B Appendix: A Closer Look at Physical Wireless Communications

B.1 Introduction

This appendix augments the discussion in Chapter 5 with more analytical treatment of wireless physical communications. The emphasis is on transmission, error detection, and error correction. The reader should review the related sections in Chapter 5 before reading this material. The material in this appendix will be expanded in the future on an as-needed basis.

B.2 Wireless Antennas and Propagation – A Closer Look

B.2.1 Antenna Gain

Antenna gain is a measure of directionality of antennas. It simply represents the power output, in a particular direction, compared to that produced in any direction by a perfect omnidirectional antenna with isotropic radiation patterns. For example, a gain of 1 indicates that the antenna is ominidirectional, while a gain of 3 in one direction indicates that the directionality of an antenna is 3 times greater than that of an omnidirectional antenna. A gain of 10 or higher requires very heavy directional and focused beams. The gain depends on wavelength and the physical size and shape of the antenna.

Antenna gain is usually represented in terms of effective area of an antenna. The effective area is related to the physical size and shape of an antenna. For example, effective area for a parabolic antenna of face area A is 0.56A. Recall that size of an antenna is roughly equal to the transmission wavelength. Relationship between antenna gain and effective area is given by:

$$G = \frac{4\pi A_e}{\lambda^2} = \frac{4\pi f^2 A_e}{c^2}$$

Where:

G = antenna gain

A_e = effective area

f = carrier frequency

c = speed of light (» $3x10^8$ m/s)

λ = carrier wavelength

Table B-1: Antenna Gains and Effective Areas

Type of Antenna	Effective Area	Power Gain (relative to Isotropic)
Isotropic	$\lambda^2/4.p$ where p = 22/7	1
Horn , mouth Area A	0.81A	$10A/\lambda^2$
Parabolic , mouth Area A	0.56A	$7A/\lambda^2$

B.2.2 Free Space Loss

As a signal traverses the free space (air), it disperses in different directions and consequently loses strength. Free space loss is the primary reason for attenuation for satellite signals because the signal attenuates as it disperses over long distances. Even if no other loss occurs, free space loss weakens the signal because it is being spread over a larger and larger area.

Free space loss L_i, for an ideal isotropic antenna is given by:

$$\frac{P_t}{P_r} = \frac{(4\pi d)^2}{\lambda^2} = \frac{(4\pi f d)^2}{c^2} \qquad \text{...... (B.1)}$$

Where

P_t = signal power at transmitting antenna

P_r = signal power at receiving antenna

λ = carrier wavelength

d = propagation distance between antennas

c = speed of light (» 3 ´ 10 8 m/s)

where d and λ are in the same units (e.g., meters)

The power loss for other types of antennas takes into account the gain of the antenna. Thus the general formula for the free space loss L is given by

$L = L_i/G_tG_r$

where

L_i = free space loss for the isotropic antenna given by equation B.1.

G_t = gain of the transmitting antenna

G_r = gain of the receiving antenna

B.2.3 More on Noise

Signals traveling through the air are distorted due to different types of noises. Intermodulation noise, a common distortion, occurs if signals with different frequencies share the same medium. It is caused by interference caused by a signal produced at a frequency that is the sum or difference of original frequencies. This is typically due to defects. In the system. Other noises are crosstalk (unwanted coupling between signal paths such as signals from one user interfering with others) and impulse noise (irregular pulses or noise spikes typically caused by external electromagnetic disturbances such as lightning or faults and flaws in the communications system).

Another type of noise, called **thermal noise**, causes distortions due to agitation of electrons (called white noise). Thermal noise, as the name implies, is present in all electronic devices and transmission media, and cannot be eliminated because it is a function of temperature. Thermal noise is particularly significant for satellite communication and causes weakness of satellite signals. The amount of thermal noise to be found in a bandwidth of 1Hz in any device or conductor is:

$$N_0 = kT\ (\text{W/Hz})$$

Where:

N_0 = noise power density in watts per 1 Hz of bandwidth

k = Boltzmann's constant = 1.3803 ´ 10^{-23} J/K

T = temperature, in kelvins (absolute temperature)

Noise is assumed to be independent of frequency. Thermal noise present in a bandwidth of B Hertz (in watts) is given by

$$N = \mathrm{k}TB$$

The common expression used to measure noise is the signal to noise ratio (SNR). The expression $\boldsymbol{E_b/N_0}$ is used in digital communications. E_b/N_0 is similar to signal/noise ratio but is more convenient for digital systems. It represents the ratio of signal energy per bit to noise power density per Hertz

$$\frac{E_b}{N_0} = \frac{S/R}{N_0} = \frac{S}{\mathrm{k}TR}$$

Where S = signal and R = bit rate. The bit error rate (BER) for digital data is a function of E_b/N_0. Given a value for E_b/N_0 to achieve a desired error rate, parameters of this formula can be selected. As bit rate R increases, transmitted signal power must increase to maintain required E_b/N_0.

B.3 Error Detection Techniques

B.3.1 Overview

Wireless systems must be designed for error detection as well as error correction. In wired systems, every time an error is detected, automatic repeat request (ARQ) protocols are used to retransmit the block of data in error. This approach is not suitable for wireless systems because wireless errors can cause numerous re-transmissions and re-transmission can also be in error due to bad weather (rain and thunderstorm), fading, attenuation, and noise. In addition, wireless systems are slower, hence retransmissions can be slow and expensive. For example, retransmissions in a satellite network can cause significant delays. It is much better to try to recover from errors as much as possible. Error correction codes, or forward correction codes (FEC), are designed to detect and correct errors.

This section gives an overview of the various error detection, ARQ, and correction schemes after a brief overview of synchronous/asynchronous communications and flow control. The discussion in this section is admittedly brief and sketchy for people who want to get a thorugh understanding of this vast and important area of development. The interested reader should consult the following books for a better understanding of the subject matter:

- J. H. Van Lint. *Introduction to Coding Theory*. Springer Verlag, 1999.
- Stephen B. Wicker. Error Control Systems for Digital Communication and Storage. Prentice Hall, 1995.
- V. S. Pless (Editor), W. C. Huffman (Editor) and Richard A. Brualdi (Editor). *Handbook of Coding Theory, Volume 1 and Volume 2.* North-Holland, 1998.
- Richard E. Blahut *Algebraic Codes for Data Transmission*. Cambridge Univ. Press, 2001.

B.3.2 Synchronous/Asynchronous Communications

Error detection and correction is based on asynchronous or synchronous. In ***asynchronous transmission***, also called start-stop transmission, data is transferred one character at a time, at an undeterministic or random time. In order for the start and end of a character to be recognized by the receiver, each character is surrounded by a start and a stop bit, hence the name start-stop (see Figure B-1a). Asynchronous communications were introduced in the old telegraphic days when the start and stop bits physically started and stopped the paper tape mechanical processes on the receiver side. They are still used very frequently in terminal communications. The sending and receiving logic needed for asynchronous communications is very simple and can be implemented economically in simple devices. The main difficulty with the asynchronous communications is that if a start bit is not recognized then the whole character may be misread. Due to this, asynchronous communications are not used at high speeds because the chances of losing data bits are greater at higher speeds.

In ***synchronous data transmission***, also known as block transmission, data is transmitted in blocks of many characters. Header and trailer characters are attached to each block so that the receiver can recognize the start or stop of a block (see Figure B-1b). Common examples of synchronous transmission are the transmission of each terminal screen as a single block and file transfers which send and receive many file records as blocks of data. Synchronous communications can operate at much higher speeds than asynchronous because the whole block can be transmitted by using one send command. The logic of generating the block headers and trailer fields is more complicated. More importantly, sophisticated error checking fields are generated and verified as we will see in the next sections. The formats of the headers, trailers, and error checking fields depend on the synchronization scheme being employed. Some systems, for example, include the start-stop bits in the data block, in addition to the block headers, trailers and error checking fields.

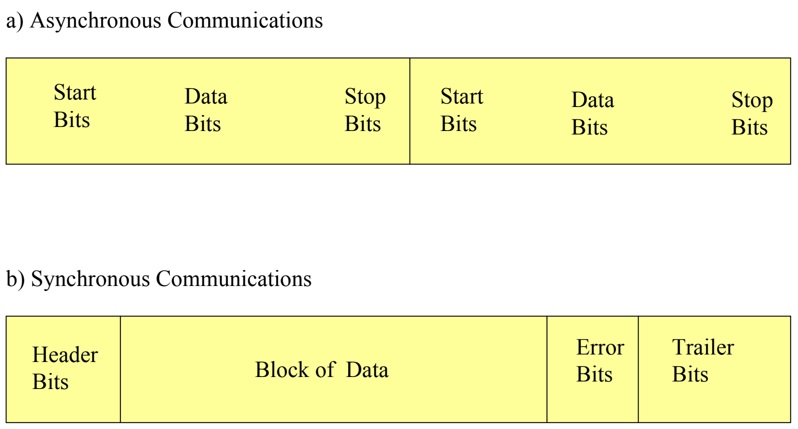

Figure B-1: Asynchronous and Synchronous Communications

B.3.3 Flow Control

The flow control feature of a protocol decides if the two entities will communicate with each other over a communication medium in one of the following modes:

- Simplex: one way communications, always. This mode is rarely used today.
- Half Duplex: two way communication, one at a time. This mode is used very commonly in many instances.
- Full Duplex: two way communications, simultaneously. This is the fastest mode of information transmission.

Error detection and correction also depends on information flows, as we will see in Section B.3.7. It should be noted that to provide a full duplex path between two end-devices, all intermediate devices and media on the communication path must operate in full duplex mode

Full duplex protocols are not trivial to implement because the two entities can send and receive information simultaneously. Full duplex protocols can use two channels (one for sending, one for receiving) or they can intermix data and control in the same message. Special "framing" bits are employed to indicate message type (data, control).

The acknowledgement control schemes can employ the following schemes:

- Acknowledgement Ignore: The sender just keeps on sending the message and does not wait for any acknowledgement from the receiver.
- Stop and Wait: The sender waits for an acknowledgement before sending the next message. The positive acknowledgement (ACK) indicates that the message has been received properly on the other side. The negative acknowledgement (NACK) indicates that the message has not been received properly due to sequencing or frame check error. This protocol is sometimes also referred to as the "ACK-NACK" protocol.
- Sliding Window: Instead of waiting for ACK-NACK for each message, the sender transmits up to n messages (n is the window size) before waiting for an acknowledgement. The receiver remembers the next message to be received within the window. In case of an error, the sender may retransmit all n messages. This facility greatly improves the speed and efficiency of the network and is especially useful in global networks where waiting for response of each message is too slow due to turnaround delays. Many protocols, in particular TCP, use a sliding window.
- Piggybacking: When two entities are operating at full or half duplex, a common situation in interactive computing, then the acknowledgements can be sent (piggybacked) with messages instead of a separate message which only carries the acknowledgement. For example, the acknowledgement from message i, system A, can be sent with the next data for system A. Piggybacking is also used in many protocols, such as SDLC.

B.3.4 Error Handling in Flow Control

The error handling schemes commonly use a timeout mechanism where the sender waits for an ACK-NACK for a while (timeout period) and then resends the message. Thus if a message is lost on its way, it is resent. However, timeout mechanisms may resend a message which was properly delivered and processed but for which the acknowledgement was lost. In this case, the same message will be reprocessed erroneously (for example, this could withdraw the same amount twice from a bank account). To circumvent this situation, the sender attaches a sequence number to each message before it is sent. The receiver checks for the sequence number with an expected sequence number to verify that the correct message has arrived. Sequence numbers are useful to detect and recover from many types of errors. For example, consider that message number 5 is being sent:

- If the message is lost before it is received and processed, the timeout mechanism resends the same message. The sequence number checking finds no jump in sequence numbers (message 5 received after message 4, 5 is resent but the receiver does not know).
- If an acknowledgement is lost on its way back after being processed, then the receiver will resend message 5. However, the receiver will find that this is a duplicate message and throw it away.

B.3.5 Sample Flow Control Protocols

Let us review a few generic protocols to illustrate how the aforementioned classification can be applied. The simplest information exchange protocol is the ***contention-based, simplex, acknowledgement-ignored protocol*** in which the sender keeps on sending the information and the receiver keeps on receiving the information. This protocol (known as the unrestricted simplex protocol), displayed in Figure B-2, is based on the following assumptions:

- Sender and receiver are always ready.

- There are no errors on the communication channel.
- There are infinite buffers on both sides or the processors at both sides are of the same speed.
- Transmission is always in one direction (sender always sends, receiver always receives).

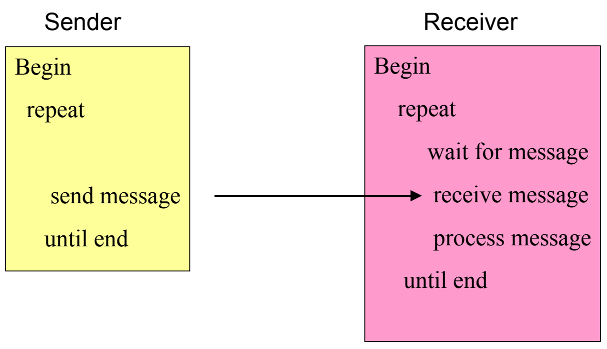

Figure B-2: Unrestricted Simplex Protocol

These assumptions are naturally very simplistic and are not satisfied in most communication systems. Let us remove the first three assumptions for a more realistic protocol. A Stop and Wait, Timeout Half-Duplex Protocol with Sequence Numbers is used commonly to handle initial handshaking, communication channel errors and the different machine capabilities of the senders and receivers. A simplified version of this protocol is shown in Figure B-3. Although this protocol satisfies many practical situations, the computer to computer communications protocols commonly employ full duplex, sliding window, and piggybacking schemes.

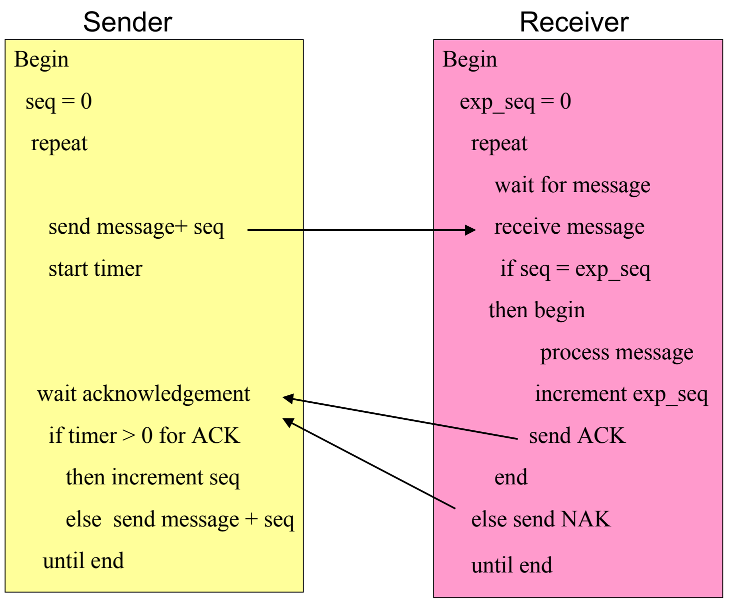

Figure B-3: Stop and Wait, Timeout Protocol With Sequence

Remember that as more functions are added in a protocol, the message format as well as the rules of the protocol become more complicated. For example, in the unrestricted simplex protocol the message format is very simple (data only). However, message formats for a full duplex protocol with piggybacking, sliding windows and sequencing must provide bits to clearly identify the message type (ACK versus data), control information (session initiation/termination commands) and sequence numbers. A generic message format is shown in Figure B-4.

The header identifies the start of a message and usually shows the sender/receiver addresses. The flag fields are used to show if the data is application data, control or an acknowledgement. The flags are used to carry several other pieces of information such as sequence number and command category and/or priority. The data field contains the application data, acknowledgement and/or commands being sent. The trailer field shows the end of message delimiters and error detection fields (checksums). A good discussion of generic protocols can be found in [Tannenbaum 1996].

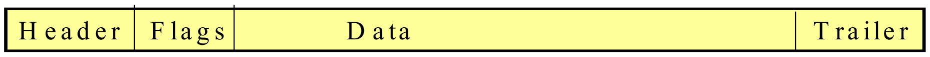

Figure B-4: A Generic Message Format

B.3.6 Error Detection Schemes

B.3.6.1 Basic Error Detection Process

The basic error detection process consists of information verification between the transmitter and the receiver. In the transmitter, an error-detecting code (check bits) is calculated from data bits for a given frame. These check bits C are appended to data bits. The receiver separates incoming frame into data bits and check bits, calculates check bits C' from received data bits, and compares calculated check bits C against received check bits C'. An error is detected if there is a mismatch, i.e., if C is not equal to C'.

Different schemes are used by communication protocols to detect if the data in a message has been distorted due to communication media errors. Examples of error control schemes are the vertical redundancy checks (VRC), which add a parity bit per byte of each data byte sent, cyclic redundancy checks (CRC), which add parity bits per block and the combined VRC and CRC schemes. Most of the current protocols use the combined VRC and CRC schemes for block (asynchronous) communications. The sender builds the check bits as a frame check sequence (FCS) by using some predetermined algorithm and sends FCS with the message. The receiver applies the same algorithm on the received message and builds a received message FCS. The receiver then compares the two FCSs and indicates an error (NACK) if the two do not match. See the following fgure.

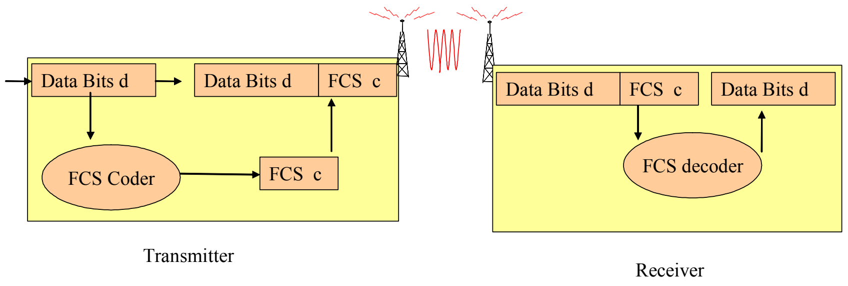

Figure B-5: FCS Transmission and Reception

B.3.6.2 Vertical Redundancy Check (Parity Check)

Parity checking is one of the oldest and most widely used error detection systems. In this method, a parity bit is appended to a block of data. In even parity systems, added bit ensures an even number of 1s and in odd parity systems, added bit ensures an odd number of 1s. For example in a 7-bit character [1110001] the even parity is [11100010] and odd parity [11100011]. The receiver of an odd parity system expects odd number of bits and even parity system expects an even number of bits. If this does not happen then an error is detected.

B.3.6.3 Cyclic Redundancy Check (CRC)

I this scheme, the transmitters generate a CRC code that is checked by the receiver. For a *k*-bit block, the transmitter generates an (*n*-*k*)-bit frame check sequence (FCS). The resulting frame of *n* bits is exactly divisible by a predetermined number. The FCS is appended to the data bits and sent to the receiver. The receiver divides incoming frame by a predetermined number. If no remainder, the receiver assumes no error. Otherwise, an error is detected.

B.3.7 Automatic Repeat Request (ARQ)

After the receiver detects an error, ARQ is used to repeat (retransmit) the blocks in error. ARQ is commonly used in data link control and transport protocols such as TCP. It relies on use of an error detection code (such as CRC) and is closely related to flow controls discussed previously.

ARQ flow control assures that transmitting entity does not overwhelm a receiving entity with data. Thus the protocols with flow control mechanism allow multiple PDUs (Packet Data Units) in transit at the same time. PDUs arrive in the same order as they are sent. In TCP and many other systems, blocks of data are broken into PDUs. There are several reasons for breaking up a block of data before transmitting:

- Limited buffer size of receiver
- Retransmission of PDU due to error requires smaller amounts of data to be retransmitted
- On shared medium, larger PDUs occupy medium for extended period, causing delays at other sending stations

Although different types of flow controls are used, sliding-window flow control is one of the most popular. In this mechanism, the transmitter maintains a list (window) of sequence numbers that are allowed to send the information without waiting for an acknowledgement. The receiver, on the other side, maintains another list that it keeps receiving before sending an acknowledgement. Thus a transmitter and a receiver can use a window that consists of 5 next PDUs that can be sent one after another. Only after all 5 have been received, an acknowledgement is generated by the receiver. This is much more efficient than the stop and wait protocol where the transmitter sends a PDU and waits for an acknowledgement before sending the next PDU.

Error control mechanisms are used to detect and correct transmission errors of the following types:

- Lost PDU : a PDU fails to arrive
- Damaged PDU : PDU arrives with errors

The receiver detects these errors and discards suspect PDUs. The following actions are taken by the receiver:

- Positive acknowledgement. Destination returns acknowledgment of received, error-free PDUs
- Retransmission after timeout. Source retransmits unacknowledged PDU

- Negative acknowledgement and retransmission. Destination returns negative acknowledgment to PDUs in error, causing a retransmission.

B.3.7.1 Error Detection Probabilities

It is interesting to look at the error detection process by reviewing the following error detection probabilities:

P_b : Probability of single bit error (BER)

P_1 : Probability that a frame arrives with no bit errors

P_2 : While using error detection, the probability that a frame arrives with one or more undetected errors

P_3 : While using error detection, the probability that a frame arrives with one or more detected bit errors but no undetected bit errors

The following interrelate these probabilities.

$$P_1 = (1 - P_b)^F$$
$$P_2 = 1 - P_1$$
$$P_3 = 0$$

where F is the number of bits per frame. The first equation indicates that the probability of no error decreases as the size of the frame increases. This is intuitively obvious. The second equation indicates that the probability that a frame arrives with undetected errors decreases as the probability of no bit errors increases. The final equation assumes that the probability that a frame arrives with no undetected bit errors is zero, i.e., all bit errors are detected.

B.4 Error Correction Mechanisms

B.4.1 Overview

In case of a transmission error, the typical approach is to retransmit the data block in error. But in wireless systems, re-transmitting data several times until the correct data is received is highly inefficient. Error correction is a much more desirable approach to ensure that the correct data is received. The goal is to correct the errors or compensate for the fading and attenuation noises discussed in a Chapter 5. The techniques include forward error correction, adaptive equalization, and diversity techniques. In these techniques, the transmitter builds a forward error correction (FEC) code that maps each d-bit block into a c-bit block codeword. The codeword is sent by using wireless transmission. On the receiver side, the incoming signal is demodulated and the block is passed through an FEC decoder (see Figure B-6). The outcomes of the FEC decoder are:

- No errors present
- Codeword produced by decoder matches original codeword
- Decoder detects and corrects bit errors
- Decoder detects but cannot correct bit errors; reports uncorrectable error
- Decoder detects no bit errors, though errors are present

Based on these outcomes, appropriate actions are taken by the receiver.

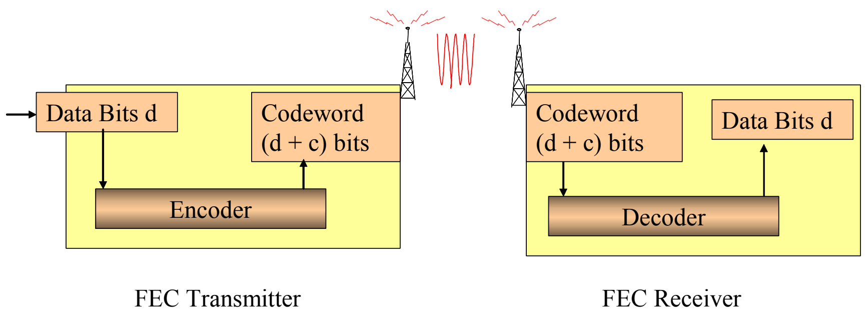

Figure B-6: Forward Error Correction -- Conceptual View

B.4.2 Forward Error Correction

In this type of method, the transmitter adds error-correcting code to data block. This code, called forward correction code (FEC), is designed to detect and correct errors and is a function of the data bits and contains redundant data. The receiver calculates error-correcting code from incoming data bits. If the calculated code matches the incoming code, no error occurred. But if error-correcting codes does not match, receiver attempts to determine bits in error and attempts to correct the error. This error correction is possible thanks to the to the redundant data in error correction code. Due to the redundant data, the total bits in a message may be 2 to 3 times the data bits. However, this redundant data is used to correct the errors and is thus quite helpful.

The foundation of error correction is classical work done by Claude Shannon [Shannon 1949]. Shannon's theorem, referred to as the Shannon's limit or Shannon's capacity, refers to the maximum rate of error free data that can theoretically be transferred over a communication channel, if the link is subject to random transmission errors. The Shannon limit states the following:

$$C = W.\log_2(1 + S/N)$$

Where:

C= net channel capacity in bits per second after error correction

W= raw channel frequency or bandwidth

S/N = Signal to noise ratio given as a straight signal not in decibels

This shows that there is a theoretical maximum amount of information that can be transmitted over a bandwidth-limited channel in the presence of background noise. The solution to overcoming the noise in communication channels, according to Shannon, was to divide the data into strings of bits and add to each string a set of extra bits, called parity bits, that would help identify and correct errors at the receiving end. Thus the whole idea is to add parity bits to the data bits for error detection and correction. The simplest technique, used for decades is the even-odd parity bits discussed previously. In the even/odd parity systems, also known as CRC/VRC systems, the sender builds the check bits as a frame check sequence (FCS) by using some predetermined algorithm and sends FCS with the message. The receiver applies the same algorithm on the received message and builds a received message FCS. The receiver then compares the two FCSs and indicates an error (NACK) if the two do not match. As we will see below, this idea has been extended significantly by different researchers for better error detection and correction.

This section gives a quick introduction to a very vast area of work. Additional details can be found in [Peterson 1972, Macwilliams 1972, Lin 1983]. .

B.4.2.1 Block Code Error Correction by using Hamming Distance

A common error correction mechanism uses Hamming distance that represents the number of *different* bits for 2 *n*-bit binary sequences; For example, consider the following two data bits v1 and v2:, and the corresponding difference bits d:

v_1=00101111;

v_2=11100111;

d(v1, v_2)=3

The Hamming distance d is 3 because the first, second, and fifth bits between v1 and v2 are different. As we will see, d is used to correct the transmission errors. The basic algorithm is:

- For each data block, create a codeword
- Send the codeword
- If the code is invalid, look for data with shortest Hamming distance (possibly correct code)

For example, consider the following datablock and and the codeword generated (note that the codeword is longer than the datablock and has many redundant bits).

Datablock (k=2)	Codeword (n=5)
00	00000
01	00111
10	11001
11	11110

Suppose you receive codeword 00100 (error). Then the receiver attempts to find the closest code. In this example, it is 00000 (only one bit different, i.e., the Hamming distance is only 1). Thus the receiver assumes that the intended code was 00000 and thus the datablock sent was 00. Thus the receiver has corrected the transmission error.

B.4.2.2 Convolutional Codes

This method generates redundant bits continuously instead of waiting for the entire data block to be read before producing a code. Convolution codes are much more efficient than block codes especially for large block sizes. Instead of generating large error correction codes, transmitting them, and then determining the error detecting code on the other side, this technique allows continuous generation, detection, and correction. The input to the convolution coder processes d data bits at a time and produces c code bits for every d input bits. Thus, given a data block of d bits, it generates a code of c bits where (c > d). Error checking and correction is carried out continuously by using a memory M that keeps track of the most recent bits. d and c are generally very small. The c-bit output of (c, d, M) code depends on the current block of d input bits and previous M-1 blocks of d input bits (memory). For details, see Robert J. McEliece, "Convolutional Codes", Theory of Information and Coding, Encyclopedia of Mathematics and its Applications.

B.4.2.3 Adaptive Equalization

Adaptive equalization is used to compensate for the fading and attenuation effects in wireless transmissions. It can be applied to transmissions that carry analog or digital information such as analog voice or video and digital data including digitized voice or video. This technique is primarily used to combat intersymbol interference (ISI). In ISI, one or more delayed copies of a pulse may arrive at the same time as the primary pulse for a subsequent bit.

Adaptive equalization involves gathering dispersed symbol energy back into its original time interval so that ISI is minimized. The adaptive equalization techniques involve a wide range of sophisticated digital signal processing algorithms. The original signal is sampled several times and transmitted with a sequence number that helps resolution on the other side.

B.4.2.4 Diversity Techniques

Diversity is based on the fact that individual channels experience independent fading events -- some more than others. To minimize errors, it provides multiple logical channels such as the following:

- Space diversity – techniques involving physical transmission path
- Frequency diversity – techniques where the signal is spread out over a larger frequency bandwidth or carried on multiple frequency carriers
- Time diversity – techniques aimed at spreading the data out over time

This technique does not eliminate errors but reduces them because the signal is spread to different channels to minimize errors. The adaptive techniques can then be used to handle any fading and attenuation errors that still may arise.

B.4.3 Turbo Codes

Turbo codes are a class of recently-developed high-performance error correction codes that come closest to approaching the Shannon limit, the theoretical limit of maximum information transfer rate over a noisy channel. As stated previously, according to Shannon, the solution to overcoming the noise in communication channels is to divide the data into strings of bits and add to each string a set of extra bits called parity bits that would help identify and correct errors at the receiving end.

Turbo codes were invented by two French communication engineers Claude Berrou and Alain Clavieux [Berrou 1993]. Turbo codes have introduced a new class of methods describing the iterative decoding of Forward Error Correction (FEC) codes. A number of research groups have been formed to experiment with turbo codes and literature in this area is growing rapidly (see the sidebar "Sources of Information on Turbo Codes"). In addition, hardware products have started appearing such as Turbo Product Code (TPC) chips.

What are these tutbo codes? Turbo codes basically split the string into more manageable components. Instead of a single decoder and encoder, turbo codes use two encoders at one end and two decoders at the other (see Figure B-7). How do they help? The main problem with the typical error correction codes is that that too many parity bits have to be added for a good code word. However, by using many parity bits a computational complexity problem arises because of the large amount of computations needed to decode the data. In fact, the closer you get to Shannon’s limit, the more complicated this process becomes. This is because more parity bits are needed as you reach the Shannon's limit and the code words get longer and longer. This results in the decoder at the receiving end having to search through a large collection of codewords.

Shannon's work demonstrated that large block-length random codes achieve channel capacity; however, large random codes require a complex decoder to decode and extract the information from the codeword. Codes with some structure permit decoding with reasonable complexity. However, it does not perform as well as random codes. Turbo code trade offs both features by providing codes that appear random, while providing enough structure to permit decoding.

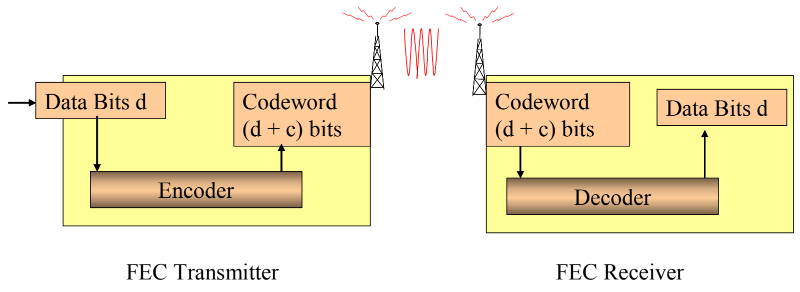

Figure B-7: Turbo Code Transmission and Reception

Figure B-7 shows the general process of turbo codes. As shown, two encoders and two decoders are used in the encoding and decoding processes. The encoding process consists of the following steps:

- A *turbo* encoder is a combination of two simple encoders. The input is a block of data bits and the output is a combination of the code output from each encoder together with the original data input.
- Each simple encoder generates parity symbols from simple recursive convolutional codes. Typically, both of the encoders are identical. Thus if you feed the same input data into each encoder, the output would be the same. However, in turbo code, one of the encoder receives the original data as its input, while another encoder receives its input directly from the interleaver.
- The interleaver is a simple device that rearranges the order of the data bits in a prescribed, but irregular, manner and provides the random-like properties of turbo code. The final output from the turbo encoder is a concatenation of the original input data stream and the parity outputs of the two encoders into one single codeword.

Two decoders and an interleaver perform the reverse operations, howver, the two decoders work with each other. Decoding of error correcting codes is basically a comparison of the probabilities for different codewords. Each constituent decoder sends *a posteriori* likelihood estimates of the decoded bits (soft, or temporary, output) to the other decoder and uses the corresponding estimates from the other decoder (soft input) as *a priori* likelihoods. The decoded information bits are available to each decoder to initialize the *a priori* likelihoods. This cycle continues for a fixed number of iterations. After the final iteration, the decoder calculates a "hard-quantized (final)" version of the likelihood estimates of either of the decoders. The performance of the turbo decoder generally improves from iteration to iteration. However, higher number of iterations result in greater amount of decoding delay and thus decay system performance.

The decoders use an algorithm known as MAP (Maximum and Posteriori). The MAP decoding process consists of four steps: forming a trellis, finding maximum likelihoods, finding the surviving path, and determining the soft output. See [Liu 2000] for details of MAP. The performance of turbo codes depend on parameters such as the number of decoding iterations, the interleaver length and the rates. Several studies of the performance have been carried out by different researchers (see, for example, [Singh 2001]). These studies confirm that turbo codes offer significant enhancement in performance over other coding schemes.

The major drawback of turbo codes is the long decoding delay -- thus they are useful mainly in systems that can tolerate such delays. Turbo codes are particularly useful in deep space communications and many experiments are underway in this area. For example, NASA is planning missions that will depend on turbo codes. NASA's future deep-space transponders, such as the ones in the Mars Reconnaissance Orbiter and Messenger, will support turbo codes. The European Space Agency (ESA), based in Paris, launched SMART-1 in 2003, the first probe to go into space using turbo codes for data transmission. Turbo codes are also used in Japan where they have been incorporated into the standards for 3G mobile phones based on the Universal Mobile Telecommunications System (UMTS). Turbo codes are used in the transmission of pictures, video and mail. In London, the Global Area Network of Inmarsat is also incorporating turbo code into digital audio broadcasting.

Turbo codes have sparked a great deal of research and industrial activity (see the sidebar "Sources of Information for Turbo Codes").

Sources of Information for Turbo Codes

Key Articles and Books

- C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: turbo codes", Proc. IEEE Int Conf. Communication, Geneva, Switzerland, May 1993.
- C.E. Shannon, "A Mathematical Theory of Communications", Bell Systems Technical Journal, January 1948.
- Chris Heegard and Stephen B. Wicker. Turbo Coding. Kluwer Academic, 1999.
- E. Guizzo, "Closing in on the Perfect Code". IEEE Spectrum, March 2004. pp. 36-42.
- Ye Liu, "MAP Algorithm for Decoding Linear Block Codes Based on Sectionalized Trellis Diagrams", IEEE Transactions on Communication, April 2000.
- William E. Ryan, "A Turbo Code Tutorial", New Mexico State University, Las Cruces, NM 88003.

Suggested Web Links

♦http://www331.jpl.nasa.gov/public/JPLtcodes.html (JPL Page on Turbo Codes)
♦http://www.vocal.com/white_paper/CF-036.pdf
♦http://www.bell-labs.com/org/1133/Research/Communications/
♦http://www.ou.edu/idp/teamlearning/docs/documentation1.pdf
♦http://www331.jpl.nasa.gov/public/TurboPerf.pdf
♦http://washingtontimes.com/upi-breaking/20040226-022207-2605r.htm
♦http://www.coe.montana.edu/ee/rwolff/EE548/papers/turbocodes/
♦http://encyclopedia.thefreedictionary.com/Shannon's%20theorem

♦http://faraday.ee.emu.edu.tr/eaince/ee562/Public/sova.html

Sample Turbo Coding Research Groups

- Block Turbo Codes Home Page, ENST de Bretagne, France
- Coding Research Group, University of Hawai'i at Monoa, USA.
- Coding Research Group, University of Notre Dame, USA.
- Communications Systems and Research Section, Jet Propulsion Laboratory, USA.
- Communications Group, Politecnico di Torino, Italy.
- Communications Laboratory, Technion - Israel Institute of Technology, Israel.
- Communications Research Group, University of York, United Kingdom.
- CRISP Turbo Coding Project, Cornell University, USA. .
- DataLab, University of California, Irvine, USA.
- Error Correcting Codes Home Page, Japan.

- Institute for Communications Engineering, Technische Universität München, Germany.
- Mobile Multimedia Research, University of Southampton, United Kingdom.
- Telecommunications Institute II, University Erlangen-Nürnberg, Germany.
- Turbo Codes in CCSR, University of Surrey, United Kingdom.
- Turbo Codes, Technische Universität Dresden, Germany.
- The Turbo Codes Home Page, University of Virginia, USA.
- Turbo Codes at West Virginia University, West Virginia University, USA.
- Wireless Systems Laboratory, Georgia Institute of Technology, USA.

B.5 Analysis of Encoding Techniques

B.5.1 Signal Encoding Criteria

The following criteria determine how successful a receiver will be in interpreting an incoming signal:

- Signal-to-noise ratio: An increase in SNR decreases bit error rate
- Data rate: An increase in data rate increases bit error rate
- Bandwidth: An increase in bandwidth allows an increase in data rate

Different factors, such as the following, can be used to compare different encoding schemes:

- Clocking: ease of determining beginning and end of each bit position.
- Signal interference and noise immunity: performance in the presence of noise
- Cost and complexity: the higher the signal rate to achieve a given data rate, the greater the cost

B.5.2 Binary Frequency-Shift Keying (BFSK) and Multiple Frequency-Shift Keying (MFSK)

In BFSK, two binary digits are represented by two different frequencies. Thus frequency f1 may represent 1 and f2 may represent a 0. BFSK is less susceptible to error than ASK (amplitude shift keying). In MFSK, more than two frequencies are used. MFSK is more bandwidth efficient but more susceptible to errors. MFSK is represented by sine wave s:

$$s_i(t) = A \cos 2\pi f_i t \quad 1 \leq i \leq M$$

where

$f_i = f_c + (2i - 1 - M)f_d$

f_c = the carrier frequency

f_d = the difference frequency

M = number of different signal elements = 2^L

L = number of bits per signal element

To match data rate of input bit stream, each output signal element is held for T_s=LT seconds, where T is the bit period (data rate = 1/T). So, one signal element encodes L bits

B.5.3 Phase-Shift Keying (PSK)

Two-level PSK (BPSK) uses two phases to represent binary digits and differential PSK (DPSK) -Phase shift with reference to previous bit, Binary 0 – signal burst of same phase as previous signal burst, Binary 1 – signal burst of opposite phase to previous signal burst.

