CHAPTER CONTENTS

7.0 Channel Coding

Objectives

This section will:

- Review some basic information theory
- Examine differential coding and delta modulation
- Introduce various coding techniques to reduce baud rate or error rate
- Examine entropy coding
- Examine block coding
- Examine convolution coding

Channel codes restructure binary information to minimize transmission errors or reduce the average bit rate. Some of the principle line codes are:

- Differential coding
- Entropy coding
  - Huffman
  - Shannon-Fano
- Block coding
  - Convolutional codes
  - CRC codes
- Hamming coding

7.1 DIFFERENTIAL CODING

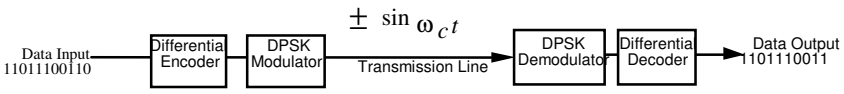
Differential coding can be performed for two basic reasons:

- To eliminate the need for an absolute reference
- To reduce the transmission bit rate

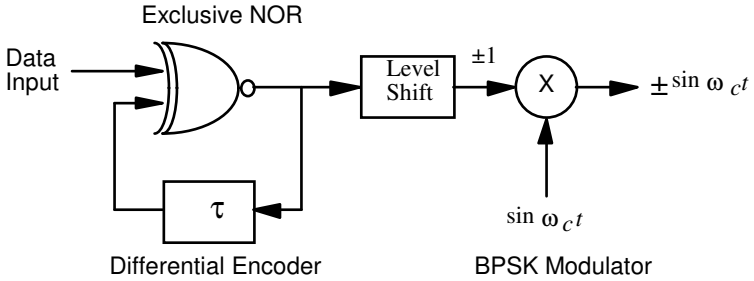
7.1.1 DPSK

Although it is relatively easy to recreate a carrier, it is not easy to set its phase since one cycle looks much like any other. Therefore, it is not possible to make a PSK detector that can measure the actual signal phase with respect to a reference phase.

By encoding information as differential phase shifts instead of absolute phase, the requirements for generating an accurate carrier reference phase, can be relaxed. However, the modem must have a memory element that remembers the phase of the last baud unit.



Differential encoding is performed prior to modulation:



The XNOR gate produces a logical 1 if the two inputs are the same and a logical 0 if the two inputs are different. In order for this circuit to operate properly, the output of the delay register must be preset to some value, say 1, before the data input is applied. [If the register is set to 0, the results will be identical except for the first bit.]

XNOR Truth Table		
Data Input	Delay Input	Output
0	0	1
0	1	0
1	0	0
1	1	1

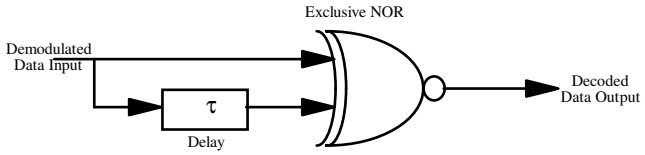
Given the following input to a differential encoder: 11011100110, the encoded pattern can be determined as follows:

Clock Pulse	1	2	3	4	5	6	7	8	9	10	11	12
Data Input	**1**	**1**	**0**	**1**	**1**	**1**	**0**	**0**	**1**	**1**	**0**	
Delayed Input	1	1	1	0	0	0	0	1	0	0	0	1
Tx XNOR Output	1	1	0	0	0	0	1	0	0	0	1	

The ones and zeros out of the encoder are then shifted to ±1 levels and used to change the phase of a sinewave carrier [± Sin(ωt)].

A squaring loop, a standard PLL or costas loop can do demodulation. In any event, a decoder follows the demodulator.

The differential decoder circuit is:



The delay input [XNOR output] must be preset to some initial value, say a logical 1 for the circuit to function properly. [If the register is set to 0, the results will be identical except for the first bit.]

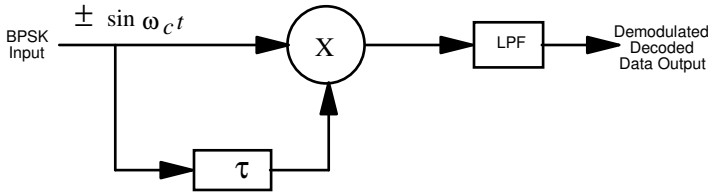
The received phase modulated signal is processed though a PLL or costas loop restoring the information to its original binary format. It can be decoded through the XNOR circuit in the receiver as follows:

Clock Pulse	1	2	3	4	5	6	7	8	9	10	11	12
Current Input	1	1	0	0	0	0	1	0	0	0	1	
Previous Input	1	1	1	0	0	0	0	1	0	0	0	1
Rx XNOR Output	**1**	**1**	**0**	**1**	**1**	**1**	**0**	**0**	**1**	**1**	**0**	

Note that the data coming out of the XNOR in the receiver corresponds exactly to the original transmitted data.

The remarkable thing is that even if the phase locked or Costas loop inverts the data by locking in the wrong state, the XNOR output remains the same except for the first bit. Consequently, it is necessary for modems to transmit a training sequence to ensure proper reception. This sequence can be completely random.

Another way to make the receiver is to incorporate the delay element into the demodulator:



There are only 4 possible combinations of current and previous conditions that can occur:

Previous Input	Current Input	Output
$\sin(\omega_i t)$	$\sin(\omega_i t)$	$\frac{1}{2}$
$-\sin(\omega_i t)$	$-\sin(\omega_i t)$	$\frac{1}{2}$
$\sin(\omega_i t)$	$-\sin(\omega_i t)$	$-\frac{1}{2}$
$-\sin(\omega_i t)$	$\sin(\omega_i t)$	$-\frac{1}{2}$

Note that the output is positive if the previous and current signals are the same, but is negative if they are different. This is differential decoding.

The low pass filter is used to remove the high frequency components created by the multiplication process. In a practical circuit, a codec may be used to digitize the incoming signal before processing. The delay unit is then simply a shift register, and the LPF a digital filter.

7.2 Information Theory

http://www-dept.cs.ucl.ac.uk/staff/S.Bhatti/D51-notes/node27.html

http://www.cs.toronto.edu/~mackay/itprnn/1997/l1/node1.html

Information theory is concerned with the optimization of information transmission.

Information is carried in symbol *m,* then two consecutive symbols, *m m*, double the amount of information, but squares the number of possible values. This can be seen if the symbol *m* can have one of several values say the numbers from 0 to 9.

$$m \times m = m^2$$

If $0 \leq m \leq 9$ then *m* x *m* represents 2 pieces of information but a possible 100 states or different values. For this reason, logarithmic units are used in describing information.

Multi-level transmission schemes are often referred to as *m*-ary systems. The most widely spread of these is the binary system that has 2 levels. It is possible to increase the information content associated with any symbol by increasing the number of possible states or levels.

General Relationship

$I = \log_b m^n = n \log_b m$ $\quad I$ = amount of information

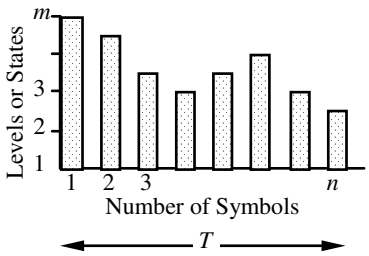
b = system base number [2 for binary, etc.]

m = number of states

n = number of symbols

Remember: $\log_x y = \dfrac{\ln y}{\ln x}$

The difference between states and symbols can be seen in the following illustration:



Example

Encoding 256 voltage levels into binary symbols requires:

$$\log_2 256 = 8 \text{ binary symbols or bits}$$

Encoding into decimal, requires only:

$$\log_{10} 256 = 2.4082 \text{ decimal symbols}$$

Since it is not possible to transmit a fraction of a symbol, 3 decimal symbols are needed in the above example. This supports a total of 10^3 = 1000 states, the vast majority of which are not used.

7.2.1 CHANNEL CAPACITY

From the forgoing, we can deduce that the information capacity [C] per unit of time [T] in a system is given by:

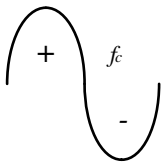
$$C = \frac{n}{T}\log_b m$$

The value n/T represents the time duration of one symbol.

For a binary system, the channel capacity is: $C = \frac{2}{T}\log_2 2 = \frac{2}{T}$. Since the smallest time period, is determined by the channel bandwidth [f_C], we observe: $T_{\min} = \frac{1}{f_c}$. Therefore the maximum capacity is $C = 2f_c$ or twice the channel bandwidth.

In an ideal low pass information channel with a bandwidth of B Hz has a maximum theoretical capacity of $2B$ bits per second. This is known as the Nyquest rate for a binary channel.

This may seem intuitively obvious. If the highest frequency, which can be placed in a given channel, is f_C, then it would seem that there would be two states in each cycle. The states could be labeled + & - in analog fashion or 1 & 0 in the more familiar digital fashion.



The information rate in a given channel, is the product of the signal entropy [H] and signaling or message rate [r]:

$$R = rH \text{ bps}$$

If the information rate R is less than the channel capacity C, then it is theoretically possible to transmit error free messages.

