CONVOLUTIONAL CODING TECHNIQUES FOR CERTAIN QUADRATIC RESIDUAL CODES

RICHARD W.D. BOOTH
MARK A. HERRO
GUSTAVE SOLOMON
TRW Systems
Redondo Beach, California

SUMMARY

The encoding and decoding of the extended Golay code and the (32, 16) quadratic residue code by convolutional coding techniques is discussed. The generation of the extended Golay code is described in detail. A modified version of the Viterbi algorithm for hard and soft decision decoding was implemented, and the simulation results are presented.

1. INTRODUCTION

Recent work by G. Solomon[1] has shown that certain block codes may be encoded and decoded by modified convolutional coding techniques. In particular, all quadratic residue codes can be coded by such techniques. The encoding is performed by first partitioning the codeword coordinates into three sets: the zeroth and infinity coordinates, the coordinates corresponding to the quadratic residues, and the nonresidues. The rate 1/2 convolutional encoder assigns one parity sequence to one set of coordinates and the other parity sequence to the remaining coordinates. The zeroth and infinity coordinates are generated by parity checks on the quadratic residue and the nonresidue coordinates, respectively. Decoding is performed by repeated use of the Viterbi algorithm. This technique is modified somewhat in order to make effective use of the zeroth and infinity coordinates. The hard decision decoding decodes all three-error patterns and some four-error patterns. The extended Golay code and the (32, 16) QR code were simulated, and the results of the simulation are presented.

2. CONSTRUCTION OF GOLAY (24, 12) CODE BY CONVOLUTIONAL TECHNIQUES

The construction of quadratic residue codes by convolutional coding techniques is illustrated in Reference 1. The particular example of the Golay (24, 12) code will be examined here.

The construction technique consists of generating 22 bits of the codeword with an appropriately chosen convolutional coder and adding two parity bits that are a function of the entire information stream. Note that this implies that most of the codeword is generated using the finite memory properties of convolutional codes and the other part of the codeword requires the entire information sequence. Consider the 12 information bits to be encoded: $i_0, i_1, i_2, ..., i_{11}$. Consider i_0 and $i_1, i_2, ..., i_{11}$. separately. Construct the following (extended) information sequence $i_1, i_2, i_3, ..., i_{10}, i_{11}, i_1, i_2, i_3$ where the first 3 bits have been repeated at the end of the information sequence. It will be shown that the following convolutional encoding structure and the appropriate parity bits will generate a permuted Golay (24, 12) code (see Figure 1).

The operation of the coder is as follows. The first four information bits of the extended information sequence are shifted into the encoder and then the parity sequences $b_0, b_1, ..., b_{10}$ and $c_0, c_1, ..., c_{10}$ are computed. Note that the machine starts and ends in the same state i_1, i_2, i_3. Finally, the parity bit s $c_{11} = \sum c_i$ and $b_{11} = \sum b_i$ are computed. The information bit i_0 complements the b_i sequence.

The relationship between this coding structure and the Golay (24, 12) code can be investigated. The cyclic portion of the (24, 12) code is considered first. Factor

$$(x^{23} + 1) = (x+ 1)\, g(x)\, f(x)$$

where

$$f(x) = x^{11} + x^{10} + x^6 + x^5 + x^4 + x^2 + 1$$

and

$$g(x) = x^{11} + x^9 + x^7 + x^6 + x^5 + x + 1.$$

The polynomial corresponding to the additive recursion rule is chosen to be $(x+ 1)f(x)$. Note that if $f(\alpha a) = 0$, then $T_r(\alpha) = 1$. Now every Golay codeword is of the form[2]

$$(d_o + Tr\, cx;\ x = \beta^i)$$

where $d_o \in GF(2)$, $c \in GF(2^{11})$, and $Tr(x) = \sum_{i=0}^{10} x^{2^i}$ and β a root of $f(x)$.

