Contents

Chapter 15 — Online Content

Articles and Data Tables

- ASCII Character Code Table
- Digital Modes — Lowest Permitted Amateur Frequency
- High-Speed Wireless Networking in the UHF and Microwave Bands by David Bern, W2LNX, and Keith Elkin, KB3TCB
- ITA2 Character Code Table
- Legacy Digital Modes
- Operating Tips for Digital Voice Using *FreeDV* by Mel Whitten, K0PFX
- The FT4 and FT8 Communication Protocols by Steve Franke, K9AN, Bill Somerville, G4WJS, and Joe Taylor, K1JT
- Varicode tables for PSK31, MFSK16 and DominoEX

Chapter 15
Digital Protocols and Modes

Since adapting radioteletype (RTTY) after World War II, radio amateurs have been creating new digital modes to communicate both as "chat-style" keyboard-to-keyboard operation or using packetized network-compatible protocols. Many new protocols have been entirely implemented using software and a computer sound card, making tools for experimentation and implementation easily available worldwide via the internet. SDR technology allows digital protocols and modes to be implemented directly without an intervening audio modulation/demodulation step. It is an exciting period for amateur radio with frequent innovations and improvements.

This chapter will focus on the protocols for transferring various data types, focusing on the processes by which data is encoded, compressed and error checked, packaged and exchanged. Joe Taylor, K1JT, and Steve Franke, K9AN, updated the sections on *WSJT-X* modes. HF Digital Voice was updated by Mel Whitten, KØPFX. The sections on VARA and ARDOP were contributed by Steve Stroh, N8GNJ. Packet radio and APRS material was updated by John Langner, WB2OSZ and Lynn Deffenbaugh, KJ4ERJ. Al Francisco, K7NHV updated the section on Winlink. The section on D-STAR was rewritten by John Davis, WB4QDX and Ed Woodrick, WA4YIH. DMR information was updated by John Burningham, W2XAB. Cory Sickles, WA3UVV updated the System Fusion section. The new section on IP Microwave Networking was contributed by Orv Beach, W6BI. Ken Humbertson, WØKAH updated the sidebar on *fldigi* software.

Modulation methods are covered in the **Modulation** chapter, and the **Digital Communications** chapter available with the online content discusses the practical considerations of operating using these modes.

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.

15.1 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.

15.1.1 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 15.1
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 15.1**.

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.

15.1.2 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.

15.1.3 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 with the online content.

Early computers used a wide variety of alphabetic 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" with the online content.

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, becoming the standard for internet, web pages, and software.

More recent schemes use varicode, where the most common characters are given shorter codes (see **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 with the online content.

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.

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 SDR Fundamentals** 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 15.2** 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

Table 15.2
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

as 15 frames per second can still be considered "full-motion" but will appear jerky and even this rate requires substantial storage. The most popular full-motion video formats are MP4, MOV, and WMV. GIF can also support limited animation.

15.1.4 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. The use of proprietary codecs is controversial in amateur radio, open-source software, and internet systems. Open-source codecs such as *CODEC2* (see the section on Digital Voice) are an active area of development both in amateur radio and the open-source community.

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. Winlink uses a compression scheme called B2F and sees an average 44% improvement in performance since most Winlink data is uncompressed previously. 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. (See the Winlink section below.)

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-frames"

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 numerous modes, including CW, as well as variations of tones, bandwidth, baud rates, number of bits, and other variations for many modes. Versions of *fldigi* are available for Windows, Linux, Unix, and OS X. See **www.w1hkj.com** for information on downloading the files.

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 **Figure 15.A** when PSK signals are present.

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 Figure 15.A, an ICOM IC-7610 is controlled by *fldigi* and thus shows the current operating frequency and mode of 14070.000 kHz USB, the actual contact frequency of 14071.465 (kHz) (in the Feq window, upper center just right of dial frequency), 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. When 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.

BPSK-31 and RTTY-45 are both modes that are well established and remain popular. However, they both need a fairly strong signal for error-free copy because they have no error correction. MFSK-16 is a mode that is popular among emcomm operators because of its robust error correction. W1AW sends digital bulletins in BPSK-31, MFSK-16, and RTTY-45, so you have the opportunity to copy the bulletins on all three modes. **Figures 15.B** and **15.C** show a portion of a decoded ARRL digital bulletin from January 18, 2022 in RTTY-45 and MFSK16, respectively.

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. An excellent guide to understanding the technical details of the different modes is also available at **www.w1hkj.com/modes/index.htm**.

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. Many radios now have either a direct digital data connector (typically a 6-pin mini-DIN connector) or an internal USB soundcard. These are the recommended digital data interfaces. A digital data interface that allows you to connect the sound card to your radio's microphone input is also an option. (See the **Assembling a Station** and **Digital Communications** chapters.)

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.

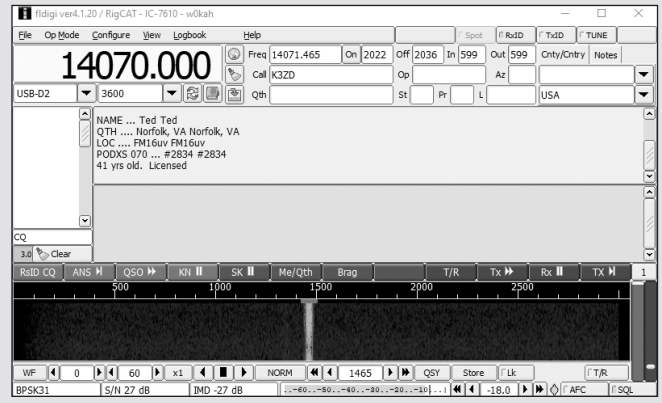


Figure 15.A — Part of a contact being conducted using BPSK-31.

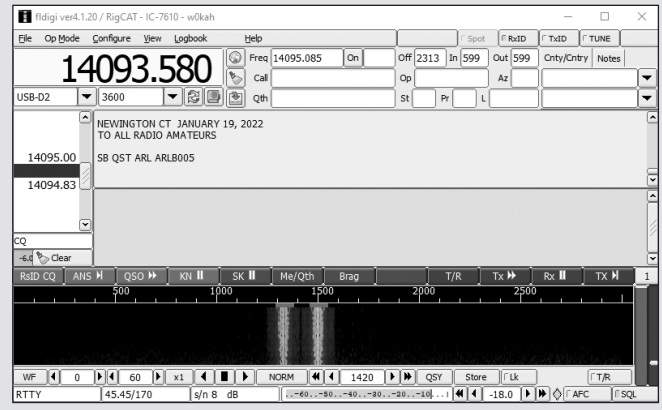


Figure 15.B — A W1AW bulletin being sent using RTTY-45.

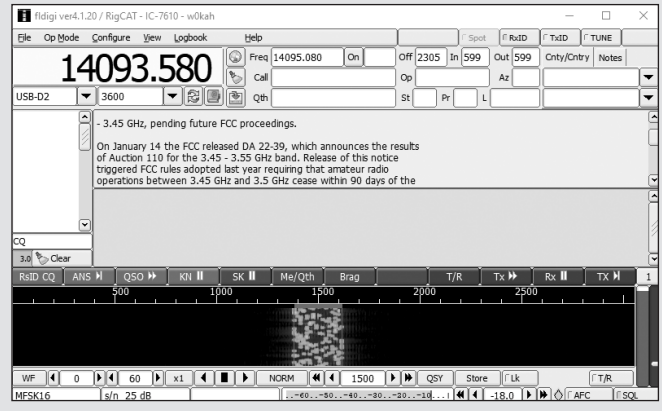


Figure 15.C — A W1AW bulletin being sent using MFSK-16.

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 15.3 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 15.3
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

15.1.5 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) without password protection 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. For example, traffic with any site accessed using the Secure Hypertext Protocol where the URL begins "https://" is encrypted. 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.

15.2 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.

15.2.1 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. **Figure 15.1** shows the character D sent by RTTY.

BAUDOT CODE

The Baudot code (see the online content) 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.

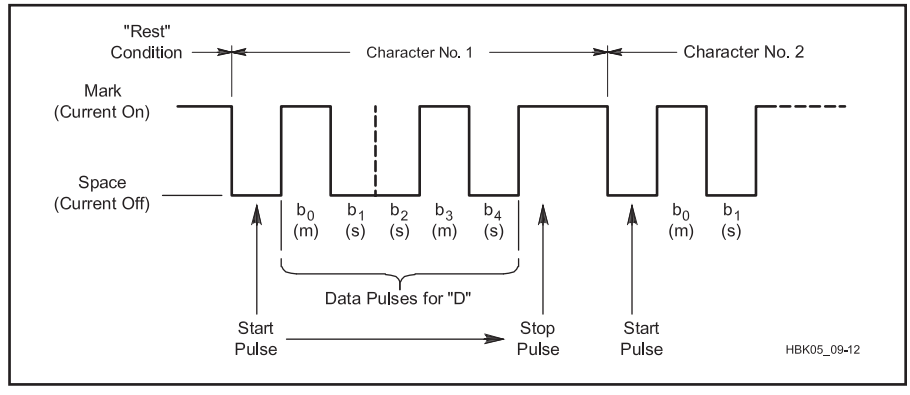


Figure 15.1 — The character "D" sent via RTTY.

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. 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.

15.2.2 PSK31 and Variants

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 encod-

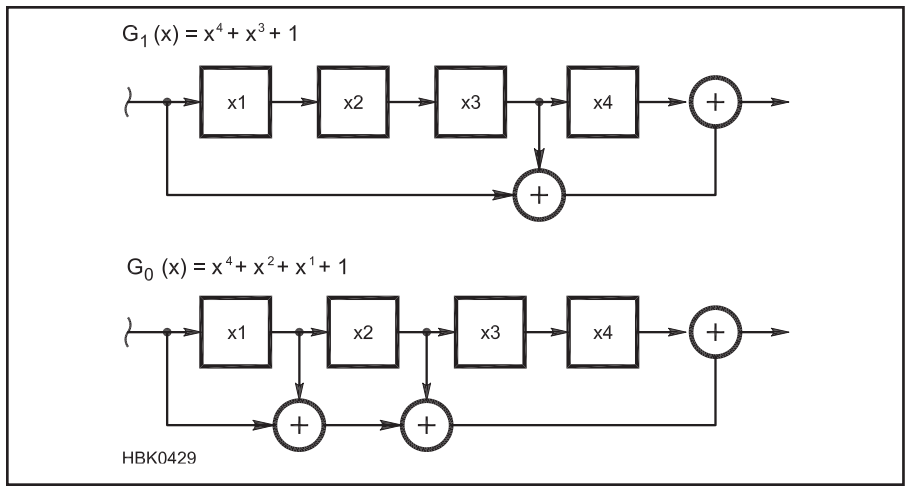


Figure 15.2 — QPSK31 convolution encoders.

ers to form the dibits that are used to modulate the QPSK31 generator. Both convolution encoders are fourth-order polynomials, shown in **Figure 15.2**.

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 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 15.4** 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.

Table 15.4
QPSK31 Modulation

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

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 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 **en.wikipedia.org/wiki/Convolutional_code**.

15.2.3 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

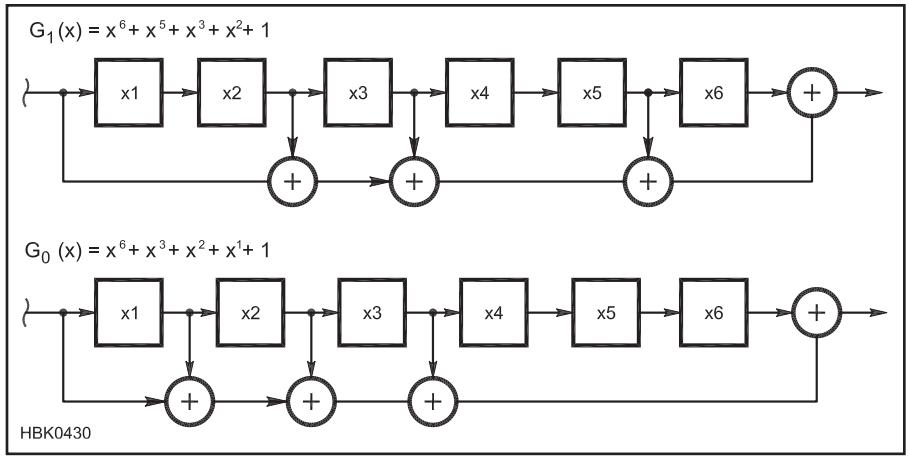


Figure 15.3 — MFSK16 convolution encoders.

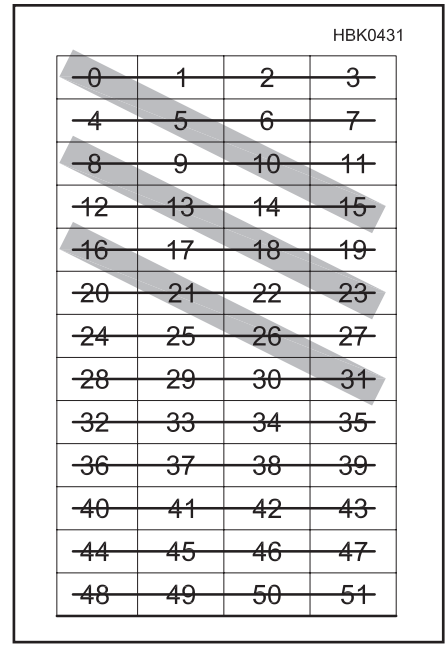


Figure 15.4 — IZ8BLY interleaver.

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 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 online content) 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 **Figure 15.3**.

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**. **Figure 15.4** 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 is equivalent to a single interpolator that is 123 bits long. **Figure 15.5** 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 **en.wikipedia.org/wiki/Gray_code** and **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 **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 clos-

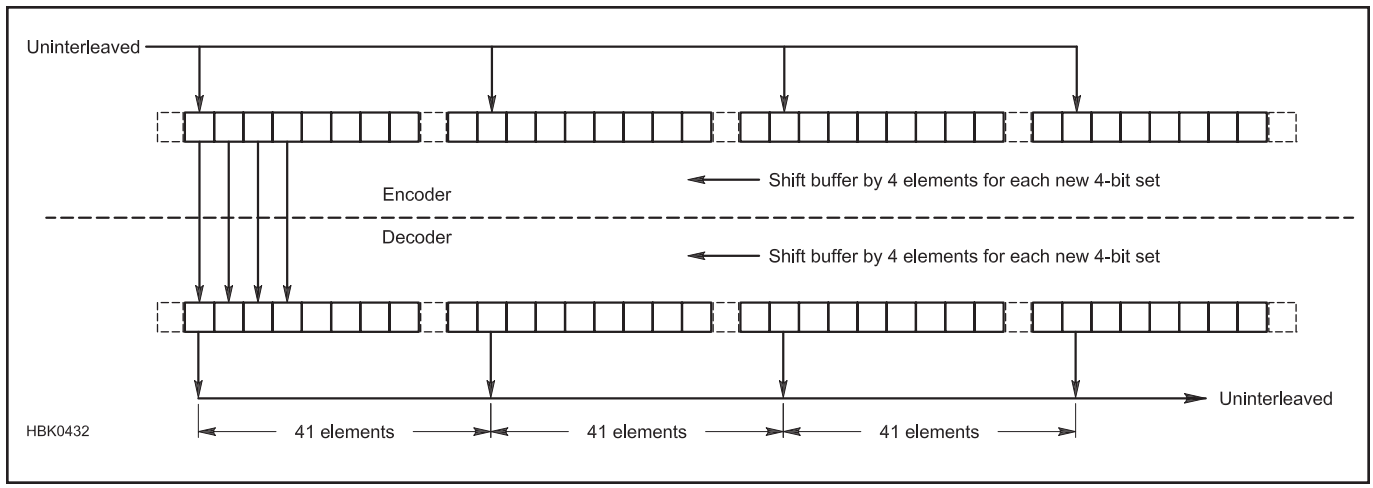


Figure 15.5 — Bit spreading through interpolation.

est FFT bin. The hard or the soft Viterbi Algorithm can be used to decode and correct errors.

15.2.4 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 15.5**) 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 MFSK15. 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 15.5. 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 with the online content.

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.

Table 15.5
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

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 by MFSK16. However, instead of a 10-stage IZ8BLY interleaver (see Figure 15.4), 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" with the online content and online at **www.w7ay.net/site/Technical/DominoEX/Measurements/**.

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 MFSK15. The software can be downloaded from **antoninoporcino.xoom.it/Chip64/** and more information is available at **www.arrl.org/technical-characteristics**. Spread spectrum is discussed in the **Modulation** chapter, as well.

15.2.5 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**).

15.2.6 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.

15.2.7 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 **n1su.com/olivia** and Olivia is supported in a number of digital multimode packages such as *MixW*, *MultiPSK* and *Ham Radio Deluxe*.

15.3 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.

15.3.1 Facsimile (WEFAX)

Also known as Weatherfax, HF-FAX, Radiofax, and Weather Facsimile, WEFAX 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 15.6**.

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).

15.3.2 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, 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** chapter with the online content.

15.3.3 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/HELL/Feld.htm** and Randy, K7AGE, has a great introduction to Hellschreiber available on YouTube at **www.youtube.com/watch?v=yR-EmyEBVqA**.

Table 15.6
Facsimile Automatic Picture Format

Signal	*Duration*	*IOC576*	*IOC288*	*Remarks*
Start Tone	5 s	300 Hz	675 Hz	200 Hz 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			

15.4 Structured Digital Modes

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

15.4.1 WSJT-X Modes

All modes described in Sections 15.4.1 through 15.4.4 are available in a multi-author, open-source software package called *WSJT-X*. Some are also available in other programs derived from *WSJT-X*. These modes are designed around structured messages that optimize the exchange of minimal contact information including call signs, signal reports, Maidenhead grid locators, and acknowledgments, even at very low signal-to-noise ratios. These modes were pioneered by Joe Taylor, K1JT, starting in 2001 when home computers were first widely equipped with sound cards and fast enough to do significant signal processing. The first modes were optimized for use over especially difficult paths such as meteor scatter and Earth-Moon-Earth (EME, or "moonbounce") on VHF and higher bands. More recently, several of the modes have become very popular for LF, MF, and HF DXing, as well.

The *WSJT-X* structured modes divide naturally into two groups, *slow* and *fast*. Slow modes send each message frame once per transmission interval, while fast modes send the message frame repeatedly, as many times as will fit into the transmission interval. The modes intended for making two-way QSOs use message formats based on compressed, fixed-length packets of 72 or 77 information bits. Modes called WSPR and FST4W allow individual stations to transmit a call sign, location and power level using 50-bit packets. As outlined in this chapter's section "Error Detection and Correction," additional bits are added to the message frames to effect error detection and correction. Details of the mathematical encoding schemes and modulation types differ from mode to mode. In all cases the forward error correction (FEC) algorithm is strong enough that false decodes are very rare: a receiving operator nearly always sees exactly the message that was transmitted, or nothing is displayed at all. All of the modes implemented in *WSJT-X* use *constant-envelope* waveforms, so they can be amplified by efficient nonlinear amplifiers without generating intermodulation products. Transmissions have a fixed duration, for example 15 seconds or one minute, and their start times are synchronized with Coordinated Universal Time (UTC) as maintained by the computer's operating system.

15.4.2 WSJT-X Message Compression

The information packets normally consist of two 28-bit call signs and a 15-bit field for a grid locator, signal report, acknowledgment, or "73". Additional bits can flag message formats containing arbitrary text (up to 13 characters), nonstandard call signs, contest exchanges, or the like. The basic aim is to compress the most common message formats used for minimal contacts into fixed-length packets. The older modes (JT4, JT9, and JT65) use 72-bit message payloads, while FST4, FT4, FT8, and Q65 use 77 bits and the propagation-probing modes WSPR and FST4W use 50.

A standard amateur call sign consists of a one- or two-character prefix, at least one of which must be a letter, followed by a decimal digit and a suffix of one to three letters. Within these rules, the number of possible call signs is equal to 37×36×10×27×27×27, or somewhat over 262 million. (The numbers 27 and 37 arise because in the first and last three positions a character may be absent, or a letter, or perhaps a digit.) Since 2^{28} is more than 268 million, 28 bits are enough to encode any standard call sign uniquely. Similarly, the number of 4-character Maidenhead grid locators on Earth is 180×180 = 32,400, which is less than 2^{15} = 32,768; thus, a grid locator requires 15 bits. Some otherwise unused values in the 28-bit fields are used for special message components such as CQ, DE, and QRZ, and some values of the 15-bit field not needed for grid locators are used for signal reports, acknowledgments, and the like. Structured messages using these 28-bit and 15-bit fields underlie the efficient, reliable exchange of basic and essential information for minimal station-to-station contacts.

15.4.3 WSJT-X Slow Modes

The slow modes in *WSJT-X* (versions 2.4.0 and later) are called FST4, FST4W, FT4, FT8, JT4, JT9, JT65, Q65, and WSPR. Each mode uses *continuous-phase frequency-shift keying* (FSK), with waveforms normally generated at audio frequency and transmitted as upper sideband by a standard SSB transceiver. Numbers in the mode names indicate the number of distinct tone frequencies used. FST4 was designed for the LF, MF and HF bands; JT4 and JT65 for EME on VHF, UHF, and microwave bands; JT9 for MF and lower HF bands; FT8 for multi-hop sporadic E at 50 MHz; FT4 for fast contest QSOs on HF and 50 MHz; and FST4W and WSPR for probing potential propagation paths using low-power transmissions. Not surprisingly, amateurs have discovered many other ways in which each mode can be used. On the HF bands, worldwide contacts are possible with these modes using power levels of a few watts (or even milliwatts) and compromise antennas. Reliable two-way contacts can be made with signal levels far too weak to be heard, 10 to 15 dB below the minimum levels required for Morse-coded CW received and decoded by ear.

Table 15.7A provides a brief summary of essential parameters for the *WSJT-X* slow modes. Column 1 gives the mode name, column 2 the message payload size and number of cyclic redundancy check (CRC) bits, and column 3 the type of forward error correction used. Each code maps a sequence of k information symbols into a longer sequence of n transmitted symbols. The results are called (n, k) *block codes*. After encoding, digital message information is modulated onto a carrier so that transfer can take place over a radio channel. The basic unit of transmitted data is called a *channel symbol*. Parameter Q (column 5 in the table) is the *alphabet size,* the number of different symbols used. Q is also the number of distinct waveforms used for conveying information. For frequency-shift keying the waveforms are sinusoids at different frequencies. Additional columns in Table 15.7A specify the keying rate, occupied bandwidth, fraction of transmitted energy devoted to synchronization, duration of transmitted waveform, and threshold signal-to-noise ratio for reliable decoding of each mode. Here and elsewhere in this section, signal-to-noise ratios are measured in a standard reference bandwidth of 2500 Hz.

Figure 15.6 illustrates the appearance of each of the *WSJT-X* slow modes in a typical waterfall-type spectral display. For comparison, this collection of simulated signals also includes an unmodulated carrier and a 25 WPM CW signal. The signals were generated with a key-down signal-to-noise ratio of 0 dB, thus simulating typical over-the-air reception of moderately weak, barely audible signals. WSPR has the narrowest occupied bandwidth of the basic *WSJT-X* modes, 5.9 Hz, while JT65 is the widest at 177.6 Hz. JT4, JT9, and JT65 use one-minute timed sequences of transmission and reception, synchronized with UTC, while WSPR uses two-minute sequences, FT8, 15 seconds, and FT4, 7.5 seconds. FST4, FST4W, and Q65 offer a wide range of sequence lengths. The following paragraphs give further details for each of the slow modes and describe their typical uses.

FT8

The FT8 protocol was developed by Steven Franke, K9AN, and K1JT. The mode became popular soon after its introduction in early

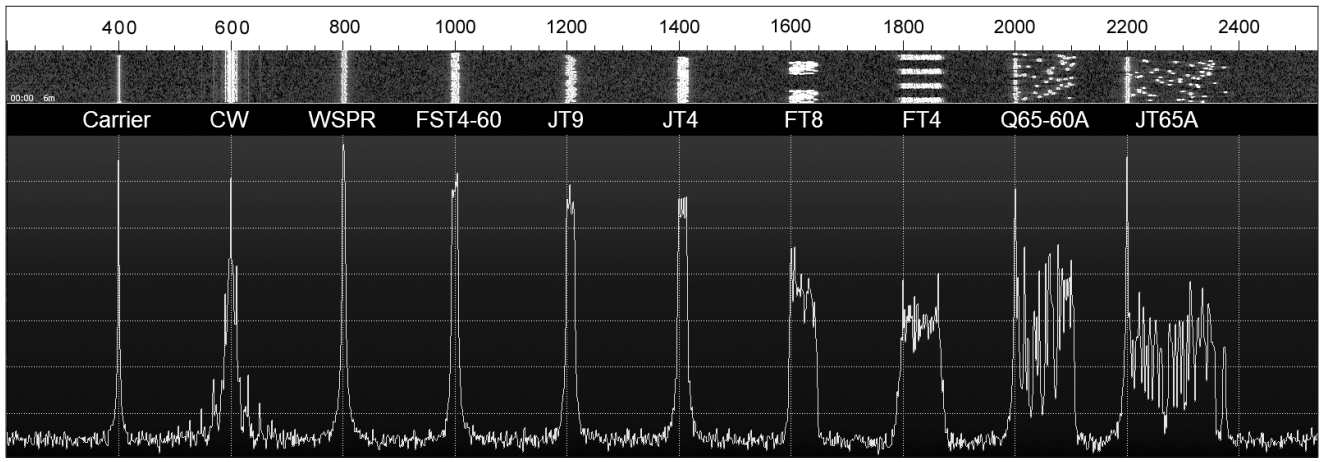


Figure 15.6 — Spectrograms of simulated signals for an unmodulated carrier, a 25 WPM CW signal, and the *WSJT-X* slow modes WSPR, JT9, JT4, FT8, FT4, QRA64, and JT65. All signals have S/N = 0 dB in a 2500 Hz reference bandwidth. Horizontal scale is frequency in Hz; vertical axis represents time over the displayed interval of about 50 seconds.

summer of 2017, and is now one of the most widely used modes of any type in amateur radio. FT8 was designed for circumstances where signals may be weak and fading, so that quick completion of reliable, confirmable QSOs is especially desirable. The terse message exchanges of call signs, locators, signal reports, and acknowledgments serve well for those engaged in country-counting, award-chasing, and the like. The short, 15-second timed sequences for transmission and reception mean that basic sensitivity for steady signals is several dB worse than the modes with one-minute transmissions. Message packets include 77 information bits and a 14-bit CRC. Three of the information bits define special message types, some of which include several subtypes. These are used for such purposes as free text and contest exchanges. A detailed description of the 77-bit message format and the FT8 protocol was published in *QEX* in 2020 (see References and Bibliography). FT8 uses a (174,91) *low-density parity check* (LDPC) code for forward error correction, and the 14-bit CRC ensures a very low false decode rate. FT8 modulation uses Gaussian filtered 8-tone FSK (8-GFSK) at 6.25 baud, and accomplishes frame synchronization with three 7 × 7 Costas arrays.

One special message subtype is used for *FT8 DXpedition Mode* (also called *Fox and Hounds*), which enables a much-sought-after station to make QSOs at high rates. In this mode *WSJT-X* allows the rare station to transmit up to five FT8 signals simultaneously, thereby conducting as many as five minimal contacts at once. The special multi-signal transmissions are subject to undesirable spectral broadening caused by intermodulation between the signals. This is true even when care is taken to avoid overdriving the transmitter audio chain because most modern transmitters produce intermodulation products at levels that can cause significant interference to weak-signal digital modes. Multi-signal transmissions are therefore not suitable for general use or for use in contests — situations where good engineering practice and common courtesy dictate that any individual station use only the minimum bandwidth necessary. Multi-signal transmission is intended only for use by DXpeditions who will operate on frequency segments that do not overlap with those

Table 15.7
Parameters of Structured Modes

A — Structured Slow Modes

Mode	*FEC Type*	*(n, k)*	*Q*	*Modulation Type*	*Keying Rate*	*Bandwidth (Hz)*	*Sync Energy (s)*	*Transmit Duration (dB)*	*S/N Threshold*
FT4	LDPC	(174,91)	4	4-GFSK	20.833	83.3	0.16	5.04	–17.5
FT8	LDPC	(174,91)	8	8-GFSK	6.25	50.0	0.27	12.6	–20.8
JT4	Convolutional	(206,72)	2	4-FSK	4.375	17.5	0.50	47.1	–23
JT9	Convolutional	(206,72)	8	9-FSK	1.736	15.6	0.19	49.0	–27
JT65	Reed Solomon	(63,12)	64	65-FSK	2.692	177.6	0.50	46.8	–25
QRA64	Q-ary Repeat Accumulate	(63,12)	64	64-FSK	1.736	111.1	0.25	48.4	–26
WSPR	Convolutional	(162,50)	2	4-FSK	1.465	5.9	0.50	110.6	–31

B — Structured Fast Modes

Mode	*FEC Type*	*(n, k)*	*Q*	*Modulation Type*	*Keying Rate*	*Bandwidth (Hz)*	*Sync Energy*	*Transmit Duration (s)*
JT9E	Convolutional	(206,72)	8	9-FSK	25	225	0.19	3.4
JT9F	Convolutional	(206,72)	8	9-FSK	50	450	0.19	1.7
JT9G	Convolutional	(206,72)	8	9-FSK	100	900	0.19	0.85
JT9H	Convolutional	(206,72)	8	9-FSK	200	1800	0.19	0.425
MSK144	LDPC	(128,90)	2	OQPSK	2000	2400	0.11	0.072

in use for regular QSOs, to avoid causing unnecessary QRM.

Recognizing a need expressed by the VHF+ contest community, the WSJT-X development team added the ability to send band-change messages as one of the Tx# messages. This change became effective with version 2.5.3 in Dec 2021. The capability is described in the WSJT-X User Guide at **www.physics.princeton.edu/pulsar/K1JT/wsjtx.html**. Read the Tx Macros section under Settings for complete information.

Summarizing, the messages now support the use of band abbreviations ABCD… etc. that represent 50 MHz, 144 MHz, 222 MHz, 432 MHz, and so forth. The band abbreviation is followed by a mode abbreviation: V (SSB Voice), W (CW), or 8 (FT8). For example, the message $DXCALL BV 228 means, "Send the call sign of the station being worked and request they QSY to 2 meter voice on 144.228 MHz." The station being queried would presumably send an acknowledgment message, although that is not required. Macros can be preprogrammed with a message that is selected from the Tx5 drop-down menu. Text can also be entered into the Tx5 message at any time.

For more information about using macros to move stations, read the article, "Moving on up with FT8" by Phil Miguelez, WA3NUF, with contributions by Joe Taylor, K1JT, in the November 2021 issue of the Mt. Airy V.H.F. Radio Club's newsletter, "Cheese sBits" at **www.packratvhf.com** in the "NEWSLETTER" section. The codes and techniques may change as operating practices and the WSJT-X software evolve.

FT4

FT4 was designed for use in contests where, under certain circumstances, it is worthwhile to trade a few dB of sensitivity for higher QSO rates. It uses the same 77-bit messages and (174,91) error-correcting code as FT8, but with transmit/receive sequences of just 7.5 seconds. Although twice as fast as FT8, with only 4 tones the occupied bandwidth of FT4 is only 83 Hz, less than twice that of FT8. The decoding threshold is –17.5 dB, 3.3 dB higher than that of FT8. FT4 uses Gaussian filtered 4-tone FSK (4-GFSK) waveforms at 20.833 baud, and frame synchronization uses four 4 × 4 Costas arrays. A *QEX* article describing how FT8 and FT4 work is part of the online information for this chapter.

FST4 and FST4W

FST4 was designed for weak-signal QSOs on LF, MF, and HF bands. It uses 77-bit messages, a 24-bit CRC and a (240,101) LDPC code for error detection and correction. Modulation is Gaussian filtered 4-tone FSK (4-GFSK) with 120 message-bearing channel symbols and 40 symbols for frame synchronization; a total of 160 channel symbols. A companion mode named FST4W (the W is for "WSPR-like") is intended for use as a propagation probe, like the popular WSPR mode. FST4W uses 50-bit messages and a (240,74) LDPC code. FST4 and FST4W use the same modulation format and frame-synchronization scheme so these signals can be detected and synchronized using the same algorithms. FST4 and FST4W offer several different keying rates. FST4 supports transmission lengths of 15, 30, 60, 120, 300, 900, and 1800 seconds and FST4W supports transmission lengths of 120, 300, 900, and 1800 seconds. In general, longer transmission lengths (lower keying rates) provide proportionally higher sensitivity, narrower bandwidth, and decreasing tolerance for Doppler frequency spread. In practice, the transmit duration in seconds is appended to the mode designator to describe a particular variant. For example, FST4-60 refers to FST4 with 60 second transmissions. On channels where Doppler spread is small compared to the keying rate, FST4 and FST4W are the most sensitive weak-signal modes implemented in *WSJT-X*.

JT4

In JT4 mode, each channel symbol carries one information bit (the most significant bit) and one synchronizing bit. The sync bits are defined by a pseudo-random binary sequence known to the software at both transmitting and receiving ends. FEC is accomplished using a strong convolutional code designed by NASA for deep space use. After the sync pattern is recognized and removed by the receiving software, the remaining signal amounts to a 2-tone FSK signal. Submodes JT4A through JT4G have tone spacings at increasing multiples 1, 2, 4, 9, 18, 36, and 72 times the basic keying rate, 4.375 baud. The wider submodes have proven very useful on propagation paths with large Doppler spread. For example, JT4F is frequently used for Earth-Moon-Earth (EME) communication on the 10 GHz band.

JT9

The JT9 mode uses eight tone frequencies to convey message information, one additional tone for synchronization, and the same convolutional code as JT4. The message frame includes 69 data symbols and 16 synchronizing symbols. Submodes JT9A-H have tone spacings at multiples 1, 2, 4, 8, 16, 32, and 64 times the 1.736-baud keying rate. JT9A (often called simply JT9) uses less than 10% the bandwidth of JT65, and for steady, undistorted signals is about 2 dB more sensitive. These characteristics made JT9 popular on the 630 meter band, but its use there and on 2200 meters is now being supplanted by FST4.

JT65

The JT65 protocol is the oldest of the weak-signal structured modes; a detailed description was published in *QEX* in 2005 (see References and Bibliography). The protocol uses a (63,12) Reed-Solomon code with alphabet size $Q = 64$. Modulation uses one synchronizing tone and 64 data tones, with data and sync information interspersed according to a pseudo-random sync pattern. In *WSJT-X* decoding is accomplished using the Franke-Taylor soft decision algorithm (see References and Bibliography). Special features can be used to convey an EME-style "OOO" signal report and short messages interpreted as RO, RRR, and 73. Submodes JT65B and JT65C use tone spacings 2 and 4 times larger than JT65A. JT65 is widely used for EME on VHF and higher bands. It was popular for low-power DXing at MF and HF, as well, but has been largely superseded on those bands by FT8.

Q65

Q65 is a jack-of-all trades mode intended for use on VHF, UHF, and microwave bands for EME and other extreme weak-signal paths. On paths with Doppler spread more than a few Hz, the weak-signal performance of Q65 is the best among all *WSJT-X* modes. Its error-correcting code was designed by Nico Palermo, IV3NWV (see References and Bibliography). Q65 uses 65-FSK with the lowest tone dedicated to synchronization. The message frame includes 63 data symbols and 22 sync symbols. Five different keying rates are available, yielding T/R sequence lengths of 15, 30, 60, 120, and 300 s. Submodes A through E provide tone spacings 1, 2, 4, 8, and 16 times the keying rate. Suitable choices of sequence length and tone spacing make Q65 a highly effective mode for tropospheric scatter, ionospheric scatter, rain scatter, trans-equatorial propagation (TEP), and EME on VHF and higher bands, as well as for other types of fast-fading signals. As a few particular examples, submodes Q65-15C and Q65-30A are widely used for TEP and ionospheric scatter, respectively, on the 50 MHz band; Q65-60C for EME on 1296 MHz; and Q65-60D for both rain scatter and EME at 10 GHz.

WSPR

The WSPR mode was designed as a propagation probe rather than for making two-way contacts. It uses the same convolutional code as JT4 but differs from the other structured modes in *WSJT-X* by using message lengths of 50 information bits and two-minute T/R sequences. Message packets normally include a 28-bit call sign, a 15-bit grid locator, and 7 bits to convey transmitter power in dBm (decibels above one milliwatt). Alternative formats can convey a compound

call sign and/or a 6-digit grid locator, using a two-transmission sequence. Typical WSPR usage was described in *QST* in 2010 (see References and Bibliography).

15.4.4 WSJT-X Fast Modes

The fast modes in *WSJT-X* aim to take advantage of brief propagation enhancements that bring a signal up to useful levels for a very short time. Keying rates and occupied bandwidths are much larger than for the slow modes so that full messages can be conveyed in these very short intervals. **Table 15.7B** lists essential parameter values for the *WSJT-X* fast modes. The last column gives the time required to transmit a message once at the specified keying rate. In these modes the transmitted information is repeated for the full duration of a T/R sequence.

