AN ERROR CORRECTION ALGORITHM OF THE CONVOLUTIONAL TYPE USING THRESHOLD DECODING

1702(H-9)

Erik DeBenedictis, Mitsuya Komamura,
Bart Locanthi and Larry White
Pioneer North America Development
Laboratory
Pasadena, CA

**Presented at
the 67th Convention
1980 Oct. 31/Nov. 3
New York**

AN ERROR CORRECTION ALGORITHM OF THE CONVOLUTIONAL TYPE USING THRESHOLD DECODING

By

Erik DeBenedictis, Consultant

Mitsuya Komamura, Bart Locanthi

Larry White

PIONEER NORTH AMERICA DEVELOPMENT LABORATORY

Pasadena, California

ABSTRACT

The application of convolutional error correcting codes to digital audio is examined. One implementation, (R 2/3, 30 taps), requiring one extra bit for each two information bits, has been simulated and would perform at about one miscorrection per hour at a mean bit error rate of 10^{-3}, and one miscorrection per 14,000 years for a mean bit error rate of 10^{-4}. The decoding hardware would consist of 20 K bits of memory and a comparatively small amount of logic. Implementation as an integrated circuit would be feasible.

INTRODUCTION

An error correction code of the convolutional type using threshold decoding will be discussed. This algorithm has a high degree of correctability and moderate overhead. It is particularly well suited to digital audio recording systems which may be characterized as having random error bursts of relatively short length with bit error rates up to 3×10^{-3} (compact PCM laser optical discs).

In one of the more complex, high rate implementations (R 2/3, 30 taps), a bit error rate of 10^{-3} will produce one miscorrection per hour. That same implementation will produce one miscorrection in 14,000 years for a bit error rate of 10^{-4}.

A much simpler implementation (R 1/2, 8 taps), will produce one miscorrection per hour for a bit error rate of 6×10^{-4} or one miscorrection in 10 months for a bit error rate of 10^{-4}.

PART I: BASIC CONVOLUTIONAL CODE (RATE 1/2)

The Basic Code is an encoding and decoding procedure with a fixed overhead and varying degrees of correctability depending on the implementation. The basic code uses two channel bits for each information bit, and is called a rate 1/2 code. The Modified Code has somewhat less correctability but uses only three channel bits for each two information bits. This is a rate 2/3 code.

Basic Code Strategy

The encoder (See Figure 1) takes a stream of data bits:

DDDDDDDDDDDDDDDDDDDDD,

interleaves parity bits with the data bits,

PDPDPDPDPDPDPDPDPDPDPDPDPDPDPDPDPDPDPDPDPD

and encodes them on the recording surface.

The decoder for the Basic Code (See Figure 2) recovers the bits from the recording surface, recalculates the parity bits, and inverts any of the data bits it believes are incorrect.

Each parity bit is calculated from the exclusive-or of a fixed number of data bits at fixed offsets from the parity bit. These offsets, in the hardware implementation, correspond to taps on a shift register memory. The correctability of the Basic Code is not determined by the overhead, which is fixed at rate 1/2, but instead is determined by the number of taps and their spacing.

Basic Code Example

In the following example we show the Basic Code with 6 taps and minimal spacing being used to correct one and two errors in the space of 40 bits. The parity bits in this example are computed by the, exclusive-or of the 3rd, 5th, 11th, 23rd, 33rd, and 37th preceeding data bit. Note that each parity bit is formed from the exclusive-or of 6 data bits, and each data bit shows up as part of 6 parity bits.

To show the correction of a single bit error, let's assume the data bit marked below with an "X" was recovered incorrectly during the playback process:

```
PDPDPDPDPDPDPDPDPDPDPDPDPDPDPDPDPDPDPDPD
 X
```

The decoder does not know which bit is incorrect, but when it recalculates the parity bits it finds a few that are different from the ones in the recording. These are shown here with an "*" just below the parity bit:

```
PDPDPDPDPDPDPDPDPDPDPDPDPDPDPDPDPDPDPDPD
 X  * *     *          *         *   *
```

After marking all the parity bits which do not agree with what it computes they should be, the decoder counts for each data bit the number of associated parity bits which are marked. This count is shown here as a number just above the data bit:

```
 6 1 1 1 1 1 1 1 0 1 1 1 0 1 1 1 1 1 0 0 0
PDPDPDPDPDPDPDPDPDPDPDPDPDPDPDPDPDPDPDPDPD
 X  * *     *           *         *   *
```

After counting marked parity bits the decoder inverts any data bit whose count is greater than or equal to some threshold, in this case 4. Any data bit whose count is less than the threshold is left unchanged. In the above example this action would properly correct the single bit error.

