is already defined as the rate of symbols/second.) Modulating a carrier increases the frequency range, or bandwidth, it occupies. The FCC currently limits digital modes by symbol rate on the various bands as an indirect (but easily measurable) means of controlling bandwidth.

The *bit rate* is the product of the *symbol rate* and the number of bits encoded in each symbol. In a simple two-state system like an RS-232 interface, the bit rate will be the same as baud. More complex waveforms can represent more than two states with a single symbol so the bit rate will be higher than the baud. For each additional bit encoded in a symbol, the number of states of the carrier doubles. This makes each state less distinct from the others, which in turn makes it more difficult for the receiver to detect each symbol correctly in the presence of noise. A V.34 modem may transmit symbols at a baud rate of 3420 baud and each symbol can represent up to 10 discrete states or bits, resulting in a *gross (*or *raw*) *bit rate* of 3420 baud × 10 or 34,200 bits per second (bit/s). Framing bits and other overhead reduce the *net bit rate* to 33,800 bit/s.

Bits per second is abbreviated here as *bit/s* for clarity but is also often seen as *bps.* Bits per second is useful when looking at the protocol but is less helpful determining how long it

Table 16.1
Data Rate Symbols and Multipliers

Name	*Symbol*	*Multiplier*
kilobit per second	kbit/s or kbps	1000 or 10^3
Megabit per second	Mbit/s or Mbps	1,000,000 or 10^6
Gigabit per second	Gbit/s or Gbps	1,000,000,000 or 10^9

takes to transmit a specific size file because the number of bits consumed by overhead is often unknown. A more useful measure for calculating transmission times is *bytes per second* or *Bps* (note the capitalization). Although there are only eight bits per byte, with the addition of start and stop bits, the difference between bps and Bps is often tenfold. Since the net bit rate takes the fixed overhead into account, Bytes per second can be calculated as $bps_{net}/8$. Higher data rates can be expressed with their metric multipliers as shown in **Table 16.1**.

Digital modes constantly balance the relationship between symbol rate, bit rate, bandwidth and the effect of noise. The Shannon-Hartley theorem demonstrates the maximum channel capacity in the presence of Gaussian white noise and was discussed in the **Modulation** chapter in the Channel Capacity section. This theorem describes how an increased symbol rate will require an increase in bandwidth and how a reduced signal-to-noise ratio (SNR) will reduce the potential *throughput* of the channel.

16.1.2 Error Detection and Correction

Voice modes require the operator to manually request a repeat of any information required but not understood. Using proper phonetics makes the information more easily understood but takes longer to transmit. If 100% accuracy is required, it may be necessary for the receiver to repeat the entire message back to the sender for verification. Computers can't necessarily distinguish between valuable and unnecessary data or identify likely errors but they offer other options to detect and correct errors.

ERROR DETECTION

The first requirement of any accurate system is to be able to detect when an error has occurred. The simplest method is *parity*. With 7-bit ASCII data, it was common to transmit an additional 8th parity bit to each character. The parity bit was added to make the total number of 1 bits odd or even. The binary representation for an ASCII letter Z is 1011010. Sent as seven bits with even parity, the parity bit would be 0 because there are already an even number of 1 bits and the result would be 01011010. The limitation of parity is that it only works with an odd number of bit inversions. If the last two bits were flipped to 01011001 (the ASCII letter Y), it would still pass the parity check because it still has an even number of bits. Parity is also rarely used on 8-bit data so it cannot be used when transferring binary data files.

Checksum is a method similar to the "check" value in an NTS message. It is generally a single byte (8-bit) value appended to the end of a packet or frame of data. It is calculated by adding all the values in the packet and taking the least significant (most unique) byte. This is a simple operation for even basic processors to perform quickly but can also be easily mislead. If two errors occur in the packet of equal amounts in the opposite direction (A becomes B and Z becomes Y), the checksum value will still be accurate and the packet will be accepted as error-free.

Cyclic redundancy check (*CRC)* is similar to checksum but uses a more sophisticated formula for calculating the check value of a packet. The formula most closely resembles long division, where the quotient is thrown away and the remainder is used. It is also common for CRC values to be more than a single byte, making the value more unique and likely to identify an error. Although other error detection systems are currently in use, CRC is the most common.

ERROR CORRECTION

There are two basic ways to design a protocol for an error correcting system: *automatic repeat request* (*ARQ*) and *forward error correction* (*FEC*). With ARQ the transmitter sends the data packet with an error detection code, which the receiver uses to check for errors. The receiver will request retransmission of packets with errors or sends an acknowledgement (ACK) of correctly received data, and the transmitter re-sends anything not acknowledged within a reasonable period of time.

With forward error correction (FEC), the transmitter encodes the data with an *error-correcting code* (*ECC*) and sends the encoded message. The receiver is not required to send any messages back to the transmitter. The receiver decodes what it receives into the "most likely" data. The codes are designed so that it would take an "unreasonable" amount of noise to trick the receiver into misinterpreting the data. It is possible to combine the two, so that minor errors are corrected without retransmission, and major errors are detected and a retransmission requested. The combination is called *hybrid automatic repeat request* (*hybrid ARQ*).

There are many error correcting code (ECC) algorithms available. Extended Golay coding is used on blocks of ALE data, for example, as described in the section below on G-TOR. In addition to the ability to detect and correct errors in the data packets, the modulation scheme allows sending multiple data streams and interleaving the data in such a way that a noise burst will disrupt the data at different points.

16.1.3 Data Representations

When comparing digital modes, it is important to understand how the data is conveyed. There are inherent limitations in any method chosen. PSK31 might seem a good choice for sending data over HF links because it performs well, but it was only designed for text (not 8-bit data) and has no inherent error correction. It is certainly possible to use this modulation scheme to send 8-bit data and add error correction to create a new mode. This would maintain the weak signal performance but the speed will suffer from the increased overhead. Similarly, a digital photo sent via analog SSTV software may only take two minutes to send, but over VHF packet it could take 10 minutes, despite the higher speed of a packet system. This doesn't mean SSTV is more efficient. Analog SSTV systems generally transmit lower resolution images with no error correction. Over good local links, the VHF packet system will be able to deliver perfect images faster or of higher quality.

TEXT REPRESENTATIONS

Morse code is well known as an early code used to send text data over a wire, then over the air. Each letter/number or symbol is represented with a varying length code with the more common letters having shorter codes. This early *varicode* system is very efficient and minimizes the number of state changes required to send a message.

The Baudot code (pronounced "bawd-OH") was invented by Émile Baudot and is the predecessor to the character set currently known more accurately as International Telegraph Alphabet No 2 (ITA2). This code is used for radioteletype communications and contains five bits with start and stop pulses. This only allows for 2^5 or 32 possible characters to be sent, which is not enough for all 26 letters plus numbers and characters. To resolve this, ITA2 uses a LTRS code to select a table of upper case (only) letters and a FIGS code to select a table of numbers, punctuation and special symbols. The code is defined in the ITA2 codes table on the CD included with this book.

Early computers used a wide variety of al-

phabetic codes until the early 1960s until the advent of the American Standard Code for Information Interchange or ASCII (pronounced "ESS-key"). At that point many computers standardized on this character set to allow simple transfer of data between machines. ASCII is a 7-bit code which allows for 2^7 or 128 characters. It was designed without the *control characters* used by Baudot for more reliable transmissions and the letters appear in English alphabetical order for easy sorting. The code can be reduced to only six bits and still carry numbers and uppercase letters. Current FCC regulations provide that amateur use of ASCII conform to ASCII as defined in ANSI standard X3.4-1977. The international counterparts are ISO 646-1983 and International Alphabet No. 5 (IA5) as published in ITU-T Recommendation V.3. A table of ASCII characters is presented as "ASCII Character Set" on the CD included with this book.

ASCII has been modified and initially expanded to eight bits, allowing the addition of foreign characters or line segments. The different extended versions were often referred to as *code pages*. The IBM PC supported code page 437 which offers line segments, and English *Windows* natively supports code page 1252 with additional foreign characters and symbols. All of these extended code pages include the same first 128 ASCII characters for backward compatibility. In the early 1990s efforts were made to support more languages directly and *Unicode* was created. Unicode generally requires 16 bits per character and can represent nearly any language.

More recent schemes use varicode, where the most common characters are given shorter codes (see **http://en.wikipedia.org/wiki/Prefix_code**). Varicode is used in PSK31and MFSK to reduce the number of bits in a message. Although the PSK31 varicode contains all 128 ASCII characters, lower-case letters contain fewer bits and can be sent more quickly. Tables of PSK and MFSK varicode characters are included on the CD included with this book.

IMAGE DATA REPRESENTATIONS

Images are generally broken into two basic types, *raster* and *vector*. Raster or *bitmap* images are simply rows of colored points known as *pixels* (picture elements). Vector images are a set of drawing primitives or instructions. These instructions define shapes, placement, size and color. Similar coding is used with plotters to command the pens to create the desired image.

Bitmap images can be stored at various *color depths* or bits per pixel indicating how many colors can be represented. Common color depths are 1-bit (2 colors), 4-bit (16 colors), 8-bit (256 colors), 16-bit (65,536 colors also called *high color*) and 24-bit (16 million also called *true color*). True color is most commonly used with digital cameras and conveniently provides eight bits of resolution each for the red, green and blue colors. Newer scanners and other systems will often generate 30-bit (2^{30} colors) and 36-bit (2^{36} colors). Images with 4- and 8-bit color can store more accurate images than might be obvious because they include a *palette* where each of their 16 or 256 colors respectively are chosen from a palette of 16 million. This palette is stored in the file which works well for simple images. The GIF format only supports 256 colors (8-bit) with a palette and lossless compression.

Digital photographs are raster images at a specific resolution and color depth. A typical low resolution digital camera image would be 640×480 pixels with 24-bit color. The 24-bit color indicated for each pixel requires three bytes to store (24 bits / 8 bits per byte) and can represent one of a possible 2^{24} or 16,777,216 colors with eight bits each for red, green and blue intensities. The raw (uncompressed) storage requirement for this image would be 640×480x3 or 921,600 bytes. This relatively low resolution image would require significant time to transmit over a slow link.

Vector images are generally created with drawing or CAD packages and can offer significant detail while remaining quite small. Because vector files are simply drawn at the desired resolution, there is no degradation of the image if the size is changed and the storage requirements remain the same at any resolution. Typical drawing primitives include lines and polylines, polygons, circles and ellipses, Bézier curves and text. Even computer font technologies such as TrueType create each letter from Bézier curves allowing for flawless scaling to any size and resolution on a screen or printer.

Raster images can be resized by dropping or adding pixels or changing the color depth. The 640×480×24-bit color image mentioned above could be reduced to 320×240×16-bit color with a raw size of 153,600 bytes — a significant saving over the 921,600 byte original. If the image is intended for screen display and doesn't require significant detail, that size may be appropriate. If the image is printed full sized on a typical 300 dpi (dots per inch) printer, each pixel in the photo will explode to nearly 100 dots on the printer and appear very blocky or pixilated.

Table 16.2
Typical Audio Formats

Audio Format	*Bits per Sample*	*Dynamic Range*	*Maximum Frequency*	*kbytes per Minute*
44.1 kHz stereo	16	98	22.050 kHz	10,584
22 kHz mono	16	98	11 kHz	2,640
8 kHz mono	8	50	4 kHz	960

AUDIO DATA REPRESENTATIONS

Like images, audio can be stored as a sampled waveform or in some type of primitive format. Storing a sampled waveform is the most versatile but can also require substantial storage capacity. MIDI (musical instrument digital interface, pronounced "MID-ee") is a common music format that stores instrument, note, tempo and intensity information as a musical score. There are also voice coding techniques that store speech as *allophones* (basic human speech sounds).

As with images, storage of primitives can save storage space (and transmission times) but they are not as rich as a high quality sampled waveform. Unfortunately, the 44,100 Hz 16-bit sample rate of an audio compact disc (CD) requires 176,400 bytes to store each second and 10,584,000 bytes for each minute of stereo audio.

The Nyquist-Shannon sampling theorem states that perfect reconstruction of a signal is possible when the sampling frequency is at least twice the maximum frequency of the signal being sampled. With its 44,100 Hz sample rate, CD audio is limited to a maximum frequency response of 22,050 Hz. If only voice-quality is desired, the sample rate can easily be dropped to 8000 Hz providing a maximum 4000 Hz frequency response. (See the **DSP and Software Radio Design** chapter for more information on sampling.)

The *bit depth* (number of bits used to represent each sample) of an audio signal will determine the theoretical dynamic range or signal-to-noise ratio (SNR). This is expressed with the formula SNR = (1.761 + 6.0206 × bits) dB. A dynamic range of 40 dB is adequate for the perception of human speech. **Table 16.2** compares the audio quality of various formats with the storage (or transmission) requirements.

VIDEO DATA REPRESENTATIONS

The most basic video format simply stores a series of images for playback. Video can place huge demand on storage and bandwidth because 30 frames per second is a common rate for smooth-appearing video. Rates as low as 15 frames per second can still be considered "full-motion" but will appear jerky and even this rate requires substantial storage. Primitive formats are less common for video

but animation systems like Adobe *Flash* are often found on the web.

16.1.4 Compression Techniques

There are a large variety of compression algorithms available to reduce the size of data stored or sent over a transmission medium. Many techniques are targeted at specific types of data (text, audio, images or video). These compression techniques can be broken into two major categories: *lossless compression* and *lossy compression*. Lossless algorithms are important for compressing data that must arrive perfectly intact but offer a smaller compression ratio than lossy techniques. Programs, documents, databases, spreadsheets would all be corrupted and made worthless if a lossy compression technique were used.

Images, audio and video are good candidates for lossy compression schemes with their substantially higher compression rates. As the name implies, a lossy compression scheme deliberately omits or simplifies that data to be able to represent it efficiently. The human eye and ear can easily interpolate missing information and it simply appears to be of lower quality.

Compression of a real-time stream of data such as audio or video is performed by software or firmware *codecs* (from *co*der-*dec*oder). Codecs provide the real-time encoding or decoding of the audio or video stream. Many codecs are proprietary and have licensing requirements. Codecs can be implemented as operating system or application plug-ins or even as digital ICs, as with the P25 IMBE and D-STAR AMBE codecs.

There are also specifically designed low bit-rate codecs for voice that are more accurately called *vocoders*. Vocoders are optimized for voice characteristics and can encode it extremely efficiently. Conversely, they cannot effectively process non-voice signals. This is easily demonstrated with a mobile phone by listening to music through a voice connection, such as when a call is placed on hold. Mobile phones use highly efficient vocoders to minimize the bandwidth of each voice channel which allows more channels per tower. This also means that non-voice sounds such as music or mechanical sounds are not well reproduced.

Good vocoders are extremely valuable and vigorously guarded by their patent holders. This has made it difficult for amateurs to experiment with digital voice techniques. David Rowe, VK5DGR, worked for several years on a project called CODEC2. This is a very effective open source vocoder made available at the end of 2012. It has been incorporated into the *FDMDV* software and there has been extensive testing and tuning made on the HF bands. More information about the project can be found at **http://codec2.org**. Digital voice is discussed elsewhere in this chapter.

LOSSLESS COMPRESSION TECHNIQUES

One of the earliest lossless compression schemes is known as *Huffman coding*. Huffman coding creates a tree of commonly used data values and gives the most common values a lower bit count. Varicode is based on this mechanism. In 1984, Terry Welch released code with improvements to a scheme from Abraham Lempel and Jacob Ziv, commonly referred to as *LZW* (Lempel-Ziv-Welch). The Lempel-Ziv algorithm and variants are the basis for most current compression programs and is used in the GIF and optionally TIFF graphic formats. *LZW* operates similarly to Huffman coding but with greater efficiency.

The actual amount of compression achieved will depend on how redundant the data is and the size of the data being compressed. Large files will achieve greater compression rates because the common data combinations will be seen more frequently. Simple text and documents can often see 25% compression rates. Spreadsheets and databases generally consist of many empty cells and can often achieve nearly 50% compression. Graphic and video compression will vary greatly depending on the complexity of the image. Simple images with solid backgrounds will compress well where complex images with little recurring data will see little benefit. Similarly, music will compress poorly but spoken audio with little background noise may see reasonable compression rates.

Run length encoding (*RLE*) is a very simple scheme supported on bitmap graphics (*Windows* BMP files). Each value in the file is a color value and "run length" specifying how many of the next pixels will be that color. It works well on simple files but can make a file larger if the image is too complex.

It is important to note that compressing a previously compressed file will often yield a larger file because it simply creates more overhead in the file. There is occasionally some minor benefit if two different compression algorithms are used. It is not possible to compress the same file repeatedly and expect any significant benefit. Modern compression software does offer the additional benefit of being able to compress groups of files or even whole directory structures into a single file for transmission.

The compression mechanisms mentioned above allow files to be compressed prior to transmission but there are also mechanisms that allow near real-time compression of the transmitted data. V.92 modems implement a LZJH adaptive compression scheme called V.44 that can average 15% better throughput over the wire. Winlink uses a compression scheme called B2F and sees an average 44% improvement in performance since most Winlink data is uncompressed previously. Many online services offer "Web Accelerators" that also compress data going over the wire to achieve better performance. There is a slight delay (latency) as a result of this compression but over a slow link, the additional latency is minimal compared to the performance gain.

LOSSY COMPRESSION SCHEMES

Lossy compression schemes depend on the human brain to "recover" or simply ignore the missing data. Since audio, image and video data have unique characteristics as perceived by the brain, each of these data types have unique compression algorithms. New compression schemes are developed constantly to achieve high compression rates while maintaining the highest quality. Often a particular file format actually supports multiple compression schemes or is available in different versions as better methods are developed. In-depth discussion of each of these algorithms is beyond the scope of this book but there are some important issues to consider when looking at these compression methods.

Lossy Audio Compression

Audio compression is based on the psychoacoustic model that describes which parts of a digital audio signal can be removed or substantially reduced without affecting the perceived quality of the sound. Lossy audio compression schemes can typically reduce the size of the file 10- or 12-fold with little loss in quality. The most common formats currently are MP3, WMA, AAC, Ogg Vorbis and ATRAC.

Lossy Image Compression

The Joint Photographic Experts Group developed the JPEG format (pronounced "JAY-peg") in 1992 and it has become the primary format used for lossy compression of digital images. It is a scalable compression scheme allowing the user to determine the level of compression/quality of the image. Compression rates of 10-fold are common with good quality and can be over 100-fold at substantially reduced quality. The JPEG format tends to enhance edges and substantially compress fields of similar color. It is not well suited when multiple edits will be required because each copy will have generational loss and therefore reduced quality.

Lossy Video Compression

Because of the massive amount of data required for video compression it is almost always distributed with a lossy compression scheme. Lossless compression is only used

when editing to eliminate generational loss. Video support on a computer is generally implemented with a codec that allows encoding and decoding the video stream. Video files are containers that can often support more than one video format and the specific format information is contained in the file. When the file is opened, this format information is read and the appropriate codec is activated and fed into the data stream to be decoded.

The most common current codecs are H.261 (video conferencing), MPEG-1 (used in video CDs), MPEG-2 (DVD Video), H.263 (video conferencing), MPEG-4, DivX, Xvid, H.264, Sorenson 3 (used by *QuickTime*), WMV (Microsoft), VC-1, RealVideo (RealNetworks) and Cinepak.

The basic mechanism of video compression is to encode a high quality "key-frame" that could be a JPEG image as a starting image. Successive video frames or "inter-

FLDIGI

By Ken Humbertson, WØKAH, and Jeff Coval, ACØSC

You may have heard of the free digital mode software *fldigi* by David H. Freese Jr., W1HKJ. It is very popular for use in emergency communications, for example, because of its multimode capabilities and its ability to work with a companion program *flmsg* that generates standard message forms to be transmitted using *fldigi*. The software supports more than 34 modes (as of early 2013) as well as variations of tones, bandwidth, baud rates, number of bits, and other variations for many modes. Versions of *fldigi* are available for *Linux*/*Free-BSD*, *Windows* XP/W2K/NT/Vista/Win7, and OS X from W1HKJ's download page at **www.w1hkj.com/download.html**.

If you have a computer with a sound card and microphone, you can begin using *fldigi* to receive with no additional hardware. A quick example would be to tune to 14.070 MHz or 21.070 MHz USB during the day. Set the radio speaker volume to a normal listening level. Start *fldigi* on the computer with sound card and microphone connected and you should see a waterfall display similar to **Fig 16.A** (left).

If your rig has computer control capability, *fldigi* can likely interface with it to display frequency and mode, as well as control the radio from the program. In Fig 16.A, an ICOM IC-7000 is controlled by *fldigi* and thus shows the current operating frequency and mode of 21.070 MHz USB, the QSO frequency of 21071.511 (kHz) (in the windows at the upper left corner), and the current mode of BPSK31 (lower left corner).

When you see multiple signals in the waterfall, *fldigi* will decode whichever one you choose when you place the mouse cursor over a signal of interest and line up the two vertical red lines on your screen with the sides of the signal, as shown near 1500 Hz. When you change modes or bandwidths within a mode, the spacing of the red lines will adjust to the new bandwidth. **Fig 16.B** (right) shows that by selecting RTTY-45 under the OP MODE tab, you will notice that the red line spacing increases to match the familiar RTTY tone shift of 170 Hz. Simply place the two red lines over the signal you wish to decode and the program does the rest.

Fig 16.B shows a decoded ARRL digital bulletin from January 2, 2013. While the examples show BPSK31 and RTTY, there are many operating modes to choose from, including CW.

Actions in *fldigi* are performed by action buttons that invoke *macros* — text scripts that control the program. For example, to call CQ use the mouse to click on an unoccupied spot in the waterfall display. This shifts the modulating tones of the signal to that offset within your receive bandwidth. Click the CQ action button on the *fldigi* display. The transmitted text is displayed in red above the waterfall display. When the text is finished, *fldigi* will return to receive mode. The tutorial "Beginner's Guide to Fldigi" at **www.w1hkj.com/beginners.html** is recommended and the program has an extensive Help file.

The program can be used with an internal sound card or external sound card adapter such as the Tigertronics Signalink USB (**www.tigertronics.com**) or West Mountain Radio Rigblaster series (**www.westmountainradio.com**). Setting up a sound card to use *fldigi* may require manipulation of the audio device configuration for your computer's operating system. Follow the instructions in the *fldigi* manuals and the manufacturer's manuals if you are using an external adapter. A digital data interface that allows you to connect the sound card to your rig's microphone input is recommended. (See the **Assembling a Station** chapter and the **Digital Communications** supplement on the CD-ROM.)

Fldigi is frequently updated to include new modes that are being developed. The program is a user-friendly way to become active on the digital modes, a rapidly expanding aspect of Amateur Radio.

