APPLICATION OF ERROR CONTROL TECHNIQUES TO DIGITAL TRANSMISSION VIA CATV NETWORKS

JOHN T. GRIFFIN

JERROLD COMMUNICATIONS
APPLIED MEDIA LAB

ABSTRACT

Digital carriage of data via CATV networks is becoming more prevalent with each passing year. Digital radio services are presently being offered by numerous cable operators. Digital compression of video is just over the horizon, and other data services will likely follow. Although these digital delivery systems do not exhibit the impairments of their analog counterparts, bit errors may cause catastrophic distortions. Forward error correction is one aspect of a solution to this problem.

This paper introduces the CATV system engineer to the concepts of forward error correction, and discusses its benefits, complexity, and limitations. It also touches on the interdependence of forward error correction with channel equalization and efficient modulation. Several important concepts, such as coding gain, are discussed in detail.

INTRODUCTION

A CATV system meets the definition of a communication system because it connects multiple information sources to users of this information. A general communication system is illustrated in Figure 1. For purposes of this paper, the source is any source of television programming; the **source encoder** might be some form of video and audio compression. The **forward error encoder** and decoder in Figure 1 are the subject of this paper.

Error control techniques can be very effective against random noise impairments, but are not a panacea for microreflections on digital transmission in a CATV network. Error control can be teamed with channel equalization, as shown in Figure 1, to develop a very robust and cost effective communication channel. Both the rate at which the errors occur and their distribution must be known before the optimum error correction scheme may be designed. This may be accomplished by a combination of simulation, laboratory tests, and field tests.

Error control applied to future CATV networks using video and audio compression (the source encoder and source decoder in Figure 1) is essential because compression eliminates the redundancy from the original analog signals. Errors occurring during transmission may cause severe impairments to the reconstituted analog signals. In general, errors will propagate through the decompression process.

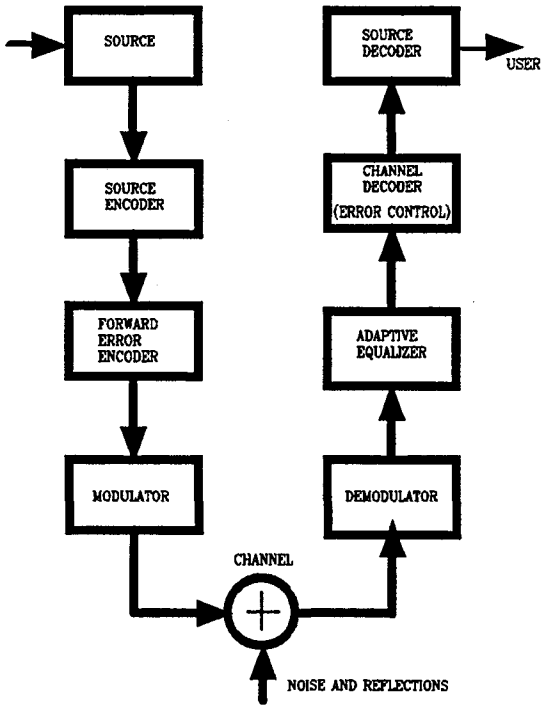
HISTORY

The history of error control began in 1948 with Claude Shannon's famous paper on channel capacity[1]. His channel capacity theorem says the following:

$$C = W \log_2(1 + S/N) \text{ bits/sec} \quad (1)$$

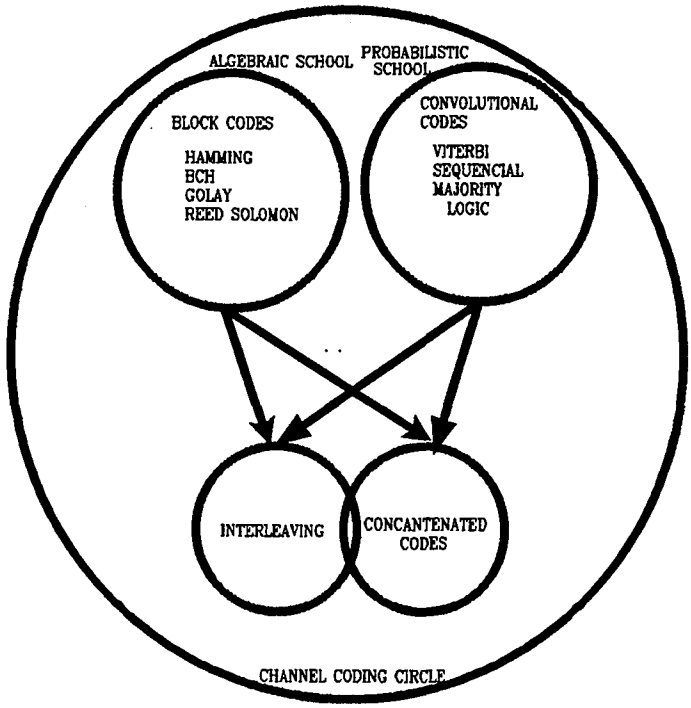
where:

C = capacity in bits/sec
W = bandwidth in Hz
S = signal power
N = noise power



DIGITAL COMMUNICATIONS SYSTEM

FIGURE 1



CODING FOR ERROR CONTROL
FIGURE 2

What Shannon said was that the noise limits the rate at which we can send information, but not the accuracy. Today designers are moving ever closer to this limit with a combination of error control and efficient modulation. Shannon's work also tells us it is more cost effective to employ error coding than to try to build an error-free channel.

