AT&T Technical Journal
Vol. 64, No. 7, September 1985
Printed in U.S.A.

Convolutional Coding for High-Speed Microwave Radio Communications

By M. KAVEHRAD*

(Manuscript received January 1, 1985)

This paper describes the design, testing, and measured performance of a rate 11/12 self-orthogonal convolutional codec that meets the bit-error-rate objective of M-state quadrature amplitude modulation systems in terrestrial radio transmission. The objective is to reduce a bit error probability of 10^{-6} to 10^{-10} or smaller. In fact, the measured output error probability is well below 10^{-10} when the channel error probability is below 10^{-5}.

I. INTRODUCTION

To maintain the low bit error rates—as required for high-quality data transmission in the face of increased demand for bandwidth efficiency—through the use of M-ary quadrature amplitude modulation (M-QAM) signaling, application of error-correction coding in terrestrial digital radio transmission may be desirable. Accumulated "randomly scattered" errors from different error sources may make it difficult to meet an objective of very low average background error probability, for example, a 10^{-10} Bit Error Rate (BER) with a high-level modulation such as 64-QAM.

As will be explained in this paper, convolutional coding was considered as a potential candidate for this task. To answer some of the implementation questions, and because the code seems very attractive for high-speed transmission, we developed a rate 11/12 double error-correcting convolutional codec and measured its actual performance

* AT&T Bell Laboratories.

in terms oi codec input/output bit error probability. The code efficiency is 0.9166 and is in the range that is justifiable for spectrally limited terrestrial radio applications. The intention is to have the process of encoding and decoding applied to data rates in the high megabit range. As such, there is a need for considerable parallel processing in the encoder and decoder realizations. In this paper we describe a procedure for such processing for a rate 11/12 convolutional code. The parallel processing can be implemented using standard logic elements.

After a general description of convolutional coding, we describe the implemented codec in Section I and then present the test setup in Section II. Finally, in Section III we present the measured performance of the codec. Section IV provides our conclusions.

1.1 Code structure

In general, a convolutional encoder can be assumed to be a linear sequential network that maps a block of k_0 parallel bits entering the circuit into an $n_0 > k_0$ bit block over a certain period of time. If the bits leaving the encoder are the original data bits plus $(n_0 - k_0)$ parity-check bits, defined by the particular code polynomials, the code is called systematic. The ratio k_0/n_0 is defined as the code rate. In convolutional coding, the parity bits in a given block have not only been affected by the data bits in the present block but also the preceding blocks. (This is not the case for block codes.)

The constraint length of the code N is the number of bits n_0 in a coded block multiplied by the number of blocks m checked by the $n_0 - k_0$ parity checks. For block codes, $m = 1$.

In general, convolutional codes have the same limitations and, roughly speaking, the same inherent capabilities as block codes.

Like block codes, convolutional codes are capable of correcting random errors, burst errors, and combinations of random and burst errors. Since the parity bits in a convolutional code check information symbols in the blocks preceding the present block, the basic parity matrix is of the form

$$h = [P_{m-1}^T \quad OP_{m-2}^T \quad O \cdots P_0^T I],$$

where $(n_0 - k_0)$ by k_0 matrices P_i^T are arbitrary, and O and I represent, respectively, the zero and identity matrices of order $n_0 - k_0$. The encoding process associates $(n_0 - k_0)$ parity checks, as specified by the matrix h, to every block of k_0 bits of the entering information, once every n_0 channel-bit times.

Decoding of a convolutional code is possible by algebraic or sequential methods. Although both techniques deal with convolutional codes, they do so in entirely different manners. As a result, most of the

definitions applied in coding are of different forms, depending on the way that the code is decoded.

In algebraic methods, the decoder bases its decisions on the check digits within a constraint length, but in the sequential methods the decisions are made based on a section of the syndrome that is much longer than the code-constraint length. This substantially improves error-correcting performance.

Decoding simply associates an error pattern with the syndrome. After the decoder calculates the syndrome, it uses it to decode the oldest block in its registers. Like block codes, a convolutional code can be modeled in terms of parity-check and generator matrices.

An n-tuple b is a code word in the convolutional code family described by h if and only if $bH^T = 0$, where H^T is the transpose of the parity-check matrix

$$H = \begin{bmatrix} P_0^T I & & & \\ P_1^T & OP_0^T I & & \\ P_{m-1}^T & OP_{m-2}^T & O \cdots & P_0^T I \end{bmatrix},$$

where P_i^T, O, I were defined in the equation for h. The sequence b can be obtained by multiplying the information-bit sequence $d = d_{k-1} \cdots d_1 d_0$ by the generator matrix of the code, which is

$$G = \begin{bmatrix} IP_0 & OP_1 \cdots & OP_{m-1} \\ & IP_0 \cdots & OP_{m-2} \\ & & IP_0 \end{bmatrix},$$

that is, $b = dG$.

It can be seen that $GH^T = 0$. The syndrome of a received n-tuple r is defined as $rH^T = s$. Clearly, all n-tuples containing errors (not a code word) have nonzero syndromes.

Self-orthogonal codes are a class of convolutional codes that are rather simply implementable, and they can also be decoded with majority-logic decoding (a nonsequential decoding approach). The disadvantage of these codes is that for large values of n_0 and k_0 the random-error-correction ability of the code decreases, that is, their minimum distance gets small. For ease of implementation only the self-orthogonal codes with $n_0 - k_0 = 1$ are usually considered for construction. In self-orthogonal codes with minimum distance d, at least $d - 1$ rows of the parity matrix H are orthogonal (in the coding sense) on each of the k_0 bits of the zero block. For these codes, it can be shown that

$$N \leq n_0 \left[\frac{(n_0 - 1)(d - 1)(d - 2)}{2} + 1 \right],$$

where N is the constraint length and d the minimum distance of the code.

Suppose that a self-orthogonal code has the ability of correcting $t = (d - 1)/2$ random errors. If t or fewer errors occur within a constraint length of the $(m - 1)$th block, it can be shown that it is possible to arrange a set of $J = d - 1$ orthogonal check sums on each data bit in the $(m - 1)$th block of this code. It has been proved that the value assumed by the majority of these check sums will always be the correct value of the noise digit that was added to the considered data bit.[1] Therefore, by adding this bit to the data bit the correction can be made.

The threshold decoder corrects errors by reencoding the received bit stream, adding the regenerated parity bit to the corresponding received parity bit, and then forming the syndrome. Then the error pattern associated with the syndrome is added to the $(m - 1)$th block of the received sequence. For practical limitations, the m-bit section of the (semi-infinite) syndrome that is within a constraint length is stored in the syndrome register. As mentioned above, $J = d - 1$ of the check sums represented by the syndrome bits are orthogonal on each data bit in the $(m - 1)$th block.

Each of the k_0 majority gates functions as a voter for one of the orthogonal check-sum sets. Therefore, the syndrome register bits are weighted according to the coefficients of the generator polynomials to form the orthogonal check-sum sets, and fed to k_0 majority gates (threshold circuits). Hence, the number of inputs to each majority gate is the same as the number of terms in each generator polynomial, that is, $J = d - 1$.

The output of the ith threshold circuit, which is a zero when more than

$$\begin{cases} J/2, & J \text{ even} \\ (J-1)/2, & J \text{ odd} \end{cases}$$

of the inputs are zeros, and one otherwise, is added to the ith data bit in the $(m - 1)$th block in the parity regenerator (reencoder) circuit. This correction bit is also used to invert each bit of the syndrome that is connected to the ith threshold circuit. After correcting all the errors, the bits are shifted out so that the next block can be processed.

With this introduction we can proceed with the design of the code for microwave radios.