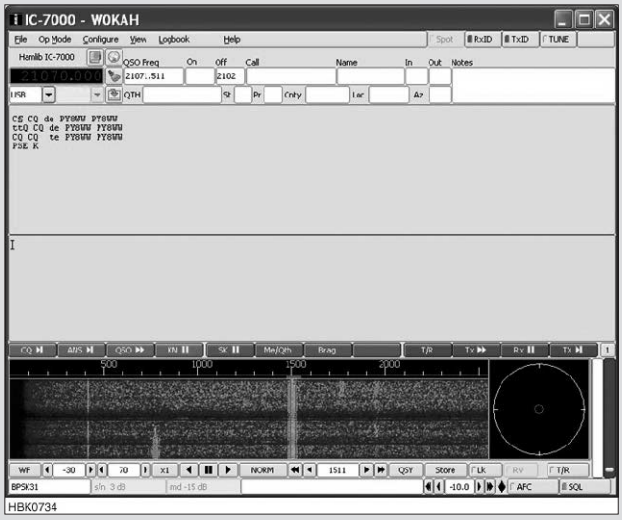

Fig 16.A

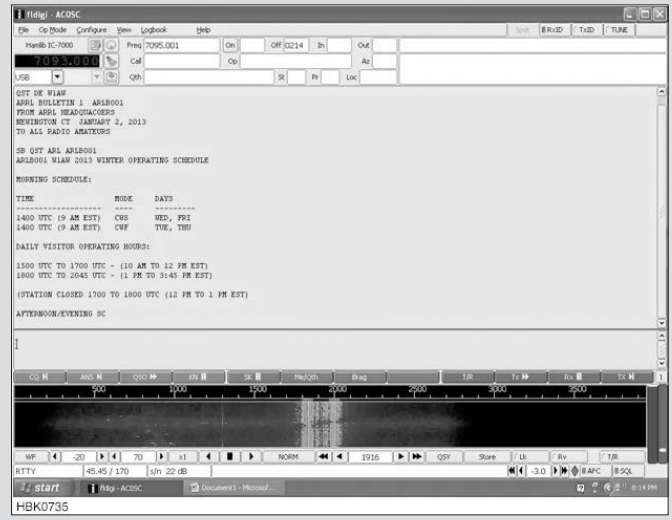

Fig 16.B

frames" contain only changes to the previous frame. After some number of frames, it is necessary to use another key-frame and start over with inter-frames. The resolution of the images, frame rate and compression quality determine the size of the video file.

BIT RATE COMPARISON

Table 16.3 provides an indication of minimum bit rates required to transmit audio and video such that the average listener would not perceive them significantly worse than the standard shown.

Table 16.3
Audio and Video Bit Rates

Audio	
8 kbit/s	Telephone quality audio (using speech codecs)
32 kbit/s	AM broadcast quality (using MP3)
96 kbit/s	FM broadcast quality (using MP3)
224-320 kbit/s	Near-CD quality, indistinguishable to the average listener (using MP3)
Video	
16 kbit/s	Videophone quality (minimum for "talking head")
128-384 kbit/s	Business videoconference system quality
1.25 Mbit/s	VCD (video compact disk) quality
5 Mbit/s	DVD quality
10.5 Mbit/s	Actual DVD video bit rate

16.1.5 Compression vs. Encryption

There is some confusion about compression being a form of encryption. It is true that a text file after compression can no longer be read unless uncompressed with the appropriate algorithm. In the United States the FCC defines encryption in part 97.113 as "messages encoded for the purpose of obscuring their meaning." Compressing a file with ZIP or RAR (common file compression methods) and transmitting it over the air is simply an efficient use of spectrum and time and is not intended to "obscure its meaning."

As amateur digital modes interact more with Internet-based services, the issue arises because many of these services utilize encryption of various types. Banks and other retailers may encrypt their entire transactions to insure confidentiality of personal data. Other systems as benign as e-mail may simply encrypt passwords to properly authenticate users. The FCC has offered no additional guidance on these issues.

16.2 Unstructured Digital Modes

The first group of modes we'll examine are generally considered "sound card modes" for keyboard-to-keyboard communications. Because each of these modes is optimized for a specific purpose by blending multiple features, they often defy simple categorization.

16.2.1 Radioteletype (RTTY)

Radioteletype (*RTTY*) consists of a *frequency shift keyed* (*FSK*) signal that is modulated between two carrier frequencies, called the *mark* frequency and the *space* frequency. The protocol for amateur RTTY calls for the mark carrier frequency to be the higher of the two in the RF spectrum. The difference between the mark and space frequencies is called the *FSK shift*, usually 170 Hz for an amateur RTTY signal.

At the conventional data speed of 60 WPM, binary information modulates the FSK signal at 22 ms per bit, or equivalent to 45.45 baud. Characters are encoded into binary 5-bit Baudot coded data. Each character is individually synchronized by adding a start bit before the 5-bit code and by appending the code with a stop bit. The start bit has the same duration as the data bits, but the stop bit can be anywhere between 22 to 44 ms in duration. The stop bit is transmitted as a mark carrier, and the RTTY signal "rests" at this state until a new character comes along. If the number of stop bits is set to two, the RTTY signal will send a minimum of 44 ms of mark carrier before the next start bit is sent. A start bit is sent as a space carrier. A zero in the Baudot code is sent as a space signal and a one is sent as a mark signal. **Fig 16.1** shows the character D sent by RTTY.

BAUDOT CODE

The Baudot code (see the *Handbook* CD) is a 5-bit code; thus, it is capable of encoding only 32 unique characters. Since the combination of alphabets, decimal numbers and common punctuations exceeds 32, the Baudot code is created as two sets of tables. One table is called the LTRS Shift, and consists mainly of alphabetic characters. The second table is called the FIGS Shift, and consists mainly of decimal numerals and punctuation marks. Two unique Baudot characters called LTRS and FIGS are used by the sender to command the decoder to switch between these two tables.

Mechanical teletypewriter keyboards have two keys to send the LTRS and FIGS characters. The two keys behave much like the caps lock key on a modern typewriter keyboard. Instead of locking the keyboard of a typewriter to upper case shift or lower case shift, the two teletypewriter keys lock the state of the teletypewriter into the LTRS table or the FIGS table. LTRS and FIGS, among some other characters such as the space character, appear in both LTRS and FIGS tables so that you can send LTRS, FIGS and shift no matter which table the encoder is using.

To send the letter Q, you need to first make sure that the decoder is currently using the LTRS table, and then send the Baudot codeword for Q, 17 (hexadecimal or "hex" value). If the same hex 17 is received when the decoder is in the FIGS shift, the number 1 will be decoded instead of the Q. Modern software does away with the need for the operator to manually send the LTRS and FIGS codes.

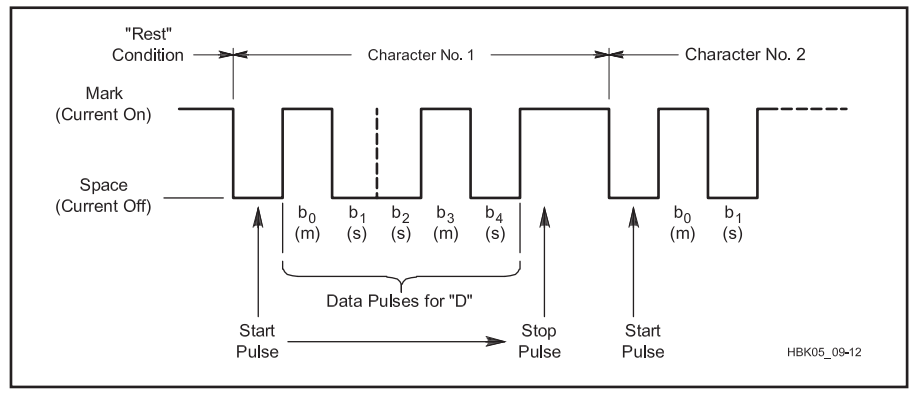

Fig 16.1 — The character "D" sent via RTTY.

When the operator sends a character from the LTRS table, the software first checks to make sure that the previous character had also used a character from the LTRS table; if not, the software will first send a LTRS character.

Noise can often cause the LTRS or FIGS character to be incorrectly received. This will cause subsequent characters to decode into the wrong Baudot character, until a correct LTRS or FIGS is received (see also USOS below). Instead of asking that the message be repeated by the sender, a trick that many RTTY operators use is to observe that on a standard QWERTY keyboard, the Q key is below the 1 key in the row above it, the W key is below the 2 key, and so on. Q and 1 happen to share the same Baudot code; W and 2 share the same Baudot code, and so forth. Given this visual aid, a printed UE can easily be interpreted by context to 73 and TOO interpreted as 599.

If the sender uses one stop bit, an RTTY character consists of a total of seven bits after adding the start and stop bits. If the sender uses 1.5 stop bits, each RTTY character has a total length of 7.5 bits. The least significant bit of the Baudot code follows the start bit of a character.

INVERTED SIGNALS

There are times when the sender does not comply with the RTTY standard and reverses the mark and space order in the spectrum. This is often called an "inverted" signal or "reverse shift." Most RTTY modulators and demodulators have a provision to reverse the shift of an inverted signal.

DEMODULATION AND DECODING

The most common way to decode an RTTY signal is to use a single sideband (SSB) receiver to first translate the two FSK carriers into two audio tones. If the carriers are 170 Hz apart, the two audio tones (called the *tone pair*) will also be 170 Hz apart. The RTTY demodulator, in the form of a terminal unit (TU) or a software modem (modulator-demodulator), then works to discriminate between the two audio tones. Some packet TNCs (terminal node controllers) can be made to function as RTTY demodulators, but often they do not work as well under poor signal-to-noise conditions because their filters are matched to packet radio FSK shifts and baud rate instead of to RTTY shift and baud rate.

As long as the tone pair separation is 170 Hz, the frequencies of the two audio tones can be quite arbitrary. Many TUs and modems are constructed to handle a range of tone pairs. If reception uses a lower sideband (LSB) receiver, the higher mark carrier will become the lower of the two audio tones. If upper sideband (USB) is used, the mark carrier will remain the higher of the tone pair. It is common to use 2125 Hz as the mark tone and 2295 Hz as a space tone. Since the mark tone (2125 Hz) is lower in frequency, the receiver will be set to LSB. In general, modem tone pairs can be "reversed" (see the Inverted Signals section), so either an LSB or a USB receiver can be used. Moreover, the tone pairs of many modems, especially software modems, can be moved to place them where narrowband filters are available on the receiver.

In the past, audio FSK demodulators were built using high-Q audio filters followed by a "slicer" to determine if the signal from the mark filter is stronger or weaker than the signal that comes out of the space filter. To counter selective fading, where ionospheric propagation can cause the mark carrier to become stronger or weaker than the space carrier, the slicer's threshold can be designed to adapt to the imbalance. Once "sliced" into a bi-level signal, the binary stream is passed to the decoder where start bit detection circuitry determines where to extract the 5-bit character data. That data is then passed to the Baudot decoder, which uses the current LTRS or FIGS state to determine the decoded character. The mark and space transmitted carriers do not overlap, although this can occur after they pass through certain HF propagation conditions. Sophisticated demodulators can account for this distortion.

A software modem performs the same functions as a hardware terminal unit, except that the software modems can apply more sophisticated mathematics that would be too expensive to implement in hardware. Modern desktop computers have more than enough processing speed to implement an RTTY demodulator. Software modems first convert the audio signal into a sequence of binary numbers using an analog-to-digital converter which is usually part of an audio chip set either on the motherboard or sound card. Everything from that point on uses numerical algorithms to implement the demodulation processes.

FSK VS AFSK MODULATION

An RTTY signal is usually generated as an F1B or F2B emission. F1B is implemented by directly shifting an RF carrier between the two (mark and space) frequencies. This method of generating FSK is often called *direct FSK*, or *true FSK*, or simply FSK. F2B is implemented by shifting between two audio tones, instead of two RF carriers. The resultant audio is then sent to an SSB transmitter to become two RF carriers. This method of first generating an audio FSK signal and then modulating an SSB transmitter to achieve the same FSK spectrum is usually called AFSK (audio frequency shift keying).

AFSK can be generated by using either an upper sideband (USB) transmitter or a lower sideband (LSB) transmitter. With a USB transmitter, the mark tone must be the *higher* of the two audio tones in the audio FSK signal. The USB modulator will then place the corresponding mark carrier at the higher of the two FSK carrier frequencies. When LSB transmission is used, the mark tone must be the *lower* of the two audio tones in the audio FSK signal. The LSB modulator will then place the corresponding mark carrier at the higher of the two FSK carrier frequencies. As when receiving, the actual audio tones are of no importance. The important part of AFSK is to have the two audio tones separated by 170 Hz, and have the pair properly flipped to match the choice of USB or LSB transmission. (See the **Modulation** chapter for more information on USB and LSB modulation and the relationship between modulating frequency and transmitted signal frequency.)

When using AFSK with older transceivers, it is wise to choose a high tone pair so that harmonic by-products fall outside the passband of the transmitter. Because of this, a popular tone pair is 2125 Hz/2295 Hz. Most transceivers will pass both tones and also have good suppression of the harmonics of the two tones. Not all transceivers that have an FSK input are FSK transmitters. Some transceivers will take the FSK keying input and modulate an internal AFSK generator, which is then used to modulate an SSB transmitter. In this case, the transmitter is really operating as an F2B emitter. This mode of operation is often called "keyed AFSK."

"SPOTTING" AN RTTY SIGNAL

By convention, RTTY signals are identified by the frequency of the mark carrier on the RF spectrum. "Spotting" the suppressed carrier frequency dial of an SSB receiver is useless for someone else unless they also know whether the spotter is using upper or lower sideband and what tone pair the spotter's demodulator is using. The mark and space carriers are the only two constants, so the amateur RTTY standard is to spot the frequency of the mark carrier.

DIDDLE CHARACTERS

In between the stop bit of a preceding character and the start bit of the next character, the RTTY signal stays at the mark frequency. When the RTTY decoder is in this "rest" state, a mark-to-space transition tells a decoder that the start of a new character has arrived. Noise that causes a start bit to be misidentified can cause the RTTY decoder to fall out of sync. After losing sync, the decoder will use subsequent data bits to help it identify the location of the next potential start bit.

Since all mark-to-space transitions are potential locations of the leading edge of a start bit, this can cause multiple characters to be incorrectly decoded until proper synchronization is again achieved. This "character slippage" can be minimized somewhat by not allowing the RTTY signal to rest for longer

than its stop bit duration. An idle or *diddle character* (so called because of the sound of the demodulated audio from an idle RTTY signal sending the idle characters) is inserted immediately after a stop bit when the operator is not actively typing. The idle character is a non-printing character from the Baudot set and most often the LTRS character is used. Baudot encodes a LTRS as five bits of all ones making it particularly useful when the decoder is recovering from a misidentified start bit.

An RTTY diddle is also useful when there is selective fading. Good RTTY demodulators counter selective fading by measuring the amplitudes of the mark and space signals and automatically adjusting the decoding threshold when making the decision of whether a mark or a space is being received. If a station does not transmit diddles and has been idle for a period of time, the receiver will have no idea if selective fading has affected the space frequency. By transmitting a diddle, the RTTY demodulator is ensured of a measurement of the strength of the space carrier during each character period.

UNSHIFT-ON-SPACE (USOS)

Since the Baudot code aliases characters (for example, Q is encoded to the same 5-bit code as 1) using the LTRS and FIGS Baudot shift to steer the decoder, decoding could turn into gibberish if the Baudot shift characters are altered by noise. For this reason, many amateurs use a protocol called *unshift-on-space* (USOS). Under this protocol, both the sender and the receiver agree that the Baudot character set is always shifted to the LTRS state after a space character is received. In a stream of text that includes space characters, this provides additional, implicit, Baudot shifts.

Not everyone uses USOS. When used with messages that have mostly numbers and spaces, the use of USOS causes extra FIGS characters to be sent. A decoder that complies with USOS will not properly decode an RTTY stream that does not have USOS set. Likewise, a decoder that has USOS turned off will not properly decode an RTTY stream that has USOS turned on.

OTHER FSK SHIFTS AND RTTY BAUD RATE

The most commonly used FSK shift in amateur RTTY is 170 Hz. However, on rare occasions stations can be found using 425 and 850 Hz shifts. The wider FSK shifts are especially useful in the presence of selective fading since they provide better frequency diversity than 170 Hz.

Because HF packet radio uses 200 Hz shifts, some TNCs use 200 Hz as the FSK shift for RTTY. Although they are mostly compatible with the 170 Hz shift protocol, under poor signal to noise ratio conditions these demodulators will produce more error hits than a demodulator that is designed for 170 Hz shift. Likewise, a signal that is transmitted using 170 Hz shift will not be optimally copied by a demodulator that is designed for a 200 Hz shift.

To conserve spectrum space, amateurs have experimented with narrower FSK shifts, down to 22.5 Hz. At 22.5 Hz, optimal demodulators are designed as *minimal shift keyed* (*MSK*) instead of frequency shift keyed (FSK) demodulators.

SOME PRACTICAL CHARACTERISTICS

When the demodulator is properly implemented, RTTY can be very resilient against certain HF fading conditions, namely when selective fading causes only one of the two FSK carriers to fade while the other carrier remains strong. However, RTTY is still susceptible to "flat fading" (where the mark and space channels both fade at the same instant). There is neither an error correction scheme nor an interleaver (a method of rearranging — interleaving — the distribution of bits to make errors easier to correct) that can make an RTTY decoder print through a flat and deep fade. The lack of a data interleaver, however, also makes RTTY a very interactive mode. There is practically no latency when compared to a mode such as MFSK16, where the interleaver causes a latency of over 120 bit durations before incoming data can even be decoded. This makes RTTY attractive to operating styles that have short exchanges with rapid turnarounds, such as in contests.

Although RTTY is not as "sensitive" as PSK31 (when there is no multipath, PSK31 has a lower error rate than RTTY when the same amount of power is used) it is not affected by phase distortion that can render even a strong PSK31 signal from being copied. When HF propagation conditions deteriorate, RTTY can often function as long as sufficient power is used. Tuning is also moderately uncritical with RTTY. When the signal-to-noise ratio is good, RTTY tuning can be off by 50 Hz and still print well.

16.2.2 PSK31

PSK31 is a family of modes that uses differentially encoded varicode (see the next section), envelope-shaped phase shift keying. BPSK31, or binary PSK31, operates at 31.25 bit/s (one bit every 32 ms). QPSK31, or quadrature PSK31, operates at 31.25 baud. Each symbol consists of four possible quadrature phase change (or *dibits*) at the signaling rate of one dibit every 32 ms. QPSK31 sends a phase change symbol of 0°, 90°, 180° or 270° every 32 ms.

Characters that are typed from the keyboard are first encoded into variable-length varicode binary digits. With BPSK31, the varicode bits directly modulate the PSK31 modulator, causing a 180° phase change if the varicode bit is a 0 and keeping a constant phase if the varicode bit is a 1. With QPSK31, the varicode bits first pass through two convolution encoders to create a sequence of bit pairs (dibits). Each dibit is then used to shift the QPSK31 modulator into one of four different phase changes.

PSK63 is a double-clock-rate version of a PSK31 signal, operating at the rate of one symbol every 16 ms (62.5 symbols per second). PSK125 is a PSK31 signal clocked at four times the rate, with one symbol every 8 ms (125 symbols per second). Although PSK63 and PSK125 are both in use, including the binary and quadrature forms, the most popular PSK31 variant remains BPSK31.

Most implementations of PSK31 first generate an audio PSK31 signal. The audio signal is then used to modulate an SSB transmitter. Since BPSK31 is based upon phase reversals, it can be used with either upper sideband (USB) or lower sideband (LSB) systems. With QPSK31 however, the 90° and 270° phase shifts have to be swapped when using LSB transmitters or receivers.

PSK31 VARICODE

Characters that are sent in PSK31 are encoded as varicode, a variable length prefix code as varicode. As described earlier, characters that occur more frequently in English text, such as spaces and the lower-case e, are encoded into a fewer number of bits than characters that are less frequent in English, such as the character q and the upper case E.

PSK31 varicode characters always end with two bits of zeros. A space character is sent as a one followed by the two zeros (100); a lower case e is sent as two ones, again terminated by the two bits of zero (1100). None of the varicode code words contain two consecutive zero bits. Because of this, the PSK31 decoder can uniquely identify the boundary between characters. A special "character" in PSK31 is the idle code, which consists of nothing but the two prefix bits. A long pause at the keyboard is encoded into a string of even numbers of zeros. The start of a new character is signaled by the first non-zero bit received after at least two consecutive zeros.

CONVOLUTION CODE

As described earlier, QPSK31 encodes the varicode stream with two convolution encoders to form the dibits that are used to modulate the QPSK31 generator. Both convolution encoders are fourth-order polynomials, shown in **Fig 16.2**.

The varicode data is first inverted before the bits are given to the convolution encoders. The first polynomial generates the most significant bit and the second polynomial generates the least significant bit of the dibit. Since we are

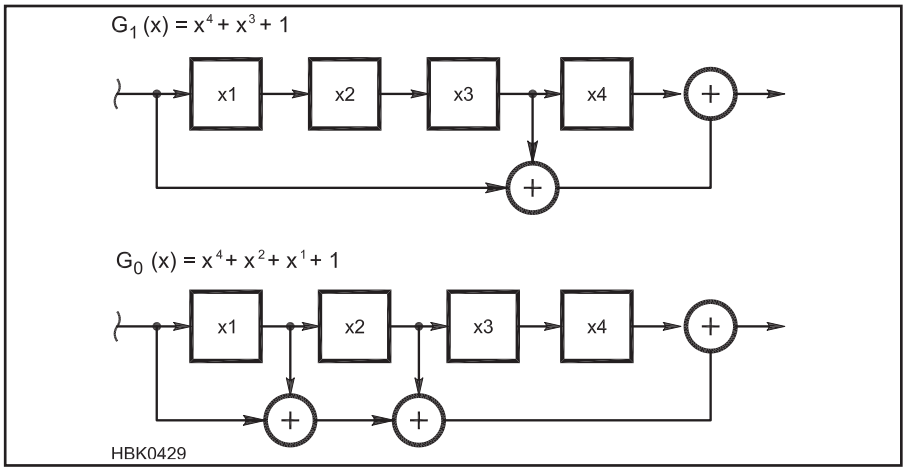

Fig 16.2 — QPSK31 convolution encoders.

working with binary numbers, the GF(2) sums shown in the above figure are exclusive-OR functions and the delay elements x1, x2, x3 and x4 are stages of binary shift registers. As each inverted varicode bit is available, it is clocked into the shift register and the two bits of the dibit are computed from the shift register taps.

MODULATION

PSK31 uses both differential phase shift modulation and envelope modulation to maintain its narrow-band characteristics. The most common way to generate an envelope-shaped PSK31 signal is to start with baseband in-phase (I) and quadrature (Q) signals. Both I and Q signals settle at either a value of +1 or a value of –1. When no phase transition is needed between symbols, the I and Q signals remain constant (at their original +1 or –1 values). See the **Modulation** chapter for information on creating I and Q signals.

To encode a 180° transition between symbols, both I and Q signals are slewed with a cosinusoidal envelope. If the in-phase signal had a value of +1 during the previous symbol, it is slewed to –1 cosinusoidally. If the in-phase signal had a value of –1 in the previous symbols, it is slewed cosinusoidally to +1. The quadrature signal behaves in a likewise manner. To encode a 90° phase shift in QPSK31, only the in-phase signal is slewed between the two symbols, the quadrature signal remains constant between the two symbols. To encode a 270° phase shift in QPSK31, only the quadrature signal is slewed between the two symbols; the in-phase signal remains constant between the two symbols.

The envelope of the real signal remains constant if there is no phase change. When the signal makes a 180° phase transition, the amplitude of a PSK31 signal will drop to zero in between the two symbols. The actual phase reversal occurs when the signal amplitude is zero, when there is no signal energy. During the 90° and 270° phase shifts between symbols of QPSK31, the amplitude of the signal does not reach zero. It dips to only half the peak power in between the two symbols.

To provide a changing envelope when the operator is idle at the keyboard, a zero in the varicode (remember that an idle varicode consists of two 0 bits) is encoded as a 180° phase change between two BPSK31 symbols. A 1 in the varicode is encoded as no phase change from one BPSK31 symbol to the next symbol. This changing envelope allows the receiver to extract bit timing information even when the sender is idle. Bit clock recovery is implemented by using a comb filter on the envelope of the PSK31 signal.