Equivalently, every Golay codeword can be written in the form

$$(d_o + Tr\, cx + i_o\, Tr\, x;\ x = \beta^i)$$

with the constraint that Tr(c) = 0. This specifies the Golay code. The function (d_o + Tr cx; x = β^i) specifies the value of the codeword at each position (coordinate) of the codeword. The coordinates are (β^0, β^1, ..., β^{22}). Now consider the roots of f(x). These roots are (α, α^2, ..., α^{2^j}, $\alpha^{2^{10}}$). The set of exponents ($j_i = 2^i \bmod 23$) of this set of roots of f(x) is (1, 2, 4, 8, 16, 9, 18, 13, 3, 6, 12). The roots of g(x) are ($\gamma, \gamma^2, \ldots, \gamma^{2^{10}}$) where $\gamma = \alpha^5$. The set of exponents of α corresponding to the roots of g(x) is (5, 10, 20, 17, 11, 22, 21, 19, 15, 7, 14). The union of these sets is the set of all exponents of β corresponding to the coordinates, except for 0.

A property of the Golay code is that a cyclic shift of the two sets of exponents corresponds to another Golay codeword. The polynomials Tr cx and Tr c^2x both correspond to Golay codewords. The codeword Tr c^2x can be determined from the codeword Tr cx by noticing that Tr $c^2\beta^{2j} = \sum(c^2\beta^{2j})^{2^i} = \sum(c\beta^j)^{2^i}$ = Tr $c\beta^j$. Thus, at position β^{2j} of the codeword corresponding to Tr c^2x is the value at position β^j of the codeword corresponding to Tr cx. The value d_o for both codewords is the same and remains at coordinate β^0. This transform $\beta^j \rightarrow \beta^{2j}$ corresponds to a cyclic shift within each set of coordinates. For example, $\beta^4 \rightarrow \beta^8$ and $\beta^{11} \rightarrow \beta^{22}$.

Similarly, the transform $\beta \rightarrow \beta^{2^j}$ also corresponds to another Golay codeword. In particular, the transform $\beta \rightarrow \beta^{16}$ corresponds to permuting the set of coordinates corresponding to f(x), yielding the following sequence (16, 3, 2, 9, 6, 4, 18, 12, 8, 13, 1). The set of coordinates corresponding to g(x) is also permuted, giving (11, 15, 10, 22, 7, 20, 21, 14, 17, 19, 5). The polynomials Tr cx and Tr c^{16}x are both generated by Golay codewords with the property that Tr $c^{16}(\beta^{16})^j$ = Tr $c\beta^j$. Thus, $\beta \rightarrow \beta^{16}$, $\beta^{16} \rightarrow \beta^3$, etc.

The convolutional encoding of the Golay codeword is performed by associating the b_i sequence with the nonresidues (except 0). A cyclic shift of the data results in c_i and b_i sequences corresponding to a shift through the two sets of indices, $\beta^{16} \rightarrow \beta^3$, $\beta^6 \rightarrow \beta^7$, $\beta^{10} \rightarrow \beta^{22}$, etc.

Now, let $\bar{a} = (a_0, a_1, \ldots, a_{22})$ be a particular Golay codeword. Choose $\bar{a}$ = (10101110001100000000000). This codeword can be generated by a four-stage convolutional encoder with tap polynomials P(x) = 1 + x + x^3 and Q(x) = 1 + x^2 + x^3 as in Figure 1 and an input data sequence of $i_4 = 1$, $i_k = 0$, $k \neq 4$. The following identification must be made:

$$b_i = a_{2 \cdot 16^i} \qquad i = 0, 1, \ldots, 10$$

$$c_i = a_{5 \cdot 16^i}$$

Thus, $b_0 = a_2$, $b_1 = a_9$, ..., $c_0 = a_5$, $c_1 = a_{11}$, ..., etc. The a_0 bit corresponds to Tr cx; x = 1. Now

$$\text{Tr } c1 = \text{Tr } c \sum_{i=0}^{10} a^{2^i} = \sum_{i=0}^{10} \text{Tr } ca^{2^i}$$

but $\text{Tr } ca^{2^i} = a_j$, where j$\epsilon$ set of quadratic residues for all i. Thus

$$\sum_{i=0}^{10} \text{Tr } ca^{2^i} = \sum_{i=0}^{10} b_i$$

since the b_i's were assigned to the coordinates corresponding to quadratic residues. Note $a_o \rightarrow b_{11}$. Finally, the overall parity on the (23, 11) code can be calculated to give the (24, 12) code

$$a_{23} = \sum_{i=0}^{22} a_i = a_o + \sum_{i \epsilon \text{ quad. res.}} a_i + \sum_{\text{nonresidues}} a_i$$