1.2 Design model

The self-orthogonal convolutional code applied here, as stated earlier, is a rate 11/12 code. That is, the encoder appends one parity bit to the end of each block of 11 information bits to form a 12-bit coded block. The code constraint length, that is, the number of bits checked by every parity bit, is $N = 1716$. Therefore, the maximum term in the

code generator polynomial is of the order of 142, which is the length of the encoder and syndrome shift registers. The code generator polynomial set is[1]

$$G_1 = 1 + D + D^3 + D^7$$
$$G_2 = 1 + D^8 + D^{24} + D^{47}$$
$$G_3 = 1 + D^9 + D^{55} + D^{73}$$
$$G_4 = 1 + D^{11} + D^{92} + D^{128}$$
$$G_5 = 1 + D^{22} + D^{43} + D^{83}$$
$$G_6 = 1 + D^{10} + D^{101} + D^{114}$$
$$G_7 = 1 + D^{59} + D^{76} + D^{103}$$
$$G_8 = 1 + D^{42} + D^{122} + D^{142}$$
$$G_9 = 1 + D^5 + D^{57} + D^{95}$$
$$G_{10} = 1 + D^{87} + D^{113} + D^{132}$$
$$G_{11} = 1 + D^{66} + D^{99} + D^{133}. \quad (1)$$

We use these polynomials to form the encoder shift register shown in Fig. 1 and the syndrome register of the decoder in Fig. 2. The details of shift register tap connections for this type of code are shown in Fig. 3. The blocks in the block diagrams in Fig. 1 through 3 will be described in the following paragraphs.

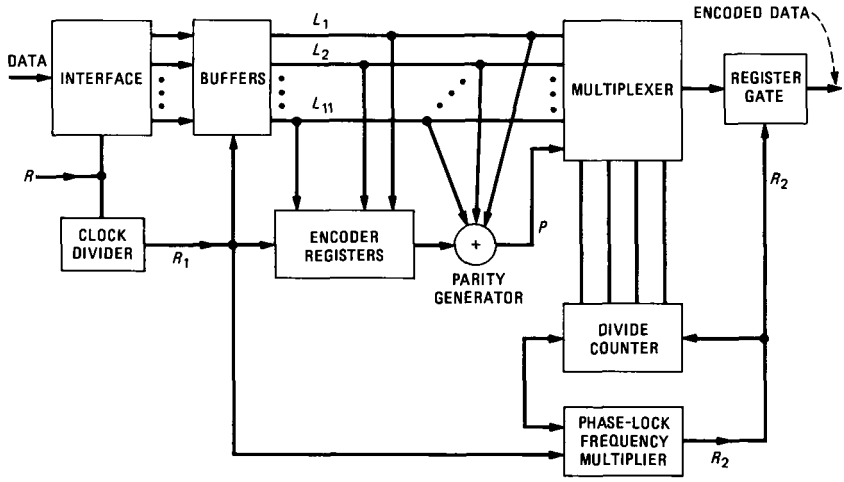


Fig. 1—Block diagram of encoder.

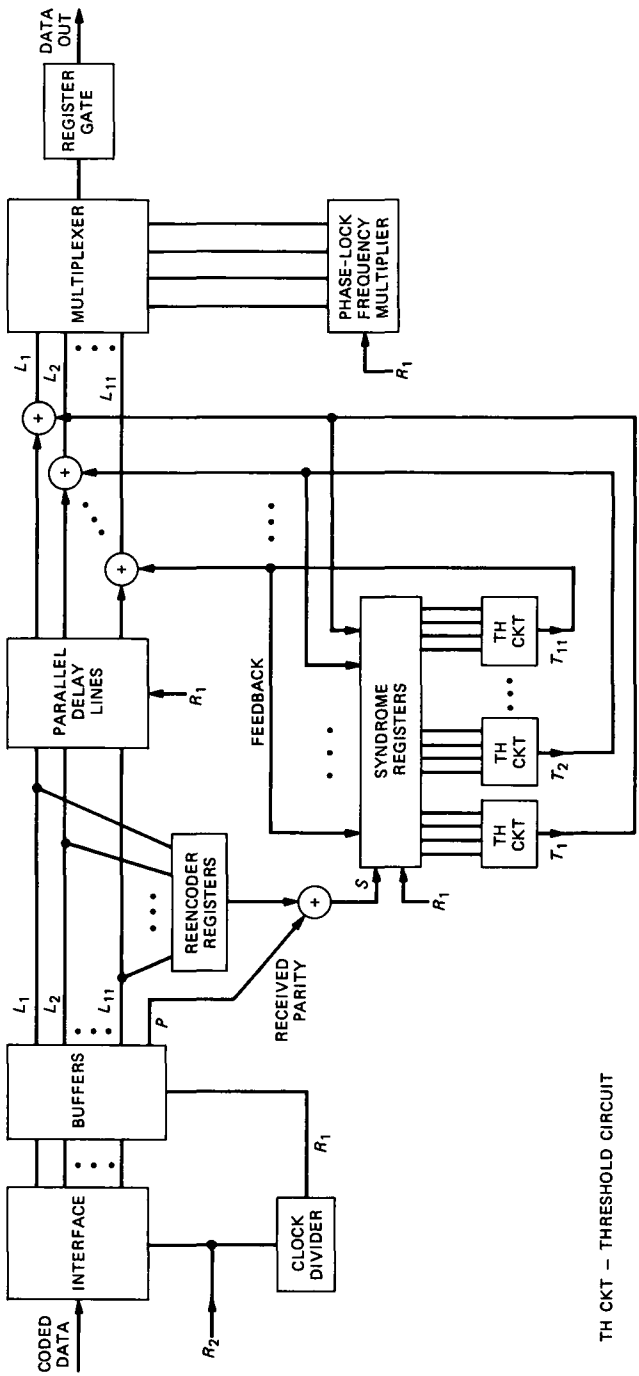


Fig. 2—Block diagram of decoder.