The convolution code that is used by QPSK31 converts a constant idle stream of zeros into a stream of repeated 10 dibits. To produce the same constant phase change during idle periods as BPSK31, the QPSK31 10 dibit is chosen to represent the 180° phase shift modulating term. **Table 16.4** shows the QPSK31 modulation.

To produce bit clocks during idle, combined with the particular convolution code that was chosen for QPSK31, results in a slightly sub-optimal (non-Gray code) encoding of the four dibits. At the end of a transmission, PSK31 stops all modulation for a short period and transmits a short unmodulated carrier. The intended function is as a squelch mechanism.

DEMODULATION AND DECODING

Many techniques are available to decode differential PSK, where a reference phase is not present. Okunev has a good presentation of the methods (reference: Yuri Okunev, *Phase and Phase-difference Modulation in Digital Communications* (1997, Artech House), ISBN 0-89006-937-9).

As mentioned earlier, there is sufficient amplitude information to extract the bit clock from a PSK31 signal. The output of a differential-phase demodulator is an estimate of the phase angle difference between the centers

Table 16.4
QPSK31 Modulation

Dibit	*Phase Change*
00	0°
01	90°
10	180°
11	270°

of one symbol and the previous one. With BPSK31, the output can be compared to a threshold to determine if a phase reversal or a non-reversal is more likely. The decoder then looks for two phase reversals in a row, followed by a non-reversal, to determine the beginning of a new character. The bits are gathered until two phase reversals are again seen and the accumulated bits are decoded into one of the characters in the varicode table.

QPSK31 decoding is more involved. As in the BPSK31 case, the phase difference demodulator estimates the phase change from one bit to another. However, one cannot simply invert the convolution function to derive the data dibits. Various techniques exist to decode the measured phase angles into dibits. The *Viterbi algorithm* is a relatively simple algorithm for the convolution polynomials used in QPSK31. The estimated phase angles can first be fixed to one of the four quadrature angles before the angles are submitted to the Viterbi algorithm. This is called a hard-decision *Viterbi decoder*.

A soft-decision Viterbi decoder (that is not that much more complex to construct) usually gives better results . The soft-decision decoder uses arbitrary phase angles and some measure of how "far" an angle is away from one of the four quadrature angles. Error correction occurs within the trellis that implements the Viterbi algorithm. References to convolution code, trellis and the Viterbi algorithm can be found at **http://en.wikipedia.org/wiki/Convolutional_code**.

16.2.3 MFSK16

MFSK16 uses M-ary FSK modulation with 16 "tones" (also known as 16-FSK modulation), where only a single tone is present at any instant. MFSK16 has a crest factor of 1, with no wave shaping performed on the data bits. The tone centers of MFSK16 are separated by 15.625 Hz. Data switches at a rate of 15.625 baud (one symbol change every 64 ms).

Characters are first encoded into the MFSK16 varicode, creating a constant bit stream whose rate is one bit every 32 ms. This results in a similar, although not identical, character rate as PSK31. The difference is due to the different varicode tables that are used by the two modes.

This bit stream is clocked into a pair of 6th-order convolution encoders. The result

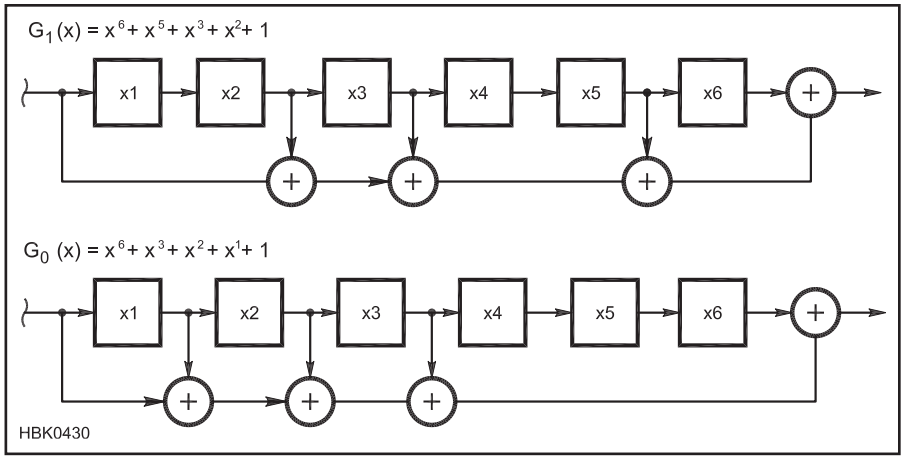

Fig 16.3 — MFSK16 convolution encoders.

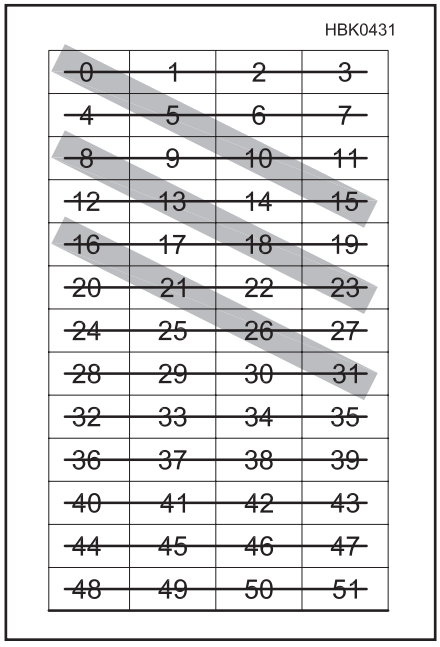

Fig 16.4 — IZ8BLY interleaver.

is a pair of bits (dibit) for each varicode bit, at a rate of one dibit every 32 ms. Consecutive pairs of dibits are next combined into sets of four bits (*quadbit* or *nibble*). Each nibble, at the rate of one per 64 ms, is then passed through an interleaver. The interleaver produces one nibble for each nibble that is sent to it. Each nibble from the interleaver is then Gray-coded and the resultant 4-bit Gray code is the ordered index of each 16 tones that are separated by 15.625 Hz. The result is a 16-FSK signal with a rate of one symbol per 64 ms. Since the symbol time is the reciprocal of the tone separation, a 16-point fast Fourier transform (FFT) can be conveniently used as an MFSK16 modulator.

MFSK16 VARICODE

Although the varicode table that is used by MFSK16 (see the *Handbook* CD) is not the same as the one used by PSK31, they share similar characteristics. Please refer to the PSK31 section of this chapter for a more detailed description of varicode. Unlike PSK31 varicode, MSK16 varicode encodings can contain two or more consecutive zero bits, as long as the consecutive zeros are at the tail of a code word. Character boundaries are determined when two or more consecutive zeros are followed by a one.

CONVOLUTION CODE

As described earlier, MFSK16 encodes the varicode stream with two convolution encoders to form the dibits that are passed on to the interleaver. Both convolution encoders are sixth order polynomial, shown in **Fig 16.3**.

The first polynomial generates the most significant bit and the second polynomial generates the least significant bit of the dibit. Since we are working with binary numbers, the GF(2) sums shown in the above figure are exclusive OR functions and the delay elements x1, x2, x3, x4, x5 and x6 are stages of binary shift registers. As each varicode bit is available, it is clocked into the shift register and the two bits of the dibit are computed from the shift register taps.

INTERLEAVER

HF fading channels tend to generate "burst" errors that are clumped together. An interleaver is used to permute the errors temporally so that they appear as uncorrelated errors to the convolution decoder. MFSK16 uses 10 concatenated stages of an IZ8BLY Diagonal Interleaver. More information on the interleaver can be found at **www.qsl.net/zl1bpu/MFSK/Interleaver.htm**. **Fig 16.4** illustrates how a single IZ8BLY interleaver spreads a sequence of bits.

The bits enter the IZ8BLY interleaver in the order 0, 1, 2, 3, 4,... and are passed to the output in the order 0, 5, 10, 15, 8, 13, ... (shown in the diagonal boxes). In MFSK16, the output of one interleaver is sent to the input of a second interleaver, for a total of 10 such stages. Each stage spreads the bits out over a longer time frame.

This concatenated 10-stage interpolator

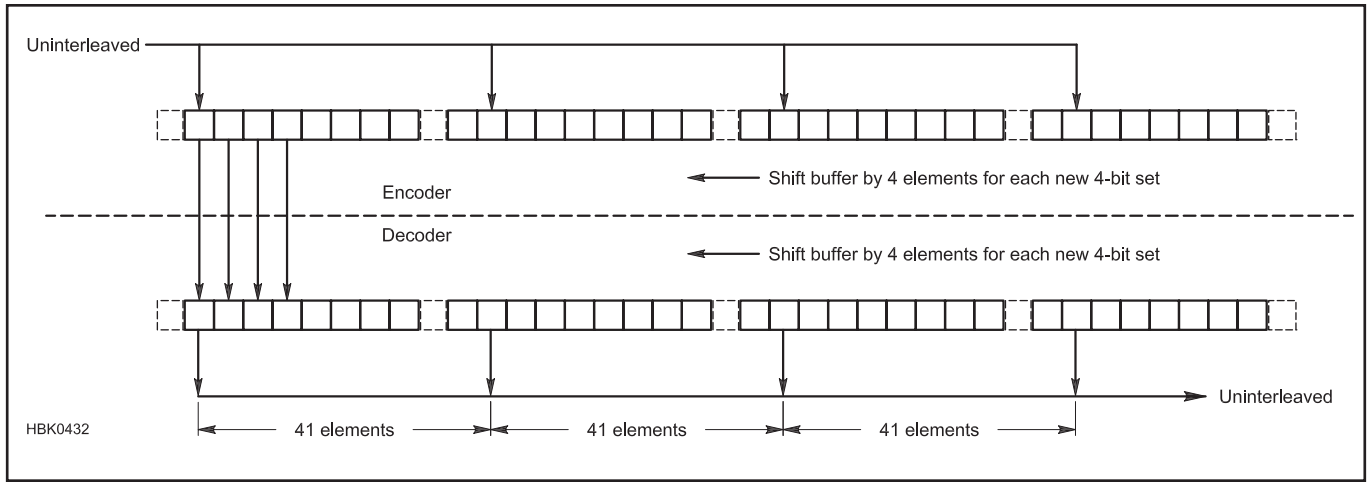

Fig 16.5 — Bit spreading through interpolation.

is equivalent to a single interpolator that is 123 bits long. **Fig 16.5** shows the structure of the single interpolator and demonstrates how four consecutive input bits are spread evenly over 123 time periods.

An error burst can be seen to be spread over a duration of almost two seconds. This gives MFSK16 the capability to correct errors over deep and long fades. While it is good for correcting errors, the delay through the long interleaver also causes a long decoding latency.

GRAY CODE

The Gray code creates a condition where tones that are next to one another are also different by a smaller *Hamming* distance. This optimizes the error correction process at the receiver. References to Gray code and Hamming distance can be found at **http://en.wikipedia.org/wiki/Gray_code** and **http://en.wikipedia.org/wiki/Hamming_distance** and **http://en.wikipedia.org/wiki/Hamming_distance**.

DEMODULATION AND DECODING

A 16-point FFT can be used to implement a set of matched filters for demodulating an MFSK16 signal once the input waveform is properly time aligned so that each transform is performed on an integral symbol and the signal is tuned so that the MFSK16 tones are perfectly centered in the FFT bins. A reference to a matched filter can be found at **http://en.wikipedia.org/wiki/Matched_filter**.

The 16 output bins from an FFT demodulator can first be converted to the "best" 4-bit index of an FFT frequency bin, or they can be converted to a vector of four numerical values representing four "soft" bits. The four bits are then passed through an MFSK16 de-interleaver. In the case of "soft decoding," the de-interleaver would contain numerical values rather than a 0 or 1 bit.

The output of the de-interleaver is passed into a convolution decoder. The Gray code makes sure that adjacent FFT bins also have the lowest Hamming distance; i.e., the most likely error is also associated with the closest FFT bin. The hard or the soft Viterbi Algorithm can be used to decode and correct errors.

16.2.4 DominoEX

DominoEX is a digital mode with MFSK (multi-frequency shift keying) designed for simplex chats on HF by Murray Greenman, ZL1BPU. It was designed to be easier to use and tune than other similar modes, offer low latency for contesting or other quick exchange situations, offer reliable copy down into the noise floor, and work well as an NVIS (near-vertical incidence skywave, see the **Propagation of Radio Signals** chapter) mode for emergency communications.

Generally MFSK requires a high degree of tuning accuracy and frequency stability and can be susceptible to multipath distortion. DominoEX specifically addresses these issues. To avoid tuning issues, IFK (*incremental frequency keying*) is used. With IFK, the data is represented not by the frequency of each tone, but by the frequency difference between one tone and the next. It also uses offset incremental keying to reduce inter-symbol interference caused by multipath reception. These techniques provide a tuning tolerance of 200 Hz and a drift of 200 Hz/minute. DominoEX also features an optional FEC mode that increases latency but provides communications over even more difficult channels. More information can be found online at **www.qsl.net/zl1bpu/MFSK/DEX.htm**.

DominoEX uses M-ary FSK modulation with 18 tones in which only a single tone is present at any instant. Information is sent as a sequence of separate 4-bit ("nibble") symbols. The value of each nibble is represented during transmission as the position of the single tone.

The position of the tone is computed as the difference of the current nibble value from the nibble value of the previously transmitted symbol. In addition, a constant offset of 2 is applied to this difference. Because there are 18 tones, any possible 4-bit value between 0 and 15 can be represented, including the offset of 2.

The additional offset of 2 tone positions ensures that a transmitted tone is separated from the previously transmitted tone by at least two tone positions. It is thus impossible for two sequential symbols to result in the same tone being transmitted for two sequential tone periods. This means sequential tones will always be different by at least two positions, an important consideration in maintaining sync.

This minimum separation of successive tones of incremental frequency keying (IFK) in DominoEX reduces the inter-symbol distortion that results from a pulse being temporally smeared when passing through an HF channel. The double-tone spacing of DominoEX modes (see **Table 16.5**) further reduces inter-symbol distortion caused by frequency smearing.

Incremental frequency keying allows the DominoEX nibbles to immediately decode without having to wait for the absolute tone to be identified. With MFSK16, a tone cannot be uniquely identified until the lowest and highest of the 16 tones have passed through the receiver. This contributes to the decoding latency of MFSK16. There is no such latency with IFK.

Since IFK depends upon frequency differences and not absolute tone frequencies, DominoEX tolerates tuning errors and drifting signals without requiring any additional automatic frequency tracking algorithms.

Like MFSK16, the DominoEX signal is not wave-shaped and has constant output power. The baud rates for DominoEX are shown in Table 16.5. The tone spacings for DominoEX 11, DominoEX 16 and DominoEX 16 have the same values as their baud rates. The tone spacings for DominoEX 8, DominoEX 5 and DominoEX 4 are twice the value of their baud rates.

Unlike PSK31 and MFSK16, characters in DominoEX are encoded into varicode nibbles instead of encoding into varicode bits. The DominoEX varicode table can be found in the CD-ROM included with this book.

BEACON MESSAGE

Instead of transmitting an idle varicode symbol when there is no keyboard activity, DominoEX transmits a "beacon" message from an alternate set of varicode (SECVAR columns in the DominoEX varicode table). This user-supplied repeating beacon message is displayed at the receiving station when the sending station is not actively sending the primary message. On average, the character rate of the beacon channel is about half of the character rate of the primary channel.

FORWARD ERROR CORRECTION (FEC)

When FEC is turned off, DominoEX has very low decoding latency, providing an interactive quality that approaches RTTY and PSK31. The first character is decoded virtually instantly by the receiver after it is transmitted. Because of that, FEC is not usually used even though it is available in most software that implements DominoEX.

DominoEX FEC is similar to the FEC that is used in MFSK16. When FEC is on, each 4-bit IFK symbol that is decoded by the receiver is split into two di-bits. The dibits enter an identical convolution coder to that used

Table 16.5
Comparison of DominoEX Modes

Mode	*Baud/ (sec)*	*BW (Hz)*	*Tones*	*Speed (WPM)*	*FEC (WPM)*	*Tone Spacing*
DominoEX 4	3.90625	173	18	~25	~12	Baud rate ×2
DominoEX 5	5.3833	244	18	~31	~16	Baud rate ×2
DominoEX 8	7.8125	346	18	~50	~25	Baud rate ×2
DominoEX 11	10.766	262	18	~70	~35	Baud rate ×1
DominoEX 16	15.625	355	18	~100	~50	Baud rate ×1
DominoEX 22	21.533	524	18	~140	~70	Baud rate ×1

by MFSK16. However, instead of a 10-stage IZ8BLY interleaver (see Fig 16.4), only 4 cascaded stages of the basic 4-bit interleaver are present in DominoEX. In the presence of long duration fading, the performance of the shortened interleaver is moderately poor when used with DominoEX 16 and DominoEX 22. However, the interleaver is quite efficient in countering fading when FEC is used with DominoEX 4, DominoEX 5 and DominoEX 8, with their longer symbol periods.

Since DominoEX works in dibit units rather than nibble units when FEC is turned on, it also switches to using the same binary varicode used by MFSK16 instead of using the nibble-based varicode. DominoEX does not implement Gray code as is used by MFSK16 FEC.

Even without FEC, DominoEX works well under many HF propagation conditions, including the ITU NVIS ("Mid-latitude Disturbed") propagation profile. However, there are conditions where DominoEX is not usable unless FEC is switched on, specifically the CCIR Flutter and ITU High Latitude Moderate Conditions profiles. DominoEX modes, especially those with tone spacings that are twice the baud rate, are very robust even under these extreme conditions once FEC is switched on.

DominoEX performance charts (character error rates versus signal-to-noise ratios) are included as the HTML document "DominoEX Performance" on the CD-ROM accompanying this book and online at **http://homepage.mac.com/chen/Technical/DominoEX/Measurements/index.html**.

CHIP64/128

Chip64 and Chip128 modes were released in 2004 by Antonino Porcino, IZ8BLY. The modes were tested on the air by IZ8BLY, Murray Greenman, ZL1BPU, (who also contributed in the design of the system), Chris Gerber, HB9BDM, and Manfred Salzwedel, OH/DK4ZC. According to IZ8BLY, "The design of this new digital mode served to introduce the spread spectrum technology among radio amateurs by providing a communication tool to experiment with. Its purpose was to prove that it's possible to take advantage of the spread spectrum techniques even on the HF channels, making the communication possible under conditions where traditional narrowband modes fail.

"Among the different possible implementations of spread spectrum, Chip64 uses the so-called Direct Sequence Spread Spectrum (DSSS). In a DSSS transmission, the low speed signal containing the data bits to be transmitted is mixed (multiplied) with a greatly higher speed signal called code. The result of this mixing operation, called s*preading*, is a high-speed bit stream which is then transmitted as a normal DBPSK. Indeed, a DSSS signal looks like nothing else than wideband BPSK.

"The system proved to be efficient and we found it comparable to the other modern digital modes. Being totally different in its architecture, it shows better performance during certain circumstances, while in others it shows no actual gain. In particular, it performs better under multipath where normal BPSK can't track arriving symbols, but in quiet environments it doesn't show any improvement over plain BPSK. This is expected because of the losses that occur due to the imperfect autocorrelation of the codes."

Chip64 has a total data rate of 37.5 bit/s and the more robust Chip128 is 21.09 bit/s. Both use the same varicode used by MFSK16. The software can be downloaded from **http://xoomer.virgilio.it/aporcino/Chip64/index.htm** and more information is available at **www.arrl.org/technical-characteristics**. Spread spectrum is discussed in the **Modulation** chapter, as well.

16.2.5 THROB

Throb is an experimental mode written by Lionel Sear, G3PPT, and gets the name from the "throbbing" sound it makes on the air. It uses either single tones or pairs from a possible nine tones spaced 8 or 16 Hz apart, resulting in a bandwidth of 72 or 144 Hz, respectively. It has three transmission speeds — 1, 2 and 4 throbs/s — resulting in data rates of 10, 20 and 40 WPM, respectively. The 1 and 2 throb/s speeds use a tone spacing of 8 Hz for a 72 Hz bandwidth and the 4-throb/s speed uses a spacing of 16 Hz for a 144 Hz bandwidth. It is implemented as a standalone application or included in a multimode package such as *MixW* (**www.mixw.net**).

16.2.6 MT63

MT63 is a mode developed by Pawel Jalocha, SP9VRC. MT63 is very complex with wide bandwidth, low speed and very high noise immunity. By using 64 different modulated tones, MT63 includes a large amount of extra data in the transmission of each character, so that the receiving equipment can work out, with no ambiguity, which character was sent, even if 25% of the character is obliterated. MT63 also features a secondary channel that operates simultaneously with the main channel that can be used for an ID or beacon.

MT63 likely has the most extensive error correction and can be quite processor intensive. It uses a Walsh function that spreads the data bits of each 7-bit ASCII character across all 64 of the tones of the signal spectrum and simultaneously repeats the information over a period of 64 symbols within any one tone. This coding takes several seconds. The combination of time domain (temporal) and frequency domain (spectral) interleaving results in superb impulse noise rejection. At the same time, in the frequency domain, significant portions of the signal can be masked by unwanted noise or other transmissions without any noticeable effect on successful reception.

On each of the 64 tones, the transmission data rate is fairly slow, which suits the nature of ionospheric disturbances. Despite the low data rate, good text speed is maintained because the text is sent on many tones at once. The system runs at several different speeds, which can be chosen to suit conditions but 100 WPM is typical of the MT63-1K mode. Although the 1 kHz bandwidth mode is typical, MT63 can also run at 500 Hz and 2 kHz bandwidth where the tone spacing and baud rate are halved or doubled and the throughput is halved or doubled, respectively.

Tuning of MT63 modes is not critical. This is because the mode can use FEC techniques to examine different combinations of the 64 tones that calculate the correct location within the spectrum. As an example, MT63-1K will still work if the decoder is off-frequency by as much as 100 Hz. MT63-2K requires even less precision and can tolerate an error of 250 Hz.

The incredible noise immunity comes at a price beyond the large bandwidth required. There is a large latency caused by the error correction and interleaving process. Quick-turnaround QSOs are not possible because there is a several second delay between typing the last character and it being transmitted.

Without confirming each transmission with some type of ARQ mode, there is no more robust digital mode than MT63. The mode was evaluated and recommended for Navy MARS message handling. The evaluation is published on the Navy MARS website (**www.navymars.org**), along with other information on this mode.

16.2.7 Olivia

Olivia is an MFSK-based protocol designed to work in difficult (low signal-to-noise ratio plus multipath propagation) conditions on the HF bands. The signal can still be copied accurately at 10 dB below the noise floor. Olivia was developed in 2003 by Pawel Jalocha, SP9VRC, and performs well for digital data transfer with white noise, fading and multipath, polar path flutter and auroral conditions.

Olivia transmits a stream of 7-bit ASCII characters. The characters are sent in blocks of five with each block requiring two seconds to transmit. This results in an effective data rate of 2.5 characters/second or 150 characters/minute. A transmission bandwidth of 1000 Hz and the baud rate of 31.25 MFSK tones/second, also known as *Olivia 1000/32*, is the most common. To adapt to different propagation conditions, the number of tones

and the bandwidth can be changed and the time and frequency parameters are proportionally scaled. The number of tones can be 2, 4, 8, 16, 32, 64, 128 or 256 and the bandwidth can be 125, 250, 500, 1000 or 2000 Hz.