7.2.2 SHANNON-HARTLEY THEOREM

The maximum theoretical channel capacity in the presence of Gaussian noise can be determined if the bandwidth and signal power is known.

$$C = B\log_2\left(\frac{S}{N} + 1\right)$$

$C =$ channel capacity in bps

$B =$ bandwidth in Hz

$\frac{S}{N}$ = signal to noise power ratio (not dB)

Example

A telephone channel has a signal to noise ratio of 30 dB and a bandwidth of 3.1 KHz. The maximum theoretical channel capacity in bps is found as follows:

$$\frac{P_S}{P_N} = 30 \text{ dB} \qquad \text{or as a ratio :} \qquad \frac{P_S}{P_N} = \text{anlog}\left(\frac{30}{10}\right) = 1000$$

$$\therefore I = 3.1 \times 10^3 \log_2(1000+1) = 30.89 \text{ Kbits/Sec}$$

This represents a maximum theoretical limit. In practice, 9600 bps is achieved since the error rate becomes excessive if the data rate is pushed much higher.

It may initially appear that if the channel bandwidth is increases to infinity, the channel capacity will also reach infinity. This however is not true. As bandwidth increases, so does the Gaussian noise admitted into the system. If the white noise has a power spectral density of $\eta/2$, the Hartley-Shannon law reduces to:

$$C = B \log_2\left(\frac{S}{\eta B} + 1\right)$$

This expression can be rewritten as:

$$C = \frac{S}{\eta}\left(\frac{\eta B}{S}\right) \log_2\left(\frac{S}{\eta B} + 1\right)$$

This allows us to take advantage of the following identity:

$$\lim_{x \to \infty} (x+1)^{1/x} = e$$

Taking the limit of capacity as bandwidth goes to infinity, results in the following simplification:

$$\lim_{B \to \infty} C = \lim_{B \to \infty} \frac{S}{\eta}\left(\frac{\eta B}{S}\right) \log_2\left(\frac{S}{\eta B} + 1\right)$$

$$= \frac{S}{\eta} \log_2 e$$

$$= 1.44 \frac{S}{\eta}$$

This expression gives the maximum information rate possible for a system with a given signal power, and unlimited bandwidth. This situation can happen over free space links. The noise power spectral density can be specified in terms of noise temperature: $\eta = kT$, where k is Boltzmann's constant and T is temperature in degrees Kelvin.

This expression suggests that any channel capacity can be achieved if the signal is powerful enough. Such things as cost and available technology determine the practical limits of this.

7.3 ENTROPY CODING

http://www.math.washington.edu/~hillman/Entropy/infcode.html
http://www.ee.uwa.edu.au/~roberto/units/itc314/

7.3.1 ENTROPY

Entropy is the average amount of information present in a source. This is defined in terms of observational probabilities. If *m* possible states have an equal probability of occurring, then the probability of any one state occurring is:

$$p = \frac{1}{m}$$

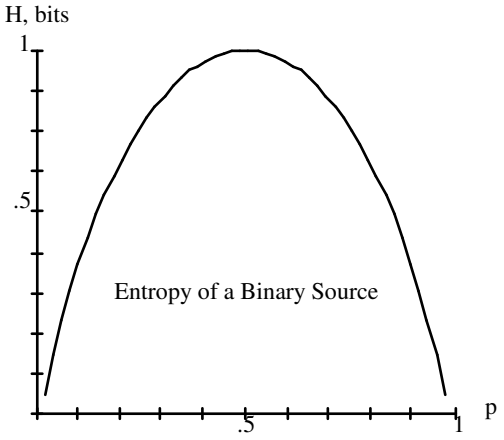
The entropy or average amount of information is defined as:

$$H \equiv I_{\text{average}} = \sum_{i=1}^{m} p_i \log_b\left(\frac{1}{p_i}\right)$$

As an example, we can determine the entropy of a binary source, where the probability of one state occurring is *p*, and of necessity, the probability of the other state occurring is 1-*p*. Applying the above equation, we obtain:

$$H = \sum_{i=1}^{2} p_i \log_2\left(\frac{1}{p_i}\right) = p\log_2\left(\frac{1}{p}\right) + (1-p)\log_2\left(\frac{1}{1-p}\right)$$

a plot of this equation is:



Note that the maximum amount of information or entropy in a binary system is 1 bit per symbol and occurs when the two possible states each have a 50% probability of occurring.

Entropy encoding reduces the bit rate by minimizing or eliminating wasted states. The result is a code that more closely approximates the actual information content.

The entropy of any signal depends upon its probability of occurrence. As an example, let's take the case where one of 8 different messages may be sent. These

messages are quite arbitrary and may represent temperature information in a data logging network, achievement scores in an exam, or any other collection of datum which is not purely random:

An arbitrary non-random collection of messages

Message or Event	Binary Representation	Probability of Occurrence
M1	001	0.09
M2	010	0.40
M3	011	0.15
M4	100	0.20
M5	101	0.03
M6	110	0.05
M7	111	0.07
M8	000	0.01

This group of messages is easily encoded into 3 bits however, they are not encoded very efficiently. Although it is not obvious, bits are being wasted.

The entropy in this example is:

$$H \equiv I_{\text{average}} = \sum_{i=1}^{N} P_i \log_s\left(\frac{1}{P_i}\right)$$

$$= .09\log_2\left(\frac{1}{.09}\right) + .4\log_2\left(\frac{1}{.4}\right) + .15\log_2\left(\frac{1}{.15}\right) + .2\log_2\left(\frac{1}{.2}\right) +$$

$$+.03\log_2\left(\frac{1}{.03}\right) + .05\log_2\left(\frac{1}{.05}\right) + .07\log_2\left(\frac{1}{.07}\right) + .01\log_2\left(\frac{1}{.01}\right)$$

$$= 2.419212$$

This means that the average 3-bit message contains only 2.419 bits of information. Needless to say, this is not optimally efficient.

Two methods used to improve transmission efficiency are Huffman and Shannon-Fano coding. Both of these methods use variable length codes. The events that are more likely to occur are given short codes while the least likely events are given the longest codes.

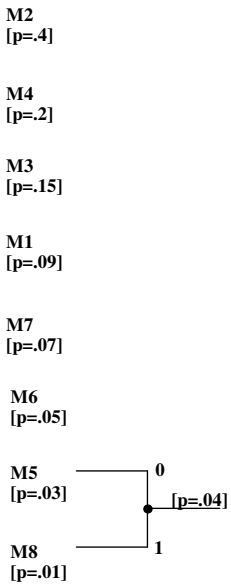
7.3.2 HUFFMAN CODE

Like Morse code, this technique utilizes a variable length format where the most common symbols are given the shortest code pattern.

The code is determined by creating an encoding tree, using the following steps:

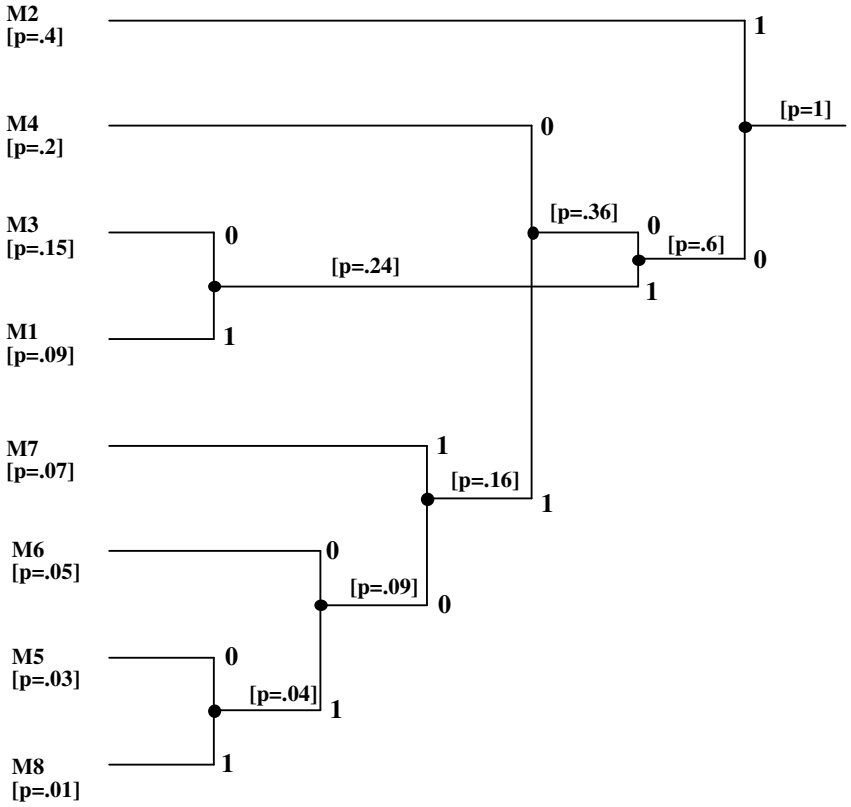
- Arrange the messages in decreasing probability of occurrence
- Combine the two branches having the lowest probability into a single branch
- Assign the value 0 to the higher probability branch and 1 to the lower probability branch. In the event that 2 branches have the same probability, assign 0 to the top branch

As an example, we shall encode the non-random messages in the previous table.



- Repeat this process until the entire tree is formed.

The code value for each message is found by retracing the path back along the tree.



Huffman Coding Animation 1

Huffman Coding Animation 2

Resulting Huffman Code Table

Message or Event	Probability of Occurrence	Binary Representation	Huffman Code
M2	0.4	010	1
M4	0.2	100	000
M3	0.15	011	010
M1	0.09	001	011
M7	0.07	111	0011
M6	0.05	110	00100
M5	0.03	101	001010
M8	0.01	000	001011

Although this code would appear to be less efficient than the standard binary format, it actually will result in 20% fewer bits being transmitted. This can be observed by calculating the message entropy.

The average number of bits transmitted is the sum of the individual message probabilities, times the number of bits per message:

$$.4\times 1+.2\times 3+.15\times 3+.09\times 3+.07\times 4+.05\times 5+.03\times 6+.01\times 6=2.49\ bits$$

This is slightly higher than the entropy but is close to optimum.

The Huffman scheme requires that a backward operation be involved. This means that all of the probabilities must be known before the codes can be assigned.

To ensure that the receiver correctly decodes the bit stream, the coding table must be forwarded. It is interesting to note however, that the individual code words can generally be identified, even without framing information.

For example, assume the following message sequence: M2M4M2M4M1M2M8M7. The digital pattern would be: 1000100001110010110011. Amazingly enough, there is only one way this bit stream can be decoded with the forgoing table. Starting from the first bit, 1 is a valid code word, but 10 is not. Therefore the first message must be M1 or 010. Carrying this process forward, 0 is not a valid code, neither is 00, but 000 is. Therefore, the next message must be M4 or 100. This process can be repeated until the entire string is decoded.

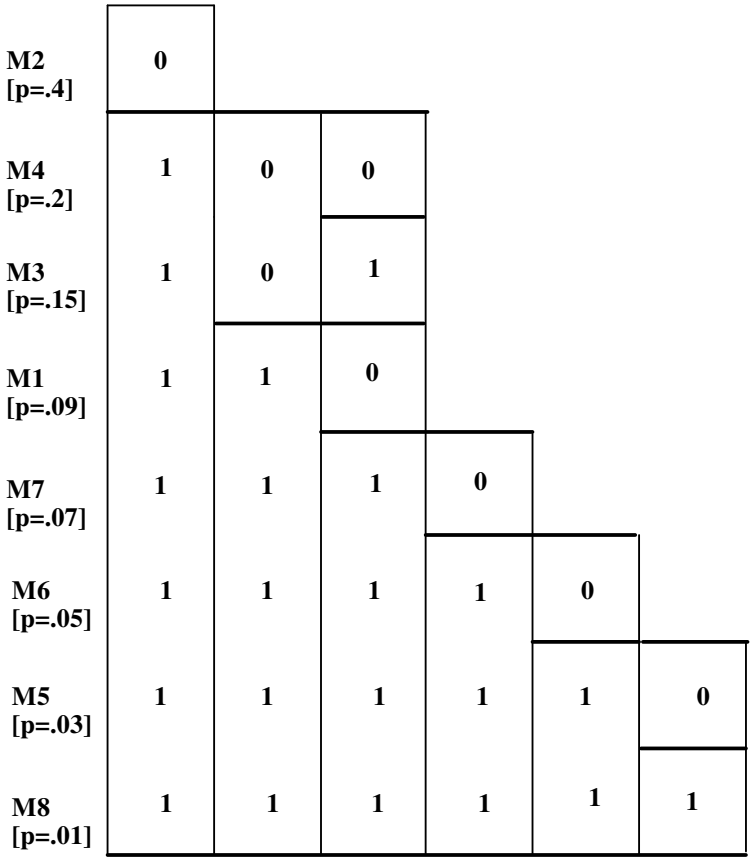
7.3.3 SHANNON–FANO CODE

This process can be done in a forward mode, which requires less memory and processing time. However, it does not always result in the same degree of compression as the Huffman process.

One way the code can be determined is by the following procedure:

- Arrange the messages in decreasing probability of occurrence
- Divide the messages into 2 equally probable groups
- If there are two possible divisions, select the one with the fewest states above the division

- Assign one group a value of 0 and the other a 1
- Continue the division process until all there is nothing left to divide



Shannon-Fano Coding Animation

Resulting Shannon-Fano Code Table

Message or Event	Probability of Occurrence	Binary Representation	Shannon-Fano Code
M2	0.4	010	0
M4	0.2	100	100
M3	0.15	011	101
M1	0.09	001	110
M7	0.07	111	1110
M6	0.05	110	11110
M5	0.03	101	111110
M8	0.01	000	111111

On the average, this code will transmit 2.5 bits per message:

$$.4 \times 1 + .2 \times 3 + .15 \times 3 + .09 \times 3 + .07 \times 4 + .05 \times 5 + .03 \times 6 + .01 \times 6 = 2.49 \; bits$$

This coding algorithm is often less efficient than that of Huffman, but can be implemented with simpler hardware.

7.3.4 RUN-LENGTH CODING

RLE takes advantage of the fact that many times adjacent data blocks or pixels will have the same value. A special escape code is inserted into the data stream to indicate that the following data or pixel value is to be repeated a certain number of times. The escape code must be a value that cannot possibly occur naturally in the data stream.