JT9 Fast

Fast submodes JT9E-H differ from their slow JT9 counterparts by using much higher keying rates and wider tone spacings. Otherwise, the coding, modulation, and synchronization schemes are the same as for the slow JT9 modes. JT9 fast modes have proven useful for such propagation types as ionospheric scatter and weak multi-hop sporadic E on the 6 meter band.

MSK144

Messages in MSK144 contain 77 information bits and a 13-bit CRC, and use the same types of user message as FT8 and FT4. Forward error correction is accomplished with a (128, 90) LDPC code. Two 8-bit synchronizing sequences are added to make a message frame 144 bits long. Modulation uses *offset quadrature phase-shift keying* (OQPSK) at 2000 baud, so the frame duration is 144/2000 = 0.72 second. With the advantages of strong error correction and an effective character transmission rate around 250 characters per second (see References and Bibliography), MSK144 has become the dominant mode for amateur meteor-scatter contacts.

15.4.5 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 2,400 bit/s for voice data with an additional 1,200 bit/s for Forward Error Correction for a total 3,600 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/others/ard9800.html**.

FREEDV

FreeDV (initial release Dec 2012) is available for *Windows*, *Linux*, and *MacOS* clients, and allows any SSB radio to be used for low-bit-rate digital voice. Since its release, *FreeDV* has added new modes to choose from based on current band conditions. In addition, new features have been added to make set up and operation easier including simultaneous decoding for all HF modes. PSK Reporter integration provides frequency, call sign, and grid square on a map-based world view of stations being copied.

FreeDV speech and call sign data is compressed, which then modulates a mode-dependent OFDM/QPSK signal which is then applied to the microphone input of an SSB transceiver. On receive, the signal is received as SSB, demodulated to audio, then further demodulated, and decoded by *FreeDV* running on a PC.

FreeDV is entirely open source — even the voice codec. This makes it unique among amateur radio digital voice systems which typically rely on a proprietary voice codec that is not available to hams for experimentation.

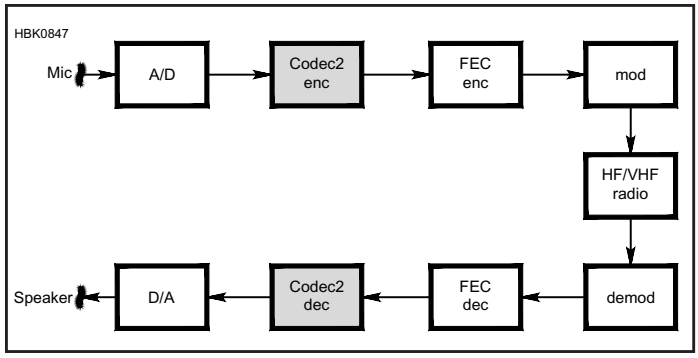
FreeDV was coded from scratch by David Witten, KDØEAG (GUI, architecture), and David Rowe, VK5DGR (*Codec2*, modem implementation, integration). Recently, additional development was coded by Mooneer Salem, K6AQ (simultaneous mode receive, PSK Reporter support, Flex 6000-series Teensy dongle support, various usability enhancements). The *FreeDV* design and user interface is based on *FDMDV* (2007), developed by Francesco Lanza, HB9TLK. The large team of reviewers and beta testers who have supported the development of *FreeDV* are credited on the **freedv.org** home page.

Key *FreeDV* Features

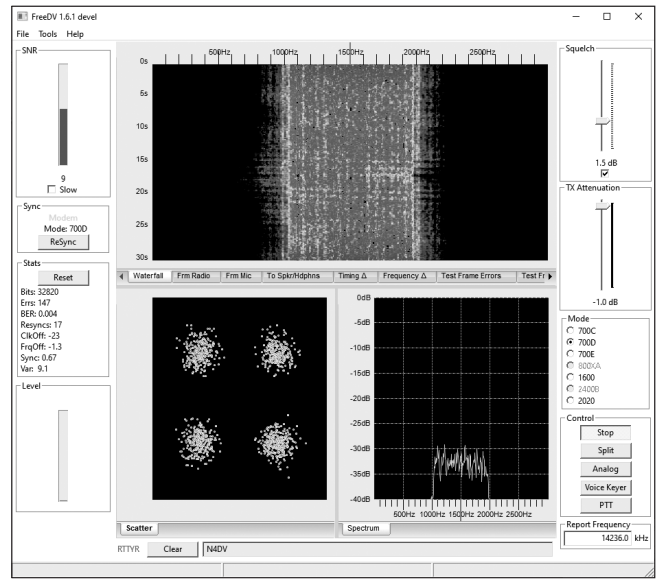
- Waterfall, spectrum, scatter, and audio oscilloscope displays
- Adjustable squelch
- TX attenuation control (drive level)
- Fast/slow SNR estimation
- Microphone and speaker signal audio equalizer
- Control of transmitter HamLib and PTT via RS-232 levels

Table 15.8
Implementations of *FreeDV*

Mode	*Codec*	*Modem*	*RF BW*	*Raw bits/s*	*FEC*	*Text bits/s*	*SNR min*	*Multipath*
1600	Codec2 1300	14 DQPSK + 1 DBPSK pilot	1125	1600	Golay (23,12)	25	4	poor
700C	Codec2 700C	14 carrier coherent QPSK + diversity	1500	1400	- -		2	good
700D	Codec2 700C	17 carrier coherent OFDM/QPSK	1000	1900	LDPC (224,112)	25	–2	fair
700E	Codec2 700C	21 carrier coherent OFDM/QPSK	1500	3000	LDPC (112,56)	25	1	good
2020	LPCNet 1733	31 carrier coherent OFDM/QPSK	1600	3000	LDPC (504,396)	22.2	2	good



(A)



(B)

Figure 15.7 — Typical *FreeDV* system architecture (A), *FreeDV* 1.61 screenshot during reception (B), and *FreeDV* SM1000 standalone adapter — no host PC needed (C).

• Works with one (receive only) or two (transmit and receive) sound cards

(For example, a built-in sound card and USB headphones)

FreeDV Overview

There are five HF implementations of *FreeDV* summarized in **Table 15.8**. The characteristics of each variation are as follows:

FreeDV 1600: 1,275 bit/s voice coding; with 300 bits/s FEC using original FDMDV modem and 75 Hz carrier spacing. Pilot carrier with 2X power useful for tuning and robust synchronization. Good voice quality. 4000 Hz speech BW.

FreeDV 700(C): 700 bit/s voice coding; with no FEC. Improves performance in fast-fading channels using a simple transmit diversity scheme that transmits the signal twice at two different frequencies. Similar voice quality to mode 1600 but at one-half the bit rate. 4000 Hz speech BW.

FreeDV 700(D) (default mode): 700 bit/s voice coding; with Low Density Parity Check (LDPC) FEC. Sensitive to tuning, must be within ±60 Hz of the transmit frequency. Most robust mode. Error free at –2 dB SNR approximates SSB quality. 4000 Hz speech BW.

FreeDV 700(E): 700 bit/s voice encoding; with shorter 80 ms frame size to reduce latency and sync time. Best in multi-path fast fading channels with up to 4 Hz Doppler spread and 6 ms delay spread.

FreeDV 2020: Experimental codec based on the LPCNet neural net (deep learning) synthesis engine developed by Jean-Marc Valin. Capable of providing 8 kHz audio BW in a narrow 1600-Hz bandwidth. Highly experimental, believed to be the first use of neural net vocoders in a real world over-the-air system.

FreeDV 800XA (Note: for VHF/UHF only): 700 bit/s codec plus 100 bits/s for sync. 4FSK modulation suitable for Class C power amplifiers.

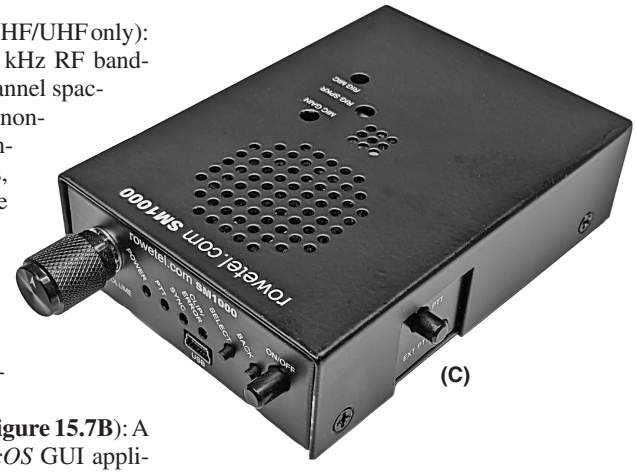
FreeDV 2400B (Note: for VHF/UHF only): 2,400 bit/s encoding with 5 kHz RF bandwidth requiring 6.25 kHz channel spacing. 4FSK modulation with non-coherent demodulation. Symbol rate of 1,200 symbols/s, tone spacing 1200 Hz, frame period 40 ms, unique word 16 bits/frame. Codec2 at 1300 with 52 bits/frame and 28 bits/frame spare.

FreeDV is composed of two primary software components:

1) The *FreeDV* GUI (see **Figure 15.7B**): A *FreeDV Windows/Linux/MacOS* GUI application. The *FreeDV* GUI software runs on any PC and requires two sound cards, for example a USB rig interface and a USB headset.

2) The *FreeDV* API: An open source, C-callable software library, that has interfaces for audio and modem samples. The API can be linked into third-party SDR programs that wish to support *FreeDV*.

A standalone hardware all open-source implementation of *FreeDV* (SM1000) is also available that does not require a PC host. Software/hardware developed by David Rowe, VK5DGR, with CAD PCB and mechanical enclosure by Rick Barnich, K8BMA. This small (see Figure 15.7C) adapter is excellent for portable and mobile work. Includes a built-in microphone and speaker with PTT switch. Easily connects to a SSB radio with RJ-45 cable or 3.5 mm plugs.



(C)

FreeDV Architecture

Figure 15.7A shows a typical *FreeDV* system using 1600 mode. Voice signals from a microphone are sampled by the ADC, typically at 8 kHz with 16-bit resolution. A speech encoder (*Codec2*) then compresses the speech to a low bit rate, for example 1,300 bit/s. FEC bits are then added to protect against errors encountered in the channel. This may increase the bit rate to 1,600 bit/s. A modulator (labeled "mod") then converts the bit stream to tones that can be passed via a radio channel. Over the channel the digital speech signal will encounter noise and other impairments such as fading and frequency offsets.

On the receive side, the demodulator (labeled "demod") then converts the tones back into a bit stream. The FEC decoder attempts to correct channel errors before passing the payload compressed voice information to the voice decoder (labeled "codec 2 dec"). The output of the voice decoder is an 8 kHz, 16-bit/sample sequence that is converted back into analog speech by the DAC and played through a speaker or headphones.

The screenshot in Figure 15.7B shows the software in operation using the 700D mode.

A waterfall display at the top shows the individual carriers in the signal. At the bottom right is a spectrum display of the signal. The constellation diagram of the QPSK signal is shown at lower left to give the operator an idea of the signal-to-noise performance along with the SNR meter at the upper left. More operating controls and status values are also displayed. More details and the software can be found at **freedv.org** as well as written and video setup guides.

15.4.6 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 15.9
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 15.9**.

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, at **hflink.com/standards/FED_STD_1045A.pdf**.

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 **hflink.com/software**. Much more ALE information and real-time data is available online at **hflink.com**.

15.5 Network Modes and Systems

This section describes both digital modes and systems that use them to build computer-to-computer networks. 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 systems such as D-STAR and System Fusion, network communication over the air uses a specific mode. Other systems such as Winlink may use several different modes to communicate. The features of these systems are described along with the modes and protocols used to implement communications within the network.

15.5.1 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 15.10**. 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.

Table 15.10
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 15.11
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	

This description is by no means exhaustive and more information can be found online and in every networking textbook. **Table 15.11** shows the placement of commonly recognized protocols within the OSI layered structure.

15.5.2 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.

15.5.3 The Terminal Node Controller (TNC)

Hardware *terminal node controllers* (TNCs) such as the venerable TAPR TNC 2 are now mostly legacy devices. Their limited processing power, modem ICs, and limited interface options have been largely rendered obsolete by the use of audio interfaces coupled with software on a more powerful host computer. Coupled with a simple audio interface and software such as *Dire Wolf*, a software modem running on a Raspberry Pi is much more capable than a hardware-based TNC.

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

Modern software-based TNCs such as *Dire Wolf* provide a virtual KISS connection, typically as a TCP/IP socket. Functionally, that is equivalent to an application such as APRS, running on a host computer, connecting over a serial link to a KISS TNC.

15.5.4 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 **Figure 15.8**, and their timing is as shown in **Table 15.12**.

• 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 15.12

• CRC: The CRC is calculated according to the CCITT standard, for the data, status and CRC.

The PACTOR acknowledgment signals are shown in **Table 15.13**. 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.

Table 15.12
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

Table 15.13
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 15.14
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

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 15.14**.

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.

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.

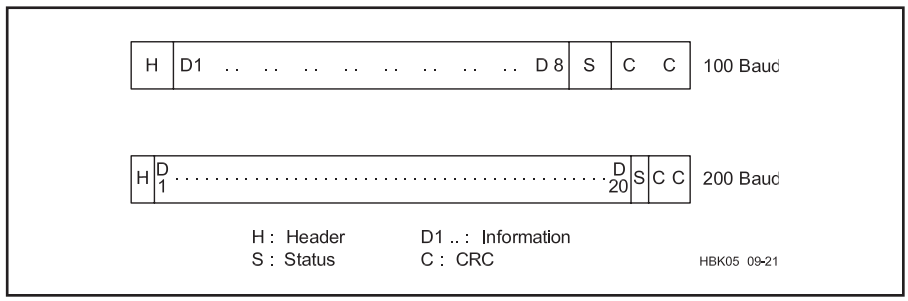


Figure 15.8 — PACTOR packet structure.

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.

15.5.5 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.

15.5.6 PACTOR-III and PACTOR-IV

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**.

PACTOR-IV

PACTOR-IV is 1.5 – 3 times faster than PACTOR-III and has 10 speed levels. Its emission designator is 2K20J2D or 2K40J2D (at a symbol rate up to 1800 symbols per second), with a maximum speed of 5513 bps.

Because PACTOR-IV uses 1800 symbols per second (baud), and current FCC rules permit a maximum of 300 baud on amateur radio bands below 28 MHz, PACTOR-IV is not allowed for use by US amateur radio operators. The FCC has allowed a number of Special Temporary Authority (STA) usages of PACTOR-IV during emergency situations such as during the aftermath of major hurricanes. Outside the US, PACTOR-IV is used by individual amateur radio operators and Winlink stations.

15.5.7 VARA

VARA was created by Jose Alberto Nieto Ros, EA5HVK as a robust and fast "sound card" data mode for reliable file transfers. VARA is most typically used by Winlink user stations and Winlink Remote Mail Servers (RMS) for faster and more reliable file transfers than previous methods such as WINMOR and packet radio. (See the Winlink section later in this chapter.)

VARA is not compatible with any other data communications mode (such as packet radio); VARA stations can only communicate with VARA stations. VARA HF 4.0 has been adopted by the Amateur Radio Safety Foundation for the worldwide Winlink Amateur Radio email system as a co-equal (with ARDOP) on HF. (**www.winlink.org/content/vara_hf_40_network_cutover_june_30_2020**) VARA FM has been similarly adopted for Winlink access on VHF/UHF. Winlink *Radio Mail Server* (RMS) stations accept connections from users using VARA, and the *Winlink Express* client software includes direct compatibility with VARA HF and VARA FM software. For example, the *Winlink Express* software has specific sections for easy integration with VARA software.

VARA achieves its robustness and speed by incorporating a number of techniques:

- Orthogonal Frequency Division Multiplexing (OFDM) generates multiple subcarriers within the audio signal. (See **Figure 15.9**.)
- Varying modulation methods (Modulation Index) – FSK through 256 QAM depending on mode and quality of channel between stations.
- Robust handshake between transmitting and receiving station to negotiate best possible speed on each transmission.
- Huffman data compression.
- Turbo Codes Forward Error Correction.

Of these techniques, the most significant is OFDM, which uses "independent subcarriers." (See the Multi-Carrier Modulation section of the **Modulation** chapter.) Subcarriers are independently managed within the transmitted audio signal. If one or more subcarriers are damaged in transmission such as a noise burst at a specific audio frequency, other subcarriers are unaffected.

Audio levels are a consistent issue with "sound card" modes and VARA addresses this by including a "Tune" function to adjust

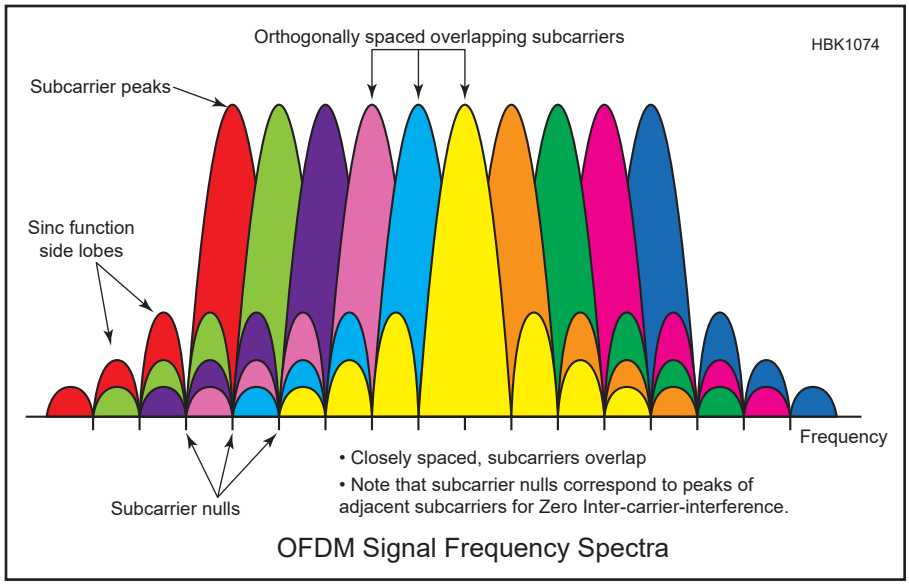


Figure 15.9 — The orthogonal frequency domain multiplexing (OFDM) structure of VARA modulation.

transmitted audio levels into the radio, and an "Autotune" function to test audio levels with another VARA station. If audio levels are too high or too low, an error message is displayed. VARA also includes a "VU Meter" to monitor the receive audio level (from the radio) into the audio interface.

