Systematic Redundant Residue Number System Codes: Analytical Upper Bound and Iterative Decoding Performance Over AWGN and Rayleigh Channels

T. H. Liew, Lie-Liang Yang, *Senior Member, IEEE*, and Lajos Hanzo, *Fellow, IEEE*

***Abstract*—The novel family of redundant residue number system (RRNS) codes is studied. RRNS codes constitute maximum–minimum distance block codes, exhibiting identical distance properties to Reed–Solomon codes. Binary to RRNS symbol-mapping methods are proposed, in order to implement both systematic and nonsystematic RRNS codes. Furthermore, the upper-bound performance of systematic RRNS codes is investigated, when maximum-likelihood (ML) soft decoding is invoked. The classic Chase algorithm achieving near-ML soft decoding is introduced for the first time for RRNS codes, in order to decrease the complexity of the ML soft decoding. