Error Correction by Redundant Bits in Constant Amplitude Multi-code CDMA

Hee-keun Song*, Sung-man Kim* ***Student Members***
Bum-gon Kim*, Tong-sok Kim*, Dae-won Ko*, Yong-cheol Kim* ***Regular Members***

ABSTRACT

In this paper, we present two methods of correcting bit errors in constant amplitude multi-code (CAMC) CDMA, which uses the redundant bits only. The first method is a parity-based bit correction with hard-decision, where the received signals despread into a two-dimensional structure with both horizontal parity and vertical parity. Then, an erroneous bit is corrected for each 4 x 4 pattern. The second method is a turbo decoding, which is modified from the decoding of a single parity check product code (SPCPC). Experimental results show that, in the second method, the redundant bits in CAMC can be fully used for the error correction and so they are not really a loss of channel bandwidth. Hence, CAMC provides both a low peak-to-average power ratio and robustness to bit errors.

Key Words : Error Correction, Constant Amplitude, Multi-code, Parity, CDMA

Ⅰ. Introduction

Multi-code CDMA (MC-CDMA)[1]-[4] can provide versatile data rates in wide-band mobile communications by assigning multiple channels to a single user. Since the transmission signal in MC-CDMA is the sum of several random binary signals, the peak-to-average power ratio (PAPR) is large and this loads a burden for a battery-operated mobile terminal. A pre-coding scheme by Wada[3] provides an effective way of reducing the PAPR when the input is either 3 bits or 9 bits. Kim[4] extended Wada's scheme into a constant amplitude multi-code (CAMC) in a recursive structure, which has no restriction in the input size. CAMC can accommodate an input of $M(=3^K)$ bits and generates $N(=4^K)$ bits of constant amplitude.

A drawback of CAMC is the low code rate of $R(=(3/4)^K)$, due to the redundant bits for achieving constant amplitude. The nature of these redundant bits is parity. Considering the fact that current codes built on [4] employ additional channel coding module, utilizing the redundant bits for correcting transmission bit errors to do without additional FEC module would compensate for the low code rate. We show that these redundant bits in CAMC can be *fully* used for the correction of bit errors and so they are not really a loss of channel bandwidth.

In this paper, two methods of utilizing parity bits in CAMC for error correction will be presented. The first method is a parity-based bit correction, following a hard decision of the received signal. In constant amplitude multi-code biorthogonal code keying (MBCK)[5], error correction is based on parity checking and searching for the most likely codeword. The proposed method has advantages over previous schemes: It provides $d_{\min}=2^k$ and so a true error correction is possible, while, in [5], $d_{\min}$ is just 2 so the error de-

※ The authors wish to thank the University Seoul for the Research Grant 2005 to support this study.
* Dept of ECE, University of Seoul (shk0802@uos.ac.kr, sungmankim@uos.ac.kr, mark3552@uos.ackr, tongsok@kt.co.kr, daewon@kti.co.kr, yckim@uos.ac.kr)
논문접수번호 : KICS2006-04-177, 접수일자 : 2006년 4월 25일, 최종논문접수일자 : 2006년 11월 6일

tection-and-search requires heavy computation.

The second method is a turbo decoding of the received signal, which was used for the decoding of single parity check product code (SPCPC). A received CAMC signal vector is despread into four quarter-sized vectors in the form of SPCPC[6]. Kim[7] showed that CAMC for $N=4^k$ is equivalent to k-dimensional SPCPC with a minimum distance of $d_{\min}=2^k$, and he presented a turbo decoding scheme with a performance near the cutoff rate. CAMC is equivalent to SPCPC and that CAMC outperforms a generic SPCPC in error correction.

In Chapter Ⅱ, CAMC signal generation is described. In Chapter Ⅲ and Chapter Ⅳ, the proposed algorithms for the parity-based decoding and the turbo-decoding for CAMC are presented. In Chapter Ⅴ, computer simulation results on BER comparison under AWGN channel is presented, followed by a conclusion in Chapter Ⅵ.