As an example, lets assume an 8-bit code and use an escape code of 255. In decimal notation, a data string may resemble:

12, 14, 22, 33, 85, 85, 85, 85, 85, 85, 85, 12, 16, 22 …

This data string can be compressed by replacing the value 85 by the escape code, the value, and the number of times the value occurred:

12, 14, 22, 33, 255, 85, 7, 12, 16, 22 …

This technique does not work for audio signals since they do not remain constant.

7.3.5 ARITHMETIC CODING

In this technique is based on the idea that the probability of a bit occurring is uniformly distributed throughout the byte. The overall probability of a 1 or 0 occurring is determined. This value is then used in each bit position and used to determine the probability of any particular sequence. The final code word is determined by finding the nearest binary fractional equivalent.

This process can be enhanced by considering intersymbol probabilities. An example of this is dynamic Markov coding.

7.3.6 LEMPEL-ZEV CODING

http://www.data-compression.com/lempelziv.html

Lempel-Zev Animation

Encoding Algorithm

1. Initialize the dictionary to contain all blocks of length one (D={a,b}).
2. Search for the longest block W which has appeared in the dictionary.
3. Encode W by its index in the dictionary.
4. Add W followed by the first symbol of the next block to the dictionary.
5. Go to Step 2.

7.4 BLOCK CODING

Block coding translates one group or block of symbols to another. If the symbol set is redefined, symbolic compression may occur. For example, octal notation organizes binary bits into blocks of 3 binary symbols and replaces them with a single digit 0 to

7. Hexadecimal notation organizes binary bits into blocks of 4 binary symbols and replaces them with a single symbol 0 to E. This type of coding reduces the number of symbols needed for transmission.

To reduce the error rates caused by channel noise, some form of redundancy or coding is used. The introduction of additional states that are not part of the original information signal creates a certain inefficiency. However, it should be appreciated that it is better to make inefficient use of the data channel and receive all of the information intact, than to maximize the channel capacity and receive no usable information at all.

Block coding can be used to reduce the error rate if more symbols are added to the original group. For example, a 3-bit binary block may be translated into a 4-bit binary block where the additional bit is defined as parity.

7.4.1 2B1Q CODING

The 2B1Q format translates 2 binary digits into a single quaternary or 4 level pulse. This code has no surplus states.

Input	Output
00	-3
01	-1
10	+1
11	+3

The 4B3T format encodes 4 binary digits into 3 ternary pulses. Since 4 binary digits can take 16 [2^4] possible states, and 3 ternary pulses represent 27 [3^3] state, there is a surplus of 11 codes. These can be used to carry additional information for error checking.

Codes that deal only with binary signals have the designation *n*B*m*B, where *n* input bits are encoded into *m* output bits. The most common of these is the 3B4B code.

7.4.2 3B4B CODING

Input	Output
000	- - + - or + + - +
001	- - + +
010	- + - +
011	- + + -
100	+ - - +
101	+ - + -
110	+ + - -
111	- + - - or + - + +

The 24B1P or 24B25B code format adds a P or parity bit to a 24-bit block.

7.4.3 CRC CODING

Cyclic redundancy codes add bits to the end of a message transaction for the purpose of error detection. The message information is divided by a generator polynomial. The remainder is then appended to the message and transmitted.

The receiver divides the message and remainder by the same generator polynomial. If no errors have occurred, the remainder is zero. The process is as follows:

- The original message polynomial is shifted to the left by the order of the generator polynomial
- The shifted message is divided by the generator polynomial, producing a remainder
- The remainder is appended to the message and transmitted.
- The receiver divides the transmitted message by the generator polynomial
- If a remainder occurs, the message is deemed in error and retransmission is requested
- If no remainder exists, the message is shifted to the right by the order of the generator polynomial

In order for this system to work, modulo 2 addition is used. This exclusive OR process makes addition and subtraction the same operation.

Example

Let the message polynomial be given by: $M(x) = 110101 = x^5 + x^4 + x^2 + 1$

Let the generator polynomial be: $G(x) = 101 = x^2 + 1$

The generator polynomial is of the second order, therefore the shifted message polynomial is:

$$M_S(x) = M(x)x^2 = \left(x^5 + x^4 + x^2 + 1\right)x^2$$
$$= x^7 + x^6 + x^4 + x^2 = 11010100$$

This is divided by the generator polynomial to produce a remainder:

$$\frac{M_S(x)}{G(x)} = x^5 + x^4 + x^3 + x^1 + 1$$

$$x^2+1 \;\overline{)\; x^7 + x^6 + x^4 + x^2}$$

$$x^7 \quad x^5$$

$$x^6 \; x^5 \; x^4 \quad x^2$$

$$x^6 \quad x^4$$

$$x^5 \quad x^2$$

$$x^5 \quad x^3$$

$$x^3 \; x^2$$

$$x^3 \quad x^1$$

$$x^2 \; x^1$$

$$x^2 \quad 1$$

Remainder $R(x)$ $\quad x^1 \; 1$

The transmitted message is:

$$T(x) = M_S(x) + R(x) = \left(x^7 + x^6 + x^4 + x^2\right) + \left(x^1 + 1\right)$$

$$= x^7 + x^6 + x^4 + x^2 + x^1 + 1$$

CRC Coding Animation

The receiver divides this message by the generator polynomial to determine if an error has occurred. There is no remainder; the message is deemed error free.