During the 1950's and 1960's the search for good codes continued. It was during this period that two mathematical bases developed to solve the error coding problem. This concept is illustrated in Figure 2. The two bases are the **algebraic** and **probabilistic** approaches. The algebraic codes are most commonly known as "block codes". The first of these were introduced in 1950 by Hamming; his are a class of single error correcting codes. Another major milestone occurred in 1960 when Bose, Ray-Chaudhuri, and Hocquenghem found a class of multiple-error-correcting codes now known as BCH codes . Reed and Solomon also developed their codes in 1960; these codes are related to the BCH, but for non-binary channels.[2,3]

The second mathematical approach to coding, the probabilistic approach, led to the development of "convolutional" or "tree" codes. In the late 1950's, studies led to the notion of sequential decoding and to the introduction of non-block codes of indefinite length. However, the most well known algorithm, the Viterbi algorithm, did not appear until 1967. Such techniques have allowed reception of digital data from deep space probes. The steady improvement in the performance of telephone modems has also resulted from advances in error coding and sophisticated modulation techniques.

ERROR CONTROL AND EQUALIZATION

Error control is very effective at mitigating the impairments caused by additive noise. A CATV channel presents other phenomena that limit channel performance. Chief among these are microreflections due to impedance mismatches at the television receiver or unterminated taps. This results in **intersymbol interference** (ISI), which is the tendency of received symbols to flow into one another. [2] This can not be overcome by increasing signal power; there is an ISI noise floor that increases with signal power. ISI may be overcome by **adaptive equalization**, which is outside the scope of this paper. Error control and adaptive equalization may be combined to result in a very robust communication system (refer to Figure 1).

DEFINITIONS

Certain terms appear throughout the literature of coding theory. These are defined here for the convenience of the reader:[2,3]

Symbol A symbol is a group of bits within an error control block. It is also defined as a signal representing a group of "k" bits in some analog manner, such as amplitude or phase. Thus, there are error control symbols and modulation symbols.

Weight The weight of a symbol, codeword, or "vector" is the number of non-zero elements.

Hamming distance The Hamming distance between two vectors having the same number of elements is defined as the number of positions in which the elements differ. This is a key concept in error control and will be discussed in more detail later in this paper.

Minimum distance The minimum distance "d" of a linear block code is the smallest distance between pairs of different codewords in the code.

Codeword A codeword or "code block" is a group of bits or symbols made up of information elements and parity (error control) elements.

Code rate Assume that a block encoder accepts information in successive "k"-bit blocks and for each k bits generates a block of "n" bits, where n > k. The code rate R = k/n is a dimensionless ratio that indicates the portion of an encoded block that carries information.

Overhead This is the percentage of parity bits that must be appended to the information bits in constructing a code.

Hard decision A hard decision demodulator makes an absolute 1/0 choice on each received bit (or symbol). The symbol is quantized to two levels.

Soft decision In making a soft decision, the demodulator makes a bit-quality measurement on each bit or symbol. The symbol is quantized to more than two levels.

Erasure This is the process of flagging a bit or symbol as unreliable. It is the result of a soft decision. This flag is passed along to the error control circuitry.

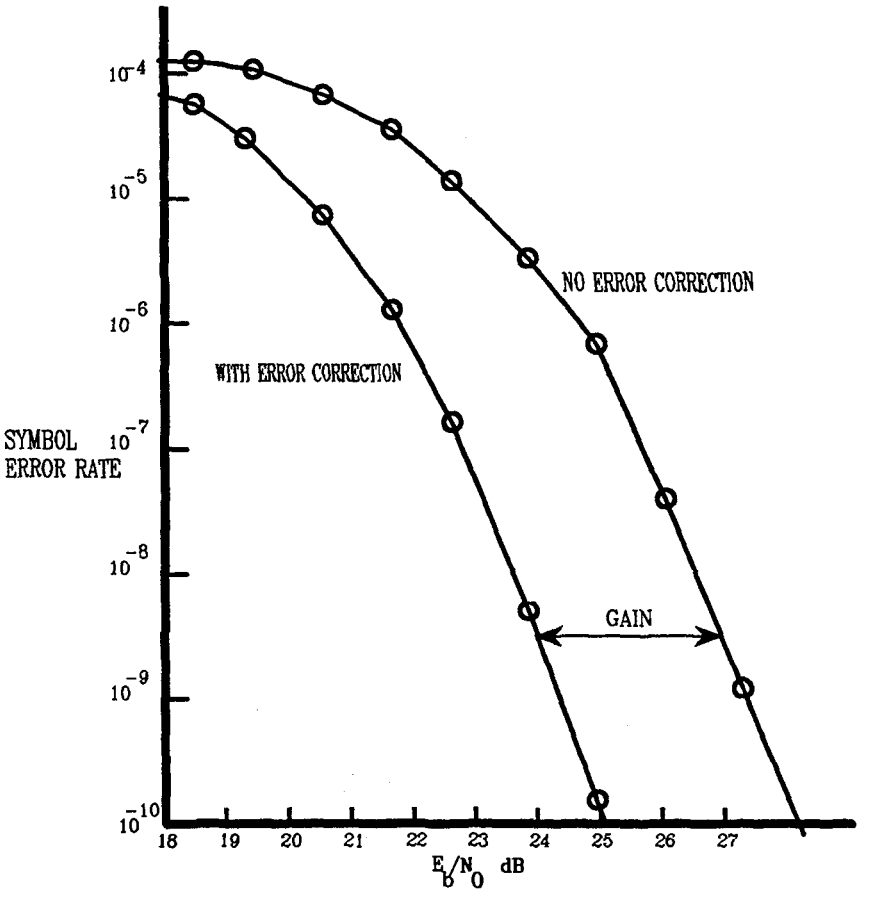
Coding gain This term describes the amount of improvement that is achieved when a particular coding scheme is used. Figure 3 illustrates coding gain on a logarithmic plot of bit error rate vs E_b/N_0 (energy/bit divided by spectral noise density). At low signal to noise ratios, the gain will become negative.