If a US call sign is used, VARA will transmit a CW identification. When a non-US call sign is used, CW identification is optional.

VARA RESOURCES

All versions of the software, along with specification documents, and guides to configuring and using VARA are maintained at the author's website, **rosmodem.wordpress.com**. Videos and presentation are also available. Besides those linked at the author's website, there are numerous videos available on YouTube that offer presentations on VARA, and demonstrate setup, troubleshooting, and operation of all three VARA variants. For discussion of VARA HF and VARA FM, there is an independent email list that is quite active (1,000+ subscribers) at **groups.io/g/VARA-MODEM** (vara-modem@groups.io).

VARA IMPLEMENTATIONS

There are three variants of VARA: VARA HF, VARA FM, and VARA SAT. Each variant is different software (not three modes of the same software). Tables showing the symbol rate, number of carriers, modulation, and net bit rate for all levels of the three VARA variants is available in the online document *VARA 4.3 quick guide.pdf*. The document is available from several repositories — an online search for the name will obtain the document from the repository.

VARA HF

VARA HF is optimized for narrow, noisy channels of the HF bands, and has three modes:

VARA HF 2750 (Tactical) is optimized for commercial and military HF radios that can use wider channels (2750 Hz) to achieve data transfer speeds up to 8,489 bps.

VARA HF 2300 (Standard) is optimized for typical amateur radio HF radios (2300 Hz channels) to achieve data transfer speeds up to 7,050 bps.

VARA HF 500 (Narrow) is optimized for use on HF bands where channel width is limited to 500 Hz. VARA HF 500 can achieve data transfer speeds up to 1,543 bps.

VARA FM

VARA FM is optimized for VHF/UHF radios and the wider, and quieter, channels available on the VHF/UHF bands (and implemented in VHF/UHF FM transceivers). VARA FM has two modes — Wide and Narrow.

VARA FM Wide is optimized for VHF/UHF radios that have a "flat audio" interface. This is usually a 6-pin Mini-DIN Data connector providing connection to the transmit modulator and receive discriminator (**www.soundcardpacket.org/7miniDIN.aspx**). This connection has been traditionally used for packet radio 9,600 bps Frequency Shift Keying (FSK) modulation.

For maximum data throughput, VARA FM Wide also requires a wide-bandwidth audio interface. For example, the Masters Communications series of Digital Radio Adapters offers various combinations of bandwidth and connectivity. (**www.masterscommunications.com/products/radio-adapter/dra/dra-features.html**)

If both the transmitting and receiving radios are configured for VARA FM Wide, and both radios have wide-bandwidth audio interfaces, data transfer speeds up to 25,210 bps are possible. Note the significant speed difference between typical "fast" packet radio (9,600 bps) versus VARA FM Wide (25,210 bps) — more than 2.5 times faster.

VARA FM Narrow is optimized for more typical VHF/UHF radios that do not have a "flat audio" interface, connecting via the microphone input and speaker output where CTCSS filtering and pre-emphasis and de-emphasis processing occurs in the audio path.

An advantage of VARA FM Narrow is that almost any audio interface is suitable, including radios with integrated audio interfaces that are not capable of VARA FM Wide. Some audio interfaces that cannot achieve full VARA FM Wide speeds *can* achieve speeds greater than VARA FM Narrow; thus, experimentation is warranted when configuring VARA FM on a particular radio/audio interface combination. A radio using VARA FM Narrow can achieve data transfer speeds up to 12,750 bps (depending on channel conditions).

A compelling feature of VARA FM is that VARA FM Wide and VARA FM Narrow are fully (and automatically) interoperable, unlike 1,200 bps and 9,600 bps packet radio. During the initial connection between two VARA FM stations, the two stations negotiate their fastest common speed. If a station configured for VARA FM Wide connects to a station configured for VARA FM Narrow, the "Wide" station "downshifts" to VARA FM Narrow for data transfer with the "Narrow" station.

Other features of VARA FM include:

- Can connect via one or two VARA FM stations acting as digipeaters (VARA digipeaters perform the same function as packet radio digipeaters but are not compatible with packet radio signals.)
- Can connect via FM voice repeaters. (Repeaters can be "optimized" for VARA but the details are not publicly available.)
- Optimizations for use with portable radios, including TX delay to accommodate slow transmit/receive switching.
- Some VHF/UHF transceivers have poor audio response at low frequencies and VARA FM Narrow mitigates that issue by "shifting the frequency range middle frequency up 200 Hz."
- Ping function to verify connectivity between two VARA FM stations. "Ping" used in VARA FM is a functional description to perform a simple connectivity test. VARA FM is not compatible with the Ping function using TCP/IP over the air.
- Monitor Mode displays all VARA FM stations heard, including their SNR.
- Transmit Delay (TXD) is automatically optimized.

VARA SAT

VARA SAT is based on VARA HF with optimizations for satellite operations, including the unique aspects of QO-100 such as significant latency. (QO-100 is the amateur payload on the geosynchronous Es'hail-2 satellite over Europe and Africa.) Optimizations in VARA SAT include variable transmit power, delays in equipment key-up (software defined transceivers, amplifier key-up, transverter inline, etc.), transponder operation, and adjustments for Doppler effects.

INTEGRATING VARA WITH OTHER SOFTWARE

VARA provides interoperability with other software using the Telnet protocol via TCP/IP socket 8300 for commands and socket 8301 for the data stream. The author provides a specification document, *VARA Protocol Native TNC Commands*, at **rosmodem.wordpress.com** to describe the various parameters and variables that can be sent and queried to interoperate with VARA.

VARA also provides a "KISS" Interface so that software designed to work with packet radio TNCs (and software implementation of TNCs) that offer a KISS interface can use VARA. (See the Packet Radio section of this chapter.) The author provides a specification document, *VARA KISS Interface*, to describe the unique aspects of the interface.

A unique aspect of VARA's KISS interface is that when other software (such as APRS) transmits (via KISS) an AX.25 Unnumbered Information (UI) frame, such transmissions are transmitted by VARA in a "robust VARA Broadcast frame" (114 bytes).

In VARA HF, KISS frames are transmitted using "VARA HF 500 Hz Level 5" — Symbol Rate 94, 2 carriers, 4PSK, data rate of 177 bps. "VARA HF 500 Hz Level 5" can be received by VARA HF 2750, VARA HF 2300, and VARA HF 500.

Note that if VARA HF is configured to work with a Winlink software client, the VARA HF's KISS interface is disabled. The author states, "This is to disable use of APRS by Winlink gateways operating under automatic control. Winlink gateways have position and frequencies published on the Winlink web site, so a position beacon is unnecessary."

In VARA FM, KISS frames are transmitted using "VARA FM Narrow Level 1" — Symbol Rate 42, 14 carriers, 4PSK, data rate of 549 bps. "Narrow Level 1" is presumably used for maximum compatibility with all VARA FM stations, including those stations that have not purchased a registration key and thus can only operate at "Level 1." This detail in the author's document "VARA Kiss Interface.pdf" "Rev, October 21th 2021" is ambiguous.

In VARA SAT, KISS frames are transmitted using "VARA SAT Level 3" — Symbol Rate 94, 20 carriers, FSK, data rate of 175 bps.

VARA-COMPATIBLE SOFTWARE

VARA's author provides two utility programs specifically for use with VARA:

• *VARA Chat* — Chat and file transfers, and

• *VARA Terminal* — Dumb terminal for use with Bulletin Board Systems.

Other software that directly supports VARA (specifically mentioned in various VARA documentation) includes *BPQ32*, *JNOS*, *Winlink Express, RMS Packet, RMS Trimode,* and *vARIM*.

LIMITATIONS OF VARA

• VARA is, essentially, a file transfer protocol. It does not have basic network functionality inherent in packet radio such as "routing" (with the exception of digipeater functionality).

• VARA is proprietary software; thus, any bug fixes, updates, and improvements are only available from the author. As described in the author's *VARA HF Modem Specification Revision 1.0.0 Oct30 2017,* VARA is a proprietary system developed by Jose Alberto Nieto Ros, EA5HVK and can be used under shareware license.

• VARA runs only on Windows but there have been some projects to run VARA FM on Linux using *WINE* (Windows emulation) software. (**k6eta.com/linux/installing-rms-express-on-linux-with-wine**)

• Full VARA functionality requires the user purchase a "registration key" for each unique station call sign. (There may be up to 15 SSIDs per call sign.) The registration key is purchased from within VARA HF, VARA FM, or VARA SAT software. One registration key will work with all three VARA variants. If a registration key is not purchased, VARA will only function at "Level 1" of each VARA variant. Example, VARA FM Level 1 is 566 bps (Wide) and 549 bps (Narrow) versus (with a registration key) 25,210 bps (Wide) and 12,750 bps (Narrow).

• Some updates of VARA software are not compatible with previous versions. For maximum compatibility within a group of VARA stations, the same version of VARA software should be used.

15.5.8 Amateur Radio Digital Open Protocol (ARDOP)

Development on ARDOP began in 2015 as a replacement for the earlier Winlink Message Over Radio (WINMOR) "transport" protocol used in Winlink clients and servers. (WINMOR is described in the paper "Legacy Digital Modes" in this chapter's online content.) Compatibility with WINMOR was not a requirement for ARDOP, and development of WINMOR has ceased.

The overall goal of ARDOP is to create a radio modem that is sufficiently capable that expensive, proprietary hardware radio modems would not generally be required except for the most demanding requirements (highest possible speeds). The key aspect of ARDOP is "Open;" the stated goal of ARDOP was, "The specification and the protocol will be released to the public domain." That goal has been met.

The details of the ARDOP protocol are well defined in *Amateur Radio Digital Open Protocol (ARDOP) Specification*, by Rick Muething, KN6KB. The current revision is 0.3.1, Mar 25, 2015.

There is an active ARDOP Support Mailing List with nearly 1,500 subscribers as of early 2022 (**ardop.groups.io/g/main**) that supports three active specialized subgroups with their own active mailing lists: Applications, Developers, and Users.

ARDOP REQUIREMENTS AND GOALS

High level requirements for ARDOP:

• Hardware compatibility — accommodate frequency accuracy of typical HF and VHF/UHF radios, majority of audio interfaces ("Sound cards"), all popular methods of Push-To-Talk (PTT) control, and flexible host interface (TCP/IP socket or serial).

• Use channel width of 200, 500, 1000, and 2000 Hz (negotiated between client and server).

• Optimize choices of channel size, modulation method, forward error correction (FEC) to maximize the fastest possible error-free data transfers.

• Compliance with the FCC symbol rate limit of 300 baud below 28 MHz (§97.307(f)(3)).

• Strong resistance to multipath distortion.

• Listen-before-transmit mechanism to avoid causing interference with a communication in progress.

• Agnostic implementation — software on host computer ("Sound card" mode) or dedicated processor/DSP (real-time or no operating system).

• Usable for both HF (single sideband, AFSK) radios and VHF/UHF (FM) radios.

• Both Forward Error Correction (FEC) and Automatic Retry Request (ARQ) for error-free data transfer (reliable transport).

• Automatic adjustment of timing parameters, including for use with VHF/UHF voice repeaters, round trip timing, latencies in the operating system (if any), TX delay, networking, etc.

Two interesting goals of ARDOP are to support both file transfers between two stations using ARQ, and a broadcast/multicast mode using FEC.

ARDOP IMPLEMENTATION

There are two stable "code" implementations of ARDOP:

• ARDOPc v1 by John Wiseman G8BPQ, for Linux (**www.cantab.net/users/john.wiseman/Documents/ARDOPC.htm**)

• WIN_ARDOP v1 by Rick Muething, for Windows (**ardop.groups.io/g/users/files**)

Progress on ARDOP development beyond these two v1.0 implementations appears to be stalled, possibly due to the rapid evolution of VARA as a competitor mode to ARDOP. (**github.com/la5nta/pat/issues/148#issuecomment-483914337**)

ARDOP has been implemented in the following Winlink software:

Pat is "Winlink anywhere," a Winlink client developed as an open-source project with versions available for Windows, Mac OS, and Linux. (**getpat.io**) The Linux version of *Pat* is popular because it has been ported to Raspberry Pi's Linux and is now included with many amateur radio software distributions for that inexpensive, capable computer. Developers for *Pat* use GitHub for *Pat* development. (**github.com/la5nta/pat**) *Pat* is unaffiliated with the Winlink Development Team.

Winlink Express (WE, **winlink.org/WinlinkExpress**) is the "preferred Winlink radio email client" because it "does it all," including many radio access methods other than ARDOP. WE is developed and supported by the Winlink Development Team. *Winlink Express* only operates on Windows (newer than Windows XP). (See the Winlink section later in this chapter.)

RMS Trimode (**www.winlink.org/content/rms_trimode**) is software to host a Winlink Radio Mail Server (RMS) — HF gateway stations in the worldwide Winlink system. ARDOP is one of several supported radio connection methods in *RMS Trimode*, including proprietary modes that require hardware modems.

ARDOP HARDWARE MODEM — TNC-PI9K6

The ARDOP goal of "Agnostic implementation … dedicated processor/DSP (real-time or no operating system" was realized. John Wiseman, G8BPQ implemented ARDOP (**www.cantab.net/users/john.wiseman/Documents/ARDOPTeensy.html**) on a Teensy 3.6 microcontroller (**www.pjrc.com/store/teensy36.html**) that was the basis for the Coastal Chipworks TNC-Pi9K6 "TNC" HAT (Hardware Attached on Top) board (**www.tnc-x.com/TNCPi9k6.htm**) for a Raspberry Pi computer. Although Coastal Chipworks is no longer in business, another version of the TNCPi-9k6 is available (**www.etsy.com/listing/775608115/tnc-pi9k6-raspberry-pi-hat-wteensy-32**) from the West Valley Amateur Radio Club (WVARC, Sun City, Arizona; **westvalleyarc.com**). It is unknown if the WVARC version of the TNC-Pi9k6 is compatible with G8BPQ's implementation on ARDOP on the Coastal Chipworks version of the TNC-Pi9k6.

ARDOP PERFORMANCE COMPARISON

The undated paper by Tom Whiteside, N5TW: "A comparison of Winlink digital mode performance based on simulation results using the Teensy IONOS Simulator" (**winlink.org/sites/default/files/downloads/a_winlink_digital_mode_performance_comparison_based_on_the_ionos_hf_vhf_channel_simulator_-_may_18_2020.pdf**) gives some perspective of ARDOP's overall performance compared to other data modes:

"Executive Summary:

"Hardware SCS modems running PACTOR 2,3 and 4 were evaluated as were sound card modes WINMOR, ARDOP and VARA across a variety of channel models and across a range of signal to noise conditions for HF. VARA testing included a new VARA 500 500 Hz mode. VARA FM and AX.25/FX.25 packet were simulated for a VHF channel.

"Spoiler alert: The SCS modems did very well as you would expect. The much less expensive VARA HF did especially well across the range of conditions tested. The SCS modems and VARA could run long test cases without losing a connection while ARDOP and WINMOR were slow and unable to run many of the cases. VARA FM crushed AX.25/FX.25 VHF cases.

…

"WINMOR and ARDOP were really breakthrough modes when they were first created at a time when computer/sound card technology was much less advanced than today, but their performance lags the other modes by a great deal. They were also much less reliable across the multipath cases with incredibly low rates and frequent connection drops."

15.5.9 Packet Radio

Amateur Packet Radio began in the late 1970s when groups of Canadian hams started experimenting with a new mode of communication. American hams had a slow start because the FCC did not authorize use of ASCII until 1980. Tucson Amateur Packet Radio (TAPR; **tapr.org**) refined the protocol and called it AX.25 because it was based on the commercial X.25 standard.

Packet radio offers several advantages over non-structured, character-based protocols:

• Transmissions are quick bursts (packets) rather than single characters

• Source and destination routing information in each packet.

• Error checking to detect corruption

• Connection-oriented protocol for guaranteed delivery or notification of failure (*reliable transport*)

• Digital repeaters (*digipeaters*) to extend range.

• Multiple packet radio communications can use the same frequency concurrently.

PACKET RADIO OVERVIEW

Packet radio is a method of exchanging data between stations that is most often used for direct, keyboard-to-keyboard connections and the transfer of small files between stations, either between two live operators or between an operator and a bulletin board system (BBS) or email servers.

A network of packet stations is built using the AX.25 protocol. The AX.25 protocol specification (version 2.2, July 1998) can be found on the TAPR website at **www.tapr.org/pdf/AX25.2.2.pdf**.

Despite its low data rate, Bell 202 modulation at 1,200 bps (bits/sec) has remained the standard for VHF/UHF operation in most areas because it is inexpensive and performs reliably over a standard FM voice channel. 9,600 bps is also popular on the VHF/UHF bands, although it is more technically demanding and requires about 5 kHz of bandwidth for higher speed data. This audio connection is usually available as a separate output on an accessory or data connector**.** At HF frequencies, Bell 103 modulation is used, due to the FCC rule (§97.307(f)(3)) limiting a data signal's symbol rate to 300 bauds below 28 MHz. (1200 bauds may be used on 10 meters.) As of early 2022, FCC action is pending on action to change the rules regarding maximum data symbol rates.

In the US, 145.01, 145.03, 145.05, 145.07, and 145.09 MHz.are the usual frequencies used for packet radio. Packet can be used on VHF/UHF channels as a transport method for applications like *Winlink Express* which require reliable transport (error-free) of small data files and messages.

TERMINAL NODE CONTROLLER (TNC)

The AX.25 protocol was first implemented with specialized hardware called a *Terminal Node Controller* (TNC), composed of a modem and a microprocessor. TAPR offered a complete kit, the TNC-1, in 1983, followed by the TNC-2 which was smaller and cheaper to manufacture. The design was then licensed to multiple manufacturers.

In the late 1980s, numerous TNC variations became available. Manufacturers added higher speed modems (2,400 and 9,600 bps), built-in bulletin board system software (BBS), transmission of telemetry and weather station data, and additional modes such as weather satellite image reception. A slide show history of TNC evolution can be found at **github.com/wb2osz/direwolf-presentation**.

KISS

The original text-based user interface was designed for operators communicating between dumb terminals. This interface was not well suited for communication with a computer application. Commands, responses, and data were all combined in a single stream. The format varied by manufacturer and depended on the operating mode of the TNC.

The KISS protocol (Keep It Simple, Stupid; **www.ka9q.net/papers/kiss.html**) was invented in 1986 and better suited for communicating with applications. Most of the protocol handling is moved to the host PC. The TNC does little more than converting between the KISS protocol and the frames (aka "packets," see section below) sent over the air. KISS-only TNCs are the standard today because the connection-oriented functions of the protocol are moved to the computer or not needed at all as with APRS (see the next section).

KISS is very simple and doesn't provide any information about the TNC status or what is happening on the radio channel. For example, an application might send a dozen packets to a TNC to be transmitted. Since KISS does not support any form of flow control, there is no standard way for the application to know whether they have been transmitted, are still in the transmit queue, waiting because of a busy channel, or discarded.

SOFTWARE TNC

The physical TNC device can be eliminated and replaced by software on a host PC with a sound card, similar to *WSJT-X*. The "software TNC" approach is very flexible. When access to a TNC is available over a computer network (Ethernet, WiFi), rather than being constrained by direct RS-232 cable connection, it is known as a *network TNC*.

As shown in **Figure 15.10**, even a single-board Raspberry Pi computer can run multiple digipeaters at the same time, send telemetry data from a weather station, and act as an internet gateway (IGate), connecting your local radio network to others around the world over the internet.

The early "soundcard modem" software produced poor results, giving this approach a bad reputation. At one time, these applications needed special hardware for acceptable results but that is no longer true with today's computers.

Currently, two software TNC implementations, *UZ7HO SoundModem* (**uz7.ho.ua/packetradio.htm**) and *Dire Wolf* (**github.com/wb2osz/direwolf**), outperform all of the hardware modem ICs. The test protocols and results are available at **github.com/wb2osz/direwolf/blob/dev/doc/WA8LMF-TNC-Test-CD-Results.pdf**.

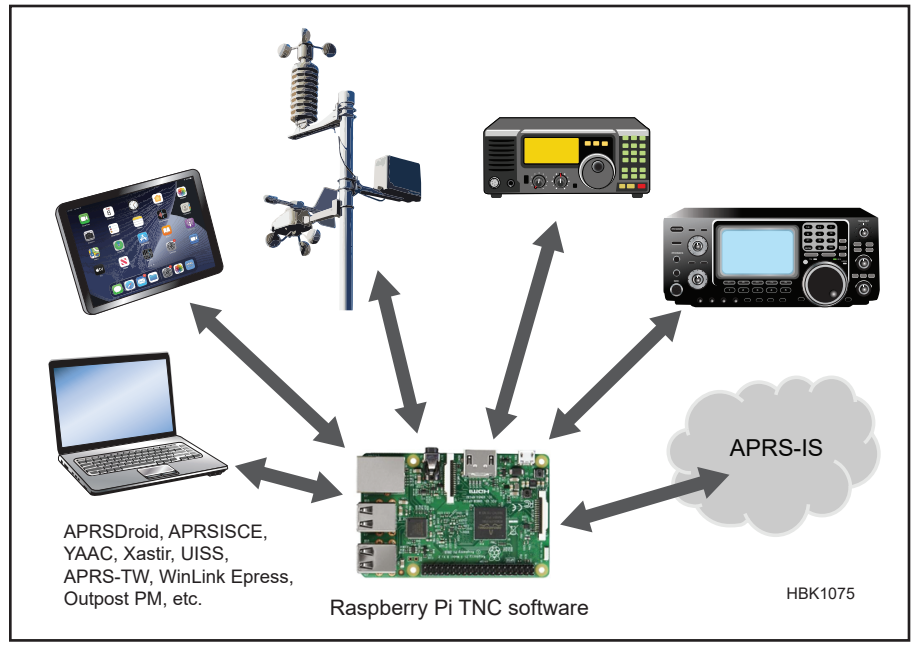


Figure 15.10 — A network TNC is not constrained by the traditional RS-232 interface to a PC and offers many interfacing possibilities with multiple applications and devices at the same time.

New Packet Radio

An IP-based high-speed version of packet radio, New Packet Radio, was created by F4HDK in 2019 (**hackaday.io/project/164092-npr-new-packet-radio**). Operating on 70 cm, the protocol supports 50 – 500 kbps and fully bi-directional links. Due to symbol rate restrictions in the US on 70 cm, a slower configuration of the protocol is available. The local interface is Ethernet so no special host software is required. The modem software is open-source, as well.

DIGIPEATERS & NETWORKING

Packet radio range can be extended by using digital repeater stations (digipeaters). Unlike voice repeaters that simultaneously receive on one frequency and transmit on another, digipeaters usually use "store and forward" on a single radio frequency. Packets are received, held in memory, and retransmitted when the frequency is clear. Digipeaters are used extensively with APRS (Automatic Packet Reporting System) covered below. Traditional full-featured TNCs included digipeating capability. A KISS-only TNC requires a separate software application to perform the digipeating function.

The original digipeating process required the user to understand the possible communication paths among the digipeaters and manually construct a path to be taken by the packets. Various networking schemes such as NET/ROM, BPQ, Xrouter, *NOS, Flex, TheNET, and ROSE were devised to solve the routing problem. The Terrestrial Amateur Radio Packet Network (TARPN; **tarpn.net**) is now vigorously promoting packet networking and provides detailed instructions on how to build a node.

AX.25 FRAMES AND ERROR DETECTION

AX.25 exchanges data as HDLC frames (High-Level Data Link Control protocol) that contain routing, identification, data, and error detection information. There are three general types of AX.25 frames:

a) Information frame (I frame);
b) Supervisory frame (S frame); and
c) Unnumbered frame (U frame).

Each frame is made up of several smaller groups, called fields. **Figure 15.11** illustrates the three basic types of frames. Note that the first bit to be transmitted is on the left side. FCS is the Frame Check Sequence field and PID is the Protocol Identifier field.

Packet radio uses the Frame Check Sequence to include a two-byte checksum in every frame. The checksum is generated using the CRC-CCITT polynomial and is sent low-byte first. If the checksum indicates an error, the frame is discarded. In AX.25 connected mode a retransmission request is made as in other ARQ modes. The FX.25 protocol extension adds forward error correction (FEC) to the protocol while retaining complete in-

Flag	Address	Control	Info	FCS	Flag
01111110	112/224 Bits	8/16 Bits	N*8 Bits	16 Bits	01111110

(A) U and S frame construction

Flag	Address	Control	PID	Info	FCS	Flag
01111110	112/224 Bits	8/16 Bits	8 Bits	N*8 Bits	16 Bits	01111110

(B) Information frame construction HBK1076

Figure 15.11 — The three types of frames or packets used in AX.25 networks. Unnumbered (U) frames are used for commands and responses to commands such as requests to connect and disconnect stations. Supervisory (S) frames are used to acknowledge or reject information frames. The Information (I) frames are used to exchange data.

teroperability with regular AX.25. **github.com/wb2osz/direwolf/blob/master/doc/AX25_plus_FEC_equals_FX25.pdf**.

TCP/IP

TCP/IP (Transmission Control Protocol/Internet Protocol) is a standardized set of protocols that form the backbone of the internet and are compatible with existing network technologies and applications. The first widely used implementation of TCP/IP for amateur radio was *NOS*, created by Phil Karn, KA9Q. *NOS* was influential in the development of TCP/IP in amateur radio because he wrote *NOS* in (highly portable) C programming language and made the *NOS* source code publicly available (what we now call open source). *NOS* and its follow-ons are now essentially obsolete as "TCP/IP stacks" are available as modules to be added into software. TCP/IP capability is an integral part of Linux and other operating systems.

The only other item necessary to use TCP/IP packet software is your own static IP address in Network 44, termed AMPRNet (AMateur Packet Radio Network). Individual IP Address Coordinators assign addresses to new TCP/IP users in internet network 44 (44.0.0.0/9 and 44.128.0.0/10) based on physical location worldwide. More information can be found on the AMPRNet website at **portal.ampr.org**.

With globally unique addresses, routing could be performed over the internet when necessary. A big advantage of using the standard TCP/IP routing and data transport protocols is that we can use ordinary internet applications for e-mail, file transfer, remote login, web browsers, Voice over IP (VoIP), routing, and so on rather than reinventing our own equivalent.

The additional overhead for TCP/IP makes it painful over AX.25 when using 1,200 or 9,600 bps modems, so it has not reached its full potential. Perhaps we will see a resurgence if higher speed (56 kbps+) data radios become available.

15.5.10 APRS

The Automatic Packet Reporting System (APRS) was developed by Bob Bruninga, WB4APR, SK, as a real time local tactical information resource for emergencies, public service, and everyday use. It is not a vehicle tracking system. APRS data is intended for integration with maps and other forms of information display.

Besides reporting positions of user stations and vehicles, packets may include weather reports, telemetry, and messages to an individual or bulletins to groups (similar to mobile text messages) with queries and responses. APRS messages support the "Ham Radio of Things" (HRoT, the amateur radio equivalent of the Internet of Things or IoT). An image format called Packet Compressed Sensing Imaging (**maqifrnswa.github.io/PCSI**) has also been defined to send images when individual packets may be lost or received out of order.

APRS uses the same AX.25 protocol and TNCs as packet radio (see the previous section), but in a different way. Rather than one-to-one data transfer, it usually operates as a one-to-many transmission without the expectation of reliable data transport. In this way, APRS uses the redundancy of many stations in the network to deliver messages.

APRS STATIONS AND BEACONS

Traditional full featured TNCs can transmit fixed location beacons and serve as digipeaters. Depending on the model, they might also send telemetry, data from a weather station, or a location from a GPS receiver. KISS-only TNCs rely on a host computer with applications to provide the functionality. Software TNCs have varying degrees of functionality (GPS tracker, digipeater, IGate, DTMF decoding, etc.) built in so they can provide services autonomously without additional applications.

Stations are generally set to transmit or *beacon* their position and other information periodically. Fixed stations should beacon infrequently to avoid crowding the channel with repetitive information. Moving mobile stations can receive their coordinates from a GPS and should report more frequently. "Smart beaconing" automatically adjusts beaconing to be more frequent at higher speeds and when making sharp changes in direction.

APRS station names are usually a call sign followed by an optional number, up to 15, called the *substation identifier* (SSID). This allows multiple stations to be operated using the same base call sign. An APRS *beacon* transmission will generally contain a call sign with 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 a short text comment.

The SSID has 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. Tactical call signs can also be used if the assigned call sign is included as part of the status text.

Some VHF/UHF transceivers (both handheld and mobile) have a built in TNC, a display, often a GPS receiver, and other features for APRS. This is very convenient because everything is in a single package. They maintain lists of stations heard and calculate the distance away and direction. They can also be used to send messages to others and receive them. Some dual-receiver models can use the frequency, offset, and tone information in a packet and switch voice operation to that channel. This might be a request to contact a person listening or an Object advertising a repeater. 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.

In North America the standard APRS frequency is 144.39 MHz. See **www.aprs.org** for more information about using APRS and the network structure. The APRS protocol specification (August 2000) can be found at **www.aprs.org/doc/APRS101.PDF**. Many enhancements since then have less formal documentation — see **www.aprs.org/aprs11.html** and **www.aprs.org/aprs12.html**.

APRS DIGIPEATERS

Although APRS is based on packet radio, APRS packets spread in all directions through the generic network of digipeaters rather than flowing toward a specific pre-determined destination. As the packet moves through the network, it often encounters an IGate station that

will relay the packet to an APRS-IS server.

The sending station determines the maximum number of times the packet can be relayed with a PATH value that sets the maximum number of "hops." (A hop is a relay between digipeaters or between a digipeater and an internet gateway or IGate.) This count is decremented each time the information is repeated in order to limit the number of hops, preventing data from being relayed too far from the originating station. APRS digipeaters remember what they transmitted in the last half minute and will not send a duplicate. This prevents the same packet from bouncing back and forth between digipeaters, as well. Digipeating for APRS is covered in much more detail at **www.aprs.net.au/vhf/aprs-digipeaters-101**.

A local group should be able to offer guidance on how the PATH should be set to avoid overloading the local network. Listening to packet 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 one or two hops to cover a wide area and find an IGate.

APRS SOFTWARE

Many applications are available to display and generate APRS information. Most involve a map to show locations of objects and capability to send/receive APRS messages, telemetry, weather reports, and function as digipeaters and IGates. Some of the most commonly used modern applications are:

APRSISCE/32 — **aprsisce.wikidot.com**

PinPoint APRS — **www.pinpointaprs.com**

SARTrack — **sartrack.co.nz** which is specifically intended for Search and Rescue and Emergency Management

UISS — **www.qsl.net/on6mu/uiss.htm** designed for packet communication with the International Space Station (ISS), and compatible satellites.

UI-View — **www.ui-view.net** (popular but no longer maintained)

Xastir — **xastir.org/index.php/Main_Page**

Yet Another APRS Client (YAAC) — **www.ka2ddo.org/ka2ddo/YAAC.html**

The app *APRS Telemetry Watcher* (APRS-TW; **aprstw.blandranch.net**) is different than other map-centric applications. Its purpose is to monitor APRS telemetry and alert the operator to predefined conditions or events, such as low battery voltage.

APRS-IS AND INTERNET GATEWAYS (IGATE)

The APRS Internet Service (APRS-IS; **aprs-is.net**) is an internet-based network which interconnects various APRS radio networks around the world and in space. Internet gateway (IGate) stations provide a two-way link between radio networks and the APRS-IS servers. This allows information from one isolated radio network to be routed to another radio network around the world. It is highly recommended that all IGate stations be bi-directional if possible. Restrictions vary by country. Receive-only IGate stations break the intended two-way communication.

The online mapping applications at **aprs.fi** and **www.aprsdirect.com** gather information from APRS-IS and display it on maps accessible over the internet. The **findu.com** website offers historical weather data, a message display, and a large list of other queries. The IGate stations do not send information directly to these websites, they are only clients of the APRS-IS network.

APRS INTEGRATION WITH OTHER TECHNOLOGIES

The Automatic Identification System (AIS; **www.navcen.uscg.gov/?pageName=aismain**) is an international tracking system for ships in coastal waters using two dedicated channels in the VHF FM Marine Radio band. AIS transponder equipment is required by law in commercial ships and is recommended for private vessels. Messages can contain position, speed, course, name, destination, status, dimensions, destination, ETA, and many other types of information. Due to its similarity to APRS, it is not hard to write a converter to translate AIS sentences to APRS format. (**github.com/wb2osz/direwolf/blob/master/doc/AIS-Reception.pdf**) This allows information about ships to be merged with APRS data and displayed on APRS mapping applications.

With billions of computers and mobile phones (handheld computers) all connected by the Internet, the large growth is expected from the "Internet of Things." What is a "thing?" It could be a temperature sensor, garage door opener, motion detector, flood water level, smoke alarm, antenna rotator, coffee maker, lights, station thermostat, or just about anything you might want to monitor or control. There exists a global APRS network with numerous digipeaters and IGate stations joined by APRS-IS servers. This is also referred to as the "APRS of Things" or the "Ham Radio of Things" (HRoT; **github.com/wb2osz/hrot**) The network is very lightly used and available for experimentation and new applications.

Two different and parallel e-mail gateways allow APRS messages to be converted to short e-mail or text messages as documented at **www.aprs-is.net/email.aspx**. When entering a message, "EMAIL" or "EMAIL-2" is used for the addressee. 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 mail server and out to the recipient, with a confirmation APRS message. For dozens of other messaging options, check **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 callsign@winlink.org
- Send short e-mail messages to any valid e-mail address or Winlink 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 **winlink.org/APRSlink**.

APRS position reports are also available from GPS-equipped D-STAR radios via DPRS gateways (see the **Repeater Systems** chapter). 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. The BrandMeister DMR Server also provides an interface from DMR radios to the APRS network.

15.5.11 Winlink

Winlink (**www.winlink.org**) 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 functionality on the crowded radio spectrum. The system provides radio interconnection services including email with attachments, position reporting, graphic and text weather bulletins, emergency/disaster relief communications, and relaying messages. This section is an overview of the Winlink system — more information is available at **www.winlink.org/content/winlink_book_knowledge**, including the Winlink FAQ.

Winlink has been assisting the maritime and RV communities continuously for many years. Emergency communications (emcomm) communities also make use of the system and the Winlink development team has responded by adding features that make the

system more reliable and redundant. The role of Winlink in emergency communications is to add another tool in the toolkit of volunteer services deploying emergency communications in their communities.

SYSTEM ARCHITECTURE

The Winlink network structure is explained in "Overview of the Winlink "Hybrid" Network" available on the Winlink website. The system is designed around *Central Message Servers* (CMS) hosted on Amazon Web Services (AWS) in multiple locations for reliability. These ensure that the system will remain operational should any piece of the internet become inoperative. Individual *Radio Message Servers* (RMS) transfer messages between the CMS (via internet) and system users (via radio). Users may also communicate directly with the CMS via the internet. The system has three basic configurations: Conventional (by far the most widely used), Radio-Only, and Peer-to-Peer. *Winlink Express* software manages how each user accesses the system. An individual message may have several destinations that include radio-to-radio addresses as well as radio-to-internet email addresses, allowing complete flexibility.

All messages exchanged within the Winlink system, including directly via the internet, are compressed according to the open-source B2F protocol to minimize transmission time. To comply with FCC rules regarding encryption, within Winlink and in all compatible user software, the modem hardware or software that handles radio modes are always used without any native or proprietary capability to compress, obscure, or encrypt message content. (See **winlink.org/B2F**.)

CONVENTIONAL CONFIGURATION

In the Conventional configuration shown in **Figure 15.12**, RMS stations connect to the CMS via the internet. The set of CMS "in the cloud" act as a hub, routing messages to and from the RMS. The RMS stations act as gateways, providing endpoint connectivity to users via HF or VHF and are located worldwide.

A number of modes can be used to connect with RMS stations, including Packet (VHF), Pactor (HF), Robust Packet, ARDOP, VARA (HF and VHF), and Iridium Go. (These protocols are discussed elsewhere in this section except for Iridium Go.) Users can also connect directly to the CMS servers over the internet via the Telnet protocol. Several slower legacy modes are sometimes used, but are not recommended in order to reduce congestion on the radio frequencies used by Winlink stations.

Because each RMS can access an identical copy of the message it does not matter which one is used to exchange messages. In addition to individual email messages, each RMS stores or has access to hundreds of 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.