The Olivia is constructed of two layers: the lower, modulation and FEC code layer is a classical MFSK while the higher layer is an FEC code based on Walsh functions. More detail on Walsh functions is available online at **en.wikipedia.org/wiki/Walsh_function**.

Assuming Olivia 1000/32 is being used, in the first layer the orthogonal functions are cosine functions, with 32 different tones. Since only one of those 32 tones is being sent at a time, the demodulator measures the amplitudes of all the 32 possible tones and identifies the tone with the highest amplitude. In the second layer every ASCII character is encoded as one of 64 possible Walsh functions. The receiver again measures the amplitudes for all 64 vectors and selects the greatest as the true value.

To avoid simple transmitted patterns (like a constant tone) and to minimize the chance for a false lock at the synchronizer, the characters encoded into the Walsh function pass through a scrambler and interleaver. The receiver synchronizes automatically by searching through time and frequency offsets for a matching pattern.

More information can be found online at **http://n1su.com/olivia** and Olivia is supported in a number of digital multimode packages such as *MixW*, *MultiPSK* and *Ham Radio Deluxe*.

16.3 Fuzzy Modes

There is a group of modes referred to as "fuzzy modes" because although they are machine generated and decoded, they are designed to be human-read. These include facsimile (fax), slow-scan TV (SSTV) and Hellschreiber.

16.3.1 Facsimile (fax)

Facsimile was developed as a mechanically transmitted technology where the source material was placed on a spinning drum and scanned line by line into an electrical signal which would be transmitted by wire or over the air. It is important that the receiving station have their drum spinning at the correct speed in order to correctly recreate the image. A value known as the *index of cooperation* (*IOC*) must also be known to decode a transmission. IOC governs the image resolution and is the product of the total line length and the number of lines per unit length divided by π. Most fax transmissions are sent with LPM (RPM) at 120 and an IOC of 576.

Facsimile is generally transmitted in single sideband with a tone of 1500 Hz representing black and 2300 Hz representing white. The *automatic picture transmission* (*APT*) format is used by most terrestrial weather facsimile stations and geostationary weather satellites. It features a start tone that triggers the receiving system, originally used to allow the receiving drum to come up to speed. It also includes a phasing signal with a periodic pulse that synchronizes the receiver so the image appears centered on the page. A stop tone, optionally followed by black, indicates the end of the transmission. The APT format is shown in **Table 16.6**.

Stations with Russian equipment sometimes use RPM 60 or 90 and sometimes an IOC of 288. Photofax transmissions such as those from North Korea use RPM 60 and an IOC 352 with gray tones, and satellite rebroadcast use also RPM 120 IOC 576, with gray tones (4 or more bit depth). For software decoding of weather fax images it is best to decode with Black and White (2-bit depth).

16.3.2 Slow-Scan TV (SSTV)

Slow-Scan TV or SSTV is similar to facsimile where a single image is converted to individual scanned lines and those lines sent as variable tones between 1500 and 2300 Hz. Modern systems use computer software and a sound card to generate and receive the required tones. (Some SSTV communication uses purely digital protocols, such as WinDRM described later, in which the picture content is sent as digital data and not directly represented in the modulation scheme.)

There are a number of different SSTV "modes" that define image resolution and color scheme. A color image takes about 2 minutes to transmit, depending on mode. Some black and white modes can transmit an image in under 10 seconds. More information about SSTV may be found in the **Image Communications** supplement on the *Handbook* CD.

16.3.3 Hellschreiber, Feld-Hell or Hell

Hellschreiber is a facsimile-based mode developed by Rudolph Hell in the 1920s. The name is German and means "bright writer" or "light writer" and is a pun on the inventor's name. In Hellschreiber, text is transmitted by dividing each column into seven pixels and transmitting them sequentially starting at the lowest pixel. Black pixels are transmitted as a signal and white as silence at 122.5 bit/s (about a 35 WPM text rate). Originally the text was printed on continuous rolls of paper so the message could be any length.

Even though each pixel is only transmitted once, they are printed twice, one below the other. This compensates for slight timing errors in the equipment that causes the text to slant. If properly in sync, the text will appear as two identical rows, one below the other or a line of text in the middle with chopped lines top and bottom. Regardless of the slant, it is always possible to read one copy of the text. Since the text is read visually, it can be sent in nearly any language and tends to look like an old dot matrix printer. More information can be found online at **www.qsl.net/zl1bpu/FUZZY/Feld.htm** and Randy, K7AGE, has a great introduction to Hellschreiber available on YouTube at **www.youtube.com/watch?v=yR-EmyEBVqA**.

Table 16.6
Facsimile Automatic Picture Format

Signal	*Duration*	*IOC576*	*IOC288*	*Remarks*
Start Tone	5 s	300 Hz	675 Hz	200Hz for color fax modes
Phasing Signal	30 s			White line interrupted by black pulse
Image	Variable	1200 lines	600 lines	At 120 LPM
Stop Tone	5 s	450 Hz	450 Hz	
Black	10 s			

16.4 Structured Digital Modes

This group of digital modes has more structured data. This provides more robust data connections and better weak signal performance or more sophisticated data. Each of these modes bundles data into packets or blocks that can be transmitted and error checked at the receive end.

16.4.1 FSK441

FSK441 was implemented in the *WSJT* software by Joe Taylor, K1JT, in 2001. FSK441 was designed for meteor scatter using four-tone frequency shift keying at a rate of 441 baud and was given the technical name FSK441. Meteor scatter presents an unusual problem because the signals are reflected by the ionization trails left by meteors that often last less than a second. The signals are weak, of short duration with widely varying signal strength and include some amount of Doppler shift. Since these meteors enter the atmosphere constantly, this mode can be used any time of day or year to make contacts on VHF between 500 and 1100 miles with a single large Yagi and 100 W.

FSK441 uses a 43-character alphabet compatible with the earlier PUA43 mode by Bob Larkin, W7PUA. This alphabet includes letters, numbers and six special characters and is shown in the **Table 16.7**. These characters are encoded by the four tones at 882, 1323, 1764 and 2205 Hz and are designated 0-3 in the table below. For example, the letter T has the code 210 and is transmitted by sending sequentially the tones at 1764, 1323 and 882 Hz. Since the modulation rate is specified as 441 baud, or 441 bit/s, the character transmission rate is 441/3 = 147 characters per second. At this speed, a ping lasting 0.1 s can convey 15 characters of text. Four signals (000, 111, 222 and 333) generate single solid tones that are easily decoded and are reserved for the standard meteor scatter messages — R26, R27, RRR and 73.

When receiving FSK441, the receiver time and frequency need to be synchronized precisely. The computer clock must be set as accurately as possible (ideally to WWV or other time standard) before attempting to transmit or receive. When the receiver hears a burst of signal the decoder attempts to identify the correct frequency shift and align the message. For reasons of transmission efficiency, no special synchronizing information is embedded in an FSK441 message. Instead, the proper synchronization is established from the message content itself, making use of the facts that (a) three-bit sequences starting with 3 are never used, and (b) the space character is coded as 033, as shown in FSK441 character code table. Messages sent by *WSJT* always contain at least one trailing space and the software will insert one if necessary.

Timed message sequences are a must, and according to the procedures used by common consent in North America, the westernmost station transmits first in each minute for 30 seconds, followed by 30 seconds of receiving. *WSJT* messages are also limited to 28 characters which are plenty to send call sign and signal report information but clearly does not make *WSJT* a good ragchew mode.

The meteor scatter procedures are also well-defined and supported directly by templates in the *WSJT* software. Signal reports are conventionally sent as two-digit numbers. The first digit characterizes the lengths of pings being received, on a 1-5 scale, and the second estimates their strength on a 6-9 scale. The most common signal reports are 26 for weak pings and 27 for stronger ones, but under good conditions reports such as 38 and higher are sometimes used. Whatever signal report is sent to the QSO partner, it is important that it not be changed. You never know when pings will successfully convey fragments of your message to the other end of the path, and you want the received information to be consistent.

More information can be found at the *WSJT* website in the user manual and other articles at **http://pulsar.princeton.edu/~joe/K1JT**.

16.4.2 JT6M

JT6M is another *WSJT* mode optimized for meteor and ionospheric scatter on 6 meters. Like FSK441, JT6M uses 30 second periods for transmission and reception to look for enhancements produced by short-lived meteor trail ionization. It will also account for Doppler shift of ± 400 Hz. JT6M includes some improvements that allow working signals many dB weaker than FSK441.

JT6M uses 44-tone FSK with a synchronizing tone and 43 possible data tones — one for each character in the supported alphanumeric set, the same set used for FSK441. The sync tone is at 1076.66 Hz, and the 43 other possible tones are spaced at intervals of 11025/512 = 21.53 Hz up to 2002.59 Hz. Transmitted symbols are keyed at a rate of 21.53 baud, so each one lasts for 1/21.53 = 0.04644 s. Every 3rd symbol is the sync tone, and each sync symbol is followed by two data symbols. The transmission rate of user data is therefore (2/3)×21.53 = 14.4 characters per second. The transmitted signal sounds a bit like piccolo music.

Contacts in FSK441 and JT6M are often scheduled online on Ping Jockey at **www.pingjockey.net**.

Table 16.7
FSK441 Character Codes

Character	*Tones*	*Character*	*Tones*
1	001	H	120
2	002	I	121
3	003	J	122
4	010	K	123
5	011	L	130
6	012	M	131
7	013	N	132
8	020	O	133
9	021	P	200
.	022	Q	201
,	023	R	202
?	030	S	203
/	031	T	210
#	032	U	211
space	033	V	212
$	100	W	213
A	101	X	220
B	102	Y	221
C	103	0	223
D	110	E	230
F	112	Z	231
G	113		

16.4.3 JT65

JT65 is another *WSJT*-supported mode designed to optimize Earth-Moon-Earth (EME) contacts on the VHF bands, and conforms efficiently to the established standards and procedures for such QSOs. It also performs well for weak signal VHF/UHF, and for HF skywave propagation. JT65 includes error-correcting features that make it very robust, even with signals much too weak to be heard. With extended transmission duration and three decoders, JT65 can reliably decode 24 to 30 dB *below* the noise floor.

JT65 does not transmit messages character by character, as done in Morse code. Instead, whole messages are translated into unique strings of 72 bits, and from those into sequences of 63 six-bit symbols. These symbols are transmitted over a radio channel; some of them may arrive intact, while others are corrupted by noise. If enough of the symbols are correct, the full 72-bit compressed message can be recovered exactly. The decoded bits are then translated back into the human-readable message that was sent. The coding scheme and robust FEC assure that messages are never received in fragments. Message components cannot be mistaken for one another, and call signs are never displayed with a few characters missing or incorrect. There is no chance for the letter O or R in a call sign to be confused with a signal report or an acknowledgment.

JT65 uses 60-second transmit-receive sequences and carefully structured messages. Standard messages are compressed so that two call signs and a grid locator can be transmitted with just 71 bits. A 72nd bit serves as a flag to indicate that the message consists of arbitrary text (up to 13 characters) instead of

call signs and a grid locator. Special formats allow other information such as call sign prefixes (for example, ZA/PA2CHR) or numerical signal reports (in dB) to be substituted for the grid locator.

The aim of source encoding is to compress the common messages used for EME QSOs into a minimum fixed number of bits. After being compressed into 72 bits, a JT65 message is augmented with 306 uniquely defined error-correcting bits. The FEC coding rate is such that each message is transmitted with a "redundancy ratio" of 5.25. With a good error-correcting code, however, the resulting performance and sensitivity are far superior to those obtainable with simple five times message repetition. The high level of redundancy means that JT65 copes extremely well with QSB. Signals that are discernible to the software for as little as 10 to 15 seconds in a transmission can still yield perfect copy.

JT65 requires tight synchronization of time and frequency between transmitter and receiver. Each transmission is divided into 126 contiguous time intervals or symbols, each lasting 0.372 s. Within each interval the waveform is a constant-amplitude sinusoid at one of 65 pre-defined frequencies. Half of the channel symbols are devoted to a pseudo-random synchronizing vector interleaved with the encoded information symbols. The sync vector allows calibration of relative time and frequency offsets between transmitter and receiver. A transmission nominally begins at t = 1 s after the start of a UTC minute and finishes at t = 47.8 s.

WSJT does its JT65 decoding in three phases: a soft-decision *Reed-Solomon* decoder, the deep search decoder and the decoder for shorthand messages. In circumstances involving birdies, atmospherics, or other interference, operator interaction is an essential part of the decoding process. The operator can enable a "Zap" function to excise birdies, a "Clip" function to suppress broadband noise spikes and a "Freeze" feature to limit the frequency range searched for a sync tone. Using these aids and the program's graphical and numerical displays appropriately, the operator is well equipped to recognize and discard any spurious output from the decoder.

The JT65 procedures are also well-defined and supported directly by templates in the *WSJT* software. More information can be found at the *WSJT* website in the user manual and other articles at **http://pulsar.princeton.edu/~joe/K1JT**.

16.4.4 WSPR

WSPR (pronounced "whisper") implements a protocol designed for probing potential propagation paths with low-power transmissions. Each transmission carries a station's call sign, Maidenhead grid locator, and transmitter power in dBm. The program can decode signals with S/N as low as –28 dB in a 2500 Hz bandwidth. Stations with Internet access can automatically upload their reception reports to a central database called WSPRnet, which includes a mapping facility.

WSPR implements a protocol similar to JT65 called MEPT_JT (Manned Experimental Propagation Tests, by K1JT). In receive mode the program looks for all detectable MEPT_JT signals in a 200-Hz passband, decodes them, and displays the results. If nothing is decoded, nothing will be printed. In T/R mode the program alternates in a randomized way between transmit and receive sequences. Like JT65, MEPT_JT includes very efficient data compression and strong forward error correction. Received messages are nearly always exactly the same as the transmitted message, or else they are left blank.

Basic specifications of the MEPT_JT mode are as follows:

- Transmitted message: call sign + 4-character-locator + dBm Example: "K1JT FN20 30"
- Message length after lossless compression: 28 bits for call sign, 15 for locator, 7 for power level; 50 bits total.
- Forward error correction (FEC): Long-constraint convolutional code, K =32, r = 1/2.
- Number of channel symbols: nsym = (50 + K – 1) × 2 = 162.
- Keying rate: 12000/8192 = 1.46 baud.
- Modulation: Continuous phase 4-FSK. Tone separation 1.46 Hz.
- Synchronization: 162-bit pseudo-random sync vector.
- Data structure: Each channel symbol conveys one sync bit and one data bit.
- Duration of transmission: (162 × 8192)/12000 = 110.6 s
- Occupied bandwidth: About 6 Hz
- Minimum S/N for reception: Around –27 dB on the *WSJT* scale (2500 Hz reference bandwidth).

The current version and documentation of *WSPR* can be found at **www.physics.princeton.edu/pulsar/K1JT/wspr.html** and WSPRnet propagation data is available at **http://wsprnet.org**.

16.4.5 HF Digital Voice

AOR

In 2004, AOR Corporation introduced its HF digital voice and data modem, the AR9800. Digital voice offers a quality similar to FM with no background noise or fading as long as the signal can be properly decoded. The AR9800 can alternatively transmit binary files and images. AOR later released the AR9000 which is compatible with the AR9800 but less expensive and only supports the HF digital voice mode.

The AR9800 uses a protocol developed by Charles Brain, G4GUO. The protocol uses the AMBE (Advanced Multi-Band Excitation) codec from DVSI Inc. to carry voice. It uses 2400 bit/s for voice data with an additional 1200 bit/s for Forward Error Correction for a total 3600 bit/s data stream. The protocol is detailed below:

- Bandwidth: 300-2500 Hz, 36 carriers
- Symbol Rate: 20 ms (50 baud)
- Guard interval: 4 ms
- Tone steps: 62.5 Hz
- Modulation method: 36 carriers: DQPSK (3.6K)
- AFC: ±125 Hz
- Error correction: Voice: Golay and Hamming
- Video/Data: Convolution and Reed-Solomon
- Header: 1 s; 3 tones plus BPSK training pattern for synchronization
- Digital voice: DVSI AMBE2020 coder, decoder
- Signal detection: Automatic Digital detect, Automatic switching between analog mode and digital mode
- Video Compression: AOR original adaptive JPEG

AOR has more information online at **www.aorusa.com/ard9800.html** and Amateur Radio Video News featured a number of digital voice modes including the AOR devices available on DVD at **www.arvideonews.com/dv/index.html**.

WinDRM

Digital Radio Mondiale (*DRM*) — not to be confused with Digital Rights Management — is a set of commercial digital audio broadcasting technologies designed to work over the bands currently used for AM broadcasting, particularly shortwave. DRM can fit more channels than AM, at higher quality, into a given amount of bandwidth, using various MPEG-4 codecs optimized for voice or music or both. As a digital modulation, DRM supports data transmission in addition to the audio channels. The modulation used by DRM is COFDM (coded orthogonal frequency division multiplex) with each carrier coded using QAM (quadrature amplitude modulation). The use of multiple carriers in COFDM provides a reasonably robust signal on HF.

Two members of Darmstadt University of Technology in Germany, Volker Fisher and Alexander Kurpiers, developed an open-source software program to decode commercial DRM called *Dream*. Typically, the DRM commercial shortwave broadcasts use 5 kHz, 10 kHz or 20 kHz of bandwidth (monaural or stereo) and require large bandwidths in receivers.

Shortly after, Francesco Lanza, HB9TLK, modified the *Dream* software to use the DRM

technology concepts in a program that only used 2.5 kHz of bandwidth for ham radio use. His software, named *WinDRM*, includes file transfer functionality and is available for download at **http://n1su.com/windrm/**.

FreeDV

FreeDV is a *Windows* and *Linux* application that allows any SSB radio to be used for low bit-rate digital voice. Speech and call sign data is compressed down to 1400 bit/s, which then modulates an 1125 Hz wide QPSK signal, which is then applied to the microphone input of an SSB radio. On receive, the signal is received as SSB, then further demodulated and decoded by *FreeDV*.

FreeDV was coded from scratch by David Witten, KDØEAG, (GUI, architecture) and David Rowe, VK5DGR, (*Codec2*, modem implementation, integration). The *FreeDV* design and user interface is based on *FDMDV*, which was developed by Francesco Lanza, HB9TLK. Francesco received advice on modem design from Peter Martinez, G3PLX. Bruce Perens, K6BP has been a thought leader on open source, patent-free voice codecs for Amateur Radio. He has inspired, promoted and encouraged the development of *Codec2* and *FreeDV*. A team of reviewers and beta testers also supported the development of *FreeDV* as credited on the **http://freedv.org** home page.

FreeDV is entirely open source — even the voice codec. This makes it unique among Amateur Radio digital voice systems that typically rely on a proprietary voice codec that is not available to ham experimentation.

FreeDV Design

- *Codec2* voice codec and *FDMDV* modem
- 14, 50 baud QPSK voice data carriers
- 1 center BPSK carrier with 2× power for fast and robust synchronization
- 1.125 kHz spectrum bandwidth (half SSB) with 75 Hz carrier spacing
- 1400 bit/s data rate with 1375 bit/s voice coding and 25 bit/s text for call sign ID
- No interleaving in time or FEC, resulting in low latency, fast synchronization and quick recovery from fades
- 44.1 or 48 kHz sample rate, sound card compatible

Key Features

- Waterfall, spectrum, scatter and audio oscilloscope displays
- Adjustable squelch
- Fast/slow SNR estimation
- Microphone and speaker signal audio equalizer
- Control of transmitter PTT via RS-232 levels
- Works with one (receive only) or two (transmit and receive) sound cards, for example a built-in sound card and USB headphones.

More details and the software can be found at **http://freedv.org** as well as written and video setup guides. A coordinating website for *FreeDV* QSOs is available at **http://qso.k7ve.org**.

16.4.6 ALE

Automatic link establishment (*ALE*) was created as a series of protocols for government users to simplify HF communications. The protocol provides a mechanism to analyze signal quality on various channels/bands and choose the best option. The purpose is to provide a reliable rapid method of calling and connecting during constantly changing HF ionospheric propagation, reception interference and shared spectrum use of busy or congested HF channels. It also supports text messages with a very robust protocol that can get through even if no voice-quality channel can be found.

Each radio ALE station uses a call sign or address in the ALE controller. When not actively in communication with another station, each HF SSB transceiver constantly scans through a list of frequencies, listening for its call sign. It decodes calls and soundings sent by other stations, using the bit error rate to store a quality score for that frequency and sender call sign.

To reach a specific station, the caller simply enters the call sign, just like dialing a phone number. The ALE controller selects the best available frequency and sends out brief digital selective calling signals containing the call signs. When the distant scanning station detects the first few characters of its call sign, it stops scanning and stays on that frequency. The two stations' ALE controllers automatically handshake to confirm that a link of sufficient quality is established and they are ready to communicate.

When successfully linked, the receiving station which was muted will typically emit an audible alarm and visual alert for the receiving operator of the incoming call. It also indicates the call sign of the linked station. The operators then can talk in a regular conversation. At the conclusion of the QSO, one of the stations sends a disconnect signal to the other station, and they each return their ALE stations to the scanning mode. Some military / commercial HF transceivers are available with ALE available internally. Amateur Radio operators commonly use the PCALE soundcard software ALE controller, interfaced to a ham transceiver via rig control cable and multi-frequency antenna.

Table 16.8
ALE Tones

Frequency	*Data*
750 Hz	000
1000 Hz	001
1250 Hz	011
1500 Hz	010
1750 Hz	110
2000 Hz	111
2250 Hz	101
2500 Hz	100

The ALE waveform is designed to be compatible with the audio passband of a standard SSB radio. It has a robust waveform for reliability during poor path conditions. It consists of 8-ary frequency-shift keying (FSK) modulation with eight orthogonal tones, a single tone for a symbol. These tones represent three bits of data, with least significant bit to the right, as shown in **Table 16.8**.

The tones are transmitted at a rate of 125 tones per second, 8 ms per tone. The resultant transmitted bit rate is 375 bit/s. The basic ALE word consists of 24 bits of information. Details can be found in Federal Standard 1045, Detailed Requirements at **www.its.bldrdoc.gov/fs-1054a/ 45-detr.htm**.

It would require a lot of time for the radio to go through the sequence of calling a station on every possible frequency to establish a link. Time can be decreased by using a "smarter" way of predictive or synchronized linking. With *Link Quality Analysis* (LQA), an ALE system uses periodic sounding and linking signals between other stations in the network to stay in touch and to predict which channel is likely to support a connection to the desired station at any given time. Various stations may be operating on different channels, and this enables the stations to find and use a common open channel.

The PCALE software developed by Charles Brain, G4GUO, is available for download at **http://hflink.com/software**. Much more ALE information and real-time data is available online at **http:/hflink.com**.

16.5 Networking Modes

The modes described in this section operate using features and functions associated with computer-to-computer networking. Even though communication using these modes may not involve the creation of a network, the modes are referred to as "networking modes" because of their structure. In cases such as Winlink 2000 and D-STAR, the most common use is to implement a networked system and those features are described along with the modes and protocols used to implement communications within the network.

16.5.1 OSI Networking Model

The Open Systems Interconnection Model or *OSI Model* is an abstract description for computer network protocol design. It defines seven different *layers* or functions performed by a *protocol stack*. In the OSI model, the highest level is closest to the user and the lowest is closest to the hardware required to transport the data (network card and wire or radio). The seven layers are described in **Table 16.9**. The modes examined previously implemented the protocols as a *monolithic stack* where all the functions are performed inside a single piece of code. The modes described in this section implement networking features in a more modular fashion. This allows greater flexibility (and complexity) when mixing features.