Vector This term is based in linear algebra and is familiar to us from physics. In coding theory vector space is one of the most important algebraic concepts. The vector provides a convenient representation of field elements that may be implemented with simple digital functions. The term is also used in matrix notation, where the vector consists of the coefficients of a polynomial. Refer to section 3.3 of reference 2.

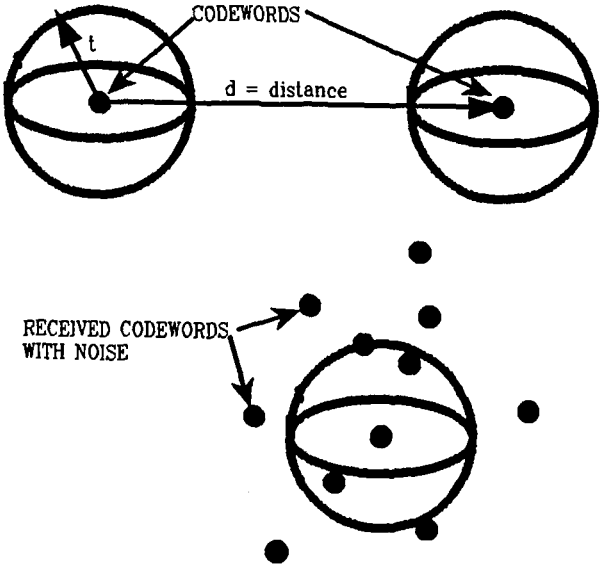
The syndrome The syndrome is defined in the dictionary as "a number of symptoms occurring together and characterizing a specific disease".[6] In coding theory, a syndrome is a sequence of discrepancies which occur when received parity bits are compared with calculated parity bits. The syndrome may take on the form of a "vector" in a matrix. Calculations of syndromes are used in many decoding algorithms to locate errors in received data.

Constraint length In a convolutional code, the constraint length is the number of data frames used in the generation of the encoded data. Each input frame may consist of one or more bits. The process occurs on a continuous basis. In terms of the actual circuit elements, the constraint length is the length of the input data shift register in the encoder.

Galois field A field having a finite number of elements is called a finite or Galois (pronounced gall-wa) field. It is denoted by GF(q), where q is the number of elements in the field. These fields are named after Evariste Galois (1811-1832), a French mathematical prodigy who established group theory mathematics by age 17.[2] Chapter 4 in reference 3 treats this theory in detail.



CODING GAIN
FIGURE 3



DECODING SPHERES
FIGURE 4

THE DISTANCE CONCEPT

This concept is crucial in visualizing the operation of error control circuitry. Figure 4 illustrates "decoding spheres" in a geometric fashion. Recall the definition of minimum distance d_m. We will define t as the number of errors that a particular code can correct. If more than t errors occur in transmission, the decoder may incorrectly decode the data or it may indicate with a flag that it can not decode the message. [3]

In Figure 4, the code is designed so that the minimum distance between codewords is defined as:

$$d \geq 2t + 1 \qquad (2)$$

where t is the number of errors that can be corrected. A codeword received error free will land at the center of a sphere. If t errors occur, the codeword will be on the surface of the sphere, and the decoder will correct the error(s). Received codewords with more than t errors may fall between spheres or within another sphere; those falling in another sphere will be incorrectly decoded. Those falling between spheres may or may not be correctly decoded, but would be erased in a soft decision decoder. One can see that the minimum distance is a critical property of a code.

BLOCK CODES

In a block (algebraic) code the encoder accepts k information bits and appends r parity-check bits to form a block of n bits, such that:

$$n = k + r \qquad (3)$$

where n = block length
r = number of parity bits

The code is referred to as an (n,k) code. The code rate R is k divided by n. Each block is independent of all others; the check bits are completely determined by the information bits within the same codeword. Also, there are 2^k codewords in the code set. The code is designed to make the codewords very different from each other to resist channel errors.

Arithmetic operations in the Galois field GF(2) are simple because no overflow or round-off error is permitted. The operations of addition and multiplication are mod-2. This is illustrated in the following tables:

+	0	1
0	0	1
1	1	0

ADDITION

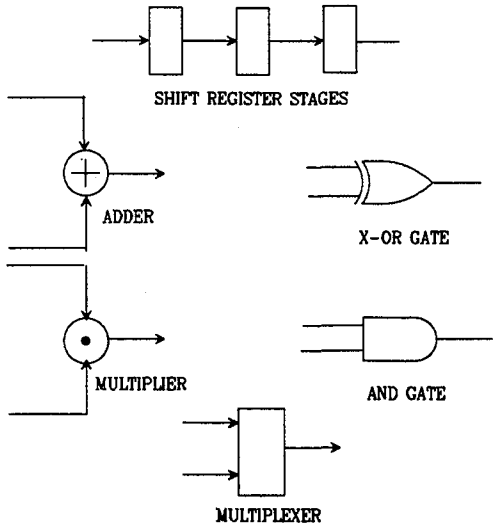
*	0	1
0	0	0
1	0	1

MULTIPLICATION

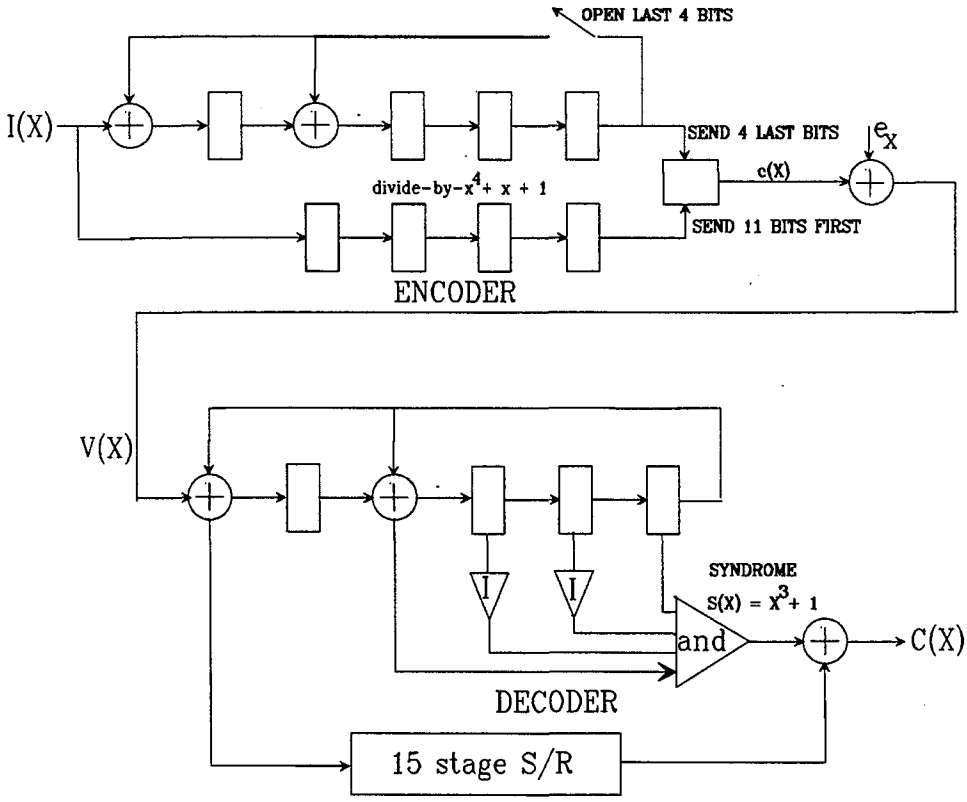
Addition bit-by-bit is accomplished with an "X-OR" gate. Multiplication is done with an "AND" gate.

Polynomial arithmetic in a Galois field (in this case GF(2)) can be used in the description of block codes. Fortunately, digital logic circuits may be constructed to mimic this special polynomial arithmetic. These circuits take the form of digital filters, and are constructed of shift register elements, X-OR gates, AND gates, and multiplexers (Figure 5). The form of the encoders and decoders are similar.

We choose for this paper "binary cyclic block codes" to illustrate the relationship of the GF(2) polynomial arithmetic to the actual circuits. We do so because these codes have proven useful and efficient in practice. Binary cyclic block codes are a subset of



CIRCUIT ELEMENTS
FIGURE 5



CYCLIC BLOCK ENCODER/DECODER

FIGURE 6

linear block codes, and fall in the "algebraic school" circle of Figure 2. A binary code must meet two criteria to be cyclic:

a. The code is linear; bit-by-bit addition of two codewords in GF(2) is again a codeword.

b. Any cyclic (end around) shift of a codeword is also a codeword. [2]

Chapters 4, 5 and 6 of reference 3 give the reader a clear understanding of the mathematical basis and implementation of cyclic block codes. The polynomial description of a codeword is also found in chapter 4 of reference 2 as follows (in general form):

$$\text{let } c(x) = c_0 + c_1x + c_2x^2 + \ldots + c_{n-1}x^{n-1} \qquad (4)$$

where n = block length, and the polynomial is of degree n-1. Now we will develop an example, as shown in Figure 6. If the information polynomial is:

$$i(x) = i_0 + i_1x + i_2x^2 + \ldots + i_{k-1}x^{k-1} \qquad (5)$$

and the generator polynomial is:

$$g(x) = x^4 + x + 1 \qquad (6)$$

(derivation of generator polynomials is given in references 2 and 3)

then the codeword takes the form:

$$c(x) = x^{n-k}i(x) + t(x) \qquad (7)$$

where t(x) is the remainder, and is equal to:

$$t(x) = -R_{g(x)}[x^{n-k}i(x)] \qquad (8)$$

this reads "t(x) is the remainder after dividing by g(x)".

and thus

$$R_{g(x)}[c(x)] = 0 \qquad (9)$$

The encoder in Figure 6 is a systematic encoder that implements a divide-by-g(x) using shift registers and X-OR gates; it produces a (15,11) Hamming code. Assume that the register stages are first cleared to zero. Eleven information bits are shifted into the circuit; division begins after four clock shifts. The circuit produces eleven information bits followed by four parity bits, to produce a fifteen bit codeword. The four parity bits are the result of the division.

Refer again to Figure 6. As the codeword passes through the channel, noise may cause bit errors. This noise is represented as the error polynomial e(x), which has degree n-1. The sum of the codeword c(x) and noise e(x) is v(x), the received codeword:

$$v(x) = c(x) + e(x) \qquad (10)$$

The decoder in the figure implements a divide by g(x), where g(x) is the same generator polynomial used in the encoder. If no error has occurred, the remainder is zero. If the remainder is non-zero, it is calculated as:

$$s(x) = x^3 + 1 \qquad (11)$$

s(x) is the **syndrome** defined earlier! The decoder circuit in the figure calculates s(x) by dividing by g(x); if s(x) is non-zero, the appropriate information bit is inverted, yielding the original information codeword c(x). The encoder and decoder of Figure 6 constitute a single-error-correcting system. Note the simplicity of the circuit, but remember it is limited to correcting single errors.