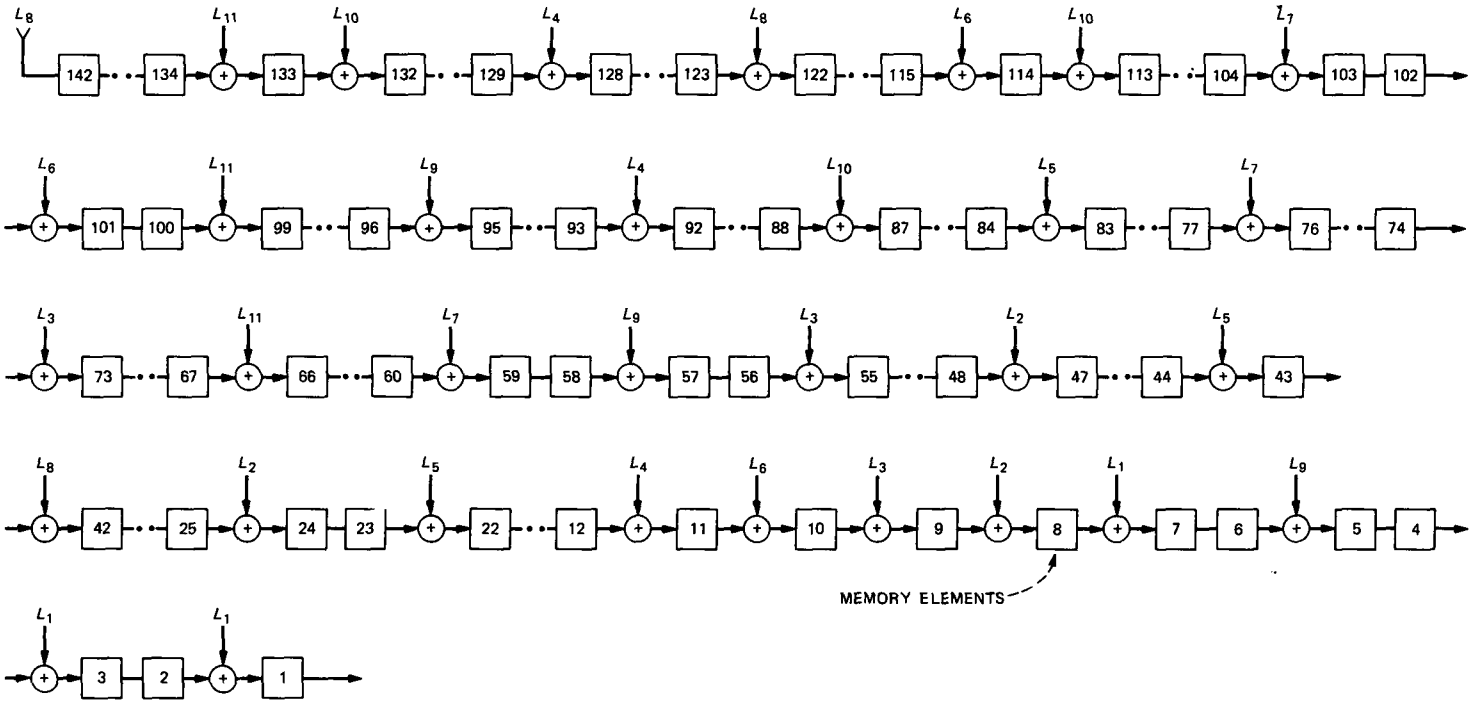


Fig. 3—Tap connection details.

The encoder circuit shown in Fig. 1 takes a serial data stream at a clock rate of R, and, via a Serial-to-Parallel (S/P) converter, the stream is converted to 11 parallel streams, each clocked at the rate $R_1 = R/11$. To realize the S/P converter circuit, an 11-bit serial-in/parallel-out shift register at the rate R was used. The clock conversion is performed by the divide-by-eleven circuit shown in Fig. 1. The timing alignment of the data streams is done through a set of flip-flops clocked at R_1. The encoder shift register consists of 142 delay elements, along with 32 exclusive OR gates placed at locations determined by the code generator polynomials. The delay elements were realized by using 8-bit Transistor–Transistor Logic (TTL) shift register Integrated Circuits (ICs). The generated parity and 11 data bits are then combined through a TTL multiplexer addressed by a 12-bit counter. The multiplexer has to be clocked at $R_2 = (12/11)R$, or more simply, $R_2 = 12R_1$. Hence, a clock multiplier was needed to generate the 12th harmonic of R_1. A digital phase-lock frequency multiplier was designed for this purpose. The circuit is shown in Fig. 4. It consists of a phase detector IC, a low-pass loop filter, a voltage-controlled multivibrator, and a divide-by-twelve counter. To generate the R_2 clock, a harmonic filtering method was tried first and discarded in favor of the phase-lock frequency multiplier. As stated earlier, the clock signal generated by the phase-lock frequency multiplier is used to address the multiplexer (mux) IC and the output of this is reclocked through a single flip-flop by the R_2 clock.

A block diagram of the decoder is shown in Fig. 2. The received coded data at rate $R_2 = (12/11)R$ is passed through an S/P converter to obtain 12 parallel bit streams. To alleviate the encoder/decoder synchronization problem in the experimental model, the R_2 clock generated on the encoder board was hard-wired to the decoder front-end circuit, where a divide-by-twelve counter was used to generate $R_1 = R_2/12$. This clock, in turn, is used to time align the 12 data streams through a set of flip-flops. The reencoder circuit is identical to the encoder shift register shown in Fig. 1. The generated parity and received parity bits are added through an exclusive OR gate to form