$$\frac{T(x)}{G(x)} = x^5 + x^4 + x^3 + x^1 + 1$$

$$x^2+1 \;\overline{)\; x^7 + x^6 + x^4 + x^2 + x^1 + 1}$$

$$x^7 \quad x^5$$

$$x^6 \; x^5 \; x^4 \quad x^2 \; x^1 \; 1$$

$$x^6 \quad x^4$$

$$x^5 \quad x^2 \; x^1 \; 1$$

$$x^5 \quad x^3$$

$$x^3 \; x^2 \; x^1 \; 1$$

$$x^3 \quad x^1$$

$$x^2 \quad 1$$

$$x^2 \quad 1$$

No Remainder

To recover the original message, the transmitted message is shifted to the right by the order of the generator polynomial:

$$M(x) = T(x)x^{-2} = \left(x^7 + x^6 + x^4 + x^2\right)x^{-2}$$

$$= x^5 + x^4 + x^2 + 1 = 110101$$

7.4.4 HAMMING CODES

http://www.rad.com/networks/1994/err_con/hamming.htm

Hamming codes are one of several FEC or forward error correcting codes. These insert extra bits into the data stream and allow the receiver to detect and correct single bit errors.

FEC codes are popular in systems where it is not possible or practical to request a retransmission.

The number of Hamming (or parity) bits that are inserted into the data path is determined by the following equation:

$$2^m \geq n + m + 1$$

$m =$ number of Hamming bits

$n =$ number of bits before coding

Hamming codes are described by the ordered pair (n+m, n). A code is called a perfect code if the equation is equal.

The number of Hamming bits is the smallest value of m that makes the above equality true. The number of Hamming bits needed for small code segments is proportionally quite high. Therefore, the most practical use for this technique is in large data blocks.

Data Length before coding	Hamming bits required	Data Length after coding
1	2	3
2 - 4	3	5 - 7
5 - 11	4	9 - 15
12 - 26	5	17 - 31
27 - 57	6	33 - 63

As long as the decoder knows where the Hamming bits have been inserted, they can be placed anywhere in the bit stream. The value of the Hamming code is dependent on the position of the marks within the sequence.

Example

Let's encode the following 12-bit sequence: 010101110101. This will require adding 5 Hamming bits. Starting from the least significant bit, we'll arbitrarily place the bits in positions 1, 2, 4, 8, and 16 where position 1 represents the least significant bit.

Hamming bits in place: 0H1010111H010H1HH

Mark Position [Decimal]	Mark Position [Binary]
3	00011
6	00110
9	01001
10	01010
11	01011
13	01101
15	01111
Odd Parity	01111

The 5-bit Hamming code is simply the parity of each column in the previous table. For odd parity, the transmitted data stream is: 0**0**10101111**1**010**1**1**11**.

If a single bit error occurs, the location of the error can be determined by checking the parity for all the locations where a mark occurred.

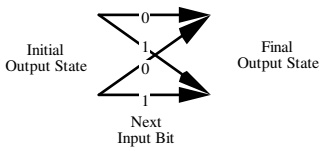
For example, if an error occurred in the 10th bit position, the code would be: 0**0**10101**0**1**1**010**1**1**11**. The mark positions, including those inserted by the Hamming code, are tabulated as before:

Mark Position [Decimal]	Mark Position [Binary]
1	00001
2	00010
3	00011
4	00100
6	00110
8	01000
9	01001
11	01011
13	01101
15	01111
Odd Parity	01010

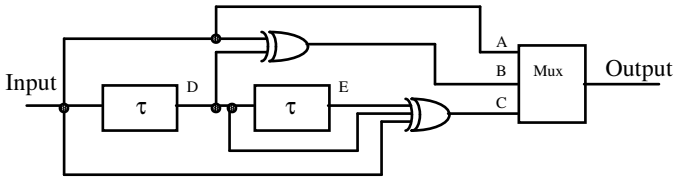
Notice that the resulting odd parity value is 01010. This number is 10 in binary notation and is the location of the error! Invert bit 10 and the data stream is restored to its original values. It should be remembered that this form of error correction only works for single bit errors.

7.5 CONVOLUTION CODING

The term convolution is used because this form of coding is essentially a convolution using modulo arithmetic. It is said to have memory, because the output code value depends upon the current and previous data sample.