But

$$\sum_{i \epsilon \, QR} a_i = \sum b_i = a_0$$

and

$$\sum_{i \epsilon \text{ nonresidues}} a_i = \sum c_i$$

Thus, $a_{23} = \sum c_i \rightarrow c_{11}$. Finally, the i_0 Tr x vector just complements the $\bar{b}$ sequence for the convolutional encoding.

In summary, it has been shown that the output sequences of a convolutional encoder are cyclic with a cyclic shift of the input data. The coordinates for a Golay codeword can be divided into three sets, the β^0 and ∞ terms, the quadratic residues, and the nonresidues. Transforms of the type $\beta \rightarrow \beta^{2^j}$ result in a permutation of the elements of each set. The outputs of the rate 1/2 encoder can be associated with each set. Cyclic shifts of the output of the encoder correspond to a shift through each permuted set.

3. ENCODING THE (32, 16) CODE

The encoding of the (32, 16) code is done in an analogous fashion to the Golay code. The cyclic part is considered first. The polynomial corresponding to the additive recursion rule is $f(x) = (x+1)(x^5+x^2+1)(x^5+x^4+x^2+x+1)(x^5+x^3+x^2+x+1)$. Let α = primitive 31st root of 1 and α a root of $x^5 + x^2 + 1$. The set of coordinates corresponding to the quadratic residue is (1, 2, 4, 8, 16, 5, 10, 20, 9, 18, 7, 14, 28, 25, 19). The polynomial representation of the codewords is $(d_o + \mathrm{Tr}\, cx + \mathrm{Tr}\, dx^5 + \mathrm{Tr}\, ex^7)$ where $d_o \in GF(2)$ and $c, d, e \in GF(2^{15})$. Equivalently, the polynomial representation is $d_o + \mathrm{Tr}\, cx + \mathrm{Tr}\, dx^5 + \mathrm{Tr}\, ex^7 + i_o(\mathrm{Tr}\, x + \mathrm{Tr}\, x^5 + \mathrm{Tr}\, x^7)$ where $c + d + e$ has $\mathrm{Tr} = 0$. The transform $x \rightarrow x^7$ yields the polynomial $(d_o + \mathrm{Tr}\, cx^7 + \mathrm{Tr}\, dx^4 + \mathrm{Tr}\, ex^{18}) = (d_o + \mathrm{Tr}\, d^8x + \mathrm{Tr}\, e^2x^5 + \mathrm{Tr}\, cx^7)$ which corresponds to another codeword. This permutes the set of quadratic residues to give the set (1, 7, 18, 2, 14, 5, 4, 28, 10, 8, 25, 20, 16, 18, 9). The codeword $\bar{a} = (a_i)$ with $a_0 = a_1 = a_2 = a_{18} = a_{21} = a_{26} = a_{27} = 1$ and $a_i = 0$ for the remainder is an element of (31, 16) code. (The overall parity is $a_{31} = 1$). The coder used to generate the Golay code can be used to generate the (32, 16) code by using the extended information sequence i_0 and $i_1, i_2, i_3, i_4, \ldots, i_{15}, i_1, i_2, i_3$. The mapping of the convolutional encoder parity sequences is

$$b_i = a_{2 \cdot 7^i} \qquad b_{15} = a_0 = \sum b_i$$

$$i = 0, \ldots, 14$$

$$c_i = a_{-5 \cdot 7^i} \qquad c_{15} = a_{31} = \sum a_i = \sum c_i$$

3. DECODING

The decoding is performed by repeated use of the Viterbi algorithm. The decoding must be done in two sections, one section based on the assumption that $i_0 = 0$ and the other based on $i_0 = 1$. Thus, two decoders are required corresponding to the complemented and uncomplemented $\bar{b}$ sequence. Since the initial encoder state is unknown, this must be estimated also. This is done by assuming an initial starting state and then decoding the received sequence with the Viterbi algorithm. Since the initial state and the final state are the same, the algorithm must force the decoder to end in the same state as the initial state. Since k = 4, there are eight initial states. Thus, the algorithm must be repeated eight times