As a data packet moves through these layers the header or preamble is removed and any required action performed before the data is passed to the next layer, much like peeling away layers of an onion until just the basic clean data is left. The OSI model does not define any interfaces between layers; it is just a conceptual model of the functions required. Real-world protocols rarely implement each layer individually and often span multiple layers.

This description is by no means exhaustive and more information can be found online and in every networking textbook. **Table 16.10** shows the placement of commonly recognized protocols within the OSI layered structure.

16.5.2 Connected and Connectionless Protocols

The protocols discussed to this point have been *connectionless* meaning they don't establish a connection with a specific machine for the purpose of transferring data. Even with packetized modes like FSK441 with a destination call sign specified, the packet is transmitted and it's up to the user to identify they are the intended recipient. In a packet-switched network, connectionless mode transmission is a transmission in which each packet is prepended with a header containing a destination address to allow delivery of the packet without the aid of additional instructions. A packet transmitted in a connectionless mode is frequently called a *datagram*.

In *connection-oriented protocols*, the stations about to exchange data need to first declare to each other they want to "establish a connection". A connection is sometimes defined as a logical relationship between the peers exchanging data. Connected protocols can use a method called *automatic repeat request* (*ARQ*) to insure accurate delivery of packets using *acknowledgements* and *timeouts*. This allows the detection and correction of corrupted packets, misdelivery, duplication, or out-of-sequence delivery of the packets.

Connectionless modes can have error correction and detection included by a higher layer of the protocol but they have no mechanism to request a correction. An advantage of connectionless mode over connection-oriented mode is that it has a low data overhead. It also allows for *multicast* and *broadcast* (net-type) operations, which may save even more network resources when the same data needs to be transmitted to several recipients. In contrast, a connected mode is always *unicast* (point-to-point).

Another drawback of the connectionless mode is that no optimizations are possible when sending several frames between the same two peers. By establishing a connection at the beginning of such a data exchange the components (routers, bridges) along the network path would be able to pre-compute (and hence cache) routing-related information, avoiding re-computation for every packet.

Many network modes incorporate both types of protocol for different purposes. In the Internet TCP/IP protocol, TCP is a connection-oriented transport protocol where UDP is connectionless.

16.5.3 The Terminal Node Controller (TNC)

While a *terminal node controller* (*TNC*) is nominally an OSI Physical layer device, the internal firmware often implements a protocol such as PACTOR that handles all the routing, and error correction through the transport layer. This greatly simplifies the coding of any protocol or application that uses these devices.

A TNC is actually a computer that contains the protocols implemented in firmware and a *modem* (modulator/demodulator). The

Table 16.9
OSI Seven Layer Networking Model

7 — Application Layer	End-user program or "application" that uses the network
6 — Presentation Layer	The format of data after transfer (code conversion, encryption)
5 — Session Layer	Manages the transfer process
4 — Transport Layer	Provides reliable data transfer to the upper layers
3 — Network Layer	Controls data routing
2 — Data Link Layer	Provides error detection and flow control
1 — Physical Layer	Signal used on the medium—voltage, current, frequency, etc.

Table 16.10
Networking Protocols in the OSI Model

Layer	*Examples*	*IP Protocol Suite*
7 — Application		NNTP, DNS, FTP, Gopher, HTTP, DHCP, SMTP, SNMP, TELNET
6 — Presentation	ASCII, EBCDIC, MIDI, MPEG	MIME, SSL
5 — Session	Named Pipes, NetBIOS, Half Duplex, Full Duplex, Simplex	Sockets, Session establishment in TCP
4 — Transport		TCP, UDP
3 — Network	AX.25	IP, IPsec, ICMP, IGMP
2 — Data Link	802.3 (Ethernet), 802.11a/b/g/n MAC/LLC, ATM, FDDI, Frame Relay, HDLC, Token Ring, ARP (maps layer 3 to layer 2 address)	PPP, SLIP, PPTP, L2TP
1 — Physical	RS-232, T1, 10BASE-T, 100BASE-TX, POTS, DSL, 802.11a/b/g/n, Soundcard, TNC, Radio	

TNC generally connects to a PC as a serial or USB device on one side and to the radio with appropriate audio and PTT cables on the other. Most of the newer rigs have dedicated data connections available that feature audio lines with fixed levels that are unaffected by settings in the radio. These jacks make swapping mike cables unnecessary when switching between voice and digital modes. Bypassing internal audio processing circuitry eliminates a number of issues that can cause problems with digital modes and makes the use of digital modes more reproducible/reliable by eliminating a number of variables when configuring equipment. These same data jacks are recommended when using a computer sound card.

Although many of the modes discussed can use a computer sound card to generate the required modulation and a separate mechanism to support push-to-talk (PTT), a TNC offers some advantages:

- TNC hardware can be used with any computer platform.
- A computer of nearly any vintage/performance level can be used.
- Data transmission/reception is unaffected by computer interruptions from virus checkers or other "inits."
- Initialization settings are held internal to the TNC and can easily be reset as needed — once working, they stay working.
- Virtually eliminates the computer as a problem/failure point.
- Offers features independent of the computer (digipeat, BBS, APRS beacon, telemetry, weather beacon, and so forth).

The majority of TNCs are designed for 300 or 1200-bit/s packet and implement the Bell 103 or Bell 202 modulation respectively. A *multimode communications processor* (*MCP*) or *multi-protocol controller* (*MPC*) may offer the capability to operate RTTY, CW, AMTOR, PACTOR, G-TOR, Clover, fax, SSTV and other modes in addition to packet. Some of these modes are only available in TNC hardware because the real-time operating system in the TNC provides a more reliable platform to implement the mode and it also helps protect proprietary intellectual property.

KISS-Mode TNCs have become popular. These devices simply provide the modem and filters to implement the baseband signals for a type of digital modulation. They rely on the computer software to generate the appropriate packet protocol and complete the mode. This means the software must be written specifically to support these TNCs by creating the entire AX.25 packet with the data embedded, rather than simply sending the data to the TNC expecting the TNC to frame the packet. By leaving the TNC to handle only the baseband signal generation and data recovery, much simpler, smaller and less expensive designs are possible while still retaining the platform independence and robustness of a separate TNC. The TNC-X from Coastal Chipworks at **http://tnc-x.com** is a good example of a KISS-mode TNC.

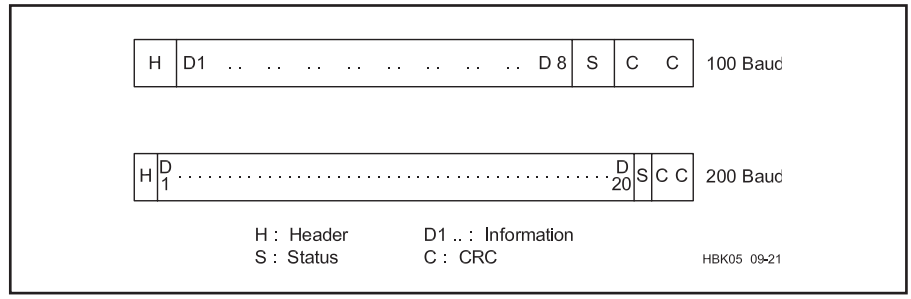

Fig 16.6 — PACTOR packet structure.

Table 16.11
PACTOR Timing

Object	*Length (seconds)*
Packet	0.96 (200 baud: 192 bits; 100 baud: 96 bits)
CS receive time	0.29
Control signals	0.12 (12 bits at 10 ms each)
Propagation delay	0.17
Cycle	1.25

16.5.4 PACTOR-I

PACTOR, now often referred to as PACTOR-I, is an HF radio transmission system developed by German amateurs Hans-Peter Helfert, DL6MAA, and Ulrich Strate, DF4KV. It was designed to overcome the shortcomings of AMTOR and packet radio. It performs well under both weak-signal and high-noise conditions. PACTOR-I has been overtaken by PACTOR-II and PACTOR-III but remains in use.

TRANSMISSION FORMATS

All packets have the basic structure shown in **Fig 16.6**, and their timing is as shown in **Table 16.11**.

- Header: Contains a fixed bit pattern to simplify repeat requests, synchronization and monitoring. The header is also important for the Memory ARQ function. In each packet that carries new information, the bit pattern is inverted.
- Data: Any binary information. The format is specified in the status word. Current choices are 8-bit ASCII or 7-bit ASCII (with Huffman encoding). Characters are not broken across packets. ASCII RS (hex 1E) is used as an IDLE character in both formats.
- Status word: See Table 16.11
- CRC: The CRC is calculated according to the CCITT standard, for the data, status and CRC.

The PACTOR acknowledgment signals are shown in **Table 16.12**. Each of the signals is 12-bits long. The characters differ in pairs in eight bits (Hamming offset) so that the chance of confusion is reduced. If the CS is not correctly received, the TX reacts by repeating the last packet. The request status can be uniquely recognized by the 2-bit packet number so that wasteful transmissions of pure RQ blocks are unnecessary.

The receiver pause between two blocks is 0.29 s. After deducting the CS lengths, 0.17 s remains for switching and propagation delays so that there is adequate reserve for DX operation.

CONTACT FLOW

In the listen mode, the receiver scans any received packets for a CRC match. This method uses a lot of computer processing resources, but it's flexible.

A station seeking contacts transmits CQ packets in an FEC mode, without pauses for acknowledgment between packets. The transmit time length, number of repetitions and speed are the transmit operator's choice. (This mode is also suitable for bulletins and other group traffic.) Once a listening station has copied the call, the listener assumes the TX station role and initiates a contact. Thus, the station sending CQ initially takes the RX station role. The contact begins as shown in **Table 16.13**.

With good conditions, PACTOR's normal signaling rate is 200 baud, but the system automatically changes from 200 to 100 baud and back, as conditions demand. In addition, Huffman coding can further increase the throughput by a factor of 1.7. There is no loss of synchronization speed changes; only one packet is repeated.

Table 16.12
PACTOR Control Signals

Code	*Chars (hex)*	*Function*
CS1	4D5	Normal acknowledge
CS2	AB2	Normal acknowledge
CS3	34B	Break-in (forms header of first packet from RX to TX)
CS4	D2C	Speed change request

All control signals are sent only from RX to TX

Table 16.13
PACTOR Initial Contact

Master Initiating Contact

Size (bytes)	1	8	6
Content	/Header	/SLAVECAL	/SLAVECAL/
Speed (baud)	100	100	200

Slave Response

The receiving station detects a call, determines mark/space polarity, and decodes 100 baud and 200-bd call signs. It uses the two call signs to determine if it is being called and the quality of the communication path. The possible responses are:

First call sign does not match slave's call sign (Master not calling this slave)	none
Only first call sign matches slave's call sign (Master calling this slave, poor communications)	CS1
First and second call signs both match the slaves (good circuit, request speed change to 200 baud)	CS4

When the RX receives a bad 200-baud packet, it can acknowledge with CS4. TX immediately assembles the previous packet in 100-baud format and sends it. Thus, one packet is repeated in a change from 200 to 100 baud.

The RX can acknowledge a good 100-baud packet with CS4. TX immediately switches to 200 baud and sends the next packet. There is no packet repeat in an upward speed change.

The RX station can become the TX station by sending a special change-over packet in response to a valid packet. RX sends CS3 as the first section of the changeover packet. This immediately changes the TX station to RX mode to read the data in that packet and responds with CS1 and CS3 (acknowledge) or CS2 (reject).

PACTOR provides a sure end-of-contact procedure. TX initiates the end of contact by sending a special packet with the QRT bit set in the status word and the call of the RX station in byte-reverse order at 100 baud. The RX station responds with a final CS.

16.5.5 PACTOR-II

This is a significant improvement over PACTOR-I, yet it is fully compatible with the older mode. PACTOR-II uses 16PSK to transfer up to 800 bit/s at a 100 baud rate. This keeps the bandwidth less than 500 Hz.

PACTOR-II uses digital signal processing (DSP) with Nyquist waveforms, Huffman and Markov compression and powerful Viterbi decoding to increase transfer rate and sensitivity into the noise level. The effective transfer rate of text is over 1200 bit/s. Features of PACTOR II include:

- Frequency agility — it can automatically adjust or lock two signals together over a ±100 Hz window.
- Powerful data reconstruction based upon computer power — with over 2 Mbyte of available memory.
- Cross correlation — applies analog Memory ARQ to acknowledgment frames and headers.
- Soft decision making — Uses artificial intelligence (AI), as well as digital information received to determine frame validity.
- Extended data block length — when transferring large files under good conditions, the data length is doubled to increase the transfer rate.
- Automatic recognition of PACTOR-I, PACTOR-II and so on, with automatic mode switching.
- Intermodulation products are canceled by the coding system.
- Two long-path modes extend frame timing for long-path terrestrial and satellite propagation paths.

This is a fast, robust mode that has excellent coding gain as well. PACTOR-II stations acknowledge each received transmission block. PACTOR-II employs computer logic as well as received data to reassemble defective data blocks into good frames. This reduces the number of transmissions and increases the throughput of the data.

16.5.6 PACTOR-III

PACTOR-III is a software upgrade for existing PACTOR-II modems that provides a data transmission mode for improved speed and robustness. Both the transmitting and receiving stations must support PACTOR-III for end-to-end communications using this mode.

PACTOR-III's maximum uncompressed speed is 2722 bit/s. Using online compression, up to 5.2 kbit/s is achievable. This requires an audio passband from 400 Hz to 2600 Hz (for PACTOR-III speed level 6). On an average channel, PACTOR-III is more than three times faster than PACTOR-II. On good channels, the effective throughput ratio between PACTOR-III and PACTOR-II can exceed five. PACTOR-III is also slightly more robust than PACTOR-II at their lower SNR edges.

The ITU emission designator for PACTOR-III is 2K20J2D. Because PACTOR-III builds on PACTOR-II, most specifications like frame length and frame structure are adopted from PACTOR-II. The only significant difference is PACTOR III's multitone waveform that uses up to 18 carriers while PACTOR-II uses only two carriers. PACTOR-III's carriers are located in a 120 Hz grid and modulated with 100 symbols per second DBPSK or DQPSK. Channel coding is also adopted from PACTOR-II's Punctured Convolutional Coding.

PACTOR-III Link Establishment

The calling modem uses the PACTOR-I FSK connect frame for compatibility. When the called modem answers, the modems negotiate to the highest level of which both modems are capable. If one modem is only capable of PACTOR-II, then the 500 Hz PACTOR-II mode is used for the session. With the MYLevel (MYL) command a user may limit a modem's highest mode. For example, a user may set MYL to 1 and only a PACTOR-I connection will be made, set to 2 and PACTOR-I and II connections are available, set to 3 and PACTOR-I through III connections are enabled. The default MYL is set to 2 with the current firmware and with PACTOR-III firmware it will be set to 3. If a user is only allowed to occupy a 500 Hz channel, MYL can be set to 2 and the modem will stay in its PACTOR-II mode.

The PACTOR-III Protocol Specification is available online at **www.scs-ptc.com/pactor.html**. More information can also be found online at **www.arrl.org/technical-characteristics** or in *ARRL's HF Digital Handbook*

by Steve Ford, WB8IMY.

The protocol specifications and equipment for PACTOR-IV have been released, but the mode is not yet legal for US amateurs. The symbol rate for PACTOR-IV is 1800 baud, but FCC rules limit US amateurs to 300 baud below the upper end of 10 meters. PACTOR-IV is being used outside the US by individual amateurs and by Winlink stations that are not subject to FCC rules. It is not known if or when this restriction will be lifted.

16.5.7 G-TOR

This brief description has been adapted from "A Hybrid ARQ Protocol for Narrow Bandwidth HF Data Communication" by Glenn Prescott, WBØSKX, Phil Anderson, WØXI, Mike Huslig, KBØNYK, and Karl Medcalf, WK5M (May 1994 *QEX*).

G-TOR is short for Golay-TOR, an innovation of Kantronics. It was inspired by HF automatic link establishment (ALE) concepts and is structured to be compatible with ALE. The purpose of the G-TOR protocol is to provide an improved digital radio communication capability for the HF bands. The key features of G-TOR are:

- Standard FSK tone pairs (mark and space)
- Link-quality-based signaling rate: 300, 200 or 100 baud
- 2.4-s transmission cycle
- Low overhead within data frames
- Huffman data compression — two types, on demand
- Embedded run-length data compression
- Golay forward-error-correction coding
- Full-frame data interleaving
- CRC error detection with hybrid ARQ
- Error-tolerant "Fuzzy" acknowledgments.

Since one of the objectives of this protocol is ease of implementation in existing TNCs, the modulation format consists of standard tone pairs (FSK), operating at 300, 200 or 100 baud, depending upon channel conditions. G-TOR initiates contacts and sends ACKs only at 100 baud. The G-TOR waveform consists of two phase-continuous tones (BFSK), spaced 200 Hz apart (mark = 1600 Hz, space = 1800 Hz); however, the system can still operate at the familiar 170 Hz shift (mark = 2125 Hz, space = 2295 Hz), or with any other convenient tone pairs. The optimum spacing for 300-baud transmission is 300 Hz, but you trade some performance for a narrower bandwidth.

Each transmission consists of a synchronous ARQ 1.92-s frame and a 0.48-s interval for propagation and ACK transmissions (2.4 s cycles). All advanced protocol features are implemented in the signal-processing software.

Data compression is used to remove redundancy from source data. Therefore, fewer bits are needed to convey any given message. This increases data throughput and decreases transmission time — valuable features for HF. G-TOR uses run-length encoding and two types of Huffman coding during normal text transmissions. Run-length encoding is used when more than two repetitions of an 8-bit character are sent. It provides an especially large savings in total transmission time when repeated characters are being transferred.

The Huffman code works best when the statistics of the data are known. G-TOR applies Huffman A coding with the upper- and lower-case character set, and Huffman B coding with upper-case-only text. Either type of Huffman code reduces the average number of bits sent per character. In some situations, however, there is no benefit from Huffman coding. The encoding process is then disabled. This decision is made on a frame-by-frame basis by the information sending station.

The real power of G-TOR resides in the properties of the (24, 12) extended Golay error-correcting code, which permits correction of up to three random errors in three received bytes. The (24, 12) extended Golay code is a half-rate error-correcting code: Each 12 data bits are translated into an additional 12 parity bits (24 bits total). Further, the code can be implemented to produce separate input-data and parity-bit frames.

The extended Golay code is used for G-TOR because the encoder and decoder are simple to implement in software. Also, Golay code has mathematical properties that make it an ideal choice for short-cycle synchronous communication. More information can also be found online at **www.arrl.org/technical-characteristics** or in *ARRL's HF Digital Handbook* by Steve Ford, WB8IMY.

16.5.8 CLOVER-II

The desire to send data via HF radio at high data rates and the problem encountered when using AX.25 packet radio on HF radio led Ray Petit, W7GHM, to develop a unique modulation waveform and data transfer protocol that is now called CLOVER-II. Bill Henry, K9GWT, supplied this description of the CLOVER-II system.

CLOVER modulation is characterized by the following key parameters:

- Very low base symbol rate: 31.25 symbols/second (all modes).
- Time-sequence of amplitude-shaped pulses in a very narrow frequency spectrum.
- Occupied bandwidth = 500 Hz at 50 dB below peak output level.
- Differential modulation between pulses.
- Multilevel modulation.

The low base symbol rate is very resistant to multipath distortion because the time between modulation transitions is much longer than even the worst-case time-smearing caused by summing of multipath signals. By using a time-sequence of tone pulses, Dolph-Chebychev "windowing" of the modulating signal and differential modulation, the total occupied bandwidth of a CLOVER-II signal is held to 500 Hz.

Multilevel tone, phase and amplitude modulation gives CLOVER a large selection of data modes that may be used (see **Table 16.14**). The adaptive ARQ mode of CLOVER senses current ionospheric conditions and automatically adjusts the modulation mode to produce maximum data throughput. When using the Fast bias setting, ARQ throughput automatically varies from 11.6 byte/s to 70 byte/s.

The CLOVER-II waveform uses four tone pulses that are spaced in frequency by 125 Hz. The time and frequency domain characteristics of CLOVER modulation are shown in **Figs 16.7, 16.8** and **16.9**. The time-domain shape of each tone pulse is intentionally shaped to produce a very compact frequency spectrum. The four tone pulses are spaced in time and then combined to produce the composite output shown. Unlike other modulation schemes, the CLOVER modulation spectrum is the same for all modulation modes.

Table 16.14
CLOVER-II Modulation Modes

As presently implemented, CLOVER-II supports a total of seven different modulation formats: five using PSM and two using a combination of PSM and ASM (Amplitude Shift Modulation).

Name	*Description*	*In-Block Data Rate*
16P4A	16 PSM, 4-ASM	750 bps
16PSM	16 PSM	500 bps
8P2A	8 PSM, 2-ASM	500 bps
8PSM	8 PSM	375 bps
QPSM	4 PSM	250 bps
BPSM	Binary PSM	125 bps
2DPSM	2-Channel Diversity BPSM	62.5 bps

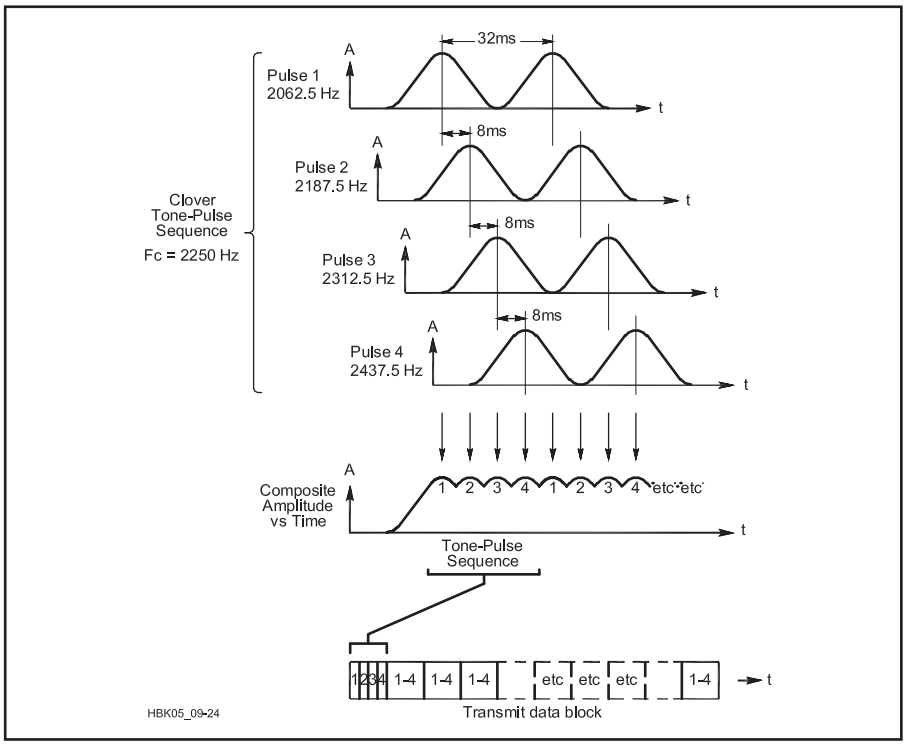

Fig 16.7 — Amplitude vs time plots for CLOVER-II's four-tone waveform.