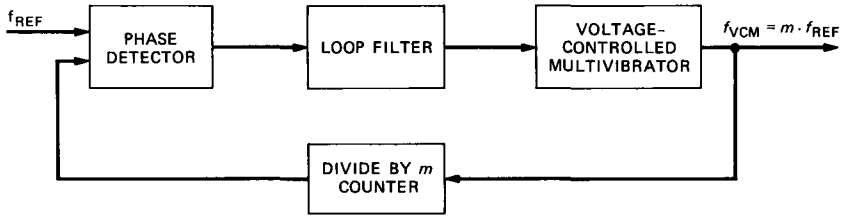


Fig. 4—Phase-lock frequency multiplier.

the syndrome bits. The syndrome register is set up the same way as the reencoder shift register. The syndrome register outputs are fed to a set of 11 threshold circuits. Each threshold circuit, as explained earlier, performs a majority logic vote; that is, if more than half the inputs to each circuit are binary ones, the output of that circuit will be a one, otherwise it will output a zero. Each binary one at the threshold circuit output indicates an error on a particular data line being checked by that threshold circuit. Because the code implemented here is a double error-correcting code, there are four inputs to each threshold circuit that operate on the orthogonal check bit set.

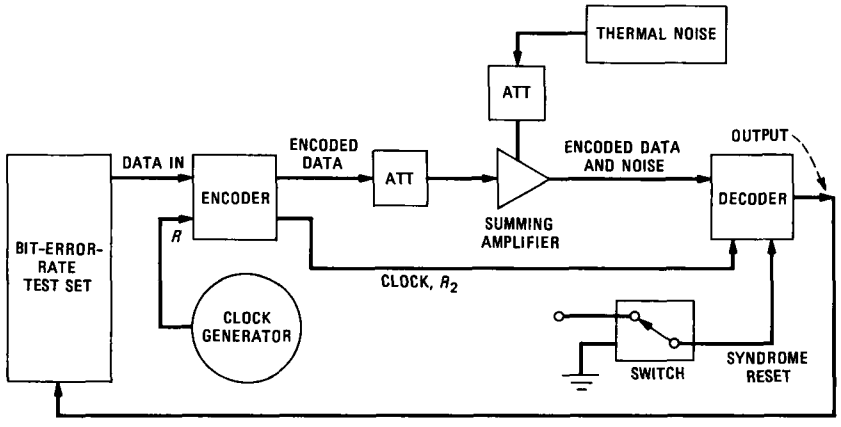
To have the error correction correctly performed, the 12 data lines have to be delayed by one syndrome register length. This can be done by using two RAMs in parallel; however, in this experimental model we used a set of shift register ICs, each containing a 128-bit delay and clocked at R_1 to acquire the delay needed. The error indicators are then modulo-2 added to the proper data bits, and the corrected data lines are multiplexed by a similar approach, as explained in the encoder circuit description. Again, a phase-lock frequency multiplier is used to provide the information clock rate at the decoder output. The multiplexer output is reclocked through a single flip-flop at the clock rate R. In addition, the error indicators are used to remove the effect of corrected errors from the check bits entering the syndrome register.

The fact that the main encoding/decoding operations here are done at a relatively low speed, because of the input serial-to-parallel conversions, makes this type of codec attractive for high-speed data transmission. Next we discuss the test procedure.

II. TESTING

To check the performance of the encoder and decoder, a test drawer was designed and built. The test drawer consisted of the encoder/decoder circuits, a thermal noise source, a summing amplifier, two attenuators, and a switch, as shown in Fig. 5. Functionally, this test drawer was to measure the error rate at the input/output of the codec.

The BER test set used here produces a random bit stream, representing the information bits, which is then encoded through the encoder. Noise is then added to the encoded signal before it enters the decoder. In order to measure both the input and output error probabilities with the same BER test set, we used the following method. The syndrome register flip-flops are set during the normal course of error correction and the decoder output stream closely resembles the encoder input data stream. However, if we reset the decoder syndrome register, the decoder error-correcting function is blocked. Consequently, the decoder outputs the unmodified noisy bit stream. There-



ATT – ATTENUATOR

Fig. 5—Block diagram of test setup.

fore, the BER test set will measure the input error probability. The syndrome register resetting operation was done by the switch shown in Fig. 5. The noise generator was a standard thermal noise source. The step attenuators were to vary the signal and noise power in order to display different error rates at the input.