in each section. The total number of times the Viterbi algorithm must be used is 16. This gives 16 candidate codewords. An alternative procedure might be to recycle the received digits through the decoder until the initial and ending states were the same. The initial state is chosen arbitrarily, and the final state (after 22 received bits) is used in the initial state for the next estimate of the codeword. The two parity bits c_{11} and b_{11} are computed for each of these 16 codewords and compared with the received bits $\hat{c}_{11}$ and $\hat{b}_{11}$. The overall metric for each of the 16 paths is calculated, including the effect of the parities, and the best path is chosen. For the hard decision decoder, the convolutional decoder has a minimum distance of 6 which lead to occasional metric = 3 ties. These cannot be resolved in an arbitrary fashion since the code must be able to correct all three error patterns. Both paths must be saved and the parity bits used to resolve the tie.

4.1 Hard Decision Decoding

Since the (24, 12) code has $d_{min} = 8$, it is required that all three error patterns be correctable. However, the convolutional part of the codeword (b_0,c_0, ..., b_{10},c_{10}) has $d_{min} = 6$. When an error pattern of weight 3 occurs within this part of the codeword, a metric tie will occur, since the minimum distance is only 6. When a tie of weight 3 is encountered, both survivors that lead up to the node which contains the tie are saved. After adding the metric due to the parity bits to both surviving paths, choose the path with the best metric. This technique allows all three error patterns to be corrected. The algorithm is illustrated below.

Step 1: Set initial state of the decoder = 000

Step 2: Assume $i_o = 0$

a) Run the first 22 quantized received numbers through the Viterbi decoder and save the estimate of the information bits ($\hat{i}_1$, $\hat{i}_2$, ..., $\hat{i}_{11}$). Store the metric generated.

b) Form the parity bits:

$$\hat{b}_{11} = \sum_{j=0}^{11} i_j$$

$$\hat{c}_{11} = \sum_{j=1}^{11} i_j$$

c) Calculate the metric due to the parity bits and add to the metric calculated in part (a). Store this cumulative metric.

Step 3: Assume $i_o = 1$.

a) Repeat Step 2.

b) Choose the decoding ($i_o = 0$ or $i_o = 1$) with the best metric from Steps 2 and 3.

Step 4:

a) Repeat Steps 2 and 3 for each of the remaining starting states. Store the metric in each case.

b) Of the eight decodings, pick the one with the best metric.

c) Output the decoded bits.

4.2 Soft Decision Decoding

Eight-level quantization and the log likelihood metric were used for the soft decision decoding. The soft decoding could be done in the same manner as the hard decision case, except for the effect of the parities. In hard decision, the parity digits could contribute a metric of at most 2, and therefore influence only the choice between two competing paths that were "close" in Hamming distance. However, with eight-level quantization, the two parity checks could conceivably contribute a metric as great as 2 x 7 = 14 and therefore influence the choice between two paths that are not close at all. Since the parity checks were intended only to help out on close decisions, the effect of the parities must be reduced as much as possible. This is done by permuting the codeword until the "weakest" received numbers are in the parity positions. By "weakest" it is meant those numbers that are closest to the border between a 0 and a 1 (i.e. , "soft" 0's or 1's). All extended quadratic residue codes are invariant under a two-point permutation such as this one.[3] This permuting procedure is outlined as follows. Let the locations of the two weakest received digits be indexed by j and k. The cyclic portion of the codeword can be cyclically shifted j times to put the j^{th} digit in the 0^{th} location. The k^{th} digit is now in the $(k-j)^{th}$ location. The codeword can now be permuted under the map $e \rightarrow -1/e$ where e is the value of the exponent of the coordinate α. This interchanges the $zero^{th}$ and infinity digits. The $(k-j)^{th}$ digit is now in location $-1/(k-j)$. The cyclic portions can now be cycled and the $-1/(k-j)^{th}$ digit put into the $zero^{th}$ location. Now the digits located at positions j and k of the original received sequence are located at position infinity and zero, respectively. The decoder operation decodes a particular Golay codeword. The inverse map must be

performed on the decoded codeword in order to recover the original codeword. This key property of the QR codes opens the door to the soft decision decoding algorithm:

Step 1: Set initial state of the decoder = 000

Step 2: Assume $i_o = 0$.

a) Find the two weakest numbers in the received codeword and move them into the parity positions.

b) Run the first 22 quantized received numbers through the Viterbi decoder and save the estimate of the information bits ($\hat{i}_1, \hat{i}_2, ..., \hat{i}_{11}$). Store the metric generated.

Step 3: Assume i_o 0 = 1

a) Repeat Step 2.

b) Choose the decoding ($i_o = 0$ or $i_o = 1$) with the best metric from Steps 2 and 3.

Step 4:

a) Repeat Steps 2 and 3 for each of the remaining starting states. Store the metric in each case.

b) Of the eight decodings, pick the one with the best metric.

c) Output the decoded bits.

4.3 Computer Simulation

A computer simulation of the decoding algorithm was performed on a CDC6500 for both the (24, 12) and the (32, 16) codes for soft decision. These curves are plotted and compared with other well known codes in Figure 2. Note that the soft decision decoding buys 1.5 to 2 dB over hard decision decoding as would be expected.

5. FINAL NOTES

The convolutional decoding technique for these codes is cumbersome since the decoding must be performed 16 times. The (48,24) code was studied in Reference 1 and is encodable using a 9-stage convolutional code structure. In spite of these limitations, it is

hoped that the connection made between algebraic block codes and convolutional codes will lead to further investigations into more readily implemented decoders for both[4] block and convolutional codes. The technique of multipliers seems to hold some promise.

REFERENCES

1. G. Solomon, "Unified Algebraic Theory of Block and Convolutional Codes," submitted J. Soc. and Appl. Math.

2. H. F. Mattson and G. Solomon, "A New Treatment of Bose-Chandhuri Codes," J. Soc. and Appl. Math., Vol 9, pp. 654-669, 1961.