Ⅱ. Generation of CAMC Signal

Fig. 1 shows the encoding process of basic-level CAMC signal vector[4]. In the following vector notations, a superscript at bold-faced letters represents the size of the vector and a subscript of {0,1,2,3}, if any, represents the index of the four quadrant vectors. An input of M bits is divided into $M/3$ groups of three bits each, $[b_0, b_1, b_2]$. At Q^4, a parity bit b_3 is appended and then $[b_0, b_1, b_2, b_3]$ is spread by 4 x 4 Hadamard matrix, $\widetilde{\mathrm{H}}^4$, into four bits of unit amplitude, $[v_0, v_1, v_2, v_3]$. The proof for constant amplitude is in *Lemma 1*.

Continuing this way, finally, the output of three $\mathrm{Q}^{N/4}$ encoders ($3N/4$ bits) and their bit-by-bit parity ($N/4$ bits) are concatenated and then are spread by $\widetilde{\mathrm{H}}^N$, into N bits of unit amplitude, $[v_0, v_1, ..., v_{N-1}]$. Expression for the parity relation and the spreading process for the top-level CAMC vector are shown in (1) and (2).

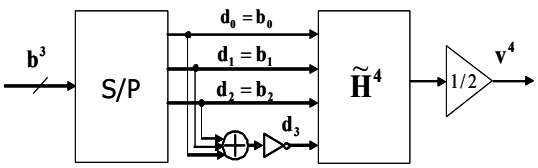


Fig. 1. Generation of the basic level 4-bit CAMC vector

$$\mathbf{v}_3^{N/4} = \overline{\mathbf{v}_0^{N/4} \oplus \mathbf{v}_1^{N/4} \oplus \mathbf{v}_2^{N/4}} \tag{1}$$

$$\mathbf{v}^N = \frac{1}{2} \cdot \left[\mathbf{v}_0^{N/4} | \mathbf{v}_1^{N/4} | \mathbf{v}_2^{N/4} | \mathbf{v}_3^{N/4}\right] \cdot \widetilde{\mathrm{H}}^4 \tag{2}$$

$\widetilde{\mathrm{H}}^N$ is an N x N pseudo-Hadamard matrix where $\mathrm{I}^{N/4}$ is an $N/4$ x $N/4$ identity matrix. When $N=4$, $\widetilde{\mathrm{H}}^N$ is identical to a regular Hadamard matrix H^4.

$$\widetilde{\mathrm{H}}^N = \begin{bmatrix} \mathrm{I}^{N/4} & \mathrm{I}^{N/4} & \mathrm{I}^{N/4} & \mathrm{I}^{N/4} \\ \mathrm{I}^{N/4} & -\mathrm{I}^{N/4} & \mathrm{I}^{N/4} & -\mathrm{I}^{N/4} \\ \mathrm{I}^{N/4} & \mathrm{I}^{N/4} & -\mathrm{I}^{N/4} & -\mathrm{I}^{N/4} \\ \mathrm{I}^{N/4} & -\mathrm{I}^{N/4} & -\mathrm{I}^{N/4} & \mathrm{I}^{N/4} \end{bmatrix}$$

Lemma 1. Each element of $\mathbf{v}^N$ is either (+1) or (-1)

Proof) Rewriting (2) for $\mathbf{v}^4 = [v_0, v_1, v_2, v_3]$ (basic level encoding) produces:

$$\begin{aligned} v_0 &= (d_0 + d_1 + d_2 + d_3)/2 \\ v_1 &= (d_0 - d_1 + d_2 - d_3)/2 \\ v_2 &= (d_0 + d_1 - d_2 - d_3)/2 \\ v_3 &= (d_0 - d_1 - d_2 + d_3)/2 \end{aligned}$$