The test was performed at an input rate of 10 Mb/s. However, because of the serial-to-parallel operation at the encoder/decoder input, up to a 250-Mb/s data rate can be handled by this codec, using standard TTL integrated circuits.

III. CALCULATED AND MEASURED RESULTS

An approximate expression on the performance of the self-orthogonal convolutional codes for low channel error rates is presented in Ref. 2, and more details can be found in Ref. 3. The result is an asymptotic, upper bound on the bit error probability of the decoded bit. The bound is given by

$$P_b \lesssim \frac{1}{NR_c} \sum_{i=t+1}^{N} \binom{N}{i} p^i (1-p)^{N-i}, \tag{2}$$

where

$\lesssim$ = asymptotically (in N_0)
N_0 = white noise spectral height
P_b = bit error probability after decoding
N = constraint length = 1716
R_c = code rate = 11/12
t = number of bit errors corrected per constraint length = 2
p = input bit error probability.

The expression for P_b in (2) is for any particular decoded bit in the first group in a constraint length, under the assumption either that (1) decoding is *direct* (without feedback syndrome correction) and the immediately preceding constraint span was free of decoder input errors, or (2) decoding is *with feedback* and the immediately preceding constraint span was free of decoder *output* errors. It happens to be valid, in general, only by virtue of the fact that the effects of prior history of the decoder are outweighed by the excess probability—included in eq. (2)—of those triple or higher weight input error patterns that do not cause output errors. This bound for the code implemented here is shown as one of the curves in Fig. 6.

As stated earlier, the set of error indicators can be used as a feedback to clean up the syndrome register. To investigate how much improve-

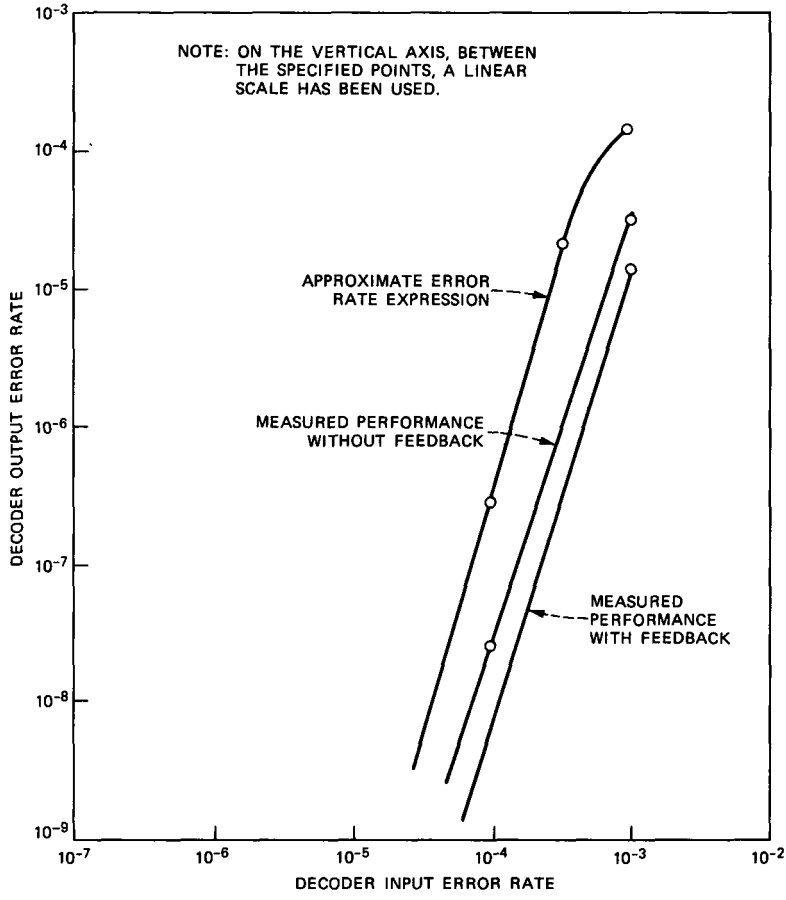


Fig. 6—Performance of a rate 11/12 codec.