3. E. R. Berlekamp, Algebraic Coding Theory, p. 359, McGraw-Hill, i1968.

4. G. Solomon, "Decoding with Multipliers," Princeton Conference on Information Sciences and Systems, March 1974.

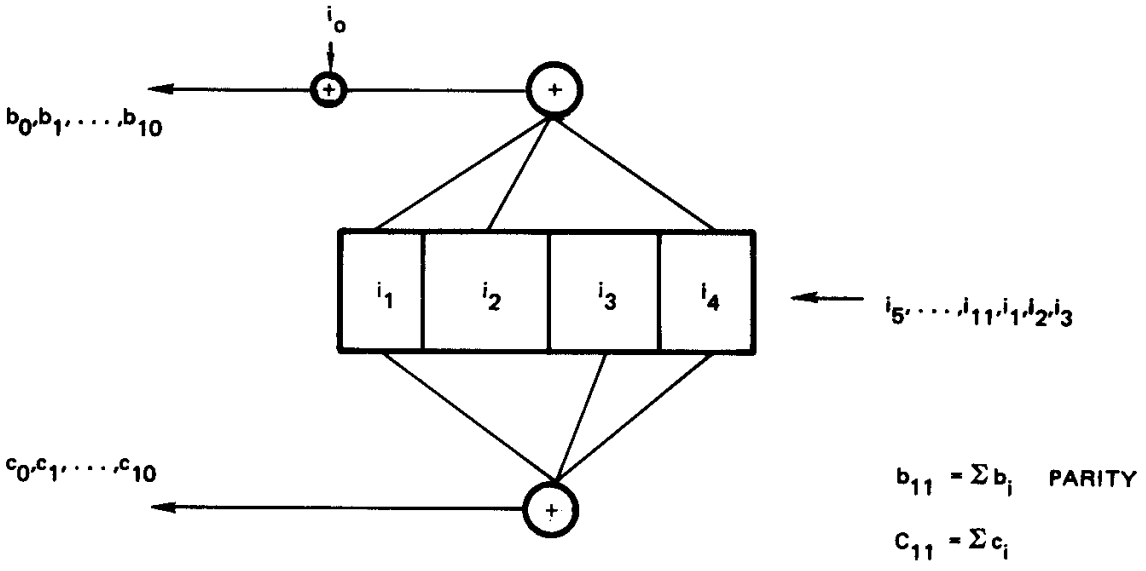

Figure 1. Convolutional Encoder for Golay (24, 12) Code

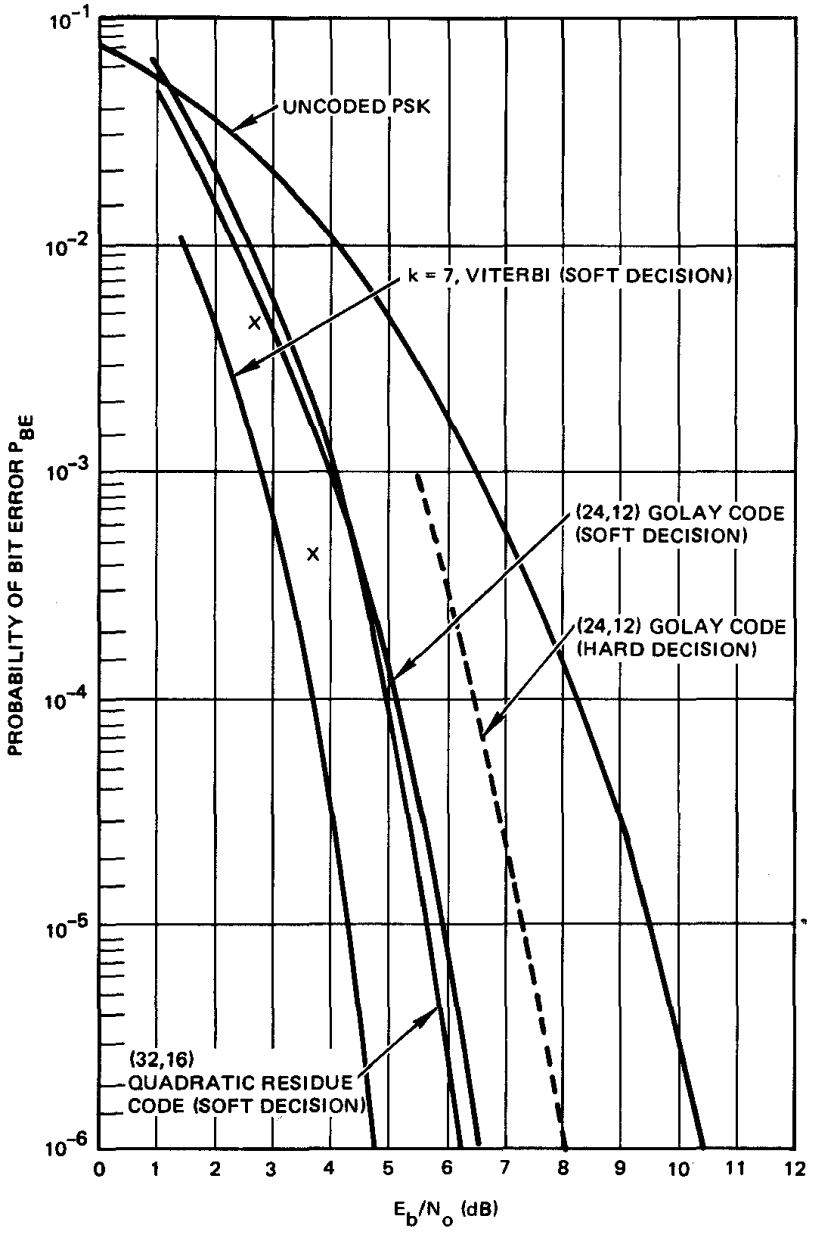

Figure 2. Rate 1/2 Code Choices