Equation (1) produces "3-to-1" distribution among the four bits, (d_0, d_1, d_2, d_3): three of them have the same value and the rest has the opposite value (*e.g.* [1 -1 1 1] or [1 -1 -1 -1]). v_0 is bound to be either $(+1/-1)$ since it is the sum of all four bits. Second, v_1 is the difference between $(d_0 + d_2)$ and $(d_1 + d_3)$. If d_0 and d_2 are the same, then $d_1 + d_3 = 0$. Hence v_1 is the sum of two identical bits and is equal to either $(+1/-1)$. If d_0 and d_2 are not the same, then $d_0 + d_2 = 0$ and $v_1 = -(d_1 + d_3)/2 = (+1/-1)$. In a similar way, we can show that v_2 and v_3 are also $(+1/-1)$. The proof for the top-level vector $\mathbf{v}^N$ is similar since the "3-to-1" distribution also holds among the corresponding bit positions in $\left(\mathbf{v}_0^{N/4}, \mathbf{v}_1^{N/4}, \mathbf{v}_2^{N/4}, \mathbf{v}_3^{N/4}\right)$. Hence, each bit in $\mathbf{v}^N$ has a value of $(+1/-1)$. (End of proof)

Ⅲ. Parity-Based Bit Correction

Since the parity bits are intermixed with information bits in the spreading process, there are no *explicit* parity bits in a CAMC vector. Nevertheless, we can extract them by despreading. a *J*-level vector, $\mathbf{v}^N$, can be despread into four (*J*-1)-level vectors, where $\mathbf{v}_3^{N/4}$ is the bit-by-bit parity vector of $\left(\mathbf{v}_0^{N/4}, \mathbf{v}_1^{N/4}, \mathbf{v}_2^{N/4}\right)$.

$$\left[\mathbf{v}_0^{N/4}|\mathbf{v}_1^{N/4}|\mathbf{v}_2^{N/4}|\mathbf{v}_3^{N/4}\right] = \frac{1}{2} \cdot \mathbf{v}^N \cdot \tilde{\mathrm{H}}^N = \frac{1}{2} \cdot \mathbf{v}^N \cdot \begin{bmatrix} \mathrm{I}^{N/4} & \mathrm{I}^{N/4} & \mathrm{I}^{N/4} & \mathrm{I}^{N/4} \\ \mathrm{I}^{N/4} & -\mathrm{I}^{N/4} & \mathrm{I}^{N/4} & -\mathrm{I}^{N/4} \\ \mathrm{I}^{N/4} & \mathrm{I}^{N/4} & -\mathrm{I}^{N/4} & -\mathrm{I}^{N/4} \\ \mathrm{I}^{N/4} & -\mathrm{I}^{N/4} & -\mathrm{I}^{N/4} & \mathrm{I}^{N/4} \end{bmatrix} \tag{3}$$

$$\mathrm{u}_3^{N/4} = \overline{\mathrm{u}_0^{N/4} \oplus \mathrm{u}_1^{N/4} \oplus \mathrm{u}_2^{N/4}} \tag{4}$$

In addition to the explicit parity, an *implicit* parity relation also holds in a CAMC vector as follows:

Lemma 2. The last segment of a CAMC vector is a bit-by-bit parity vector of the preceding 3 segments.

$$\mathrm{u}_3^{N/4} = \overline{\mathrm{u}_0^{N/4} \oplus \mathrm{u}_1^{N/4} \oplus \mathrm{u}_2^{N/4}} \tag{5}$$