If, for example, the error had occurred in a parity bit instead of a data bit, the only marked parity bit would be the one with the error, and all of its associated data bits would get a count of 1. No correction would be done when this happens because all error counts would be less than the threshold.

To show the correction of a double error, let's assume the data bits marked below with the letter "X" were recovered incorrectly during the playback process:

```
PDPDPDPDPDPDPDPDPDPDPDPDPDPDPDPDPDPDPDPDPD
 X X
```

The decoder would mark and count bits as follows:

```
 5 5 2 2 1 2 2 2 1 1 2 2 1 1 2 2 2 2 1 0 0
PDPDPDPDPDPDPDPDPDPDPDPDPDPDPDPDPDPDPDPDPD
 X X* 0 *   * *         * *       * * * *
```

Note that the parity bit above the "0" has two incorrect data bits associated with it, and so is not marked by the decoder with an "*".

We have shown in these examples how the Basic Code, implemented with 6 taps and minimal spacing, would operate to correct one and two bit errors. With these taps this algorithm can correct any three incorrect bits which occur in the space of 74 bits.

Basic Algorithm Error Expectation

In the following calculations we assume the spacing of taps is far enough apart that the effects of bursts are so small that they can be left out of the calculations. The validity of this assumption can be tested by computer simulation with Gilbert noise.

Let p = Probability of a bit being incorrect before decoding

C = The number of taps

T = The threshold - generally T = INT((C+3)/2) for optimum performance (See Figure 3)

D = An arbitrary data bit

A. Assume D is incorrect before decoding. D will be corrected only if T or more parity bits associated with D are marked. Each parity bit will be marked if an even number of other data bits, or itself, are incorrect before decoding.

Let s_i, the probability that a parity bit associated with an incorrect data bit is marked, be computed as follows:

$$s_i = \sum_{j=0 \text{ by } 2\text{'s}}^{C} p^j (1-p)^{C-j} \binom{C}{j}$$

So then q_i, the probability that an incorrect data bit is corrected is:

$$q_i = \sum_{j=T}^{C} s_i^j (1-s_i)^{C-j} \binom{C}{j}$$

B. Assume D is correct before decoding. D will be made incorrect if T or more parity bits associated with D are marked. Each parity bit will be marked if an odd number of other data bits, or itself, are incorrect before decoding.

Let s_c, the probability that a parity bit associated with a correct data bit is marked, be computed as follows:

$$s_c = \sum_{j=1 \text{ by } 2\text{'s}}^{C} p^j (1-p)^{C-j} \binom{C}{j}$$

So then q_c, the probability that a correct data bit is changed during decoding to be incorrect, is:

$$q_c = \sum_{j=T}^{C} s_c^j (1-s_c)^{C-j} \binom{C}{j}$$

C. Let D be an arbitrary data bit. D will be incorrect after decoding only if (A) it was incorrect before decoding and was left uncorrected, or (B) it was correct before decoding and was made incorrect during decoding.

So let q, the probability that a data bit is incorrect after decoding, be computed as follows:

$$q = p(1-q_i) + (1-p)\, q_c$$

According to this formula, 1 error per hour (at 1.6 megabits per second of audio data) is achieved with the following error rate and taps:

TAPS	BIT ERROR RATE	MISCORRECTIONS/HOUR
6	2.93×10^{-4}	1
8	6.10×10^{-4}	1
10	9.64×10^{-4}	1
12	1.31×10^{-3}	1
16	1.90×10^{-3}	1
24	2.65×10^{-3}	1
32	3.01×10^{-3}	1

Figures 4 and 5 show miscorrections per hour for various combinations of taps, thresholds, and bit error rates.

PART II: CONVOLUTIONAL CODE (RATE 2/3)

Modified Rate 2/3 Code Strategy

The Modified Code uses only one parity bit for every two data bits. This increases the coding efficiency from 1/2 to 2/3, with a corresponding reduction in correctability per tap, as shown in the next section.

If 32 taps are used when implementing the Modified Code, then each parity bit will have 32 associated data bits, but each data bit will have only 16 associated parity bits. The threshold is based on the number of parity bits associated with each data bit, but the amount of memory necessary to implement the encoder and decoder is based on the number of taps. Figure 6 shows the optimum thresholds for various numbers of taps.

Modified Algorithm Error Expectations

In these calculations we again assume the spacing of taps is large enough that the effects of bursts may be left out of the calculations.

The calculations of s_i and s_c for the modified algorithm are identical to that for the Basic Algorithm because these calculations are based on the number of data bits associated with each parity bit, which is the same as C, the number of taps.

The calculations for q_i and q_c, however, are modified because the number of parity bits associated with each data bit is only half the number of taps.

Let H = C/2

$$q_i = \sum_{j=T}^{H} s_i^j (1-s_i)^{H-j} \binom{H}{j}$$

$$q_c = \sum_{j=T}^{H} s_c^j (1-s_c)^{H-j} \binom{H}{j}$$

$$q = p \quad (1-q_i) + (1-p) \; q_c$$

Where q is again the probability of a data bit being incorrect after decoding.

TAPS, THRESHOLD	BIT ERROR RATE	MISCORRECTIONS/HOUR
12,4	1.50×10^{-4}	1
16,5	3.20×10^{-4}	1
20,6	4.87×10^{-4}	1
32,9	9.54×10^{-4}	1

Figures 7 and 8 show miscorrections per hour for various combinations of taps and thresholds and bit combinations.

Table 1 shows register memory length vs number of taps for codes of rate 1/2 and 2/3.

For evaluating the performance of various error correcting and/or concealment processes, we use a PDP 11/60 computer for operating on digital music recordings stored in its memory.

These processes consist of encoding the original data, operating on the encoded data with Gilbert noise, decoding the noisy data and storing it in a new file. We can then compare the new file with the original for error examination. The final operation consists of performing A/B listening tests of the files to determine the level of acceptability of a particular process.

From our tests to date:

1. Noise resulting from the convolutional process failing at a rate of one per second is less objectionable than that from a new high quality LP phonograph record.

2. For digital noise typical of a compact optical laser disc we should expect only one miscorrection every 10,000 years from a convolutional encoding/decoding process as outlined in the paper. Furthermore, the hardware implementation of such a process is amenable to currect solid state digital technology.

3. When we have used codes of lower correctability and tried error concealment (previous sample holding, previous slope holding, or averaging) the effects of concealment were audible.

CONCLUSION

We have demonstrated the feasibility of using a convolutional code of simple implementation which can be used for compact disc PCM optical laser recording which has a high degree of correctability which should not require concealment.

GENERAL REFERENCES

James Singer, TRANSACTIONS OF THE AMERICAN MATHEMATICS SOCIETY, Vol. 43, pp. 377-385, 1938.

S. Y. Tong, IEEE TRANSACTIONS ON INFORMATION THEORY, Vol. IT-16 #5, September, 1970

J. P. Robinson, A. J. Bernstein, IEEE TRANSACTIONS ON INFORMATION THEORY, Vol. IT-13, #1, January, 1967.

James L. Massey, THRESHOLD DECODING, MIT Press, 1963

MEMORY REQUIREMENTS FOR RATE 1/2 CODE (AVERAGE BURST LENGTH 10 BITS)

NO. TAPS	BITS OF SHIFT REGISTER MEMORY
6	1080
8	2100
10	3360
12	5160
16	10,800
20	17,040
24	25,560

MEMORY REQUIREMENTS FOR RATE 2/3 CODE

NO. TAPS	BITS OF SHIFT REGISTER MEMORY
12	2400
16	4740
20	7860
24	11,760
28	17,340
30	20,220

TABLE 1

RATE 1/2 ENCODER

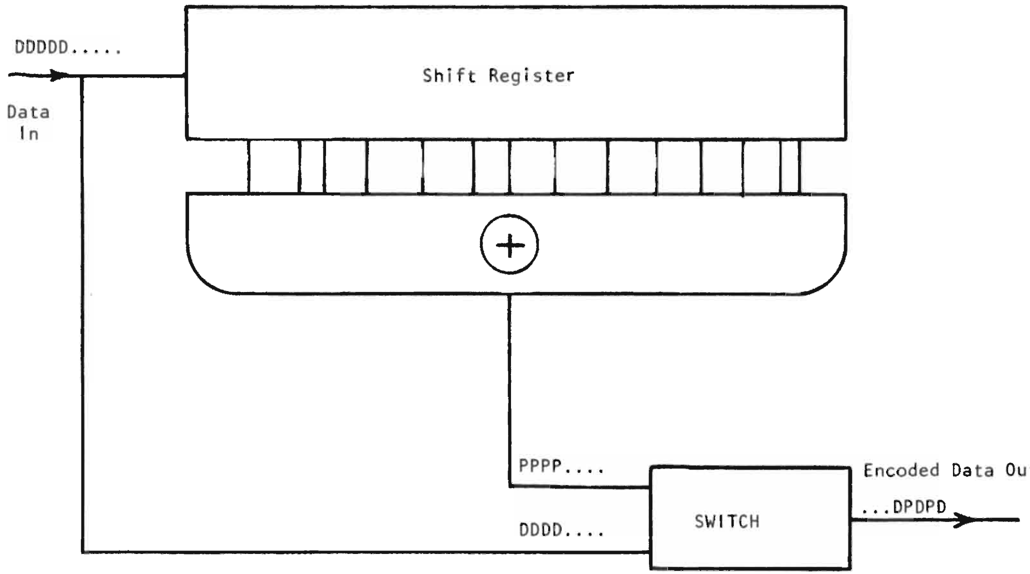


FIGURE 1

RATE 1/2 DECODER

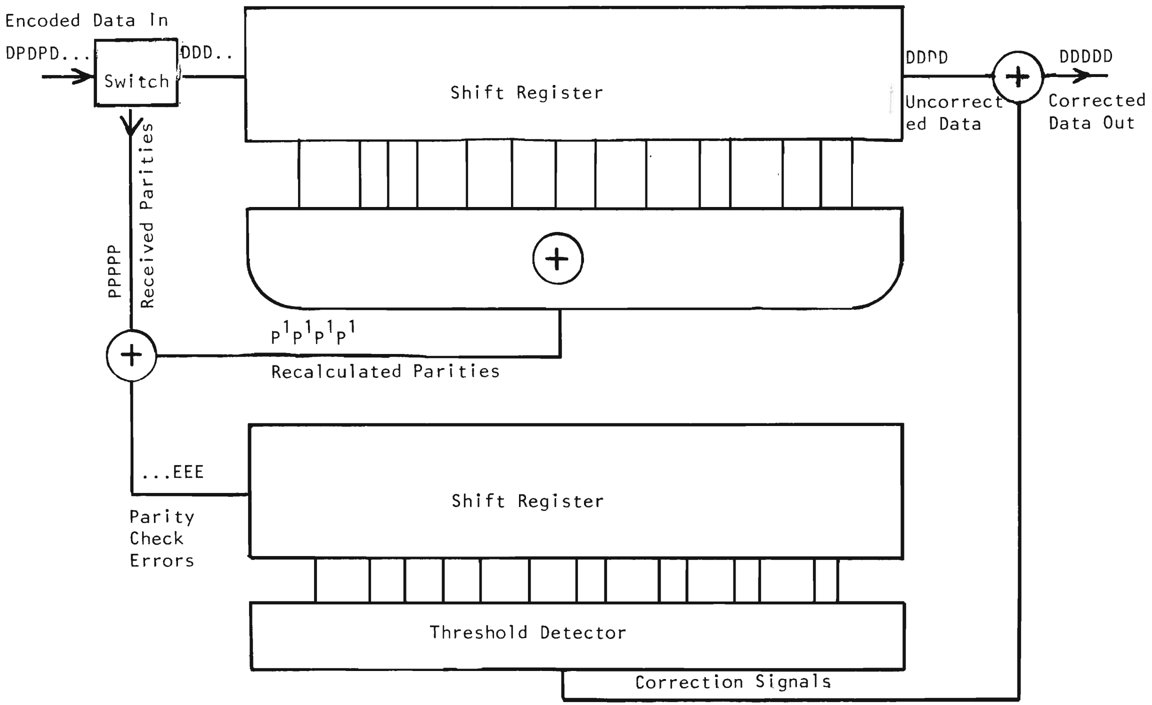
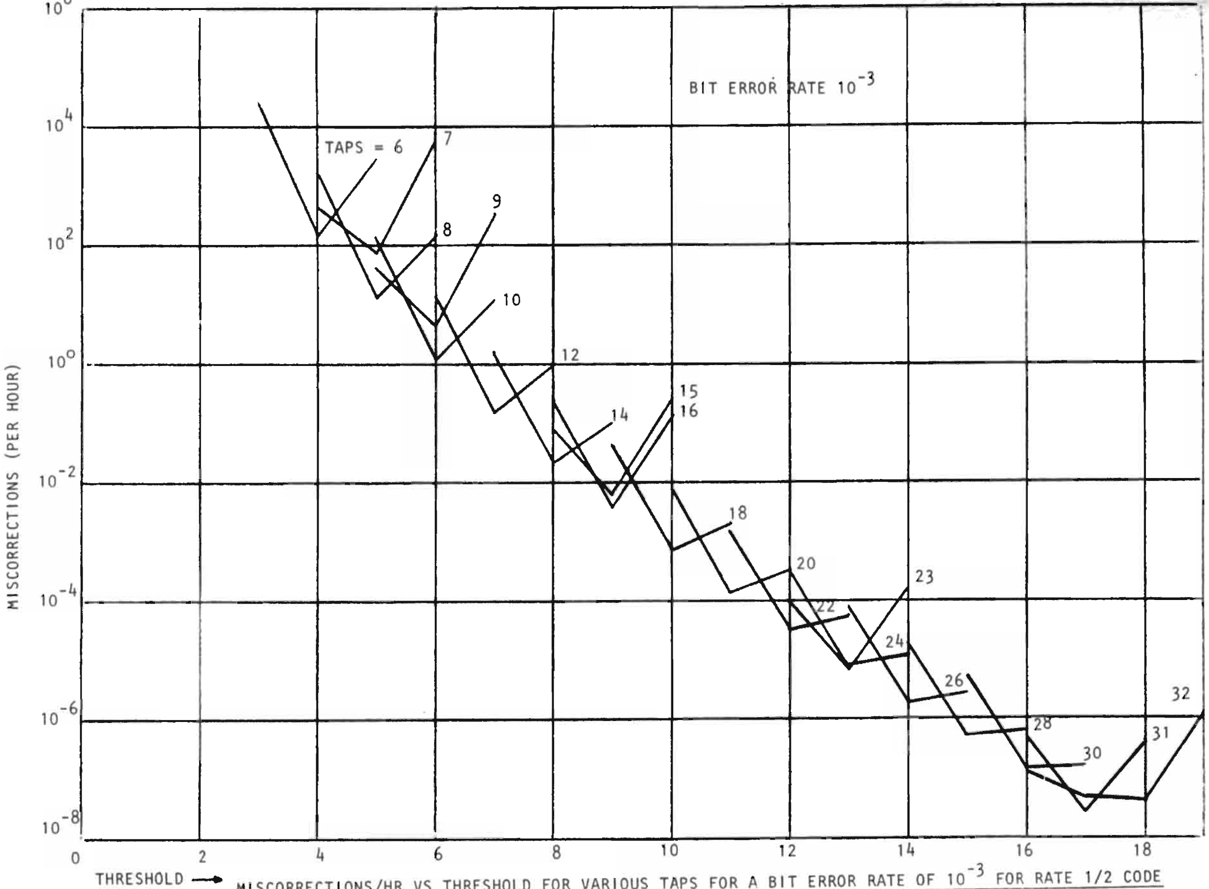
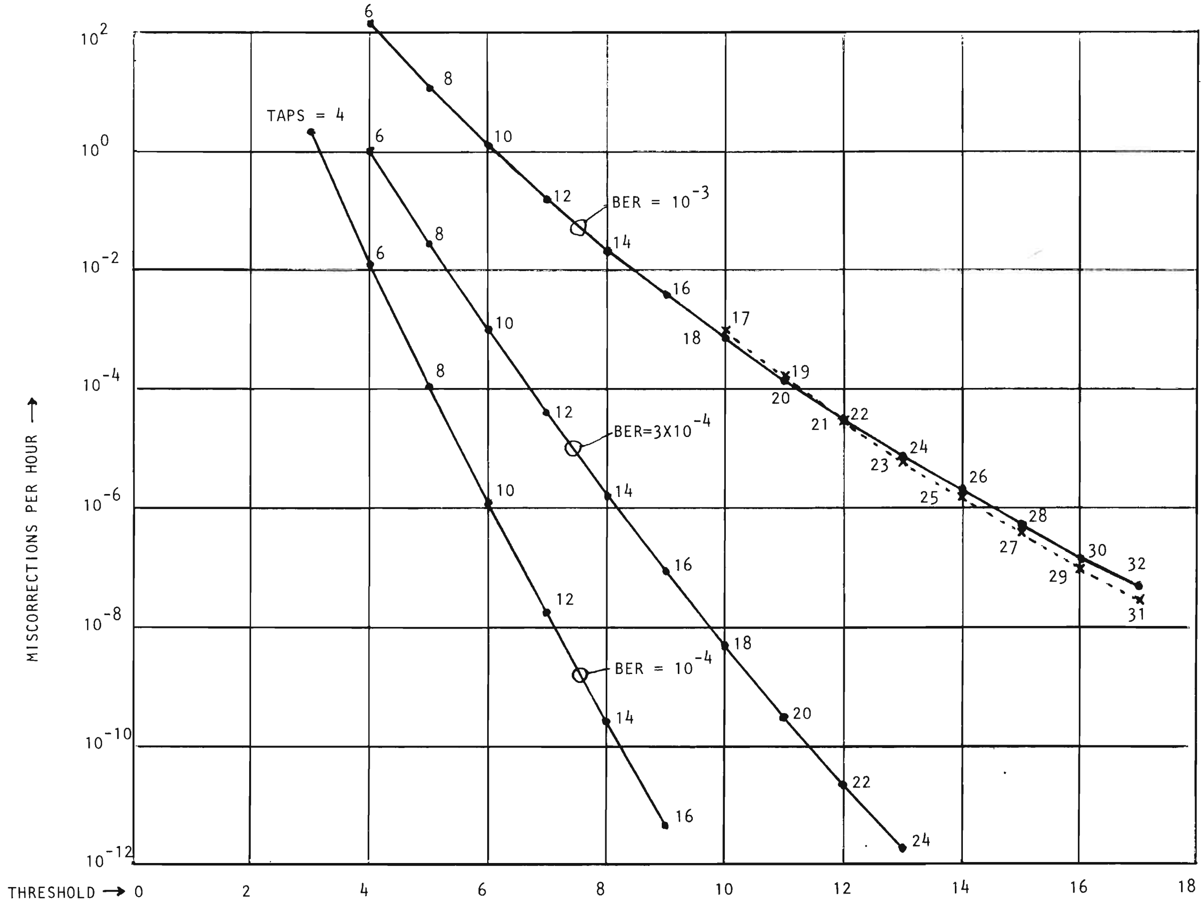


FIGURE 2



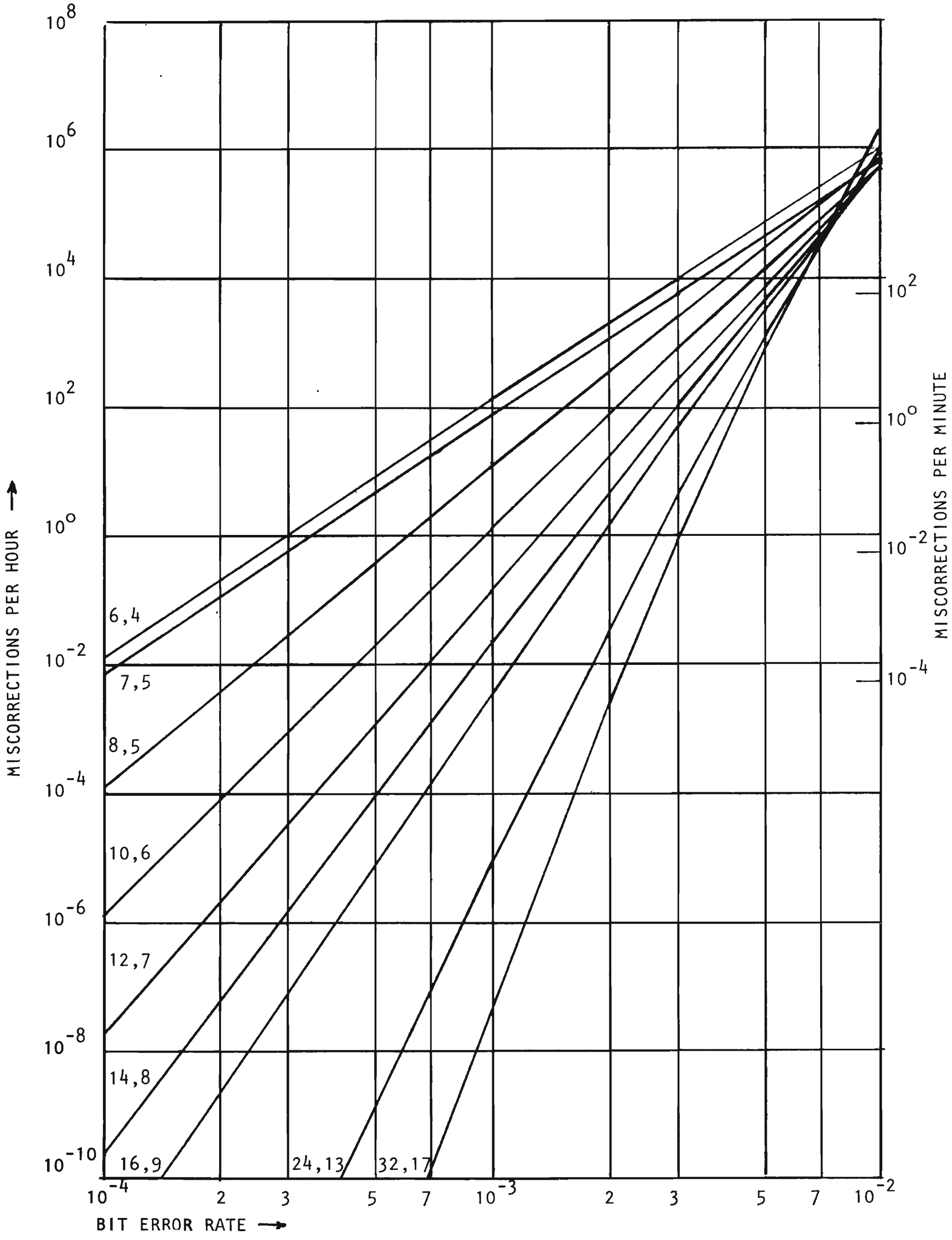
MISCORRECTIONS/HR VS THRESHOLD FOR VARIOUS TAPS FOR A BIT ERROR RATE OF 10^{-3} FOR RATE 1/2 CODE

FIGURE 3



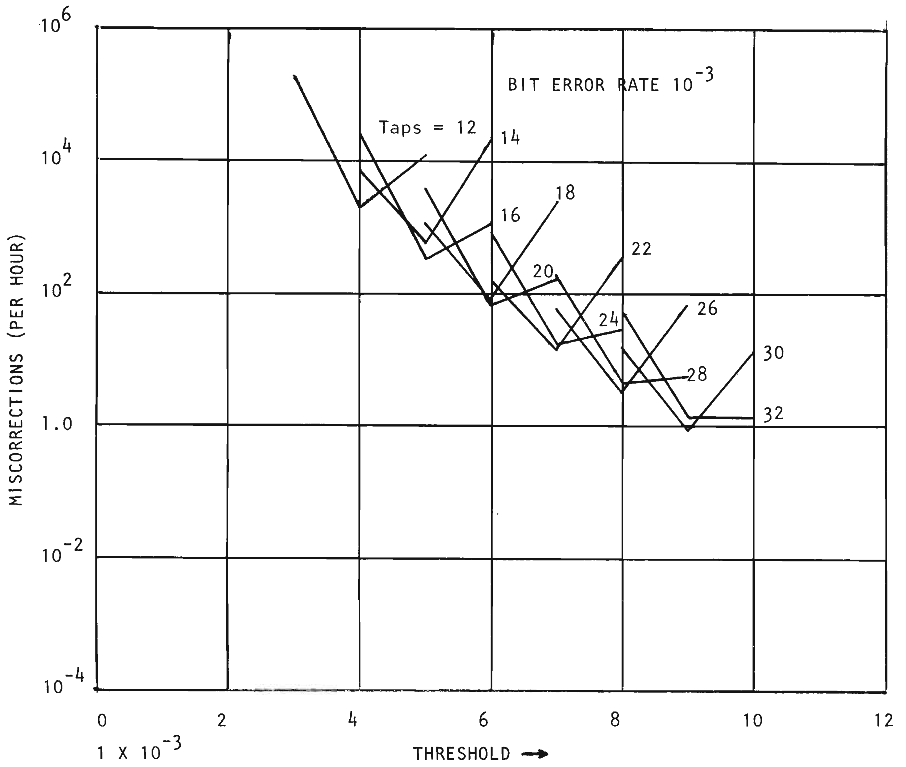
MISCORRECTIONS PER HOUR VS THRESHOLD FOR 3 BIT ERROR RATES AND VARIOUS TAPS FOR RATE 1/2 CODE

FIGURE 4



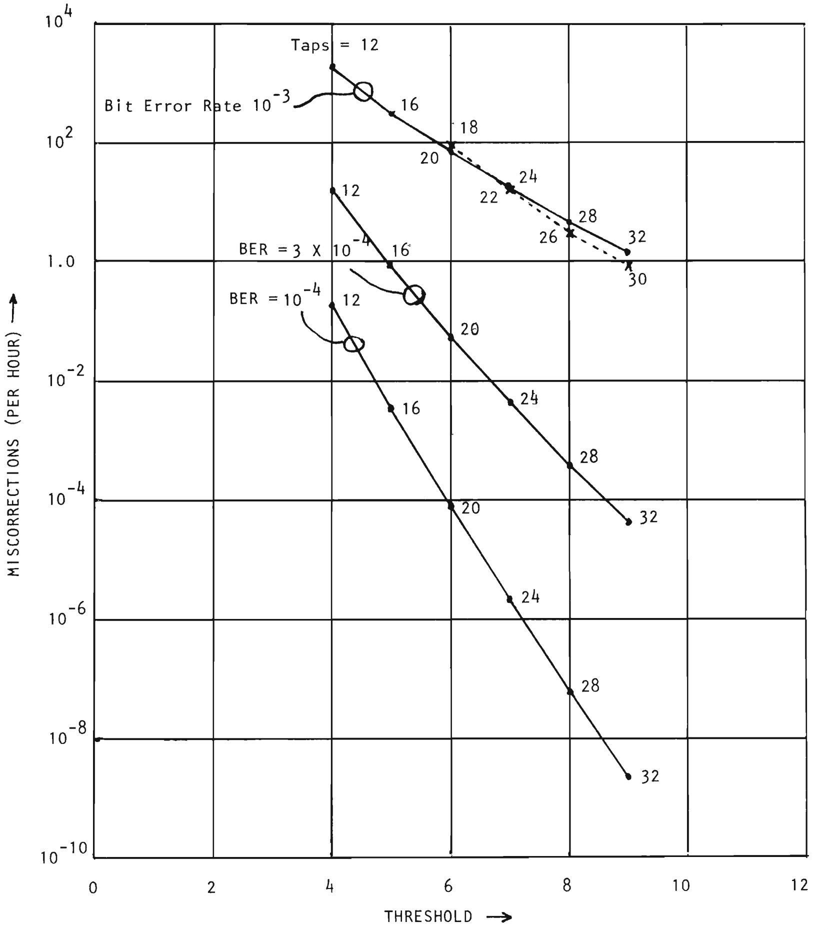
ISCORRECTIONS/HR VS BIT ERROR RATE FOR SEVERAL (TAP,THRESHOLD) COMBINATIONS (RATE 1/2)