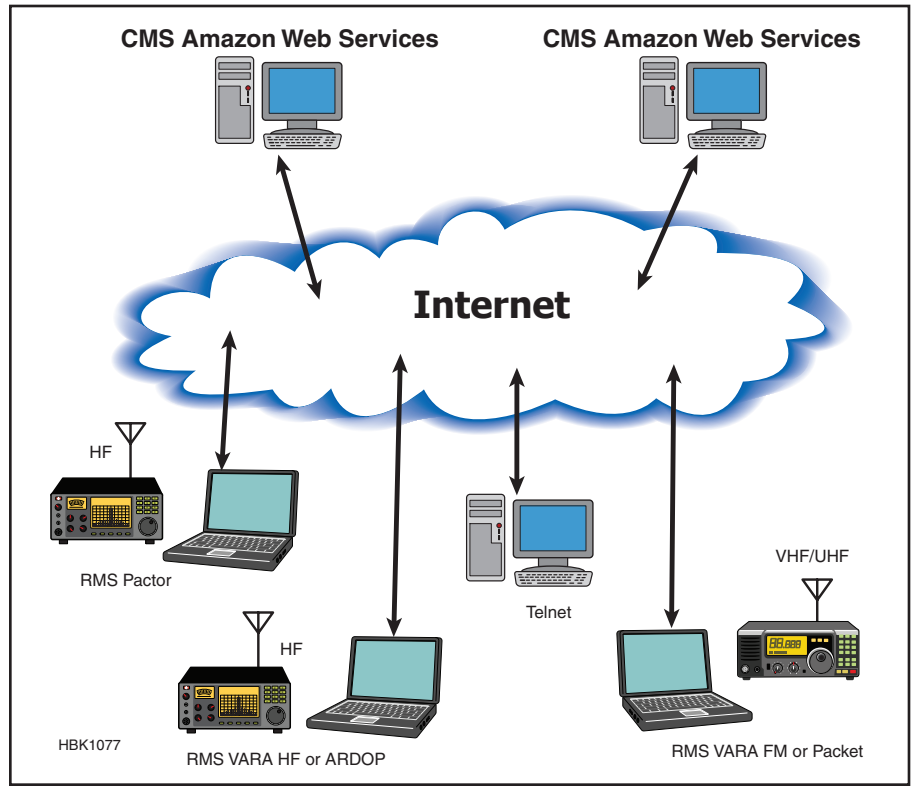


Figure 15.12 — Winlink system. Central Message Servers (CMS) are hosted on Amazon Wireless Services (AWS) servers.

RADIO-ONLY CONFIGURATION

In case of an internet outage, the Radio-Only configuration allows traffic to be exchanged directly between RMS stations. Because the CMS are not available in this configuration, each user must designate at least one *Message Pickup Station* (MPS). (Up to three MPSs may be designated by a user.) When a message is sent to that user, it will be automatically forwarded from one RMS to another until it reaches the recipient's chosen MPS. Because all traffic is over the air, message handling will take longer than in the Conventional configuration and internet email addresses are not available.

Radio-only forwarding is an option that can be selected when you generate a message. It means that message will not be sent over the internet, even if the internet is available. Any message sent by radio-only forwarding will reach each MPS and will be picked up when the recipient next connects to that station.

PEER-TO-PEER CONFIGURATION

Finally, a direct radio-to-radio configuration (Peer-to-Peer) allows stations to exchange messages directly without going through an RMS station or CMS. This requires both stations to be able to communicate using the same protocol on a single frequency. Internet and message relay functions are not available in this configuration.

MAXIMIZING DATA RATE

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, so by limiting the forwarding of messages between Winlink RMS stations to the internet, a great deal of radio air time is eliminated, making the time and spectrum available to others for message handling or other operations. Many RMS stations are limited to higher speed modes and have a much higher ratio of traffic minutes and message counts to connect times than do the RMS stations that operate at slower speeds. In other words, the amount of traffic that is passed with a higher speed connection is much greater for an equivalent amount of connect time with approximately the same number of connections.

The fastest HF mode is PACTOR IV, but it is not normally available in the United States because the protocol's data symbol rate is greater than is permitted below 30 MHz by FCC rules. Occasionally, PACTOR IV is temporarily authorized by the FCC during disaster responses to clear HF traffic. It is used regularly by stations outside the United States.

CLIENT ACCESS TO WINLINK

Winlink supports a clean, simple interface to the internet SMTP email system through the *Winlink Express* software available at **winlink.org/WinlinkExpress**. You can obtain a Winlink email address by using *Winlink Express* to connect to the system either over the radio or through Telnet. There is no charge, although a $25 donation (one time) is suggested. You can't obtain a Winlink address over the web.

Any message sent or received may include multiple recipients and multiple binary attachments. Mail addressed to a Winlink address from a non-Winlink email address, however, must be known to the radio user or the message will be rejected. Spam would clog the system otherwise. The system maintains a whitelist of non-Winlink email addresses you've sent mail to recently and you will receive messages sent to you from addresses on that list. You can also add email addresses to your account at **winlink.org/content/how_manage_your_whitelist_spamcontrol**.

In case of an emergency where local services are interrupted, the system may be used by non-amateur groups as an alternative to normal email. Connecting to any of the RMS can automatically allow a local amateur station to pass emergency traffic. Winlink uses no external source for sending or receiving internet email. 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 program for sending and receiving messages using Winlink. Once connected to a Winlink station, message transfer is completely automatic. On the amateur bands, *Airmail* can transfer messages automatically with other stations using *Airmail* and any station supporting Winlink.

When used with Winlink, *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. To obtain a copy of *Airmail*, including the installation and operating instructions, download the program from **www.siriuscyber.net/ham**.

When web access is available, Winlink messages can be retrieved or sent via a web interface (webmail). The web browser access is limited to text-based messages without the use of bulletins or file attachments. Password-protected access to your Winlink mailbox is available at **www.winlink.org/webmail**.

Winlink can be accessed 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 services normally available over radio channels, but at internet speeds.

Both *Winlink Express* and *Airmail* support Telnet. This allows regular use of the system with the same software configuration at high speed via internet, without using RF bandwidth and provides a full-featured mechanism to access Winlink. An Emergency Operations Center (EOC) with internet access can easily use Telnet with RF as a fallback. Similarly, an RV or marine user can use the software from an internet café or home via Telnet and switch the software back to radio when mobile, with no change in functionality.

WINLINK FEATURES

Bulletins

To address the needs of mobile users for near real-time data, Winlink uses an "on-demand" bulletin distribution mechanism. Users select requested bulletins from an available "catalog" list managed in *Airmail* or *Winlink Express*. When bulletin requests are received by an RMS, 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 sent to the user.

The global catalog includes hundreds of weather, propagation, and information bulletins, including instructions for using the system, world news, and piracy reports. All Winlink 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 needed by specific agencies in any community emergency.

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

Attachments

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 *Airmail* or *Winlink Express*. Email message attachments sent through the Winlink system must be limited in size. Users can set the maximum file size. 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 to protect the user from viruses and malware.

Standard Form Templates

The U.S. government Federal Emergency Management Agency (FEMA) publishes standard forms which must be used by local agencies for official communications, such as to request resources. To repeatedly transmit form graphics over the radio would be a great waste of limited spectrum. To avoid this, the *Winlink Express* software includes a regularly updated library of forms — called *templates* — with each copy of the software. Using templates means the radio traffic only consists of the form name and the text used to fill out the form. This greatly reduces transmission time for the message. Since the same template is available to the recipient of the message, the completed form appears to the recipient exactly as it was filled in by the originator.

If the recipient software doesn't include a specific template, the text is transmitted in human-readable form. This means the text of the message transmitted can easily be read by the recipient even though it isn't displayed in the original form.

Position Reporting

The bulletin services mentioned above is beyond normal messaging, but Winlink 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 operators. The default time per any 24-hour period is 30 minutes, however, the user may request more time from the RMS operator should it be needed. The time limit is individual to each RMS station. Utilization of higher speed modes such as PACTOR III and PACTOR IV are a great timesaver.

Network Maintenance

The system has a number of other secondary network maintenance features. Extensive traffic reports are collected, the state of in-

dividual RMS stations is monitored and reported if one becomes inactive. Daily backups are performed automatically at all RMS and CMS to ensure the system integrity. Security is maintained through the vigorous updating of virus definitions and automatic virus screening for internet mail and files. The system has the ability to block any user by both radio (by frequency band) and internet (by email address) to prevent abuse of the system.

15.5.12 D-STAR

D-STAR (Digital Smart Technologies for Amateur Radio) is a digital voice and data protocol specification developed from research by the Japan Amateur Radio League (JARL) into digital technologies for amateur radio in 2001. While other digital on-air technologies for other services are being used by amateurs, D-STAR was one of the first on-air protocols designed specifically for amateur service use to be widely deployed. As of early 2022, radios supporting D-STAR are sold by Icom and Kenwood.

D-STAR is a bit-streaming protocol that encapsulates digitized voice and low speed data (DV) or higher speed Ethernet data (DD). D-STAR radios can communicate either simplex or via repeaters. D-STAR equipment is available for the 2 meter (VHF), 70 cm (UHF) and 23 cm (1.2 GHz) bands.

D-STAR's GMSK modulation similarly to FM modulation: Only one station can talk at a time. Since Forward Error Correction (FEC) is included in D-STAR. FEC will correct bit errors in the data stream. This provides a higher sound quality and slightly greater range than FM modulation. With FM signals, signal quality gradually degrades as signal strength decreases. With D-STAR, the receiving station hears perfect signals down to low signal levels before the signal is no longer received.

An overview of the D-STAR system, "D-STAR — For the Second Century of Amateur Radio" is available at **www.icomamerica.com/en/downloads/DownloadDocument.aspx?Document=366**. A detailed discussion of the D-STAR specification is contained in a three-part *QEX* article by John Gibbs, KC7YXD, in the Jul/Aug, Sep/Oct, and Nov/Dec 2003 issues. These articles are available at **www.icomamerica.com/en/products/amateur/dstar/dstar/default.aspx** in the Downloads area. More information about D-STAR repeater systems may be found in the **Repeater Systems** chapter.

DD AND DV MODE

The D-STAR protocol specifies two modes; Digital Voice (DV) and Digital Data (DD). The DV mode provides both voice and low speed data channel on 2 meters, 70 cm and 23 cm over a 4,800-bps (bit/s) data stream. The high-speed Digital Data (DD) mode operates at 128 kbps on the 23 cm band. The user's amateur radio call sign is included in all transmissions for signal routing and identification.

The DV data stream is broken down to three main packages: voice, forward error correction (FEC) and data. (See **Figure 15.13A**.) The largest portion of the data stream is the voice package, which is a total of 3,600 bits/s with 1,200 bps dedicated to forward error correction, leaving 1,200 bps for data. This additional data contains various data flags as well as the data header, leaving about 950 bps 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. A second data option has been made within the DV protocol in which the serial data rate is 3,480 bps and utilizes the entire channel.

The focus of D-STAR's design was the most efficient way to conserve RF spectrum. While D-STAR's 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 later models of D-STAR radios, a higher speed data protocol was implemented in DV mode which "borrows" some of the voice bits for data transmission of pictures and files at a maximum rate of 3,600 bits/s increasing data throughput.

In addition to DV mode, the D-STAR protocol outlines the high-speed Digital

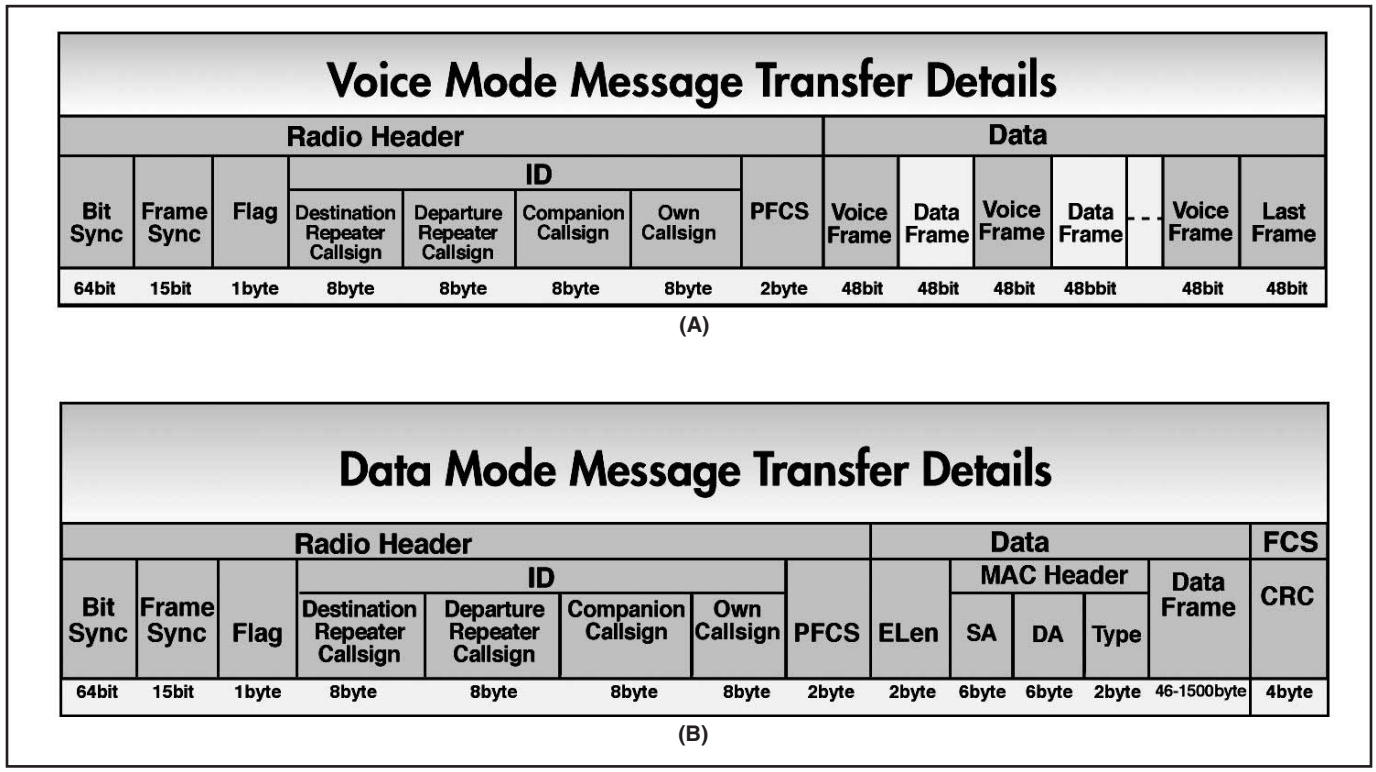


Figure 15.13 — D-STAR protocol frame structure for DV (A) and DD (B).

Data (DD) mode. (See Figure 15.13B.) This higher speed data, 128 kbps, is available only on the 23 cm band because it requires a 150 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 kbps is reduced to approximately 100 kbps — still more than double a dial-up connection speed with significant range.

Another consideration is the data rate specified at 128 kbps 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.

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. The Icom 23cm DD repeater module operates at an output of 10 watts.

Radios currently providing DV mode data service use a serial port for low-speed data (1,200 bps), while the DD mode radio offers a standard Ethernet connection for high-speed (128 bps) 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 Ethernet.

D-STAR VOICE CODEC

The AMBE voice codec (encoder/decoder, also referred to as a *vocoder*) for D-STAR uses a voice data rate (2,400 bps) optimized for use by public service and satellite communications services. These characteristics made it suitable for amateur radio, as well. The AMBE codec is proprietary to Digital Voice Systems, Inc. (DVSI). AMBE stands for Advanced Multi-Band Excitation and as firmware, it is embedded in ICs for radio and other communications equipment.

There are several AMBE products with varying rates of voice compression. JARL tested the different rates and determined that one of the lowest rates would be acceptable to amateur radio operators. While there has been considerable discussion about the proprietary nature of AMBE, at the time of development of D-STAR, there were no open protocols available that could be implemented in a handheld radio with the processors available at that time, at a reasonable cost. AMBE was already in service in many applications, including P25 and later, DMR.

D-STAR REPEATERS

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 **Figure 15.14**. Repeaters only retransmit the data stream and do not convert the voice data back to analog form.

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.

GATEWAY CONFIGURATION

In DV mode, user registration is not required but allows for additional capabilities such as repeater linking/unlinking and access via hotspots (discussed in the **Repeater Systems** chapter). An unregistered user may access a local repeater and if it is linked to another repeater or reflector the user will be included in the local conversation as well as the transmission "sent" to other linked repeaters or reflectors. Only registered users may link and unlink the repeater. Users accessing the D-STAR network through hotspots must be registered to connect to repeaters or reflectors.

In a Gateway configuration, *all* DD mode 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.

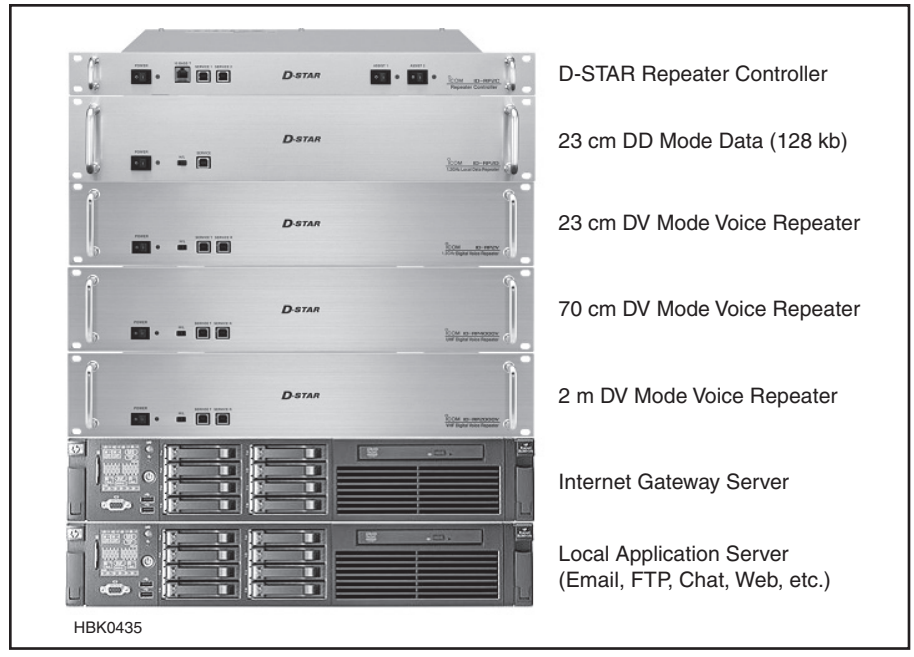


Figure 15.14 — Full D-STAR repeater system showing the controller and complement of DV and DD repeater modules.

15.5.13 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 4,800 baud and two bits per symbol, yielding 9,600 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, 2,400 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 4,400 bit/s. An additional 2,800 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 4,800 bit/s. Phase 2 radios will still offer backward compatibility with Phase 1. As of 2022, Phase 3 access to the 700 MHz band in the US has been implemented.

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.apcointl.org/technology/interoperability/project-25**.