ment is achieved by this operation, the output error probability measurements were taken for two cases: with and without syndrome error correction. These results are also shown in Fig. 6. As one can observe, having the set of feedback error indicators connected improves performance. In this case, the output probability of error is improved by almost one half of an order of magnitude at an input error rate of 10^{-4} by having the feedback links in Fig. 2 connected.

As stated earlier, the difference between the approximate bound and the measured performance of this double-error-correcting code can be due to the fact that a self-orthogonal, double-error-correcting, convolutional codec can correct many—triple, quadruple and longer—error patterns. However, the bound in eq. (2) only takes into account double-error correction.

Note that the BER test setup simulated only thermal noise effects. The possible effects of modem implementation and other nonthermal effects on a real channel need to be characterized. If these effects merely increase the decoder input error rate for a given bit energy to noise density (E_b/N_0), but maintain a pure Poisson distribution of those errors, then the output BER versus input BER results of the decoder test will still apply. Conversely, if the other effects cause significant departure from a Poisson arrival of decoder input errors, then the output BER performance versus input BER performance of the decoder will degrade from the test results previously described. In particular, if there is a tendency towards error clustering, a degradation could occur. For instance, clusters of three errors or more in a constraint span that are more frequent than that predicted by a Poisson model could be a source for performance degradation.

For example, in a gray coded 16-QAM modem, even a very small residual phase offset error in detection would significantly increase the probability of 2 bit errors in a 4-bit baud. Then both errors are prone to erroneous decoding if a third input error happens to occur nearby. When the objective is a 10^{-10} output BER, even a very slight effect of this sort can quickly result in an order-of-magnitude degradation in the codec performance.

IV. CONCLUSIONS

This paper has described the design, testing, and performance of a rate 11/12 self-orthogonal convolutional codec that meets the BER performance objective of M-QAM radio systems. The objective was to convert a bit error rate of 10^{-6} to an equivalent error rate of 10^{-10} or better. The measured error rate is well below 10^{-10} at an input error rate of 10^{-6}.

V. ACKNOWLEDGMENT

The author wishes to thank P. Dollard, P. J. McLane, and C.-E. Sundberg for the careful review of the manuscript and their useful suggestions.

REFERENCES

1. W. W. Wu, "New Convolutional Codes—Part I" IEEE Trans. Commun., *COM-23*, No. 9 (September 1975), pp. 942–56.
2. M. Kavehrad, "Implementation of a Self-Orthogonal Convolutional Code Used in Satellite Communications," IEEE, Trans. Elec. Circuits and Syst., *3*, No. 3 (May 1979), pp. 134–8.
3. S. Lin and D. J. Costello, Jr., *Error Control Coding: Fundamentals and Applications*, New Jersey: Prentice Hall, 1983, p. 413.

AUTHOR

Mohsen Kavehrad, B.S. (Electrical Engineering), 1973, Tehran Polytechnic Institute; M.S. (Electrical Engineering), 1975, Worcester Polytechnic Institute; Ph.D. (Electrical Engineering), 1977, Polytechnic Institute of New York; Fairchild Industries, 1977–1978; GTE, 1978–1981; AT&T Bell Laboratories, 1981—. At AT&T Bell Laboratories Mr. Kavehrad is a member of the Communications Methods Research Department at Crawford Hill Laboratory. His research interests are digital communications and computer networks. Technical Editor, IEEE Communications Magazine; Chairman, IEEE Communications Chapter of New Hampshire (1984); Member, IEEE, Sigma Xi.