Proof. Let $\mathbf{v}^N$ be represented as a concatenation of 4 segments $\left(\mathbf{u}_0^{N/4}, \mathbf{u}_1^{N/4}, \mathbf{u}_2^{N/4}, \mathbf{u}_3^{N/4}\right)$. Substitution of $\mathbf{v}^N$ by $\left[\mathbf{u}_0^{N/4}|\mathbf{u}_1^{N/4}|\mathbf{u}_2^{N/4}|\mathbf{u}_3^{N/4}\right]$ in (3) produces

$$\mathbf{v}_0^{N/4} = \frac{1}{2}(\mathbf{u}_0^{N/4} + \mathbf{u}_1^{N/4} + \mathbf{u}_2^{N/4} + \mathbf{u}_3^{N/4}) \tag{6}$$

All the elements of $\mathbf{u}_0^{N/4}, \mathbf{u}_1^{N/4}, \mathbf{u}_2^{N/4}, \mathbf{u}_3^{N/4}$ and $\mathbf{v}_0^{N/4}$ are $(+1/-1)$. The corresponding bits in $\left(\mathbf{u}_0^{N/4}, \mathbf{u}_1^{N/4}, \mathbf{u}_2^{N/4}, \mathbf{u}_3^{N/4}\right)$ should have "3-to-1" distribution: Three of them are the same and the rest is the opposite (*e.g.* [-1 +1 -1 -1]). Otherwise, the elements of $\mathbf{v}_0^{N/4}$ cannot have a value of $(+1/-1)$ since it is half the sum of $\mathbf{u}_0^{N/4}, \mathbf{u}_1^{N/4}, \mathbf{u}_2^{N/4}$ and $\mathbf{u}_3^{N/4}$. The "3-to-1" distribution among $\mathbf{u}_0^{N/4}, \mathbf{u}_1^{N/4}, \mathbf{u}_2^{N/4}$ and $\mathbf{u}_3^{N/4}$ means that a bit-by-bit odd parity relation holds among them.

(End of proof)

Now we describe the details of error correction. Define $\mathbf{W}^N$ as a 4 x (*N*/4) matrix, the rows of which are the 4 quadrant vectors, $\left(\mathbf{v}_0^{N/4}, \mathbf{v}_1^{N/4}, \mathbf{v}_2^{N/4}, \mathbf{v}_3^{N/4}\right)$. The fourth row is the vertical parity and the last segment of each row is the horizontal parity. *i.e.*, $\mathbf{W}^N$ is in the form of SPCPC with both vertical and horizontal parity. Details of error correction are in *Step 1* ~ *Step 4*. Each of error-corrected $\left(\mathbf{v}_0^{N/4}, \mathbf{v}_1^{N/4}, \mathbf{v}_2^{N/4}\right)$ is recursively despread into four (*J*-2)-level vectors, $\left[\mathbf{v}_0^{N/16}|\mathbf{v}_1^{N/16}|\mathbf{v}_2^{N/16}|\mathbf{v}_3^{N/16}\right]$. Note that the fourth vector $\mathbf{v}_3^{N/4}$ cannot be despread into genuine lower level vectors since it has been generated not by spreading some lower level vectors, but just by a parity operation. This process continues until error correction is finally performed among four basic-level vectors. An example for *N*=16 is shown in Fig 2. A single bit error in a 16-bit Rx vector is diffused into 3 bits in the despreading process. Nevertheless, these erroneous bits get corrected into a clean $\mathbf{W}^{16}$ by vertical and horizontal parity checking.

Step 1. Check vertical parity among the 4 bits for each column of $\mathbf{W}^N$.

$$\mathrm{P_V}(i) = \begin{cases} 1, & \text{if } \mathbf{v}_3^{N/4}(i) = \overline{\mathbf{v}_0^{N/4}(i) \oplus \mathbf{v}_1^{N/4}(i) \oplus \mathbf{v}_2^{N/4}(i)} \\ 0, & \text{otherwise} \end{cases}, i = 0, 1, \cdots, \left(\frac{N}{4} - 1\right) \tag{7}$$

Step 2. Check horizontal parity in (5) among the 4 segments of the first row.

$$\mathrm{P_H}(j) = \begin{cases} 1, & \text{if } \mathbf{u}_3^{N/16}(j) = \overline{\mathbf{u}_0^{N/16}(j) \oplus \mathbf{u}_1^{N/16}(j) \oplus \mathbf{u}_2^{N/16}(j)} \\ 0, & \text{otherwise} \end{cases}, j = 0, 1, \cdots, \left(\frac{N}{16} - 1\right) \tag{8}$$

Step 3. Do from i=0 to (*N*/4-1)

/* Correct a bit error for each 4 x 4 pattern */

If both $\mathrm{P_V}(i)$ and $\mathrm{P_H}(i \bmod N/16)$ are 0, then replace the suspicious bit by the value inducible from the horizontal parity.

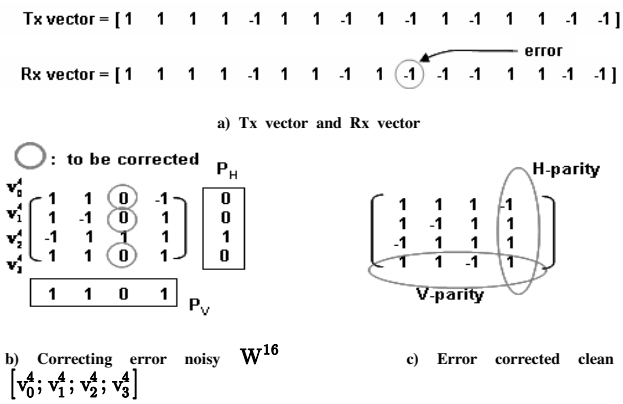


Fig. 2. An illustration of parity-based bit correction

Step 4. Repeat *Step 2* and *Step 3* for the second/third/fourth row of $\mathbf{W}^{N}$.

Ⅳ. Turbo Decoding for CAMC

4-1. Iterative Decoding of SPCPC

Hagenauer[8] developed a soft input, soft output-based decoding for a multi-dimensional product code. Rankin[7] extended the works of [8] and proposed an iterative decoding algorithm for SPCPC. In the encoder, a parity bit is appended to each of (n-1)-bit sequence along all the dimensions of an interleaved hypercube of $(n\text{-}1)^Q$ information bits. The encoded output of n^Q bits, is a Q-dimensional product code with a code rate of $(1\text{-}1/n)^Q$. The code rate of a K-level CAMC is the same as that of K-dimensional SPCPC with n=4.

The LLR, $L_q(X_k)$, for the k-th bit in the q-th dimension is iteratively refined by exchanging the extrinsic information between dimensions. LLR consists of three terms: the channel reliability which is proportional to the signal strength, the a priori information (API, $A_q(X_k)$), and the extrinsic information (EI, $E_q(X_k)$). $\mathbf{X}=[X_0, X_1, ..., X_{N-1}]$ is the input vector and $\mathbf{Y}=[Y_0, Y_1, ..., Y_{N-1}]$ is the received signal through AWGN channel.

$$L_q(X_k)=\log\frac{\Pr\{X_k=+1|\mathbf{Y}\}}{\Pr\{X_k=-1|\mathbf{Y}\}}=\frac{2}{\sigma^2}Y_k+E_q(X_k)+A_q(X_k) \tag{9}$$

$$E_q(X_k)=2\tanh^{-1}\left(\prod_{j=0,\,j\neq k}^{N-1}\tanh\left(\frac{A_q(X_k)+2/\sigma^2 Y_j}{2}\right)\right) \tag{10}$$

$$A_q(X_k)=\sum_{i=1,i\neq q}^{Q}E_i(X_k) \tag{11}$$

4-2. Proposed Turbo-Decoding

The parities in CAMC are different from those in SPCPC. Since an explicit parity relation holds only among four quadrant vectors at the same level, the parity relation among four J-level CAMC vectors holds only in the context of the J-th dimension of SPCPC. For parities in other dimensions, we need to spread/despread CAMC vectors. Likewise, EI and API associated with J-level CAMC vectors are valid only in the J-th dimension. Hence, when EI is exchanged between dimensions, it needs to be spread/despread to fit into the four quadrant structure in the corresponding dimensions.

Fig. 3 illustrates the block diagram for the turbo-decoding of CAMC with N=64, which is equivalent to 3-dimensional SPCPC. **S(.)** and **D(.)** represent the spreading process and the despreading process, respectively. Expressions for LLR ($L_q(X_k^q)$), API ($A_q(X_k^q)$) and EI ($E_q(X_k^q)$) for CAMC decoding are shown in (12), (13) and (14). X_k^q is the k-th bit of the vector reconfigured into the q-th dimension. $[X_0^q, X_1^q, ..., X_{N-1}^q]$ and $[Y_0^q, Y_1^q, ..., Y_{N-1}^q]$, ($N$=$4^q$) is an input vector and

the received vector which are reconfigured into the q-th dimension. The final decision on, X_k^Q, is obtained by hard-limiting of the top-level LLR.

$$L_q(X_k^q) = \frac{2}{\sigma^2} Y_k^q + E_q(X_k^q) + A_q(X_k^q) \tag{12}$$

$$E_q(X_k^q) = 2\tanh^{-1}\left(\prod_{j=0,\, j\neq k}^{N-1} \tanh\left(\frac{A_q(X_k^q) + 2/\sigma^2 Y_j^q}{2} \right) \right) \tag{13}$$

$$A_q(X_k^q) = \sum_{i=1}^{q-1} \mathcal{S}(E_i(X_k^q)) + \sum_{i=q+1}^{Q} \mathcal{D}(E_i(X_k^q)) \tag{14}$$

V. Computer Simulation Results

We tested the performance of CAMC for the two proposed methods, by computer simulation. In the simulation, BPSK modulated signal is transmitted through a binary-input AWGN channel with varying E_b/N_0. After correctible errors are removed from the received signal, we despread it back to the original information bits, for which BER is measured.

In the test for the parity-based bit correction method (labeled as CAMC-P), the received signal is demodulated and hard-limited into a binary value. Then, correctible errors are removed and we compared the BER of CAMC-P, with CAMC-None (CAMC without any error correction) in Fig. 4. BER comparison with MBCK-P (MBCK with parity-based error correction)[5] is presented for N=16, where MBCK and CAMC have equal code rates. For N=16, CAMC-P and MBCK-P appear to have equal performance. The gain in E_b/N_0 obtained by CAMC-P, for BER of 10^{-5}, is 1.4 dB (N=256) and 0.7 dB (N=64), compared to CAMC-None.

In the test for the turbo-decoded CAMC (labeled as CAMC-T), the BER performance is compared in two ways. First, we compared it with that of SPCPC (n=4), from 2-dimension to 4-dimension. The code rates are R_2=9/16 (2-d), R_3=27/64 (3-d) and R_4=81/256 (4-d). Second, the performance of CAMC-T is compared with the cutoff rate[9], which represents a practical upper limit of the transmission rate.

BER for CAMC-T and SPCPC is shown in Fig. 5. For BER of 10^{-5}, the E_b/N_0 in CAMC-T is 1.4 dB lower (4-d) and 1.3 dB lower (3-d) than that in SPCPC. CAMC outperforms SPCPC when the dimension is larger than two. This is an expected result since the error correction in CAMC is boosted by the processing gain. On the other hand, the fourth quadrant vector of CAMC is not decomposable into lower level quadrants and so CAMC has fewer parity bits than SPCPC. This accounts for why the performance of CAMC is lower than SPCPC in low dimensions.

In Fig. 6, the performance of turbo-decoded CAMC is compared with the cutoff rate. Comparing E_b/N_0 at 10^{-5} for CAMC (0.4 dB for 4-d) to the cutoff rate (2.0 dB for R_4=81/256), CAMC is 1.6 dB better than the cutoff rate. Hence, the redundant bits in CAMC with turbo-decoding are fully utilized for error correction.

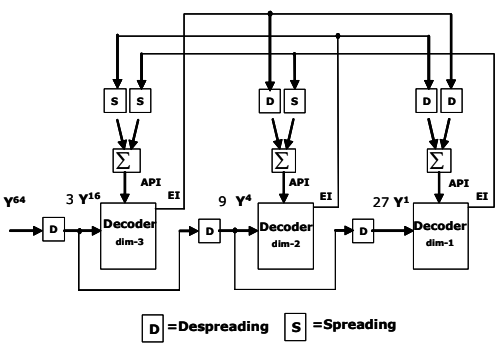


Fig. 3. Block diagram of CAMC decoding for N=64

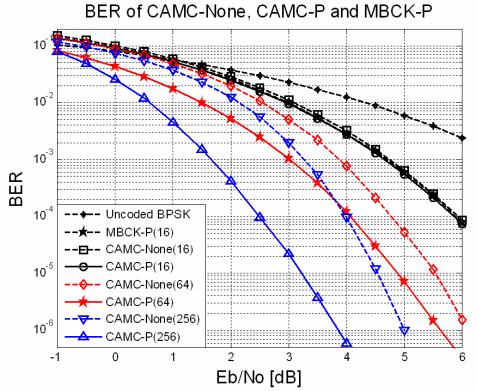


Fig. 4 BER of CAMC-P with CAMC-None and MBCK-P

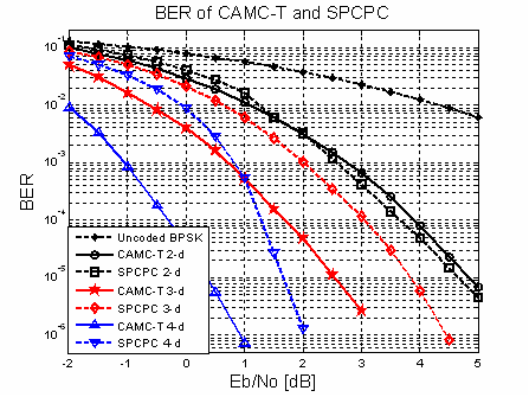


Fig. 5. BER of CAMC-T and SPCPC (2-d, 3-d and 4-d)

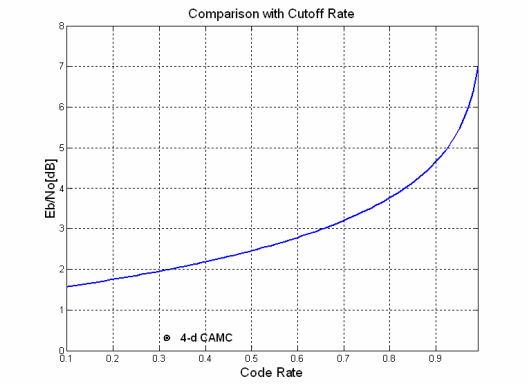


Fig. 6. Cutoff rate (solid line) and 4-d CAMC (circle)

Ⅵ. Conclusions

We presented two methods of correcting bit errors in CAMC, by using the redundant bits only. The first method is a parity-based bit correction, which makes use of the two-dimensional parity structure of the decomposed CAMC signal. We showed that a decomposed CAMC vector is provided with both explicit vertical parity and implicit horizontal parity. Though the improvement in BER is not that high, this method is simpler and shows better or equal performance, when compared to other number-crunching methods.

The second method is a turbo decoding, which is similar to the one used for the decoding of SPCPC. Though the computational complexity of this method is higher than the first method, the performance of error correction reaches the limit of the cutoff rate and so the redundant bits are not really a loss of channel bandwidth. Thus, the channel coding module used in the current codes based on [4], can be substituted by the proposed method.