Table 16.15
Data Bytes Transmitted Per Block

Block	*Reed-Solomon Encoder Efficiency*			
Size	*60%*	*75%*	*90%*	*100%*
17	8	10	12	14
51	28	36	42	48
85	48	60	74	82
255	150	188	226	252

Table 16.16
Correctable Byte Errors Per Block

Block	*Reed-Solomon Encoder Efficiency*			
Size	*60%*	*75%*	*90%*	*100%*
17	1	1	0	0
51	9	5	2	0
85	16	10	3	0
255	50	31	12	0

Data is modulated on a CLOVER-II signal by varying the phase and/or amplitude of the tone pulses. Further, all data modulation is differential on the same tone pulse — data is represented by the phase (or amplitude) difference from one pulse to the next. For example, when binary phase modulation is used, a data change from 0 to 1 may be represented by a change in the phase of tone pulse one by 180° between the first and second occurrence of that pulse. Further, the phase state is changed only while the pulse amplitude is zero. Therefore, the wide frequency spectra normally associated with PSK of a continuous carrier is avoided. This is true for all CLOVER-II modulation formats. The term *phase-shift modulation* (PSM) is used when describing CLOVER modes to emphasize this distinction.

CLOVER-II has four "coder efficiency" options: 60%, 75%, 90% and 100% ("efficiency" being the approximate ratio of real data bytes to total bytes sent). 60% efficiency corrects the most errors but has the lowest net data throughput. 100% efficiency turns the encoder off and has the highest throughput but fixes no errors. There is therefore a tradeoff between raw data throughput versus the number of errors that can be corrected without resorting to retransmission of the entire data block.

Note that while the In Block Data Rate numbers listed in the table go as high as 750 bit/s, overhead reduces the net throughput or overall efficiency of a CLOVER transmission. The FEC coder efficiency setting and protocol requirements of FEC and ARQ modes add overhead and reduce the net ef-

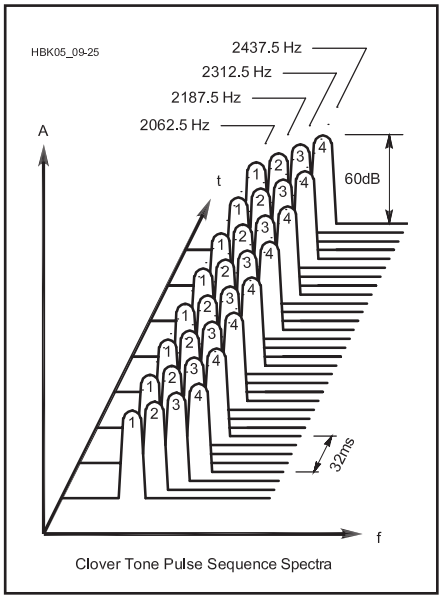

Fig 16.8 — A frequency-domain plot of a CLOVER-II waveform.

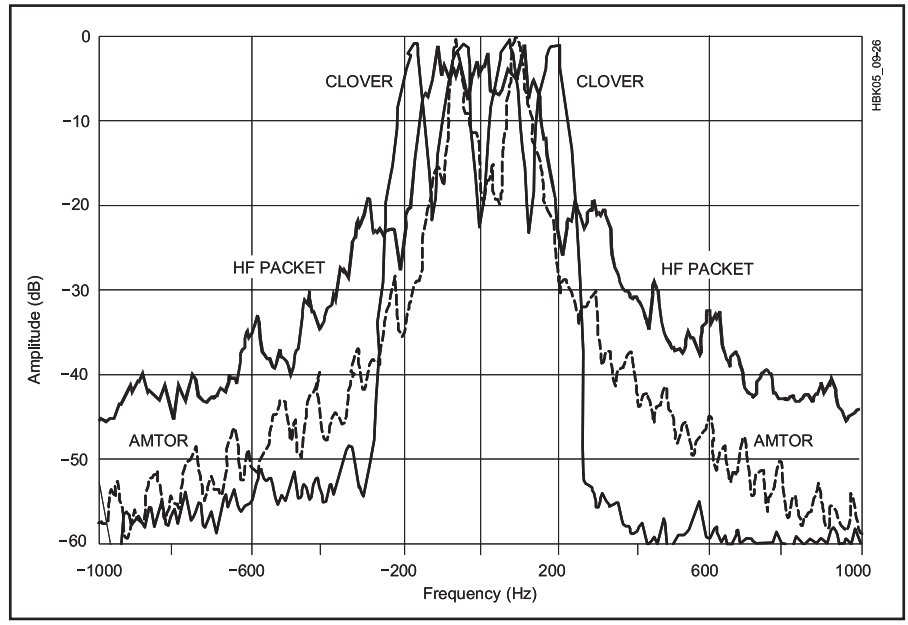

Fig 16.9 — Spectra plots of AMTOR, HF packet-radio and CLOVER-II signals.

ficiency. **Tables 16.15** and **16.16** detail the relationships between block size, coder efficiency, data bytes per block and correctable byte errors per block.

With seven different modulation formats, four data-block lengths (17, 51, 85 or 255 bytes) and four Reed-Solomon coder efficiencies (60%, 75%, 90% and 100%), there are 112 ($7 \times 4 \times 4$) different waveform modes that could be used to send data via CLOVER. Once all of the determining factors are considered, however, there are eight different waveform combinations that are actually used for FEC and/or ARQ modes.

16.5.9 CLOVER-2000

CLOVER-2000 is a faster version of CLOVER (about four times faster) that uses eight tone pulses, each of which is 250 Hz wide, spaced at 250 Hz centers, contained within the 2 kHz bandwidth between 500 and 2500 Hz. The eight tone pulses are sequential, with only one tone being present at any instant and each tone lasting 2 ms. Each frame consists of eight tone pulses lasting a total of 16 ms, so the base modulation rate of a CLOVER-2000 signal is always 62.5 symbols per second (regardless of the type of modulation being used). CLOVER-2000's maximum raw data rate is 3000 bit/s.

Allowing for overhead, CLOVER-2000 can deliver error-corrected data over a standard HF SSB radio channel at up to 1994 bit/s, or 249 characters (8-bit bytes) per second. These are the uncompressed data rates; the maximum throughput is typically doubled for plain text if compression is used. The effective data throughput rate of CLOVER-2000 can be even higher when binary file transfer mode is used with data compression.

The binary file transfer protocol used by HAL Communications operates with a terminal program explained in the HAL E2004 engineering document. Data compression algorithms tend to be context sensitive — compression that works well for one mode (say, text), may not work well for other data forms (graphics, for example). The HAL terminal program uses the PK-WARE compression algorithm, which has proved to be a good general-purpose compressor for most computer files and programs. Other algorithms may be more efficient for some data formats, particularly for compression of graphic image files and digitized voice data. The HAL Communications CLOVER-2000 modems can be operated with other data compression algorithms in the users' computers.

CLOVER-2000 is similar to the previous version of CLOVER, including the transmission protocols and Reed-Solomon error detection and correction algorithm. The original descriptions of the CLOVER Control Block (CCB) and Error Correction Block (ECB) still apply for CLOVER-2000, except for the higher data rates inherent to CLOVER-2000. Just like CLOVER, all data sent via CLOVER-2000 is encoded as 8-bit data bytes and the error-correction coding and modulation formatting processes are transparent to the data stream — every bit of source data is delivered to the receiving terminal without modification.

Control characters and special "escape sequences" are not required or used by CLOVER-2000. Compressed or encrypted data may therefore be sent without the need to insert (and filter) additional control characters and without concern for data integrity. Five different types of modulation may be used in the ARQ mode — BPSM (Binary Phase Shift Modulation), QPSM (Quadrature PSM), 8PSM (8-level PSM), 8P2A (8PSM + 2-level Amplitude-Shift Modulation) and 16P4A (16 PSM plus 4 ASM).

The same five types of modulation used in ARQ mode are also available in Broadcast (FEC) mode, with the addition of 2-Channel Diversity BPSM (2DPSM). Each CCB is sent using 2DPSM modulation, 17-byte block size and 60% bias. The maximum ARQ data throughput varies from 336 bit/s for BPSM to 1992 bit/s for 16P4A modulation. BPSM is most useful for weak and badly distorted data signals, while the highest format (16P4A) needs extremely good channels, with high SNRs and almost no multipath.

Most ARQ protocols designed for use with HF radio systems can send data in only one direction at a time. CLOVER-2000 does not need an OVER command; data may flow in either direction at any time. The CLOVER ARQ time frame automatically adjusts to match the data volume sent in either or both directions. When first linked, both sides of the ARQ link exchange information using six bytes of the CCB. When one station has a large volume of data buffered and ready to send, ARQ mode automatically shifts to an expanded time frame during which one or more 255 byte data blocks are sent.

If the second station also has a large volume of data buffered and ready to send, its half of the ARQ frame is also expanded. Either or both stations will shift back to CCB level when all buffered data has been sent. This feature provides the benefit of full-duplex data transfer but requires use of only simplex frequencies and half-duplex radio equipment. This two-way feature of CLOVER can also provide a back-channel "order-wire" capability. Communications may be maintained in this chat mode at 55 WPM, which is more than adequate for real-time keyboard-to-keyboard communications.

More information can also be found at **www.arrl.org/technical-characteristics** or in *ARRL's HF Digital Handbook* by Steve Ford, WB8IMY.

16.5.10 WINMOR

While the various PACTOR modes currently dominate and generally represent the best available performance HF ARQ protocols suitable for digital messaging, PC sound cards with appropriate DSP software can now begin to approach PACTOR performance. The WINMOR (Winlink Message Over Radio) protocol is an outgrowth of the work SCAMP (Sound Card Amateur Message Protocol) by Rick Muething, KN6KB. SCAMP put an ARQ "wrapper" around Barry Sanderson's RDFT (Redundant Digital File Transfer) then integrated SCAMP into a Client and Server for access to the Winlink message system. (More on Winlink in a later section.) SCAMP worked well on good channels but suffered from the following issues:

- The RDFT batch-oriented DLLs were slow and required frame pipelining, increasing complexity and overhead.
- RDFT only changed the RS encoding on its 8PSK multi carrier waveform to achieve a 3:1 range in speed/robustness which is not enough.
- RDFT was inefficient in Partial Frame recovery (no memory ARQ).
- RDFT was a 2.4 kHz mode and limited to narrow HF sub bands.
- SCAMP's simple multi-tone ACK/NAK did not carry session ID info, increasing chances of fatal cross session contamination.

WINMOR is an ARQ mode generated from the ground up to address the limitations of SCAMP/RDFT and leverage what was learned. Today, a viable message system (with the need for compression and binary attachments) requires true "error-free" delivery of binary data. To achieve this there must be some "back channel" or ARQ so the receiving station can notify the sender of lost or damaged data and request retransmission or repair. **Table 16.17** outlines the guidelines used in the development of WINMOR.

Perhaps the most challenging of these requirements are:

- The ability to quickly tune, lock and acquire the signal which is necessary for practical length ARQ cycles in the 2-6 s range.
- The ability to automatically adapt the modulation scheme to changing channel conditions. An excellent example of this is Pactor III's extremely wide range of speed/robustness (18:1) and is one reason it is such an effective mode in both good and poor channel conditions.

The most recent development effort has focused on 62.5 baud BPSK, QPSK and 16QAM and 31.25-baud 4FSK using 1 (200 Hz), 3 (500 Hz) and 15 (2000 Hz). With carriers spaced at twice the symbol rate. These appear to offer high throughput and

Table 16.17
WINMOR Development Guidelines

Absolute Requirements	*Desirable Requirements*
Work with standard HF (SSB) radios	Modest CPU & OS demands
Accommodate Automatic Connections	Bandwidth options (200, 500, 3000 Hz)
Error-free transmission and confirmation	Work with most sound cards/interfaces
Fast Lock for practical length ARQ cycles	Good bit/s/Hz performance ~ P2 goal
Auto adapt to a wide range of changing channel conditions	Efficient modulation and demodulation for acceptable ARQ latency
Must support true transparent binary to allow attachments and compression	Selective ARQ & memory ARQ to maximize throughput and robustness.
Must use loosely synchronous ARQ timing to accommodate OS and DSP demands	Near Pactor ARQ efficiency (~70% of raw theoretical throughput)

robustness especially when combined with multi-level FEC coding.

WINMOR uses several mechanisms for error recovery and redundancy.

1) FEC data encoding currently using:

• 4,8 Extended Hamming Dmin = 4 (used in ACK and Frame ID)

• 16-bit CRC for data verification

Two-level Reed-Solomon (RS) FEC for data:

• First level Weak FEC, for example RS 140,116 (corrects 12 errors)

• Second level Strong FEC, for example RS 254,116 (corrects 69 errors)

2) Selective ARQ. Each carrier's data contains a Packet Sequence Number (PSN). The ACK independently acknowledges each PSN so only carriers with failed PSNs get repeated. The software manages all the PSN accounting and re-sequencing.

3) Memory ARQ. The analog phase and amplitude of each demodulated symbol is saved for summation (phasor averaging) over multiple frames. Summation is cleared and restarted if max count reached. Reed-Solomon FEC error decoding done after summation.

4) Multiple Carrier Assignment (MCA). The same PSN can be assigned to multiple carriers (allows tradeoff of throughput for robustness). Provides an automatic mechanism for frequency redundancy and protection from interference on some carriers.

5) Dynamic threshold adjustment (used on QAM modes) helps compensate for fading which would render QAM modes poor in fading channels.

In trying to anticipate how WINMOR might be integrated into applications they came up with a "Virtual TNC" concept. This essentially allows an application to integrate the WINMOR protocol by simply treating the WINMOR code as just another TNC and writing a driver for that TNC. Like all TNCs there are some (<10) parameters to set up: call sign, timing info, sound card, keying mechanism, etc. A sample image of the virtual TNC appears in **Fig 16.10**.

The WINMOR software DLL can even be made to appear as a physical TNC by "wrapping" the DLL with code that accesses it through a virtual serial port or a TCP/IP port. Like a physical TNC WINMOR has a "front panel" with flashing lights. But since operation is automatic with no front panel user interaction required the WINMOR TNC can be visible or hidden.

WINMOR looks promising and the testing to date confirms:

• Sound card ARQ is possible with a modern CPU and OS while making acceptable CPU processing demands. (CPU Loading of < 20% on a 1.5 GHz Celeron/Win XP)

• Throughput and robustness can be adjusted automatically to cover a wide range of bandwidth needs and channel conditions. (10:1 bandwidth range, 57:1 throughput range)

• ARQ throughput in excess of 0.5 bit/s/Hz is possible in fair to good channels (0.68 - 0.82 bit/s/Hz measured)

• Good ARQ efficiency — 70-75%

• Throughput is currently competitive with P2 and P3 and significantly better than P1

More information about WINMOR will be available as it develops at **www.winlink.org/WINMOR**.

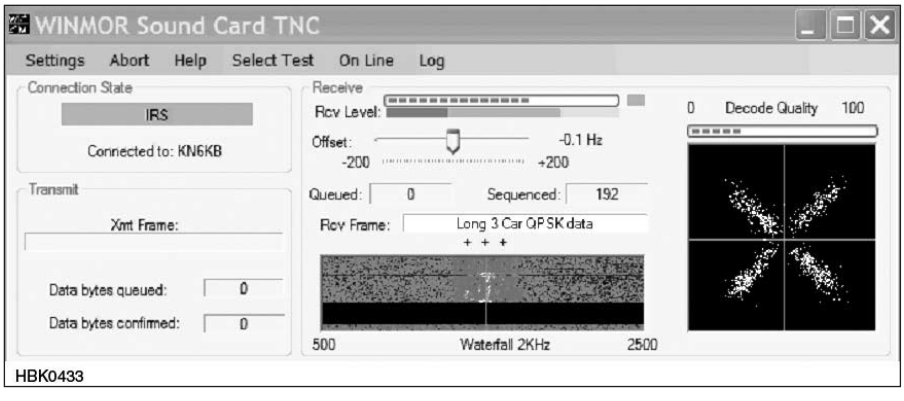

Fig 16.10 — WINMOR sound card TNC screen.

16.5.11 Packet Radio

Amateur packet radio began in Canada after the Canadian Department of Communications permitted amateurs to use the mode in 1978. The FCC permitted amateur packet radio in the US in 1980. In the first half of the 1980s, packet radio was the habitat of a small group of experimenters who did not mind communicating with a limited number of potential fellow packet communicators. In the second half of the decade, packet radio took off as the experimenters built a network that increased the potential number of packet stations that could intercommunicate and thus attracted tens of thousands of communicators who wanted to take advantage of this potential.

Packet radio provides error-free data transfer via the AX.25 protocol. The receiving station receives information exactly as the transmitting station sends it, so you do not waste time deciphering communication errors caused by interference or changes in propagation. This simplifies the effort required to build other protocols or applications on top of packet technology since much of the data transfer work is handled seamlessly by the TNC in the lower layers. There are a number of other advantages to the packet radio design:

• Packet uses time efficiently, since packet bulletin-board systems (PBBSs) permit packet operators to store information for later retrieval by other amateurs.

• Packet uses the radio spectrum efficiently, since one radio channel may be used for multiple communications simultaneously, or one radio channel may be used to interconnect a number of packet stations to form a "cluster" that provides for the distribution

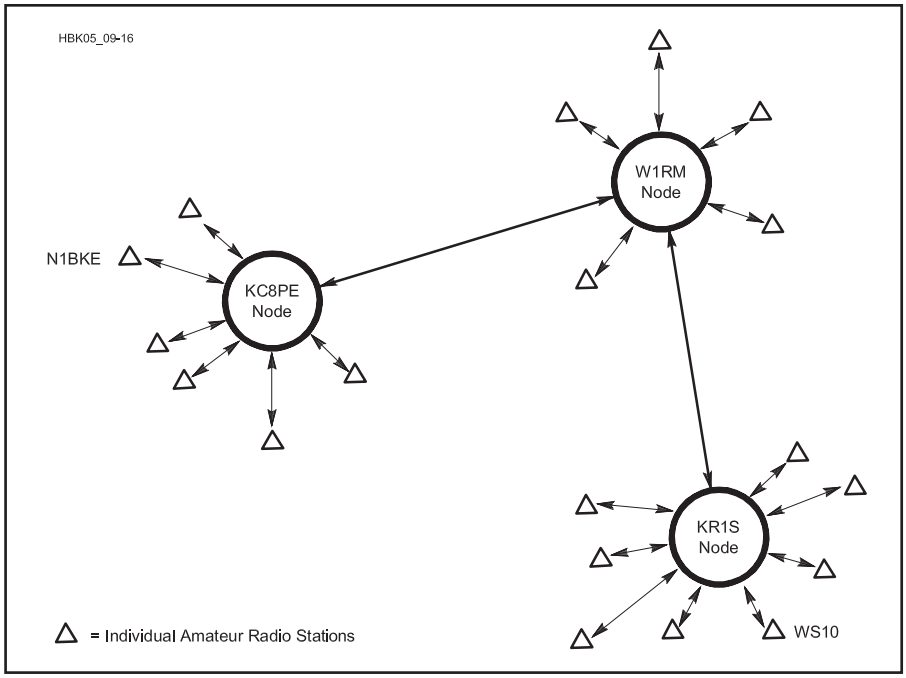

Fig 16.11 — DX spotting clusters (based on *PacketCluster* software) are networks comprised of individual nodes and stations with an interest in DXing and contesting. In this example, N1BKE is connected to the KC8PE node. If he finds a DX station on the air, he'll post a notice — otherwise known as a spot — which the KC8PE node distributes to all its local stations. In addition, KC8PE passes the information along to the W1RM node. W1RM distributes the information and then passes it to the KR1S node, which does the same. Eventually, WS1O — who is connected to the KR1S node — sees the spot on his screen. Depending on the size of the network, WS1O will receive the information within minutes after it was posted by N1BKE. Many such networks can also be found on the web or are accessible by the use of *TELNET* software.

of information to all of the clustered stations. The DX PacketCluster nodes are typical examples (see **Fig 16.11**). Each local channel may be connected to other local channels to form a network that affords interstate and international data communications. This network can be used by interlinked packet bulletin-board systems to transfer information, messages and third party traffic via HF, VHF, UHF, satellite and the Internet.

• Packet uses other stations efficiently, since any packet-radio station can use one or more other packet-radio stations to relay data to its intended destination.

• Packet uses current station transmitting and receiving equipment efficiently, since the same equipment used for voice communications may be used for packet communications. The outlay for the additional equipment necessary to add packet capability to a voice station may be less than $100.

DIGIPEATERS

A *digipeater* is a packet-radio station capable of recognizing and selectively repeating packet frames. An equivalent term used in the network industry is *bridge*. Virtually any TNC can be used as a single-port digipeater, because the digipeater function is included in the AX.25 Level 2 protocol firmware. The digipeater function is handy when you need a relay and no node is available, or for on-the-air testing. Digipeaters are used extensively with APRS (Automatic Packet Reporting System) covered below.

TCP/IP

Despite its name, TCP/IP (Transmission Control Protocol/Internet Protocol) is more than two protocols; it's actually a set of several protocols. TCP/IP provides a standardized set of protocols familiar to many and compatible with existing network technologies and applications. TCP/IP has a unique solution for busy networks. Rather than transmitting packets at randomly determined intervals, TCP/IP stations automatically adapt to network delays as they occur. As network throughput slows down, active TCP/IP stations sense the change and lengthen their transmission delays accordingly. As the network speeds up, the TCP/IP stations shorten their delays to match the pace. This kind of intelligent network sharing virtually guarantees that all packets will reach their destinations with the greatest efficiency the network can provide.

With TCP/IP's adaptive networking scheme, you can chat using the TELNET protocol with a ham in a distant city and rest assured that you're not overburdening the system. Your packets simply join the constantly moving "freeway" of data. They might slow down in heavy traffic, but they will reach their destination eventually. This adaptive system is used for all TCP/IP packets, no matter what they contain.

TCP/IP excels when it comes to transferring files from one station to another. By using the TCP/IP File Transfer Protocol (FTP), you can connect to another station and transfer computer files — including software. As might be imagined, transferring large files can take time. With TCP/IP, however, you can still send and receive mail (using the SMTP protocol) or talk to another ham while the transfer is taking place.

When you attempt to contact another station using TCP/IP, all network routing is performed automatically according to the TCP/IP address of the station you're trying to reach. In fact, TCP/IP networks are transparent to the average user. To operate TCP/IP, all you need is a computer, a 2 meter FM transceiver and a TNC with KISS (keep it simple, stupid) capability. As you might guess, the heart of your TCP/IP setup is software. The TCP/IP software set was written by Phil Karn, KA9Q, and is called NOSNET or just NOS for short. There are dozens of NOS derivatives available today. All are based on the original NOSNET. NOS takes care of all TCP/IP functions, using your "KISSable" TNC to communicate with the outside world. The only other item necessary is your own IP address in Network 44, termed AMPRNet (AMateur Packet Radio Network). Individual IP Address Coordinators assign addresses to new TCP/IP users in the 44.x.x.x subnet based on physical location worldwide. Your local coordinator can be found on AMPR.org on the list at **http://noh.ucsd.edu/~brian/amprnets.txt**.

More packet information can be found in the annual ARRL/TAPR Digital Communications Conference proceedings and on the TAPR (Tucson Amateur Packet Radio) website at **www.tapr.org**. TAPR also maintains the AX.25 protocol specification at **www.tapr.org/pub_ax25.html**.

16.5.12 APRS

APRS (**http://aprs.org**) was developed by Bob Bruninga, WB4APR, for tracking and digital communications with mobile GPS equipped stations with two-way radio. APRS is different from regular packet in four ways:

• Integration of maps and other data displays to organize and display data

• By using a one-to-many protocol to update everyone in real time

• By using generic digipeating so that prior knowledge of the network is not required

• Provides worldwide transparent Internet

backbone, linking everyone worldwide

APRS turns packet radio into a real-time tactical communications and display system for emergencies, public service applications and global communications. Normal packet radio has shown usefulness in passing bulk message traffic (e-mail) from point to point. It has been difficult to apply conventional packet to real time events where information has a very short life time and needs to get to everyone.