Multilevel PSK. Using multiple phase angles with each angle having more than one amplitude, multiple signals elements can be achieved

$$D = \frac{R}{L} = \frac{R}{\log_2 M}$$

where

D = modulation rate, baud

R = data rate, bps

M = number of different signal elements = 2^L

L = number of bits per signal element

B.5.4 Performance of Different Schemes

Bandwidth of modulated signal (B_T)

ASK, PSK B_T=(1+r)R

FSK B_T=2DF+(1+r)R

Where:

R = bit rate

$0 < r < 1$; related to how signal is filtered

DF = f_2-f_c=f_c-f_1

These expressions show that FSK consumes more bandwidth than ASK and PSK. But ASK suffers more distortions (amplitudes are distorted easily by noise) than PSK, thus PSK is more favorite. Let us then look at the bandwidths of multilevel PSK and FSK (MPSK and MFSK.).

Bandwidth of modulated signal (B_T) for MPSK

$$B_T = \left(\frac{1+r}{L}\right)R = \left(\frac{1+r}{\log_2 M}\right)R$$

For MFSK

$$B_T = \left(\frac{(1+r)M}{\log_2 M}\right)R$$

where

L = number of bits encoded per signal element

M = number of different signal elements

B.5.5 Quadrature Amplitude Modulation

QAM is a combination of ASK and PSK and combines the good features of both. Basically two different signals are sent simultaneously on the same carrier frequency, one is shifted 90 degrees from the other (hence phase shifting). The signals are AM modulated and are sent simultaneously on the same medium. The transmitted signal can be represented as following:

$$s(t) = d_1(t)\cos 2\pi f_c t + d_2(t)\sin 2\pi f_c t$$

At the receiver, the twp signals are demodulated and the results are combined to produce the original signal.

B.5.6 Pulse Code Modulation (PCM)

PCM is based on the sampling theorem (sample rate should be higher than twice highest frequency). Each analog sample is assigned a binary code --analog samples are referred to as pulse amplitude modulation (PAM) samples. The digital signal consists of block of *n* bits, where each *n*-bit number is the amplitude of a PCM pulse. For PCM of voice communications (BW = 4K), 8000 samples per second are taken and 8 bits are used to represent 256 levels. On the receiver, quantizing the PCM pulse takes place and the original signal is approximated. Quantizing basically converts each *n*-bit number into the amplitude of the PCM pulse This leads to quantizing noise. In PCM systems, signal-to-noise ratio is calculated for quantizing noise.

B.5.7 Delta Modulation (DM)

Analog input is approximated by staircase function that moves up or down by one quantization level (δ) at each sampling interval. The bit stream approximates derivative of analog signal (rather than amplitude), i.e., 1 is generated if function goes up, 0 otherwise. This significantly reduces the data carried, i.e., instead of 8 bits per signal, 1 bit is carried (this can increase data rate 8 times). Another advantage of DM over PCM is the simplicity of its implementation.

Two important parameters are important for delta modulations: a) size of step assigned to each binary digit (δ), and b) sampling rate. The accuracy is improved by increasing the sampling rate. However, this increases the data rate.

B.6 References

Berrou, C., Glavieux, A., and Thitimajshima, P., "Near Shannon limit error-correcting coding and decoding: turbo codes", *Proc. IEEE Int. Conf. Communication*, Geneva, Switzerland, May 1993.

Guizzo, E. "Closing in on the Perfect Code." *IEEE Spectrum*, March 2004, pp. 36-42.

Heegard, C. and Wicker, S, *Turbo Coding*. Kluwer Academic, 1999.

Liberti, J. and Rappaport, T. *Smart Antennas for Wireless Communications*. Prentice Hall, 1991.

Mark, J. and Zhuang, W. *Wireless Communications and Networking*. Prentice Hall, 2003.

Liu, Y., ”MAP Algorithm for Decoding Linear Block Codes Based on Sectionalized Trellis Diagrams”, *IEEE Transactions on Communication*, April 2000.

Lin, S. and Costello, D., *Error Control Coding: Fundamentals and Applications*, Prentice Hall, NJ, 1983.

MacWilliams, F. and Sloane, N., *The Theory of Error-Correcting Codes*, North-Holland: New York, NY, 1977.

Peterson, W. and Weldon, E.J., *Error-Correcting Codes*, 2nd edition, MIT Press: Cambridge, Mass., 1972.

Rappaport, T. *Wireless Communications*: Principles and Practice. 2nd ed. Prentice Hall, 2001.

Shannon, C. E., ”A Mathematical Theory of Communications”, Bell Systems Technical Journal, January 1948.

Singh, D., ”Turbo code study and simulation for mobile communication systems”, MSc Thesis, University College Cork, September 2001.

Stallings, W. *Wireless Communications and Networks*. Prentice Hall, 2002.

Stojmenovi, I., et al. H*andbook of Wireless Networks and Mobile Computing*. John Wiley, 2002.

Walker, J., ed. *Advances in Mobile Information Systems*. Artech House Mobile Communications Library, December 1998.