15.5.14 Yaesu System Fusion

Yaesu released a specification for System Fusion in 2013, becoming the most recent DV methodology. System Fusion supports digital voice and data in a 12.5 kHz narrow-band channel at 9,600 bps, using C4FM modulation over VHF (144-148 MHz) and UHF (440-450 MHz). System Fusion supports three modes of operation: voice full rate (Voice FR) mode, data full rate (Data FR) mode, or voice/data (V/D) mode. Voice FR mode is typically displayed on System Fusion transceivers as "VW," with voice/data mode appearing as "DN." The two full rate modes use the entire 9,600 bps channel for their respective voice or data payloads, whereas the V/D mode splits the channel into two 4,800 bps payloads, with voice information on one and data on the other.

The voice modes also include robust forward error correction (FEC) and utilize the AMBE+2 (Advanced MultiBand Excitation) CODEC from DVSI. In V/D mode, the voice and FEC data use the AMBE+2 "Enhanced Half-Rate" mode where voice and FEC payloads consume individual 3,600 bps totals, with header, synchronization, and routing data consuming much of the remainder of the 4,800 bps available. The data component of the V/D mode can be used for features like GPS information, in order to provide APRS-like functionality. The Voice FR mode encodes using the AMBE+2 "Enhanced Full-Rate" mode, where voice information and FEC consume 7,200 bps. The Voice FR mode provides the highest quality voice exchange.

Each transmission comprises 960 byte (100 ms) packets. Each transmission begins with a header packet (HC), followed by communication channel packets (CC), and ends with a terminator packet (TC). The HC and TC packets synchronization information, source, and destination call signs, plus uplink and downlink call signs, as used for routing. In the V/D mode, each of the CC packets contain 40 bytes of frame sync, then 200 bytes of frame information, followed by 72 bytes of interleaved voice and data frames, for a total of 960 bytes. In the Voice FR mode, the first CC packet contains some overflow call sign information, but is generally segmented into 144-byte frames with voice information.

Yaesu repeaters offer the System Fusion environment with an "Automatic Mode Select" (AMS), operating in both FM analog and System Fusion digital modes. The repeaters can be configured in a combination of modes. While the repeaters are capable of operation in a purely analog mode, this prevents digitally equipped users from taking advantage of the enhancements that digital operation offers. With a similar discouragement toward digital enhancements, the repeaters can be configured in such a way as to allow analog or digital reception, but forcing the output to analog only. While this configuration allows both analog and digital users to coexist, it does so in a constrained fashion, negating the digital advantages of a white noise and hiss free experience, plus strips away GPS and call sign information. Operating in purely digital mode is also possible, offering enhancements for digital users, but effectively "locking out" analog FM operations, as with other DV methodologies. This is typically implemented when there is more than one System Fusion repeater operating in a given area.

More interesting (and common) is the hybrid AMS configuration, allowing analog FM in to analog FM out and digital in to digital out. In this configuration, analog users are not suddenly "disconnected" from a repeater and other analog-only capable operators. Also, digital users are free to take full advantage of the enhancements available with their transceivers. With the users' System Fusion transceivers configured in AMS, digital users can hear an analog call placed on the repeater, in between transmission exchanges. The transceivers will switch to analog, automatically allowing them to communicate with the analog station. This furthers an environment of inclusion for both "camps" to enjoy and allows analog users to upgrade to System Fusion digital at a time of their choosing, rather than forcing a transition, all at once.

In order for analog FM transceivers to remain quiet during digital transmissions on the selected receive frequency, enabling the tone squelch feature to match the repeater's transmitted continuous tone coded subaudible squelch (CTCSS) or digital coded squelch (DCS) signal, is required. Watching for a visual "channel busy" indicator on the analog users' transceivers or enabling a Busy Channel Lockout feature will prevent accidental interference, when digital communications are taking place.

15.5.15 Digital Mobile Radio — DMR

Digital Mobile Radio (DMR) is divided into three tiers, Tier 1 is FDMA (Frequency Division Multiple Access), Tier 2, is TDMA (Time Division Multiple Access), and Tier 3 is TDMA Trunking. Amateurs utilize Tier 2.

Digital Mobile Radio (DMR) is a worldwide standard defined by the European Telecommunications Standards Institute (ETSI) based on Motorola Solutions MOTOTRBO Tier 2 and is used in commercial products from a number of manufacturers. Tier 2 and Tier 3 transmissions fit in a standard 12.5 kHz narrowband channel structured as two-time-slot TDMA. The two channels result in a spectral efficiency of 6.25 kHz per channel making it ultra-narrowband.

DMR uses 4FSK modulation at 4,800

symbols/second, corresponding to 9,600 bps (bits/second) for both time slots. Each DMR timeslot is 30 msec long with 2.5 msec of guard time between time slots resulting in a 27.5 msec frame of 264 bits; 108-bit payload; 48-bit SYNC or embedded signaling; and a second 108-bit payload for a total of 216 bits of payload per frame.

While not specified in the ETSI DMR standard, manufacturers have followed Motorola Solutions' lead and utilize the DVSI AMBE+2 codec to encode and decode the voice audio. The vocoder compresses 60 msec of audio with FEC (forward error correction) into 216 bits of data for transmission.

Since a user radio (as opposed to a repeater) only transmits during a single time slot for 27.5 msec of each 60 msec time period, the result is an approximately 50% transmit duty cycle, unlike an FM or FDMA-based digital mode transmitter. This translates into longer battery life on handheld radios. It should be noted that typical transmit power measurements are not possible for TDMA transmitters.

Synchronization of TDMA transmitters is critical. When using a repeater, the repeater is the time source for synchronization of the user radios. During simplex communications the first transmitter becomes the time source for synchronization. The critical nature of DMR TDMA limits the practical range between user radios and repeaters to about 50 miles because of synchronization. Simplex range between user radios does not have any distance limitation. For amateur DMR repeaters the 50-mile limitation is only an issue if users are active on both time slots at the same time and one is near the repeater and the other is beyond the 50-mile limit.

The original Tier 2 specification for DMR covered conventional simplex or repeater operation. In 2013, the Tier 3 standard for trunking operation was approved by ETSI. While the ETSI standards allow for interconnection of repeaters over IP networks, beaconing by repeaters allows roaming by user radios properly configured in a homogeneous network of repeaters.

Like many digital modes, the DMR specification only covers the air interface to allow the radios to interoperate. It does not cover the network linking protocols, leaving that to the manufacturer. Motorola Solutions uses their IP Site Connect (IPSC) protocol to interconnect repeaters over IP networks. These IPSC networks have been expanded using gateway/conference bridging software on servers to expand networks worldwide. The main gateway/conference bridging products are RavenNet's cBridge (sold by Rayfield Communications) and TL-NET (sold by Bridgecomm Systems), DMRplus (IPSC2), and Brandmeister. The cBridge/TL-NET product supports only MOTOTRBO repeaters and analog interfaces. Brandmeister and DMRplus support a variety of other repeater manufacturers as well as hotspots.

Table 15.15

Wireless Networking Frequencies

airMAX	*Ubiquiti*	*ISM*	*Amateur*
M900 900 MHz	902 – 928	902 – 928	902 – 928
M2 2.4 GHz	2402 – 2462	2400 – 2500	2390 – 2459
M3 3 GHz[1]	3370 – 3730		3300 – 3500[3]
M5 5 GHz	5725 – 5850	5725 – 5875[2]	5650 – 5925

[1]For export from USA
[2]U-NII: 5150 – 5350, 5470 – 5825 MHz
[3]ARRL Band Plan

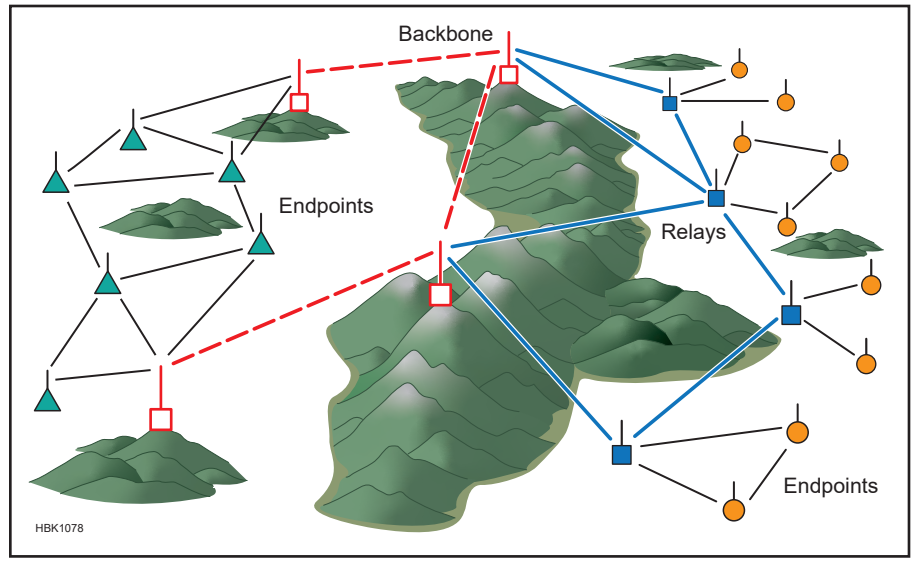


Figure 15.15 — A mesh network is created by connecting the individual user stations (endpoints) either directly or through relay nodes that communicate through backbone links. In this way an entire region can be connected using standard networking technology.

Conferencing *bridges* are used to connect groups of users. DMR uses the term *Talk Group* (TG) to identify a specific bridge. System Fusion uses the term *Room* and D-STAR uses the term *Reflector*. The amateur use of these conference bridges also allows for interconnecting other digital and analog networks together. Many amateur DMR networks utilize one time slot for local traffic and the other for wider area communications. (See the **Repeater Systems** chapter for more information about bridges in these networks.)

15.5.16 IP-Based Microwave Networking

Implementing an IP-based network using amateur radio allows implementation of any IP-based service (subject to the restrictions of FCC Part 97) on amateur frequencies. Microwave bands are used because wide-bandwidth signals are permitted on amateur frequencies above 902 MHz (33 cm). Some examples of services that can be supported over an IP-based network include:

- Keyboard to keyboard (text)
- Voice (VOIP phone systems, chat servers)
- Video (video chat, webcams)
- Email/messaging
- Document editing/management
- File sharing services
- Web servers
- Repeater linking
- Games

NETWORK TOPOLOGIES

There are two primary network topologies used to implement amateur radio microwave IP networks. Amateur Radio Emergency Data Network (AREDN) can create a peer-to-peer mesh or be used in point-to point or point-to-multipoint topologies while HamWAN, HamNet, and Mi6WAN implement star topologies. **Figure 15.15** illustrates the basic mesh network components.

All are TCP/IP-based and the same applications and services can be supported regardless of which networking technology is used. The

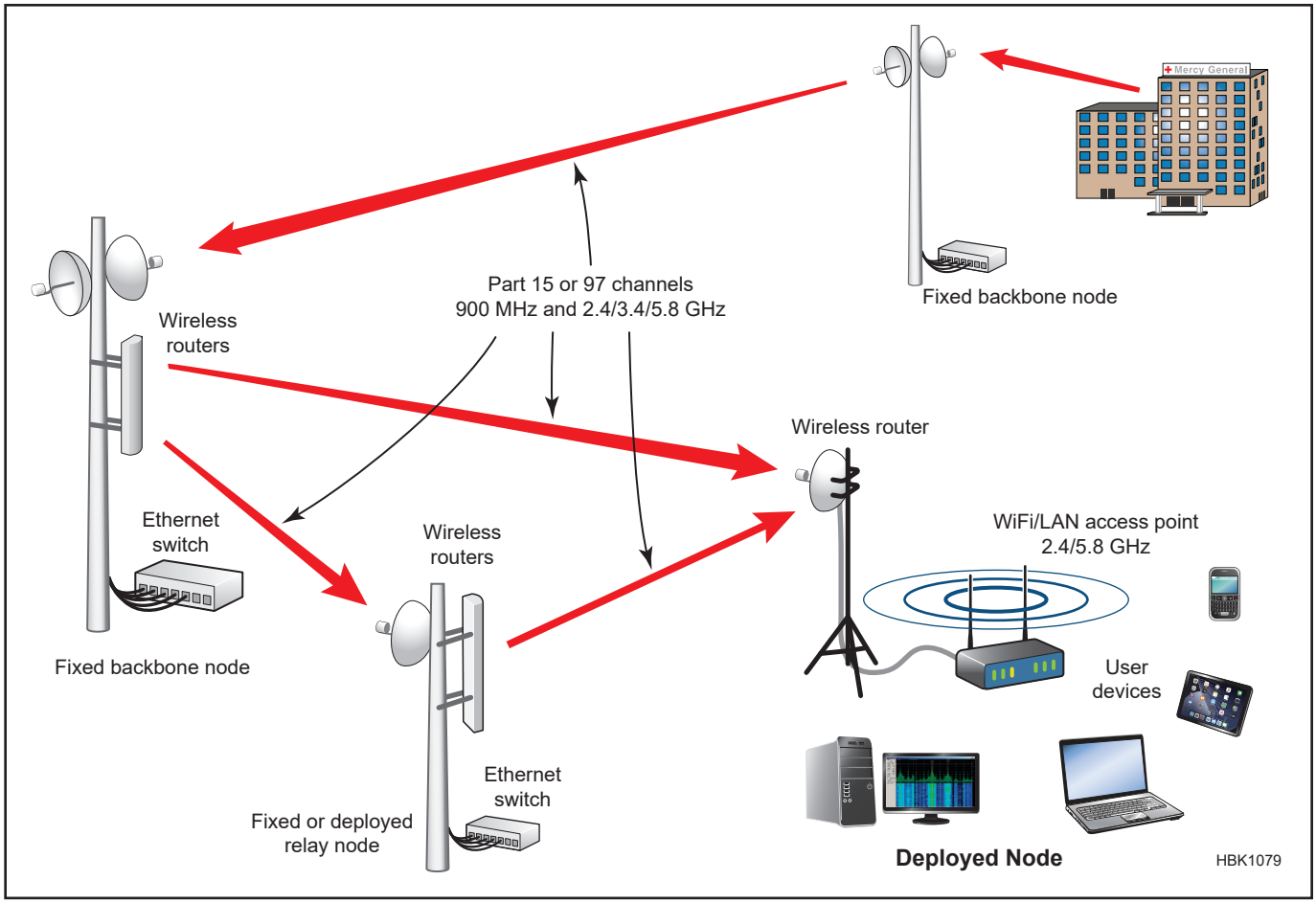


Figure 15.16 — The components of a typical AREDN network.

systems adapt commercial hardware operating in the amateur allocations of microwave frequency bands as shown in **Table 15.15**.

Additional information on amateur networking is available at the following websites:

- AREDN: **www.arednmesh.org**
- HamWAN: **hamwan.org**
- Hamnet (European): **hamnetdb.net**
- Mi6WAN (Central Michigan): **w8cmn.net**

AREDN (AMATEUR RADIO EMERGENCY DATA NETWORK)

AREDN develops software for the Ubiquiti, Mikrotik, TP-Link and Gl.iNet series of wireless routers, in the 900 MHz, 2.4 GHz, 3.4 GHz and 5.8 GHz amateur bands. (See **www.arednmesh.org/content/supported-platform-matrix** for the current list of supported devices.) The stock firmware in the routers is replaced to become a fully functional node in a peer-to-peer network. (See **Figure 15.16**) The map of a typical AREDN network in Western Oregon is shown in **Figure 15.17**.

In an AREDN network, nodes are self-discovering, self-configuring, and self-advertising. This greatly simplifies the establishment of a network since minimal networking knowledge is required to set up a node. For complete information on AREDN network design and technologies, an extensive set of documentation is available at **arednmesh.readthedocs.io**.

A typical AREDN endpoint node consists of a wireless router to connect to the network and a wireless data access point, usually sup-

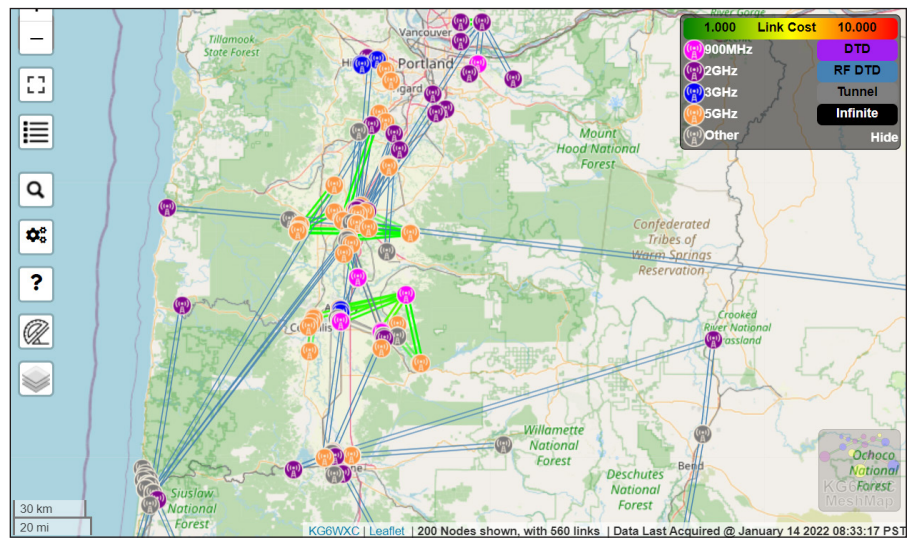


Figure 15.17 — The Western Oregon AREDN network as of early January.

Figure 15.18 — A backbone site (right tower) in Southern California. User access points for 2 and 5 GHz are shown, along with a 5 GHz county backbone link. A 3 GHz local backbone node and a Part 15 wireless access point are also shown. (Photo courtesy Orv Beach, W6BI)

porting both WiFi and Ethernet LAN connections. The access point provides connectivity that is the same as in a typical home network and many AREDN nodes are part of a ham's home network. Relay nodes and backbone nodes consist of two or more wireless routers connected together with an Ethernet switch. No other equipment is required. Installed in advantageous locations, backbone nodes support long-distance links between relay nodes in different areas. **Figure 15.18** shows the K6PVR backbone site in the mountains of Southern California.

The current firmware also includes tunneling capability to allow RF "islands" to be connected through the internet. In this configuration, individual groups of nodes (see **Figure 15.19**) communicate with each and one or more has regular internet access. Tunneling protocols are then used to connect the groups of nodes together as if they were using a purely RF system of relays and backbone links.