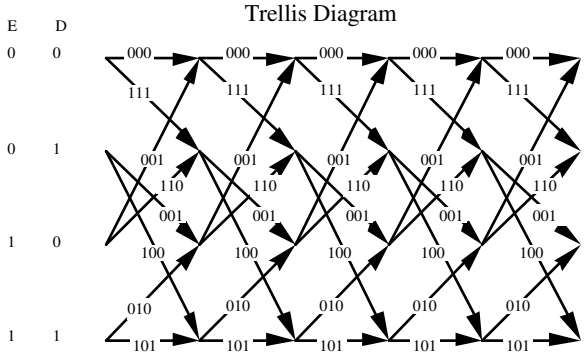
This form of encoding has a very low efficiency because it produces several binary coded symbols for every input bit. This inefficiency provides error detection and correcting capabilities.



3 Bit Convolutional Coder

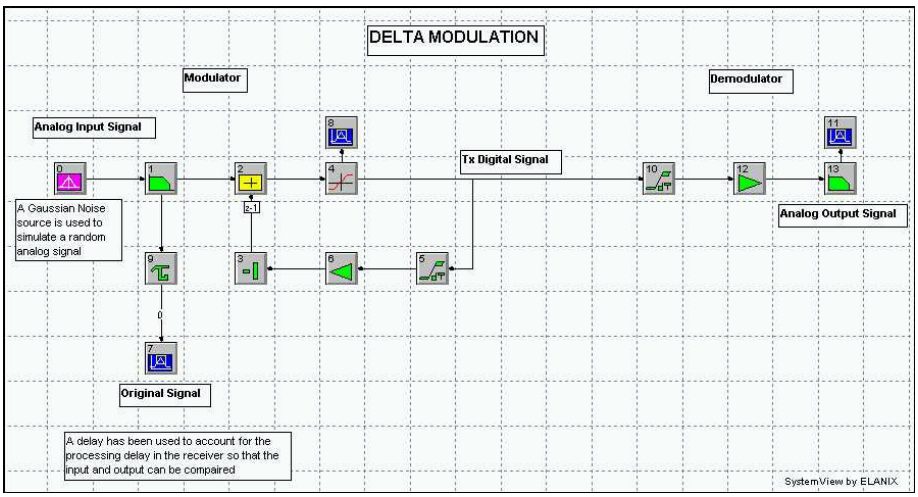
Current State		New Input	Output	Next State	
E	D	A	ABC	E	D
0	0	0	000	0	0
0	0	1	111	0	1
0	1	0	011	1	0
0	1	1	100	1	1
1	0	0	001	0	0
1	0	1	110	0	1
1	1	0	010	1	0
1	1	1	101	1	1

Trellis Diagram



SYSTEMVIEW MODELS

Delta Modulator Model



Review Questions

QUICK QUIZ

1. It requires [3.5, 4.7, 5.6] bits to encode the English alphabet in binary.
2. The primary purpose of convolution coding is to reduce the [error, baud] rate.
3. The primary purpose of the Huffman coding scheme is to reduce the [error, baud] rate.
4. The [Huffman, Shannon-Fano] coding scheme is implemented in a forward mode.
5. The Shannon-Hartley Theorem states that channel capacity is a function of transmitter power. [True, False]

ANALYTICAL PROBLEMS

1. Given the following characteristics of the Voyager spacecraft as it approached Uranus:

   - Transmit power 18.2 watts
   - Transmit antenna gain 48 dB
   - Space loss -301 dB
   - Receive antenna gain 72 dB
   - Temperature 10^{o} K

   a) Find the maximum theoretical data rate from the probe.

   b) Explain why the actual data rate was only 29.9 Kbps.

2. Encode the following message states using the Huffman and Shannon-Fano coding techniques and calculate the resulting entropy.

Message	Binary Value	Probability
1	001	.07
2	010	.13
3	011	.17
4	100	.20
5	101	.16
6	110	.14
7	111	.10
8	000	.03

3. Calculate the entropy of the following ternary systems:

   a) All three states have an equal probability of occurrence

   b) Calculate the entropy if state 1 occurs twice as often as state 2 and three times as often as state 3

4. Given the following:

   Message polynomial: $M(x) = 110101$

   Generator Polynomial: $G(x) = 101$

   Develop the transmitted CRC code.

COMPOSITION QUESTIONS

1. What are the advantages of DPSK over PSK?
2. How can channel capacity be increased according to the Shannon-Hartley Theorem?
3. Why doesn't an infinite bandwidth transmission system have infinite capacity?
4. Explain the process of CRC coding.

For Further Research

Digital Communications – Design for the Real World, Andy Bateman, Addison-Wesley

Digital Signal Transmissions, Bissell and Chapman, Cambridge University Press

http://www.tns.lcs.mit.edu/~turletti/gsm-overview/node9.html

http://www.aydinvector.com/pcm_1.html

http://www.rad.com/networks/1994/err_con/intro.htm