The APRS network consists of five types of APRS stations:

- Basic — Full transmit and receive capability
- Tracker — Transmit-only device (portable or weather station)
- RELAY — Provides basic digipeating
- WIDE — Dedicated digipeaters with specific coverage areas
- IGate — Internet gateways to repeat APRS packets to servers on the Internet

APRS stations are generally set to beacon their position and other information at frequent intervals. Fixed stations don't require a GPS and should beacon infrequently to avoid crowding the channel with needless reports. Moving mobile stations can receive their coordinates from a GPS and should report more frequently. More sophisticated APRS devices support "smart beaconing" and "corner pinning" where a station will beacon more frequently when it moves faster or automatically beacon when changing direction more than a predefined angle. An APRS beacon will generally contain a call sign with *Secondary Station ID* (*SSID*), position report (lat/long), heading, speed, altitude, display icon type, status text and routing information. It may also contain antenna/power information, weather data or short message data.

Call signs generally include an SSID that historically provided information about the type of station. For example, a call sign of N7SS-9 would generally indicate a mobile station. Since there are several systems in use, the SSID information is not definitive. Recently a number of dedicated APRS radios have appeared that feature an internal TNC and APRS software. These radios include a method of displaying and entering messages without requiring the use of a computer meaning their user can directly respond to a message sent. There is value in knowing someone can respond and it is recommended they use an SSID of -7. Tactical call signs can also be used if the assigned call sign is included as part of the status text.

The APRS packet includes PATH information that determines how many times it should be digipeated. A local group should be able to offer guidance on how the PATH should be set to not overload the local network. Listening to squawking signals on the national APRS frequency of 144.390 MHz will provide some idea how busy the channel is. In much of the country the APRS network is well developed and a beacon doesn't need more than a couple hops to cover a wide area and find an IGate.

APRS SOFTWARE

There is APRS software available for most any platform with varying levels of support. Some of the most common are DOSAPRS (DOS), WinAPRS / APRS+SA / APRSPoint / UI-View (*Windows*), MacAPRS (Macintosh), PocketAPRS (Palm devices), APRSce (*Windows* CE devices) and Xastir / X-APRS (*Linux*). Because APRS beacons find their way to the Internet via IGates, it is also possible to track station or view the network online. There is a good network view available at **http://aprs.fi** and individual station information can be found at **http://map.findu.com/call sign**. The Findu.com site also offers historical weather data, a message display and a large list of other queries.

APRS INTEGRATION WITH OTHER TECHNOLOGIES

APRS is very similar to the *AIS* (*Automatic Identification System*) used to track ships in coastal waters. The AIS transponder equipment is becoming more common in commercial ships and private vessels. The AIS information can be displayed simultaneously with APRS data on the aprs.fi website. More information about AIS can be found at **http://en.wikipedia.org/wiki/Automatic_Identification_System**.

APRS RF networks worldwide are interconnected via APRS-IS. APRS-IS is an ad hoc network of Amateur Radio servers, gateways (IGates), and clients (display software). APRS-IS facilitates world-wide messaging without having to define special paths. Because all clients have access to all gated packets, databases such as **http://aprs.fi** and **http://findu.com** can store and parse the packets for later retrieval from browsers and other clients. APRS-IS messaging support allows many interfaces for the RF ham running an APRS client such as call sign lookups, e-mail, and calling CQ world-wide. More information on many of these features can be found at **www.aprs-is.net**.

APRS position reports are also available from GPS-equipped D-STAR radios via DPRS gateways (more on D-STAR in a later section). The DPRS specification defines how to translate the continuous stream of GPS position reports in the D-STAR DV stream to individual APRS packets. This definition restricts the number of APRS packets generated to prevent overrunning a local RF network because the D-STAR radios continuously stream new position reports while the user is talking. Most D-STAR repeaters have DPRS IGates running on the gateway providing the D-STAR radio positions to APRS-IS. Those positions can be gated to local RF by APRS IGates if the APRS IGate sysop enables this feature.

Keith Sproul, WU2Z, operates an e-mail gateway that allows APRS messages to be converted to short e-mail or text messages. When entering a message, E-MAIL is used for the destination address. The message body must then contain the actual e-mail address followed by a space and very short message. If the message is picked up by an IGate it will be sent to the WU2Z mail server and out to the recipient, with a confirmation APRS message. For dozens of other messaging options, check online at **http://aprs.org/aprs-messaging.html**.

There is connectivity between APRS and the Winlink system via APRSLink. APRSLink monitors all APRS traffic gated to the Internet, worldwide, and watches for special commands that allow APRS users to:

- Read short e-mail messages sent to their call sign@winlink.org
- Send short e-mail messages to any valid e-mail address or Winlink 2000 user
- Perform e-mail related maintenance (see commands below)
- Be notified of pending Winlink e-mail via APRS message
- Query APRSLink for information on the closest Winlink RMS packet station

Attention must be paid to the APRS traffic generated when using these features but they can be very handy. Details on the APRSLink system are available at **www.winlink.org/aprslink**.

16.5.13 Winlink 2000

Winlink 2000 or *WL2K* is a worldwide radio messaging system that takes advantage of the Internet where possible. It does this in order to allow the end-user more radio spectrum on the crowded spectrum. The system provides radio interconnection services including: e-mail with attachments, position reporting, graphic and text weather bulletins, emergency / disaster relief communications, and message relay. The PACTOR-I, II, and III protocols are used on HF, and AX.25 Packet, D-STAR and 802.11 are used on VHF/UHF.

Winlink 2000 has been assisting the maritime and RV community around the clock for many years. More recently there has been an increasing interest in emergency communications (emcomm), and the Winlink 2000 development team has responded by adding features and functions that make the system more reliable, flexible and redundant. The role of Winlink 2000 in emergency communications is to supplement existing methodologies to add another tool in the toolkit of

the volunteer services deploying emergency communications in their communities.

The Winlink 2000 system is a "star" based network containing five mirror-image, redundant Common Message Servers (CMS) located in San Diego (USA), Washington DC (USA), Vienna (Austria), Halifax (Canada) and Perth (Australia). These ensure that the system will remain in operation should any piece of the Internet become inoperative. Each Radio Message Server node (RMS) is tied together as would be the ends of a spoke on a wheel with the hub function performed by the Common Message Servers. Traffic goes in and out between the CMS and the Internet e-mail recipient, and between the end users and the Radio Message Server (RMS) gateways. Multiple radio-to-radio addresses may be mixed with radio-to-Internet e-mail addresses, allowing complete flexibility.

Because each Radio Message Server gateway is a mirror image of the next, it does not matter which station is used. Each can provide over 700 text-based or graphic weather products, and each can relay the user's position to a web-based view of reporting users and interoperates with the APRS system.

One of the most important objectives in the eyes of the Winlink development team was to reduce the use of the HF spectrum to only that required to exchange messages with a user, and to do that at full "machine" speeds. The HF spectrum is very crowded, and limiting the forwarding of messages between WL2K RMS stations to the Internet, a great deal of radio air time is eliminated, making the time and spectrum available to individual users either for message handling or for other operations. A number of the RMS PACTOR servers found on HF restrict their protocols to PACTOR II (400 to 800 bit/s) and PACTOR III (1400 to 3600 bit/s.) In doing so, these RMS stations typically have a much higher ratio of traffic minutes and message counts to connect times than do the RMS stations that also receive the slower PACTOR I (100 to 200 bps) protocol. In other words, the amount of traffic that is passed with an Express station is much greater for an equivalent amount of connect time with approximately the same number of connections. On average, this translates to a PACTOR I station downloading an 80,000 byte file in approximately 80 minutes while on PACTOR III, the same download takes approximately six minutes. Note that PACTOR IV is not yet legal for US amateurs as discussed earlier in the PACTOR section.

The RMS servers provide endpoint connectivity to users via HF or VHF and can be found worldwide. When connecting via the Internet, users can connect directly to the CMS server operational closest to them via TELNET. Formerly, the full-featured message servers on HF were known as PMBOs (Participating Mail Box Offices) and there was a packet radio to TELNET bridge component for VHF/UHF called Telpac. These have been replaced with the newer RMS PACTOR (HF) and RMS Packet (VHF/UHF) software although the older terms are still in use. Over the air, PACTOR is used on HF, AX.25 packet on VHF/UHF and both employ the B2F compressed binary format to maximize transmission efficiency. A diagram of the Winlink 2000 architecture is shown in **Fig 16.12**.

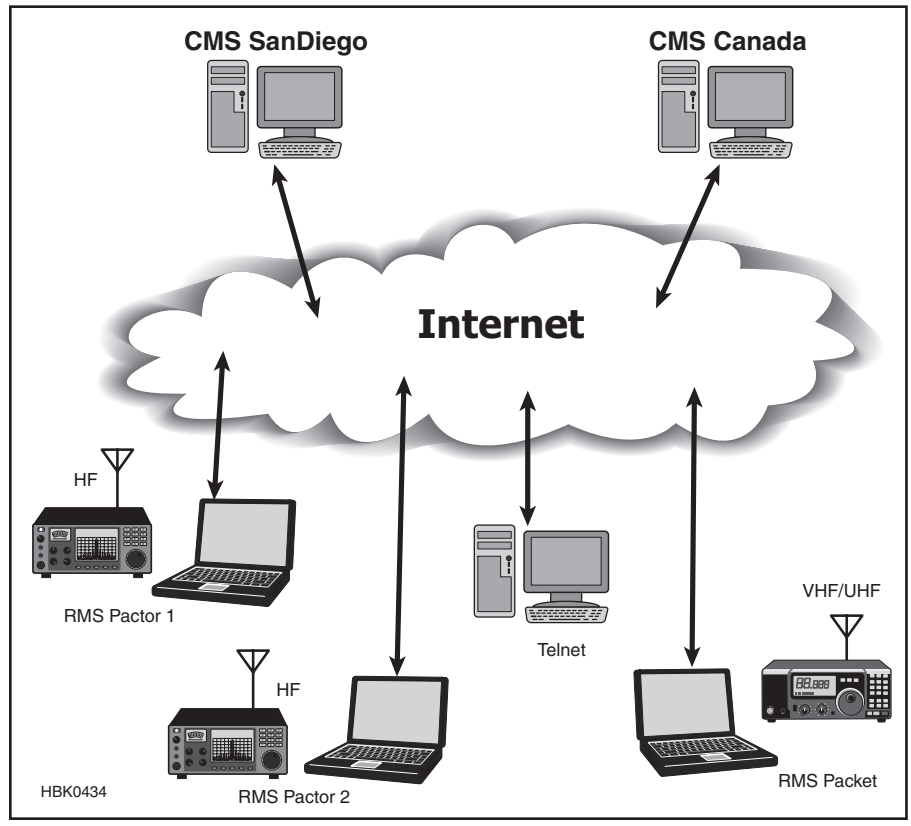

Fig 16.12 — Winlink 2000 system.

CLIENT ACCESS TO WINLINK

The primary purpose of the Winlink 2000 network system is to assist the mobile or remotely located user and to provide emergency e-mail capabilities to community agencies. Because of this, WL2K supports a clean, simple interface to the Internet SMTP e-mail system. Any message sent or received may include multiple recipients and multiple binary attachments. The radio user's e-mail address, however, must be known to the system as a radio user or the message will be rejected. This simple Internet interface protocol has an added benefit in case of an emergency where local services are interrupted and the system must be used by non-Amateur groups as an alternative to normal SMTP e-mail.

Connecting to any one of the WL2K publicly-used RMS Packet stations via HF or the specialized non-public emcomm RMS stations, can immediately and automatically connect a local amateur station to the Internet for emergency traffic. Using a standard SMTP e-mail client, the *Paclink* mini-e-mail server can replace a network of computers (behind a router) as a transparent substitute for normal SMTP mail. WL2K uses no external source for sending or receiving Internet e-mail. It is a stand-alone function which interacts directly with the Internet rather than through any external Internet service provider.

Airmail is independently developed, distributed and supported by Jim Corenman, KE6RK. It is the oldest and most widely used program for sending and receiving messages using the WL2K system. Airmail may be used for HF Pactor, VHF/UHF Packet, and for TELNET connections over any TCP/IP medium including the Internet and high-speed radio media, such as D-STAR and HSMM. Once connected to a WL2K station, message transfer is completely automatic. On the ham bands, Airmail can transfer messages automatically with any station supporting the BBS or F6FBB protocols, such as Winlink 2000, F6FBB, MSYS and other Airmail stations. When used with WL2K, Airmail also contains position reporting capabilities, and a propagation prediction program to determine which of the participating Winlink stations will work from anywhere on Earth. Airmail also contains a limited mail server that allows it to host e-mail

independently from the Winlink network.

To obtain a copy of Airmail, including the installation and operating instructions, download the program from the Airmail web page at **www.siriuscyber.net/ham** and support is available from the Airmail user group at **http://groups.yahoo.com/group/airmail2000**.

When an RF connection is not available (or necessary) but web access is available, Winlink 2000 messages can be retrieved or sent via a web interface. The web browser access is limited to text-based messages without the use of bulletins or file attachments. Password-protected access to the Winlink mailbox is available through the Winlink page at **www.winlink.org/webmail**. There is also a terminal mode for interactive keyboard commands, allowing a terminal rather than computer-based software to connect to an RMS via RF. Because of the inefficient use of airtime, this method is discouraged but may be used for the listing and deletion of messages only.

Airmail provides a super-fast replica of WL2K radio operations while directly connected through the Internet to one of the CMS servers. This method of obtaining messages over the Internet allows multiple attachments, catalog bulletins, and all other Winlink 2000 services normally available over radio channels, but at Internet speeds. In order to use this service, a user must currently be listed as a radio user, and obtain the connection information for the closest CMS server. Both Paclink and Airmail support the TELNET client service. This operation allows regular use of the system with the same software configuration at high speed, without using RF bandwidth and provides a full-featured mechanism to access the system where no RF connection is available. An Emergency Operations Center with Internet access can use the TELNET path with RF as a fallback with just a minor change in the software. Similarly, an RV or marine user can use the software from an Internet café or home via TELNET and switch the software back to the TNC when mobile, with no change in functionality.

WINLINK FEATURES

To address the needs of mobile users for near real-time data, WL2K uses an "on-demand" bulletin distribution mechanism. (Note that such bulletins are not the same as traditional AX.25 Packet Bulletins.) Users must first select requested bulletins from an available "catalog" list managed in Airmail. When bulletin requests are received by an RMS station, a fresh locally cached copy of the requested bulletin is delivered. If no fresh locally-cached version is available, the RMS accesses the Internet and finds the bulletin which is then downloaded to the RMS and then sent to the user. The global catalog currently includes over 700 available weather, propagation, and information bulletins, including, instructions for using the system, World news, and piracy reports. All WL2K RMS stations support a single global catalog which ensures users can access any bulletin from any RMS. Bulletins can contain basic text, graphic fax or satellite images, binary or encoded files like GRIB or WMO weather reports. Local processing is used to re-process images to sizes suitable for HF Pactor transmission. The system prevents bulletin duplication and automatically purges obsolete time-sensitive weather bulletins and replaces them with the current version.

The system also has the ability to contain bulletins with attachment information which is local to each RMS. This is especially useful for the non-public emcomm RMS which may house valuable procedural information pertinent to complex information or instruction needed by specific agencies in any community emergency.

Multiple binary or text-based file attachments of any type or number may be attached to a message by simply selecting the file to be sent from a *Windows* selection dialog in the user's HF AirMail or the Winlink 2000 VHF/UHF Paclink server with a standard SMTP e-mail client. E-mail message attachments sent through the Winlink 2000 system must be limited in size. Users are provided an option to allow this limit to be determined. When using the default B2F format, the protocol chosen by the user usually determines the file size of an attachment. A user may also turn off the ability to receive file attachments. Certain file attachment types are blocked from the system for the protection of the user from virus attacks.

The WL2K network administrator may post notices that are delivered to all individual WL2K users as a private message. This is a valuable tool for notifying users of system changes, outages, software upgrades, emergencies, etc.

The integrated nature of the system makes possible other services beyond just simple messaging. The bulletin services mentioned above is beyond normal messaging, but WL2K also provides rapid position reporting from anywhere in the world. This facility is interconnected with the APRS, ShipTrak, and YotReps networks. It supports weather reporting from cruising yachts at sea and an interconnection with the YotReps network which is used by government forecasters for weather observations in parts of the world where no others are available. It also allows the maritime user to participate in the National Weather Service's NOAA MAROB a voluntary marine observation reporting program.

To ensure equitable access to the system individual users are assigned daily time limits on HF frequencies by RMS sysops. The default time per any 24-hour period is 30 minutes, however, the user may request more time from the RMS sysop should it be needed. The time limit is individual to each RMS station. Utilization of the PACTOR-II and PACTOR-III protocols are a great timesaver, allowing the user up to 18 times the volume of messages over that of PACTOR-I for the same period of time.

The system has a number of other secondary features to help keep it healthy. Extensive traffic reports are collected, the state of individual RMS stations are monitored and reported if it becomes inactive and daily backups are performed automatically at all RMS stations as well as the Common Message Servers to ensure the system integrity. Security is ensured through the vigorous updating of virus definitions and automatic virus screening for all Internet mail and files. The system has the ability to block any user by both radio (by frequency band) and Internet (by e-mail address) to prevent abuse of the system. Spam is controlled through the use of a secure "acceptance list" or "white list" methodology. More information about the Winlink 2000 system is available on the Winlink website at **www.winlink.org**.

16.5.14 D-STAR

D-STAR (Digital Smart Technologies for Amateur Radio) is a digital voice and data protocol specification developed as the result of research by the Japan Amateur Radio League (JARL) to investigate digital technologies for Amateur Radio in 2001. While there are other digital on-air technologies being used by amateurs that were developed for other services, D-STAR is one of the first on-air protocols to be widely deployed and sold by a major radio manufacturer that is designed specifically for amateur service use. D-STAR transfers both voice and data via a data stream over the 2 meter (VHF), 70 cm (UHF) and 23 cm (1.2 GHz) Amateur Radio bands either simplex or via repeater.

One of the interesting features about the D-STAR protocol is the fact the system uses Amateur Radio call signs not only as an identifier, but also for signal routing. In the most common configuration, the most vital part of a D-STAR system is the gateway server, which networks a single system into a D-STAR network via a *trust server*. The trust server provides a central, master database to look up users and their associated system. This allows Amateur Radio operators to respond to calls made to them, regardless of their location on the D-STAR network. While almost all documentation references the Internet as the connection point for a network, any IP network connectivity will work, depending on signal latency.

Additionally, D-STAR can provide a zero infrastructure support system utilizing point-to-point "backbone" 10 GHz connections. Currently, the global D-STAR trust server is maintained by a group of dedicated D-STAR enthusiasts from Dallas, Texas — the Texas Interconnect Team.

The D-STAR protocol specifies two modes; Digital Voice (DV) and Digital Data (DD). In the protocol, the DV mode provides both voice and low speed data channel on 2 meters, 70 cm and 23 cm over a 4800-bit/s data stream. In the protocol, the DV mode uses a data rate of 4800 bit/s. This data stream is broken down to three main packages: voice, forward error correction (FEC) and data. The largest portion of the data stream is the voice package, which is a total of 3600 bits/s with 1200 bit/s dedicated to forward error correction, leaving 1200 bit/s for data. This additional data contains various data flags as well as the data header, leaving about 950 bit/s available for either GPS or serial data. This portion of the data stream does not provide any type of error correction, which has been overcome by implementing error correction in the application software.

While there are various techniques of encoding and transporting a DV signal, the focus of D-STAR's design was the most efficient way to conserve RF spectrum. While D-STAR's "advertised" occupied bandwidth is 6.25 kHz, tests reveal a band plan of 10 kHz spacing is adequate to incorporate the D-STAR signal as well as provide space for channel guards.

In addition to DV mode, the D-STAR protocol outlines the high speed Digital Data (DD) mode. This higher speed data, 128 kbit/s, is available only on the 23 cm band because it requires an advertised 130 kHz bandwidth, only available at 23 cm in world-wide band plans. Unlike the DV mode repeaters, the DD mode module operates as an "access point" operating in half duplex, switching quickly on a single channel.

As with the DV mode, there is a portion of the data stream used for signal identification with the data header as well as various system flags and other D-STAR related items. Once this portion of the data stream is taken into consideration, the 128 kbit/s is reduced to approximately 100 kbit/s — still more than double a dial-up connection speed with significant range. Another consideration is the data rate specified at 128 kbit/s is the gross data rate. Therefore, the system developers are challenged by the area coverage/potential user issue. This means the higher the elevation of the system, the more potential users and the slower the system will become as all the users split the data bandwidth. Finally, there is an issue from the days of packet radio. While technically, the opportunity for "hidden transmitter" issues does exist and collisions do occur, the TR switching is very fast and this effect is handled by TCP/IP as it is for WiFi access points.

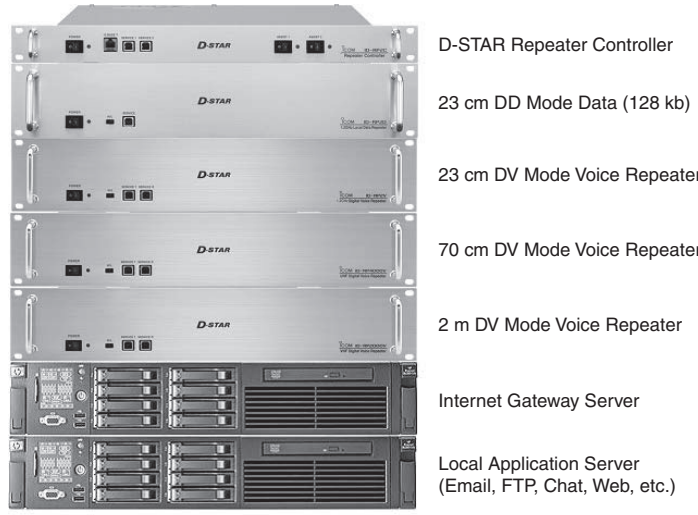

Fig 16.13 — Full D-STAR system.

The simplex channel eliminates the need for duplexers at a repeater site if only the DD mode system is installed. It is still recommended to have filtering, such as a band-pass filter, in place to reduce possible interference from other digital sources close to the 23 cm band as well as to reduce RF overload from nearby RF sources. While some DD mode system owners would like more sensitivity or more output power (10 W), at the time of print, no manufacturer has developed pre-amps or RF power amplifiers with an adequate TR switching time to boost the signals.

Radios currently providing DV mode data service use a serial port for low-speed data (1200 bit/s), while the DD mode radio offers a standard Ethernet connection for high-speed (128 kbit/s) connections, to allow easy interfacing with computer equipment. The DD mode Ethernet jack allows two radios to act as an Ethernet bridge without any special software support required. This allows standard file sharing, FTP, TELNET, HTTP/web browsing, IRC chat or even remote desktop connections to function as if connected by wire.

In a Gateway configuration, *all* users must be registered in the network. This provides the DD mode sysop a layer of authorization, meaning that if someone wants to use a DD mode system, and they have not received authorization to use the gateway, their DD mode access will be denied. Any gateway registered user, on the common network, can use any DD mode system, even if the registration was not made on that system. While we are not able to use encryption in the Amateur Radio service, security can be implemented in standard software or consumer routers and firewalls.