The example just presented is of a **binary** block code; the coefficients of all the polynomials are either binary 0 or 1. As you may recall from our brief history lesson, Reed and Solomon developed multiple error correcting codes in a 1960 paper. These codes (and there are many) are are very effective in the presence of burst errors. The **overhead** of these codes is typically 10% or less, making them very efficient. However the decoding hardware is far more complex than described above for the binary code. Algorithms for decoding of R/S codes must calculate two syndromes, one for error location, and one for error magnitude. This is because the mathematics is over a Galois field $GF(2^m)$, where m is a small integer on the order of 7 or 8. In hardware,a parallel bus m bits wide is required. The data bits are arranged into "symbols" of m bits, and the arithmetic calculations are done on these symbols. A number of sophisticated decoding algorithms have been developed for the many Reed Solomon codes. They have found many practical applications, such as compact discs.

CONVOLUTIONAL CODES

These codes are based on a probabilistic approach to the problem of error control. They were originally called **recurrent** codes, and are also referred to as **tree** codes, from the use of a tree or trellis diagram used to visualize the sequence of events. A convolutional code does not have a simple block structure, with each codeword independent from all others. Rather, the codewords are generated using a **sliding window** over the information symbols. A continuous stream of encoded symbols is produced, where successive codeword frame are coupled together by the encoder.

Figure 7 illustrates a generic convolutional encoder, and will be used to define terms common in the literature. The input information is broken into **information frames** of k_0 symbols; **m** is the number of these frames stored in the encoder shift register. The length of the shift register is **m** X k_0, which is the **constraint length**, denoted by **v**. The output **codeword frame** is made up of n_0 symbols. The code is referred to as an (n_0,k_0) **code**. **K** is the **wordlength** of the code and is equal to $(m + 1)k_0$. **Blocklength N** is equal to $(m + 1)n_0$, and is the length of the output code that may be influenced by an input frame k_0. The rate **R** of the code is k_0/n_0. The input to the encoder is data at a rate of k_0 symbols per second, and the output is data at a rate of n_0 symbols per second.[3]

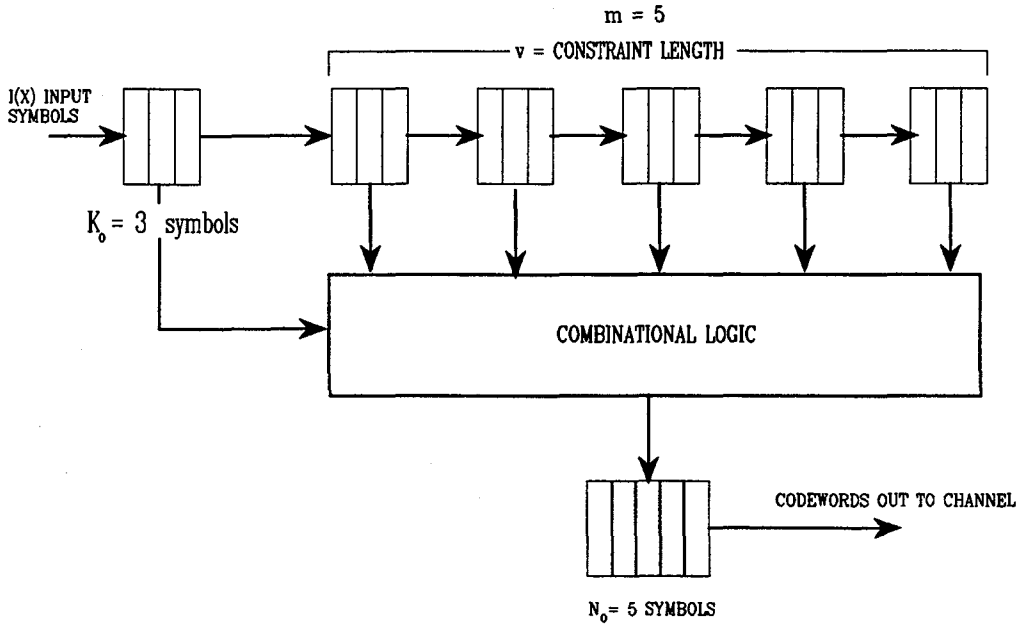
Next we will consider the mathematical basis for these codes. We used a **generator polynomial** in constructing a block code. Convolutional codes require a set of multiple polynomials to describe them; these are best described by a mathematical **matrix**. Matrix notation provides a means of writing a number of simultaneous equations (polynomials) in compact form. Appendix A of reference 2 presents a summary of matrix definitions and manipulations.

A matrix is made up of row and column **vectors**, whose elements are the coefficients of the polynomials.

The generator-polynomial matrix is given by:

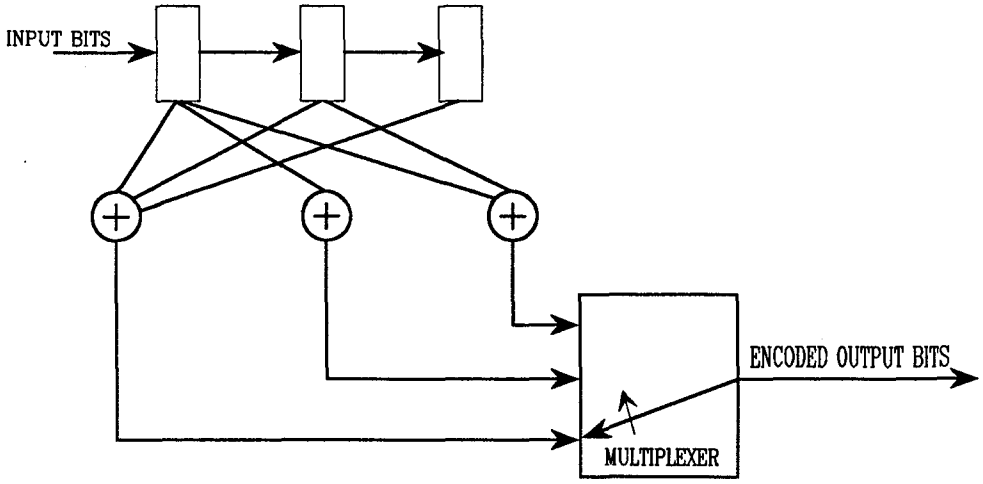
$$G(x) = [g_{ij}(x)] \qquad (12)$$

This is a k_0 by n_0 matrix of polynomials. Further, if d(x) is a



GENERIC CONVOLUTIONAL ENCODER

FIGURE 7



(3,1) CONVOLUTIONAL ENCODER

FIGURE 8

set of k_0 information polynomials and c(x) is a matrix of n_0 codeword ploynomials, then:

$$c(x) = d(x)g(x) \qquad (13)$$

Also, there is a parity check matrix H(x) that satisfies

$$G(x)H(x)^T = 0 \qquad (14)$$

and there is a **syndrome-polynomial** vector given as

$$s(x) = v(x)H(x)^T \qquad (15)$$

where v(x) represents the received codewords.

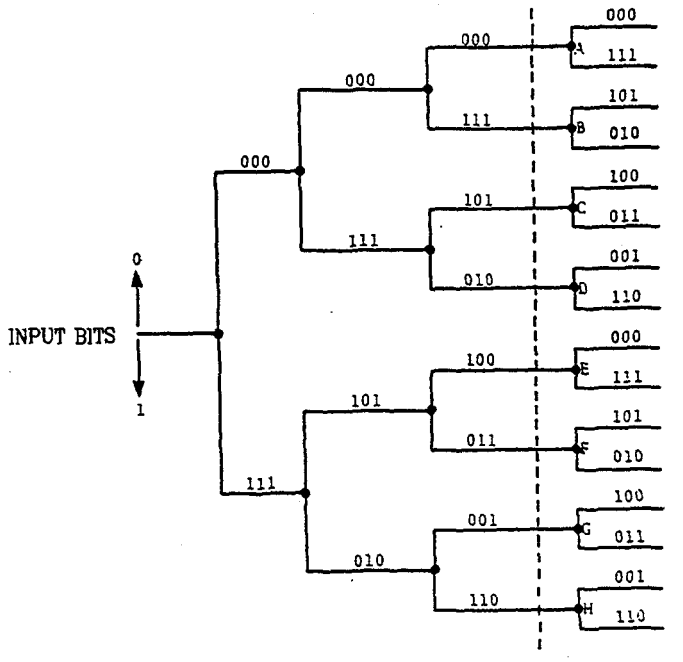
We will now go on to describe the **tree** and **trellis** structures, as these are very useful in visualizing the generation of the convolutional code. We will use an example taken from a paper by Batson. [4] Figure 8 shows a simple encoder made up of a three stage shift register, three XOR gates, and a multiplexer. This is a (3,1) tree encoder. The coefficients of the generator polynomials specify which stages of the shift register are connected to each modulo 2 adder. In Figure 9, we see the tree that describes the operation of the encoder. Assume the shift register contains all zero's to start the sequence. In the tree, an input of zero causes the circuit to follow the upper branch, a logic one the lower branch. The labels on the branches indicate the resulting output code. An input sequence of 1011 results in an output sequence of 111 101 011 010. It can be seen that this tree would grow very large after a relatively short input stream, and would be unwieldy.

An alternate structure which is much more compact is the **trellis diagram.** This may be seen in Figure 9. The state of the encoder is the most recent contents of the k_0-1 stages of the encoder shift register. Time is from left to right. The circuit steps from state to state at clock times. The path is up for a logic zero, and down for a logic 1. The encoded output bits are shown on the branches. The repetitive structure of the trellis is immediately apparent.

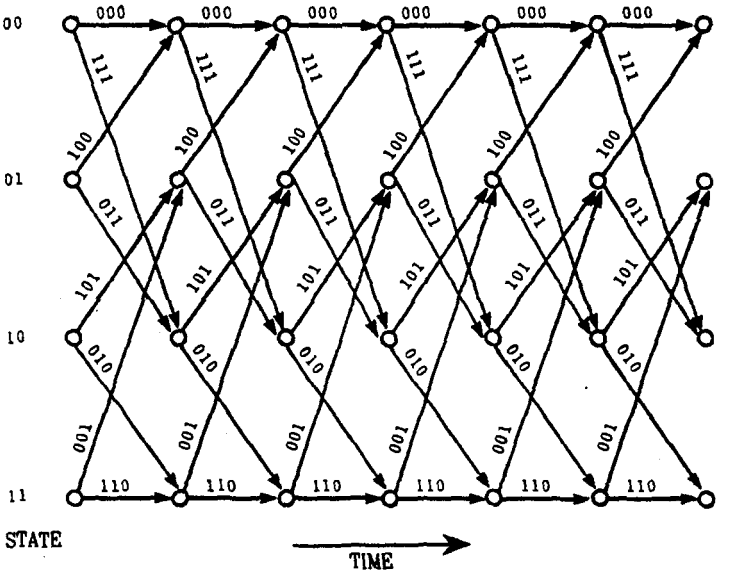
The trellis structure is useful is understanding decoding algorithms for convolutional codes, such as the Viterbi algorithm[4], which has found wide application. It basically attempts to find a valid path through the trellis that is as close as possible to the received sequence. This method is very effective, but it should be noted that the hardware requirements for the Viterbi decoder grow exponentially with constraint length.

The coding gain of the Viterbi algorithm may be improved by the use of a **soft decision** demodulator. Such a demodulator takes into account the distance of a received symbol from the center of it's decoding sphere. This is accomplished with an analog-to-digital converter (A/D) to quantize the received signal.

We made brief mention of the **syndrome vector** earlier. Next, we will illustrate its use in a syndrome feedback decoder of a convolutional code. Figure 10 illustrates both the encoder and decoder for such a code. The encoder and decoder both calculate the same parity bits if the data is received error-free. However, if a data bit is received in error, the locally calculated parity will differ from the received parity. When this difference occurs, a logic 1 will appear in the syndrome register. It is the function of the decoder decision table to find the **most likely** bit error location. [2]

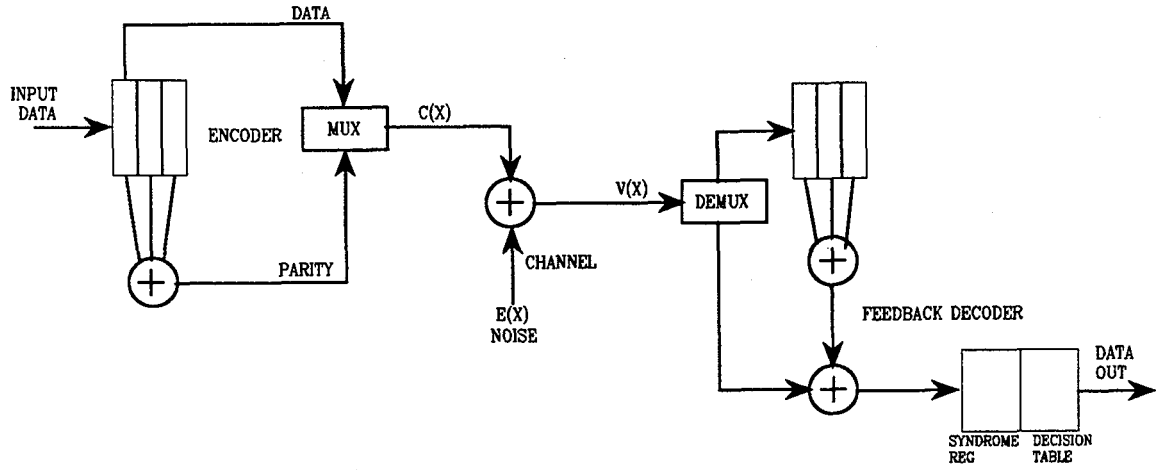


CODE TREE



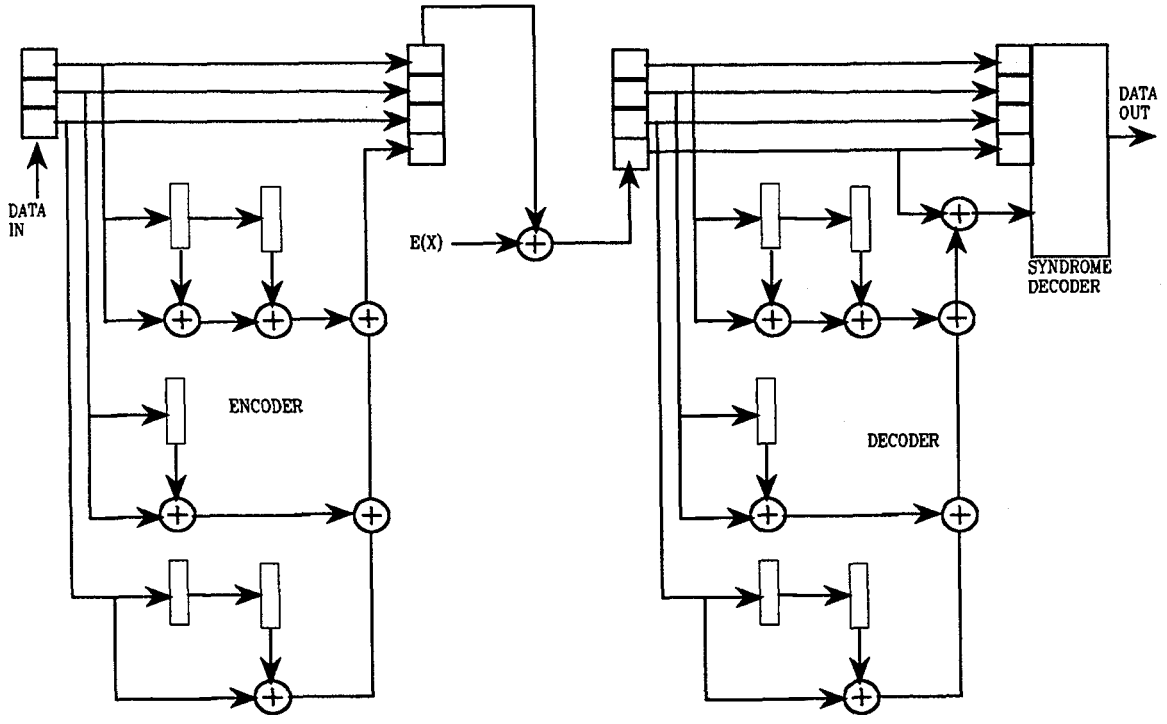
TRELLIS DIAGRAM

FIGURE 9



CONVOLUTIONAL SYNDROME FEEDBACK DECODER

FIGURE 10



WYNER-ASH ENCODER AND DECODER

FIGURE 11

More than one error pattern can result in the same syndrome; the decoder will choose the pattern with the least errors and compensate for that pattern.

One more example of a convolutional encoder/decoder combination is illustrated in Figure 11; the Wyner-Ash code is used here.[3,5] The decoder uses the syndrome concept. The Viterbi decoder described earlier is a better method of improving coding gain.

INTERLEAVING AND CONCANTENATED CODES

Figure 2 illustrates how these two techniques relate to block codes and convolutional codes. Figure 12 depicts a hardware block diagram of a system employing these techniques.

Interleaving is used to transform a bursty channel into an independent error channel by scrambling the encoded symbols before transmission. An interleaver structure is built from semiconductor memory in a rectangular array. Encoded data is written into the array by rows and out by columns before transmission. After reception and decoding by the decoder, the process is reversed. This techniques has proven effective in satellite links that are subject to long bursts of errors.

Concantenated codes are used to increase coding gain. A Reed-Solomon code is used with the Viterbi decoder in Figure 12 due to the bursty nature of uncorrected errors out of the Viterbi decoder.[2,3]

CONCLUSIONS

This paper is intended as an introduction to error control theory for the CATV system engineer. In order to determine the optimum strategy to develop a practical digital delivery system for CATV, a number of factors should be considered:

1. The worst-case allowable symbol-error-rate. This determines the required coding gain.

2. The environment in which the system will operate, to include channel C/N, expected reflections, and other impairments.

3. The type of digital modulation selected (eg, 16QAM, 64QAM, etc). The digital channel bandwidth, the allowable signal power relative to AM channels, and any effect on those AM channels must be considered.

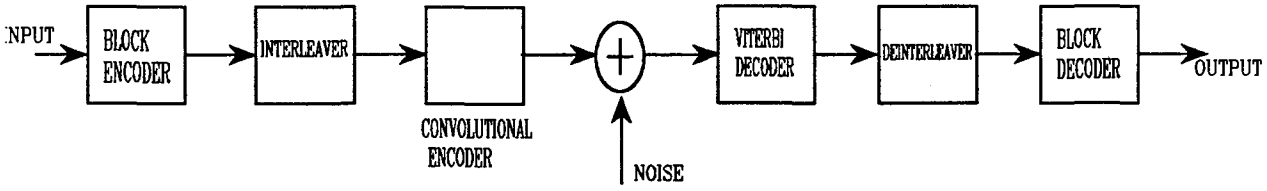
4. The required parity overhead, which increases the symbol rate.

5. The distribution of errors in the channel. A CATV channel is subject to random errors, not burst errors.

6. The behavior of the required adaptive equalizer under various conditions.

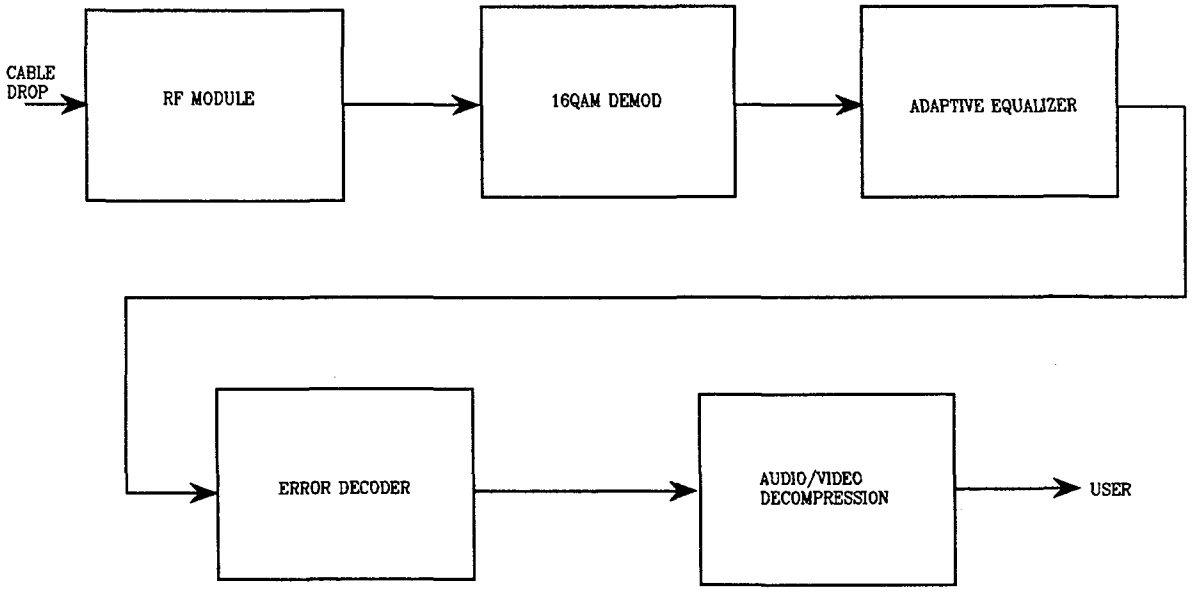
7. The circuit complexity and cost of the hardware, especially in the subscriber terminal.

Figure 13 illustrates the likely functional blocks in a CATV subscriber terminal employing digital data delivery. The demodulator, adaptive equalizer, error decoder, and decompression hardware must be designed to operate in concert. A properly designed system promises to deliver consistent high quality video and audio to all subscribers.



INTERLEAVING AND CONCANTENATED CODES

FIGURE 12



SUBSCRIBER TERMINAL

FIGURE 13

REFERENCES

1. Shannon, C.E., A Mathematical Theory of Communication, Bell Syst. Tech. J., Vol XXVII (1948)

2. Michelson, A, and Levesque, A., Error-Control Techniques For Digital Communication, John Wiley & Sons, New York, 1985

3. Blahut, R., Theory and Practice of Error Control Codes, Addison-Wesley Publishing Co., Reading Mass., 1983

4. Batson, B., A Description of The Viterbi Decoding Algorithm, NASA Report EE70-8008(U), May 1970

5. Wyner, A., and Ash, R., Analysis of Recurrent Codes, IEEE Trans. Inf. Theory, IT-9 (1963): 143-156

6. Webster's New World Dictionary, The World Publishing Company, Cleveland, 1964