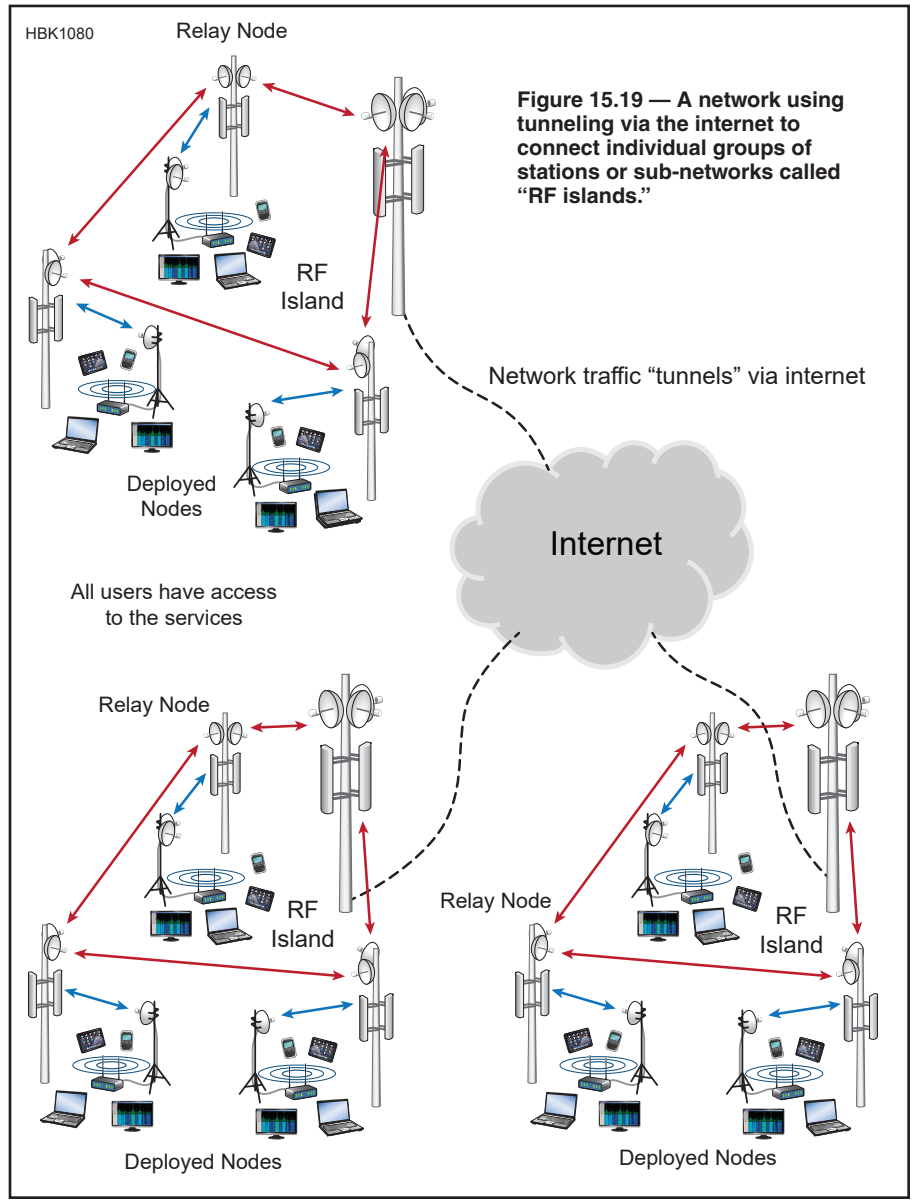


Figure 15.19 — A network using tunneling via the internet to connect individual groups of stations or sub-networks called "RF islands."

HAMWAN

HamWAN utilizes the international version of MikroTik wireless hardware. It's based on a star network topology in which all of the user nodes (also known as *client nodes*) connect directly to a central node, also known as a *cell site* or *distribution node*. Each distribution node communicates over an *uplink path* to a *ring node* connected in a backbone called a *data ring*. (See **Figure 15.20**) Each ring node consists of at least two wireless routers supporting the backbone and one providing the link to the distribution nodes, similarly to the AREDN backbone nodes. A current implementation of the ring-and-distribution network in the Western Washington area is shown in **Figure 15.21**.

HamWAN distribution nodes communicate to the ring nodes over a dedicated link at 3.4 or 5.8 GHz. HamWAN client nodes use MicroTik Nv2 command protocols which support *time division multiple access* (TDMA) technology. HamWAN also uses digital certificates to authenticate users. Unlike AREDN, HamWAN requires pre-coordination to be arranged between the nodes and more extensive network knowledge to configure the nodes. Complete documentation of the HamWAN network is available at **hamwan.org**.

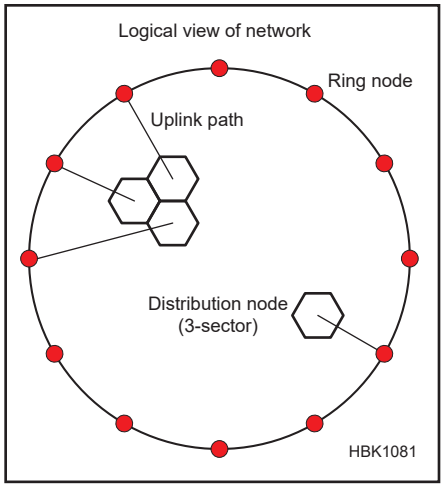


Figure 15.20 — The HamWAN ring network topology is built from distribution nodes that serve a small area and a backbone of ring nodes that connect all of the distribution nodes together. (Graphic from hamwan.org)

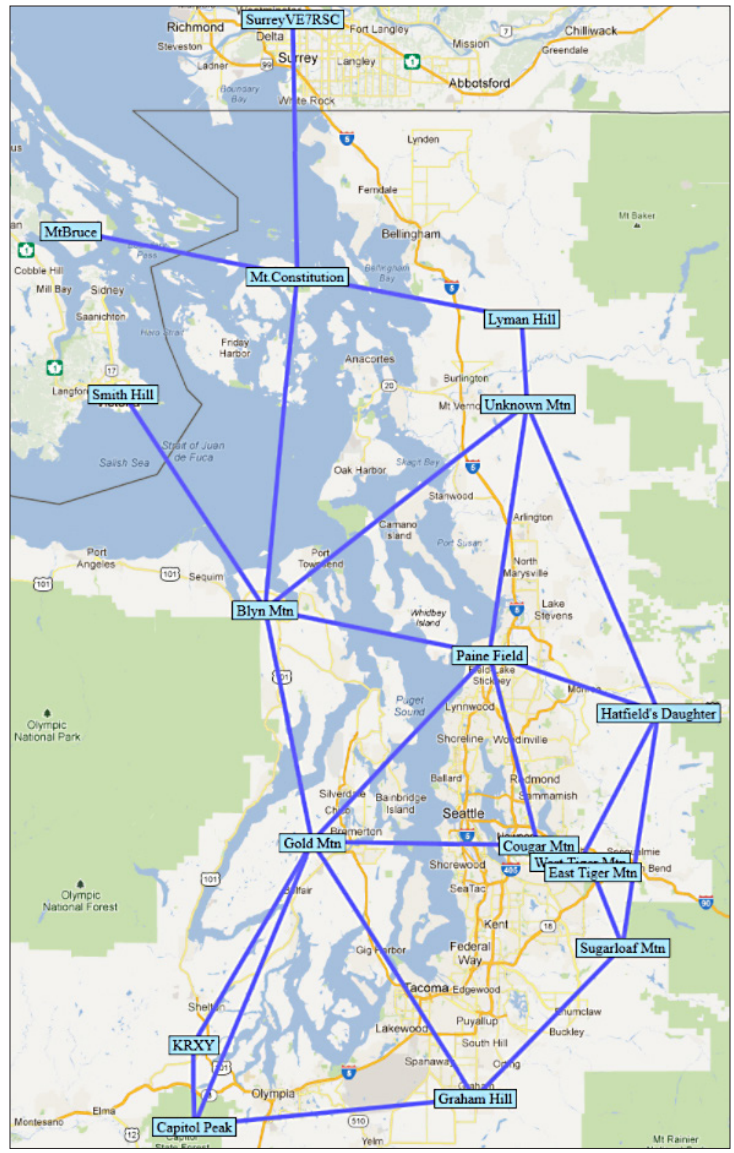


Figure 15.21 — The Puget Sound Data Ring is an HamWAN network in the Western Washington area. (Graphic from hamwan.org)

USE OF FCC PART 15 OR COMMERCIAL CHANNELS

Many amateur networks operating in the microwave spectrum use Part 15 channels for their backbones between groups of nodes which communicate between each other on Part 97 channels. (See Figure 15.16) Other networks are built purely of nodes using only Part 97 channels. There are pros and cons to each approach:

Part 97-only networks use channels that are much less crowded and often have a lower noise floor and thus potentially higher throughput or longer links. However, Part 97 rules prohibit encryption so protocols such as TLS (Transport Layer Security) and SSL (Secure Sockets Layer) cannot be used. Websites that require the use of HTTPS (Secure HTTP) cannot be accessed. Part 15 and other commercial channels are free of rules prohibiting encryption but the channels can be busy with commercial traffic from WISPS (Wireless Intranet Service Providers) operating on these channels

If you can "hitch a ride" on an existing commercial network as a backbone link, the only requirement for supporting an AREDN-based network is that it provides VLAN 2 from point to point. VLAN 2 is the VLAN (Virtual Local Area Network) that AREDN uses to carry DtD (Device to Device) traffic between nodes.

15.6 Digital Mode Table

Table 15.16 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 table and a detailed listing showing the lowest permitted frequency for a number of digital mode variants are also available as a PDF file with the online content.

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).

Table 15.16

Digital Modulation Modes and Formats Used in Amateur Radio

See the text of this chapter for definitions and abbreviations

Lowest Permitted Frequency is the lowest amateur frequency at which this emission may be used according to Part 97.307 limits on symbol rate and bandwidth. If no frequency is listed, the emission is permitted in all data segments.

See also the online PDF file "Digital Modes - Lowest Permitted Amateur Frequency" in the online content accompanying this book.

Mode or Format Name	*Developer*	*Principal Freq*	*Principal Application*	*Lowest Permitted Frequency*	*Data Rate (bits/sec)*	*Bit rate (bits/sec)*
ALE	MIL-STD-188-141, FED-STD-1045	HF	Data		<375	375
AMTOR-A	G3PLX	HF	Data		53	114
AMTOR-B	G3PLX	HF	Data		57	114
AOR AMBE	AOR Corp	HF	Voice, Data		2400	3600
APCO P25	APCO	VHF	Voice	50 MHz	6800	9600
ARDOP	KN6KB	HF	Winlink email		25-167	46-2296
Chip64	IZ8BLY	HF	Keyboarding		21.1/37.5	300
CLOVER-II	Hal Comm.	HF	Data		< 37.5-750	62.5-750
CLOVER-2000	Hal Comm.	HF	Data		108-1994	500-3000
DMR	Motorola	VHF	Voice	50 MHz		9600
Domino	ZL2AFP	HF	Keyboarding		31/44/62?	31/44/62
DominoEX	ZL1BPU	HF	Keyboarding		(<15.63)-86.13	15.63-86.13
D-Star (DV)	JARL	VHF	Voice & data	28 MHz	D:960 V:2400	4800
D-Star (DD)	JARL	UHF	Data	902 MHz	72k-124k	128000
Facsimile		HF	Image			
FDMDV	G3PLX/HB9TLK	HF	Voice		1450	1450
FSK441	K1JT	VHF	Meteor scatter	28 MHz	882	882
FFST4*	K9AN, G4WJS, K1JT	LF-MF	Weak signal		1.49	6.18
FST4W*	K9AN, G4WJS, K1JT	LF-MF	Weak signal beacon		0.457	2.92
FT4	K9AN, K1JT	HF	Contests		15.3	41.67
FT8	K9AN, K1JT	HF	DXing		6.1	18.75
G-TOR	Kantronics	HF	Data		35/75/115	80/160/240
Hellschreiber (Feld)	Rudolf Hell	HF	Keyboarding		2.5 char/s	122.5
JT4*	K1JT	V/UHF	Moonbounce		1.64	8.75
JT9*	K1JT	MF-VHF	Weak signal		1.47	5.21
JT65*	K1JT	V/UHF	Moonbounce		1.54	16.1
MFSK16	ZL1BPU/IZ8BLY	HF	Keyboarding, Data		31.25	62.5
MSK144	K9AN, K1JT	VHF	Meteor scatter	50 MHz	1069.4	2000
MT63	SP9VRC	HF	Keyboarding		35/70/140	320/640/1280
Olivia	SP9VRC	HF	Keyboarding		8.75/17.5	78.13/156.25
Packet (Bell202)		VHF	Data	28 MHz	<1200	1200
PACTOR-I	DL6MAA/DF4KV	HF	Data, Winlink email		51.2/128	100/200
PACTOR-II	Spec. Comm. Sys.	HF	Data, Winlink email		100-700	200-800
PACTOR-III	Spec. Comm. Sys.	HF	Data, Winlink email		85-2722	200-3600
PSK31	G3PLX	HF	Keyboarding		31.25	31.25
Q65*	IV3NWV, K1JT	V/UHF	Scatter, Moonbounce		1.51	10.0
QPSK31	G3PLX	HF	Keyboarding		31.25	62.5
PSK63/125		HF	Keyboarding		62.5/125	62.5/125
QPSK62/126		HF	Keyboarding		62.5/125	125/250
Q15X25	SP9VRC	HF	Data		300/1200/2400	2500
RTTY (Baudot)		HF	Keyboarding, contests		30.3	45.45
SSTV (traditional)	WØORX	HF	Image		8 s/frame	
SSTV Martin M1	Martin Emmerson	HF	Image		114 s/frame	
SSTV Scottie S1	Eddie Murphy	HF	Image		110 s/frame	
System Fusion	Yaesu	VHF	Voice, data	50 MHz		9600
Throb	G3PPT	HF	Keyboarding		10/20/40 wpm	
VARA	EA5HVK	HF-VHF	Winlink email		18 – 12,750	
WSPR (MEPT-JT)	K1JT	HF-VHF	Weak signal beacon		0.45	2.93

* A number of submodes are available.

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.

Symbol rate (baud)	*Modulation ("N-"means multi-carrier)*	*Error handling*
125	8FSK	FEC
114	FSK	ARQ
114	FSK	FEC
50	36-QPSK	FEC
4,800	4FSK/QPSK	FEC
300	MFSK, MPSK	FEC
300	DBPSK-DSSS	FEC
31.25	4-(2-16)DPSK/(2-4)DASK	FEC/FEC+ARQ
62.5	8-(2-16)DPSK/(2-4)DASK	FEC/FEC+ARQ
4800	4FSK/QPSK/C4FM	FEC
7.8/11/15.6	16FSK	None
3.9-21.5	18FSK	None/FEC
4800	0.5 GMSK/QPSK/4PSK	FEC
128000	0.5 GMSK/QPSK/4PSK	FEC
120 lpm	FM, 1500-2300 Hz	None
50	15-QPSK	None
441	4FSK	None
3.09	4-GFSK	FEC + CRC
1.46	4-GFSK	FEC + CRC
20.833	4-GFSK	FEC + CRC
6.25	8-GFSK	FEC + CRC
100/200/300	FSK, 170/200 Hz shift	FEC + ARQ
122.5	ASK	None
4.375	4FSK	FEC
1.736	9FSK	FEC
2.69	65FSK	FEC
15.625	16FSK	FEC
2000	OQPSK	FEC + CRC
5/10/20	64-DPSK	FEC
15.63/31.25	32-FSK	FEC
1200	FSK	ARQ
100/200	FSK, 200 Hz shift	ARQ
100	2-DBPSK/PI-4DQPSK/8,16DPSK	FEC + ARQ
100	(2-18)-DBPSK/DQPSK	FEC + ARQ
31.25	BPSK	None
1.667	65FSK	FEC + CRC
31.25	QPSK	FEC
62.5/125	BPSK	None
62.5/125	QPSK	FEC
83.33	15-QPSK	FEC + ARQ
45.45	FSK, 170 Hz shift	None
120-line B/W	FM, 1200-2300 Hz	None
240-line RGB	FM, 1200-2300 Hz	None
240-line RGB	FM, 1200-2300 Hz	None
4800	4FSK/QPSK/C4FM	FEC
1/2/4	9FSK/2-9FSK	None
23-94	OFDM,MFSK, MPSK, QAM	FEC
1.46	4FSK	FEC

15.7 References and Bibliography

Bern, D., W2LNX, and Elkin, K., KB3TCB, "High-Speed Wireless Networking in the UHF and Microwave Bands," *QST*, Nov. 2015, pp. 33 – 37.

Bern, D., W2LNX, and Elkin, K., KB3TCB, "High-Speed Wireless Networking in the UHF and Microwave Bands," **files.tapr.org/meetings/DCC_2014/DCC2014-High-Speed-Wireless-Networking-UHF-Microwave-Bands-W2LNX-presentation.pdf**.

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

Ford, S., WB8IMY, *Get on the Air with HF Digital*, 3rd Edition (ARRL, 2022).

Ford, S., WB8IMY, *HF Digital Handbook*, 4th Edition (ARRL, 2007).

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

Franke, S., K9AN, and J. Taylor, K1JT, "Open Source Soft-Decision Decoder for the JT65 (63,12) Reed-Solomon Code," *QEX*, May/June 2016, pp. 8 – 17.

Franke, S., K9AN, and Taylor, J., K1JT, "The MSK144 Protocol for Meteor-Scatter Communication," *QEX*, Sep./Oct. 2018, pp. 30 – 36.

Franke, S., Somerville, B., and Taylor, J., "The FT4 and FT8 Communication Protocols," *QEX*, July/Aug. 2020, pp. 7 – 17.

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

"Manual of Transmission Methods – Reference Document," Rohde & Schwarz, 2014, available at **www.udxf.nl/R&S-Manual-of-Transmissing-Methods.pdf**.

Palermo, N., IV3NWV, "Q-ary Repeat-Accumulate Codes for Weak Signals Communications."

Popiel, G., KW5GP, *High-Speed Multimedia for Amateur Radio* (ARRL, 2016).

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

Yaesu System Fusion Standard, "Amateur Radio Digital Standards," v1.0 January 15, 2013, rev 1.01 April 18, 2013, available for download from **www.yaesu.com**.

Taylor, J., K1JT, "The JT65 Communications Protocol," *QEX*, Sep./Oct. 2005, p. 3.

Taylor, J., K1JT, and Walker, B., W1BW, "WSPRing Around the World," *QST*, Nov. 2010, p. 30.

Taylor, J., K1JT, "WSJT: New Software for VHF Meteor-Scatter Communication," *QST*, Dec. 2001, pp. 36 – 41.