A D-STAR repeater system consists of at least one RF module and a controller. While any combination of RF modules can be installed, typically a full system includes the three voice modules (2 meters, 70 cm and 23 cm) and the 23 cm DD mode module shown in **Fig 16.13**. A computer with dual Ethernet ports, running the Gateway software, is required for Internet access to the global network. An additional server, as shown in the diagram, can be incorporated for local hosting of e-mail, chat, FTP, web and other services. In a D-STAR system installation, the standard repeater components (cavities, isolators, antennas and so forth) are not shown but are required as with any analog system. Some groups have removed analog gear and replaced it with D-STAR components on the same frequency with no additional work beyond connecting the power and feed lines. More information about D-STAR systems may be found in the **Repeaters** chapter and in the **Digital Communications** supplement on the *Handbook* CD.

USER-CREATED FEATURES AND TOOLS

D-STAR has already benefitted from a strong user community that has been developing applications, products and upgrades at a rapid pace. The *DPlus* gateway add-on provides a number of new features, such as an echo test, voicemail, simulcast to all modules, a linking feature, playback of voice or text messages and more. There are also a number of "D-STAR Reflectors" around the world that act like conference bridges for connecting multiple repeater systems.

One of the other projects that came from

the Open D-STAR project is the DV Dongle by Robin Cutshaw, AA4RC. The DV Dongle contains the DSVI AMBE codec chip and a USB interface to allow a computer user to talk with other D-STAR voice users much the way EchoLink users can connect to the system with a PC. The DV Tool software is written in Java to provide cross-platform support. The DV Dongle is available at ham radio dealers and the latest software and manual can be downloaded from **www.opendstar.org/tools**.

Pete Loveall, AE5PL, created a DPRS-to-APRS gateway that moves D-STAR GPS data to the APRS-IS network. Although it doesn't natively appear on the APRS RF network (unless IGated specifically), it does appear in all the online APRS tools. DPRS has been implemented in software with *DStarMonitor* running on many D-STAR gateways and *DPRS Interface* available for most computers at **www.aprs-is.net/dprs**. DPRS has been implemented in hardware with the µSmartDigi manufactured by Rich Painter, ABØVO. The µSmartDigi offers a compact, portable TNC designed to act as a gateway for position packets between a D-STAR digital network and a conventional analog APRS RF network via a D-STAR radio and a conventional FM radio. More DPRS and µSmartDigi information is available at **www.aprs-is.net/dprs** and **www.usmartdigi.com** respectively.

Satoshi Yasuda, 7M3TJZ/AD6GZ, created a D-STAR node adapter or "hot spot" that connects to a standard simplex FM radio and allows a D-STAR radio user to access the D-STAR network. The system requires the FM rig be accurately configured for a clean signal and loses the D-STAR benefit of only consuming 6.25 kHz of bandwidth. It does allow D-STAR users access to the network via RF where there is Internet access and no repeater infrastructure available. More information can be found online at **http://d-star.dyndns.org/rig.html.en**.

Brian Roode, NJ6N, wrote one of the earliest end-user applications for D-STAR. His *d*Chat* is a *Windows*-based keyboard to keyboard communication application that uses the D-STAR DV mode data capability. *d*Chat* is used to enable text-based communication between multiple stations on a simplex frequency or through a repeater. More information can be found online at **http://nj6n.com/dstar/dstar_chat.html**.

Dan Smith, KK7DS developed D-RATS with public service users in mind. In addition to chat, D-RATS supports file/photo downloads bulletins, forms, email and a sophisticated mapping capability. It also has a "repeater" functionality that allows sharing a D-STAR radio data stream over a LAN. The repeater feature is interesting in an environment where the radio would be physically separated from the software user, like in an emergency operations center. Dan has current developments and downloads available at **www.d-rats.com/wiki**.

D-STAR TV was developed by John Brown, GM7HHB, and provides still images over D-STAR DV mode and streaming video over DD mode. The limited bandwidth of DV mode data and the lack of error correction mean the images take a while and may have some corruption (just like SSTV). The DD mode has much more bandwidth and can support live video. More information is available online at **www.dstartv.com**.

This is just a sample of the development taking place and there will undoubtedly be more applications coming as the user base increases. The two websites with the most current D-STAR information are **www.dstarinfo.com** and **www.dstarusers.org**.

16.5.15 APCO Project 25 (P25)

Project 25 (P25) or APCO-25 is a suite of standards for digital radio communications for use by public safety agencies in North America. P25 was established by Association of Public-Safety Communications Officials (APCO) to address the need for common digital public safety radio communications standards for first responders and Homeland Security/Emergency Response professionals. In this regard, P25 fills the same role as the European Tetra protocol, although not interoperable with it.

P25 has been developed in phases with Phase 1 completed in 1995. P25 Phase 1 radio systems operate in 12.5 kHz analog, digital or mixed mode. Phase 1 radios use Continuous 4-level FM (C4FM) modulation for digital transmissions at 4800 baud and two bits per symbol, yielding 9600 bit/s total channel throughput. In the case of data transmission, data packets basically consist of a header, containing overhead information, followed by data. In the case of digitized voice transmission, after the transmission of a header containing error protected overhead information, 2400 bit/s is devoted to periodically repeating the overhead information needed to allow for late entry (or the missed reception of the header).

After evaluating several candidates, Project 25 selected the IMBE (Improved MultiBand Excitation) vocoder from DVSI for phase 1, operating at 4400 bit/s. An additional 2800 bit/s of forward error correction is added for error correction of the digitized voice. An 88.9-bit/s low-speed data channel is provided in the digitized voice frame structure and several forms of encryption are supported.

P25 Phase 2 requires a further effective bandwidth reduction to 6.25 kHz by operating as two-slot TDMA (Time Division Multiple Access) with support for trunking systems. This provides two channels from the same 12.5 kHz spectrum when working through a repeater to provide time synchronization. The TDMA timing requirements does limit range to a published 35 km but this is plenty for public safety users. It uses the newer AMBE (Advanced MultiBand Excitation) codec from DVSI to accommodate the bit rate reduction to 4800 bit/s. Phase 2 radios will still offer backward compatibility with Phase 1. Phase 3 work has started and will support access to the new 700 MHz band in the US.

P25 radios offer a number of features of interest to public service agencies such as an emergency mode, optional text messaging, over the air programming/deactivation. More P25 information can be found at **www.apco911.org/frequency/project25/information.html#documents** and **www.tiaonline.org/standards/technology/project_25**.

Table 16.18
Digital Modulation Modes and Formats Used in Amateur Radio

See the text of Chapter 16 for definitions and abbreviations
Developed by Alan Bloom, N1AL and edited by Scott Honnaker, N7SS; March 2013

Mode or Format Name	*Developer*	*Principal Freq*	*Principal Application*	*Data rate (bit/s)*	*Bit rate (bit/s)*	*Symbol rate (baud)*	*Modulation ("N-" means multi-carrier)*	*Error handling*
ALE	MIL-STD-188-141, FED-STD-1045	HF	Data	<375	375	125	8FSK	FEC
AMTOR-A	G3PLX	HF	Data	53	114	114	FSK	ARQ
AMTOR-B	G3PLX	HF	Data	57	114	114	FSK	FEC
AOR AMBE	AOR Corp	HF	Voice, Data	2400	3600	50	36-QPSK	FEC
APCO P25	APCO	VHF	Voice	6800	9600	4,800	4FSK/QPSK	FEC
Chip64	IZ8BLY	HF	Keyboarding	21.1/37.5	300	300	DBPSK-DSSS	FEC
CLOVER-II	Hal Comm.	HF	Data	< 37.5-750	62.5-750	31.25	4-(2-16)DPSK/(2-4)DASK	FEC/FEC+ARQ
CLOVER-2000	Hal Comm.	HF	Data	108-1994	500-3000	62.5	8-(2-16)DPSK/(2-4)DASK	FEC/FEC+ARQ
Domino	ZL2AFP	HF	Keyboarding	31/44/62?	31/44/62	7.8/11/15.6	16FSK	None
DominoEX	ZL1BPU	HF	Keyboarding	(<15.63)-86.13	15.63-86.13	3.9-21.5	18FSK	None/FEC
D-Star (DV)	JARL	VHF	Voice & data	D:960 V:2400	4800	4800	0.5 GMSK/QPSK/4PSK	FEC
D-Star (DD)	JARL	UHF	Data	72k-124k	128,000	128,000	0.5 GMSK/QPSK/4PSK	FEC
Facsimile		HF	Image			120 lpm	FM, 1500-2300 Hz	None
FDMDV	G3PLX/HB9TLK	HF	Voice	1450	1450	50	15-QPSK	None
FSK441	K1JT	VHF	Meteor scatter	882	882	441	4FSK	None
G-TOR	Kantronics	HF	Data	35/75/115	80/160/240	100/200/300	FSK, 170/200 shift	FEC + ARQ
Hellschreiber (Feld)	Rudolf Hell	HF	Keyboarding	2.5 char/s	122.5	122.5	ASK	None
JT6M	K1JT	50 MHz	Meteor scatter	77.9	116.8	21.53	44FSK	None
JT65	K1JT	V/UHF	Moonbounce	1.54	16.1	2.7	65FSK	FEC
MFSK16	ZL1BPU/IZ8BLY	HF	Keyboarding, Data	31.25	62.5	15.625	16FSK	FEC
MT63	SP9VRC	HF	Keyboarding	35/70/140	320/640/1280	5/10/20	64-DPSK	FEC
Olivia	SP9VRC	HF	Keyboarding	8.75/17.5	78.13/156.25	15.63/31.25	32-FSK	FEC
Packet (Bell202)		VHF	Data	<1200	1200	1200	FSK	ARQ
PACTOR-I	DL6MAA/DF4KV	HF	Data, Winlink e-mail	51.2/128	100/200	100/200	FSK, 200 Hz	ARQ
PACTOR-II	Spec. Comm. Sys.	HF	Data, Winlink e-mail	100-700	200-800	100	2-DBPSK/PI-4DQPSK/8,16DPSK	FEC + ARQ
PACTOR-III	Spec. Comm. Sys.	HF	Data, Winlink e-mail	85-2722	200-3600	100	(2-18)-DBPSK/DQPSK	FEC + ARQ
PSK31	G3PLX	HF	Keyboarding	31.25	31.25	31.25	BPSK	None
QPSK31	G3PLX	HF	Keyboarding	31.25	62.5	31.25	QPSK	FEC
PSK63/125		HF	Keyboarding	62.5/125	62.5/125	62.5/125	BPSK	None
QPSK62/126		HF	Keyboarding	62.5/125	125/250	62.5/125	QPSK	FEC
Q15X25	SP9VRC	HF	Data	300/1200/2400	2,500	83.33	15-QPSK	FEC + ARQ
RTTY (Baudot)		HF	Keyboarding, contests	30.3	45.45	45.45	FSK, 170 Hz	None
SSTV (traditional)	WØORX	HF	Image	8 s/frame		120-line B/W	FM, 1200-2300 Hz	None
SSTV Martin M1	Martin Emmerson	HF	Image	114 s/frame		240-line RGB	FM, 1200-2300 Hz	None
SSTV Scottie S1	Eddie Murphy	HF	Image	110 s/frame		240-line RGB	FM, 1200-2300 Hz	None
Throb	G3PPT	HF	Keyboarding	10/20/40 wpm		1/2/4	9FSK/2-9FSK	None
WinDRM	HB9TLK	HF	Voice, Data				COFDM (n-QAM)	FEC/FEC+ARQ
WINMOR	KN6KB	HF	Winlink email		62.5-3750	31.25/62.5	(1-15)-QPSK/16QAM/4FSK	FEC + ARQ
WSPR (MEPT-JT)	K1JT	HF-VHF	Weak signal beacon	0.45	2.93	1.46	4FSK	FEC

16.6 Digital Mode Table

Table 16.18 is a summary of common digital modes used by amateurs as of early 2013 and their primary characteristics. The following text is intended to make high-level comparisons. For more information on these modes see the earlier sections of this chapter. This page is also available as a PDF file on the CD-ROM accompanying this book.

Many modes have a number of variants, with the most common shown in the table. High reliability, low data rate modes are more common on HF. Higher data rates are available on VHF/UHF, where the bands have less noise and require less effort to make them reliable. Some modes have very specific intended uses like the meteor scatter, moonbounce and beaconing modes created by Joe Taylor, K1JT as *WSJT* (see the section Structured Digital Modes).

There can be a significant difference between data rate and bit rate in high reliability modes. The data rate is the amount of user data transmitted where the bit rate includes the packet and error correction overhead. Some of these rates are approximate and can vary based on conditions and the variant of the mode used. The symbol rate is a function of the modulation scheme and how many simultaneous carriers are used to transmit the data.

The error handling mechanism is critical to note when sending data. Some modes include no error handling which means errors are not detected and must be addressed in a higher protocol layer (as in AX.25 packet). Forward error correction (FEC) makes a best effort to address errors in real time as part of the data sent in each packet. FEC can add substantial overhead to each packet and does not guarantee error-free delivery but does make the mode robust in high noise environments. Automatic Retry Request (ARQ) can guarantee error-free delivery of data but has no ability to actually correct received data. The combination of FEC and ARQ allows minor errors to be corrected in real time with major errors generating a retry request.

16.7 Glossary

ACK — Acknowledgment, the control signal sent to indicate the correct receipt of a transmission block.

Address — A character or group of characters that identifies a source or destination.

AFSK — Audio frequency-shift keying.

ALE — Automatic link establishment.

Algorithm — Numerical method or process.

APCO — Association of Public Safety Communications Officials.

ARQ — Automatic Repeat reQuest, an error-sending station, after transmitting a data block, awaits a reply (ACK or NAK) to determine whether to repeat the last block or proceed to the next.

ASCII — American National Standard Code for Information Interchange, a code consisting of seven information bits.

AX.25 — Amateur packet-radio link-layer protocol.

Baud — A unit of signaling speed equal to the number of discrete conditions or events per second. (If the duration of a pulse is 20 ms, the signaling rate is 50 baud or the reciprocal of 0.02, abbreviated Bd).

Baudot code — A coded character set in which five bits represent one character. Used in the US to refer to ITA2.

Bell 103 — A 300-baud full-duplex modem using 200-Hz-shift FSK of tones centered at 1170 and 2125 Hz.

Bell 202 — A 1200-baud modem standard with 1200-Hz mark, 2200-Hz space, used for VHF FM packet radio.

BER — Bit error rate.

BERT — Bit-error-rate test.

Bit stuffing — Insertion and deletion of 0s in a frame to preclude accidental occurrences of flags other than at the beginning and end of frames.

Bit — Binary digit, a single symbol, in binary terms either a one or zero.

Bit/s or ***bps*** — Bits per second.

Bitmap — See **raster.**

Bit rate — Rate at which bits are transmitted in bit/s or bps. **Gross** (or **raw**) **bit rate** includes all bits transmitted, regardless of purpose. **Net bit rate** (also called **throughput**) only includes bits that represent data.

BLER — Block error rate.

BLERT — Block-error-rate test.

BPSK — Binary phase-shift keying in which there are two combinations of phase used to represent data symbols.

Byte — A group of bits, usually eight.

Cache — To store data or packets in anticipation of future use, thus improving routing or delivery performance.

Channel — Medium through which data is transmitted.

Checksum — Data representing the sum of all character values in a packet or message.

CLOVER — Trade name of digital communications system developed by Hal Communications.

COFDM — Coded Orthogonal Frequency Division Multiplex, OFDM plus coding to provide error correction and noise immunity.

Code — Method of representing data.

Codec — Algorithm for compressing and decompressing data.

Collision — A condition that occurs when two or more transmissions occur at the same time and cause interference to the intended receivers.

Compression — Method of reducing the amount of data required to represent a signal or data set. **Decompression** is the method of reversing the compression process.

Constellation — A set of points that represent the various combinations of phase and amplitude in a QAM or other complex modulation scheme.

Contention — A condition on a communications channel that occurs when two or more stations try to transmit at the same time.

Control characters — Data values with special meanings in a protocol, used to cause specific functions to be performed.

Control field — An 8-bit pattern in an HDLC frame containing commands or responses, and sequence numbers.

Convolution — The process of combining or comparing signals based on their behavior over time.

Cyclic Redundancy Check (CRC) — The result of a calculation representing all character values in a packet or message. The result of the CRC is sent with a transmission block. The receiving station uses the received CRC to check transmitted data integrity.

Datagram — A data packet in a connectionless protocol.

Data stream — Flow of information, either over the air or in a network.

DBPSK — Differential binary phase-shift keying.

Dibit — A two-bit combination.

DQPSK — Differential quadrature phase-shift keying.

Domino — A conversational HF digital mode similar in some respects to MFSK16.

DRM — Digital Radio Mondiale. A consortium of broadcasters, manufacturers, research and governmental organizations which developed a system for digital sound broadcasting in bands between 100 kHz and 30 MHz.

DV —Digital voice.

Emission — A signal transmitted over the air.

Encoding — Changing data into a form represented by a particular code.

Encryption — Process of using codes and encoding in an effort to obscure the meaning of a transmitted message. **Decryption** reverses the encryption process to recover the original data.

Error Correcting Code (ECC) — Code used to repair transmission errors.

Eye pattern — An oscilloscope display in the shape of one or more eyes for observing the shape of a serial digital stream and any impairments.

Fast Fourier Transform (FFT) — An algorithm that produces the spectrum of a signal from the set of sampled value of the waveform.

FEC — Forward error correction.

Frame — Data set transmitted as one package or set.

Facsimile (fax) — A form of telegraphy for the transmission of fixed images, with or without half-tones, with a view to their reproduction in a permanent form.

FCS — Frame check sequence. (see also **CRC**)

FDM — Frequency division multiplexing.

FDMA — Frequency division multiple access.

FEC — Forward error correction, an error-control technique in which the transmitted data is sufficiently redundant to permit the receiving station to correct some errors.

FSK — Frequency-shift keying.

Gray code — A code that minimizes the number of bits that change between sequential numeric values.

G-TOR — A digital communications system developed by Kantronics.

HDLC — High-level data link control procedures as specified in ISO 3309.

Hellschreiber — A facsimile system for transmitting text.

Host — As used in packet radio, a computer with applications programs accessible by remote stations.

IA5 — International Alphabet, designating a specific set of characters as an ITU standard.

Information field — Any sequence of bits containing the intelligence to be conveyed.

Interleave — Combine more than one data stream into a single stream or alter the data stream in such a way as to optimize it for the modulation and channel characteristics being used.

IP — Internet Protocol, a network protocol used to route information between addresses on the Internet.

ISO — International Organization for Standardization.

ITU — International Telecommunication Union, a specialized agency of the United Nations. (See **www.itu.int**.)

ITU-T — Telecommunication Standardization Sector of the ITU, formerly CCITT.

Jitter — Unwanted variations in timing or phase in a digital signal.

Layer — In communications protocols, one of the strata or levels in a reference model.

Least significant bit (LSB) — The bit in a byte or word that represents the smallest value.

Level 1 — Physical layer of the OSI reference model.

Level 2 — Link layer of the OSI reference model.

Level 3 — Network layer of the OSI reference model.

Level 4 — Transport layer of the OSI reference model.

Level 5 — Session layer of the OSI reference model.

Level 6 — Presentation layer of the OSI reference model.

Level 7 — Application layer of the OSI reference model.

Lossless (compression) — Method of compression that results in an exact copy of the original data

Lossy (compression) — Method of compression in which some of the original data is lost

MFSK16 — A multi-frequency shift communications system

Modem — Modulator-demodulator, a device that connects between a data terminal and communication line (or radio). Also called **data set**.

Most significant bit (MSB) — The bit in a byte or word that represents the greatest value.

Multicast — A protocol designed to distribute packets of data to many users without communications between the user and data source.

MSK — Frequency-shift keying where the shift in Hz is equal to half the signaling rate in bit/s.

MT63 — A keyboard-to-keyboard mode similar to PSK31 and RTTY.

Nibble — A four-bit quantity. Half a byte.

Node — A point within a network, usually where two or more links come together, performing switching, routine and concentrating functions.

OFDM — Orthogonal Frequency Division Multiplex. A method of using spaced subcarriers that are phased in such a way as to reduce the interference between them.

OSI-RM — Open Systems Interconnection Reference Model specified in ISO 7498 and ITU-T Recommendation X.200.

PACTOR — Trade name of digital communications protocols offered by Special Communications Systems GmbH & Co KG (SCS).

Packet — 1) Radio: communication using the AX.25 protocol. 2) Data: transmitted data structure for a particular protocol (see also **frame.**)

Parity (parity check) — Number of bits with a particular value in a specific data element, such as a byte or word or packet. Parity can be odd or even. A **parity bit** contains the information about the element parity.

Pixel — Abbreviation for "picture element."

Primitive — An instruction for creating a signal or data set, such as in a drawing or for speech.

Project 25 — Digital voice system developed for APCO, also known as P25.

Protocol — A formal set of rules and procedures for the exchange of information.

PSK — Phase-shift keying.

PSK31 — A narrow-band digital communications system developed by Peter Martinez, G3PLX.

QAM — Quadrature Amplitude Modulation. A method of simultaneous phase and amplitude modulation. The number that precedes it, for example, 64QAM, indicates the number of discrete stages in each symbol.

QPSK — Quadrature phase-shift keying in which there are four different combinations of signal phase that represent symbols.

Raster (image) — Images represented as individual data elements, called pixels.

Router — A network packet switch. In packet radio, a network level relay station capable of routing packets.

RTTY — Radioteletype.

Sample — Convert an analog signal to a set of digital values.

Shift — 1) The difference between mark and space frequencies in an FSK or AFSK signal. 2) To change between character sets, such as between LTRS and FIGS in RTTY.

SSID — Secondary station identifier. In AX.25 link-layer protocol, a multipurpose octet to identify several packet radio stations operating under the same call sign.

State — Particular combination of signal attributes used to represent data, such as amplitude or phase.

Start (stop) bit — Symbol used to synchronize receiving equipment at the beginning (end) of a data byte.

Symbol — Specific state or change in state of a transmission representing a particular signaling event. **Symbol rate** is the number of symbols transmitted per unit of time (see also **baud**).

TAPR — Tucson Amateur Packet Radio Corporation, a nonprofit organization involved in digital mode development.

TDM — Time division multiplexing.

TDMA — Time division multiple access.

Throb — A multi-frequency shift mode like MFSK16.

TNC — Terminal node controller, a device that assembles and disassembles packets (frames).

Trellis — The set of allowed combination of signal states that represent data. (see also **Viterbi coding** and **constellation**.)

Turnaround time — The time required to reverse the direction of a half-duplex circuit, required by propagation, modem reversal and transmit-receive switching time of transceiver.

Unicast — A protocol in which information is exchanged between a single pair of points on a network.

Varicode — A code in which the different data values are represented by codes with different lengths.

Vector — Image represented as a collection of drawing instructions.

Word — Set of bits larger than a byte.

16.8 References and Bibliography

A. Bombardiere, K2MO, "A Quick-Start Guide to ALE400 ARQ FAE," *QST*, Jun 2010, pp 34-36.

B.P. Lathi, *Modern Digital and Analog Communication Systems*, Oxford University Press, 1998.

B. Sklar, *Digital Communications, Fundamentals and Applications*, Prentice Hall, 1988.

S. Ford, WB8IMY, *HF Digital Handbook*, 4th edition, ARRL, 2007.

S. Ford, WB8IMY, *VHF Digital Handbook*, ARRL, 2008.