FIGURE 5



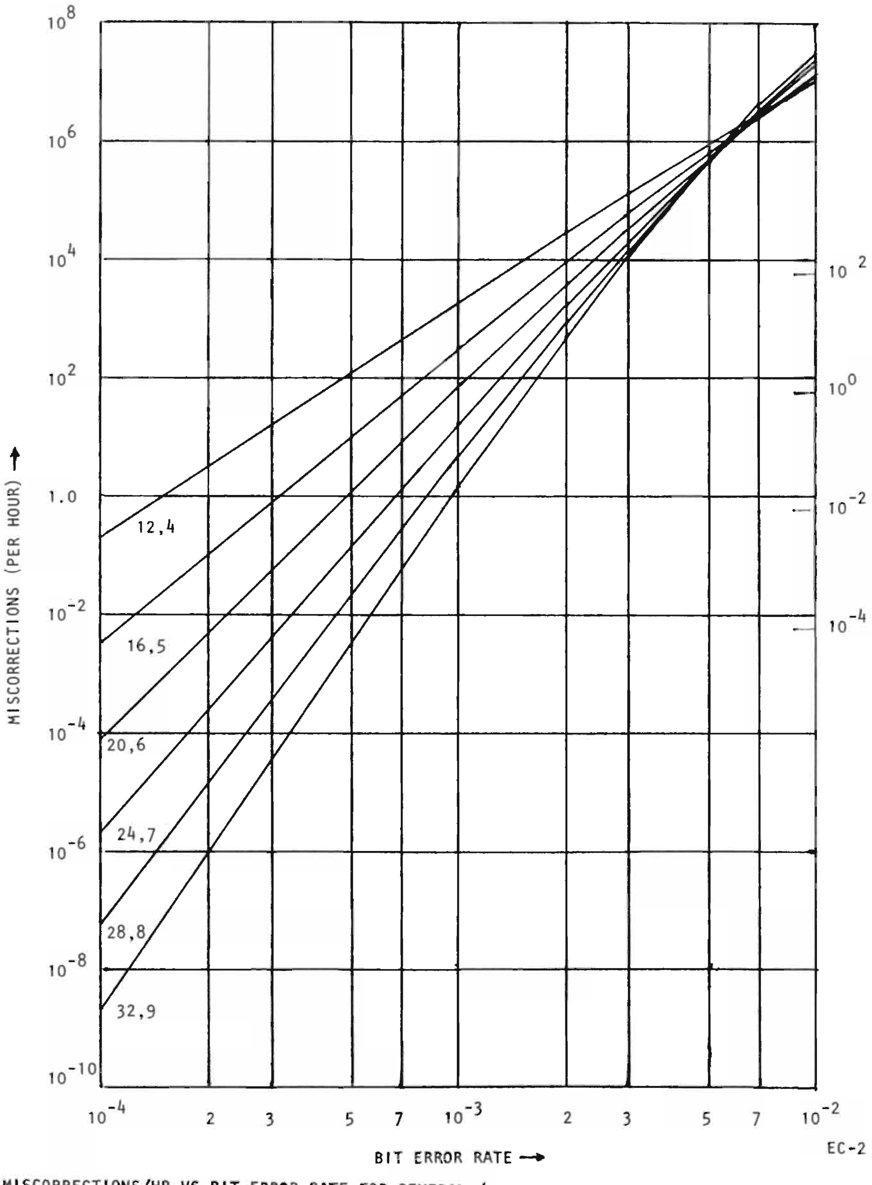
MISCORRECTIONS/HR VS THRESHOLD FOR VARIOUS TAPS FOR A BIT ERROR RATE OF 10^{-3} FOR RATE 2/3

FIGURE 6



EC-2

FIGURE 7: MISCORRECTIONS PER HOUR VS THRESHOLD FOR THREE BIT ERROR RATES AND VARIOUS TAPS (RATE 2/3)



MISCORRECTIONS/HR VS BIT ERROR RATE FOR SEVERAL (TAP,THRESHOLD) COMBINATIONS (RATE 2/3)

FIGURE 8