Since CAMC has a good error correction performance with a simple coding scheme, many applications may be found in wide-band mobile communications and in signal storage.

“”

References

〔1〕 I. Chih-Lin and R. Gitlin, “Multi-code CDMA wireless personal communications networks,” *IEEE Proc. ICC*, pp.1060-1064, June, 1995.

〔2〕 D. Kim and V. Bhargava, “Combined multi-dimensional signaling and transmit diversity for high-rate wide-band CDMA,” *IEEE Trans. Commun.,* Vol.50, No.2, pp.262-275, Feb. 2002.

〔3〕 T. Wada *et al*, “A constant amplitude coding for orthogonal multi-code CDMA systems,” *IEICE Trans. Fund,* Vol.E80-A, pp.2477-2484, Dec. 1997.

〔4〕 Y. Kim, “Recursive generation of constant amplitude multi-code DS-CDMA signal,” *IEE Electronics Letters*, Vol.39, No.25, pp.1782-1783, Dec. 2003.

〔5〕 M. Kim, *et al*, “A Multi-code Biorthogonal Code Keying with Constant Amplitude Coding”, *IEICE Trans. Commun.,* Vol. E88-B, pp.2928-2936, Jul. 2005.

〔6〕 Y. Kim, “Constant Amplitude Multi-Code CDMA with Built-in Single Parity Check Product Code,” *IEEE Communications Letters*, pp.4-6, Jan. 2006.

〔7〕 D. Rankin and T. Gulliver, “Single parity check product codes,” *IEEE Trans. Commun.*, Vol.49, No.8, pp.1354-1362, Aug. 2001.

〔8〕 J. Hagenauer, E. Offer and L. Papke, “Iterative decoding of binary block and convolutional codes,” *IEEE Trans. Inform. Theory*, Vol.42, pp.429-445, Mar. 1974.

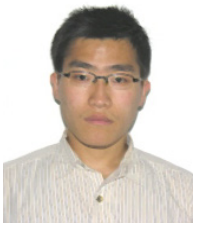
〔9〕 S. Benedetto and E. Biglieri, “Principles of digital communications with wireless applications,” *Kluwer Academic*, Chapter.10, 1999.

송 희 근 (Hee-keun Song) 학생회원

2004년 서울시립대학교 전자전기공학부 졸업

2006년~현재 서울시립대학교 전자전기컴퓨터공학부 석사과정

김 성 만 (Sung-man Kim) 학생회원



2003년 충주대학교 컴퓨터공학과 졸업

2005년~현재 서울시립대학교 전자전기컴퓨터공학부 석사과정

김 범 곤 (Bum-gon Kim) 정회원

1997년 서울시립대학교 전자공학과 졸업

2005년 서울시립대학교 전자전기컴퓨터공학부 석사

2005년~현재 서울시립대학교 전자전기컴퓨터공학부 박사과정

김 동 석 (Tong-sok Kim) 정회원

1982년 전북대학교 전자공학과 졸업

1990년 전북대학교 전자공학과 석사

1998년~현재 서울시립대학교 전자전기컴퓨터공학부 박사과정

1984년~1988년 ㈜인켈

1990년~현재 KT 연구원

고 대 원 (Dae-won Ko) 정회원

1983년 성균관대학교 전자공학과 졸업

1997년 서울시립대학교 제어계측공학과 석사

1998년~현재 서울시립대학교 제어계측공학과 박사과정

1995년 정보통신기술사

1986년~1989년 금성반도체

1989년~현재 ㈜PIVOTEC

김 용 철 (Yong-cheol Kim) 정회원

1981년 서울대학교 전자공학과 졸업

1983년 KAIST 전기전자공학과 석사

1993년 USC 박사

1983년~1986년 금성연구소

1993년~1996년 LG이노텍연구소 전문팀장

1996~현재 서울시립대학교 전자전기컴퓨터공학부 교수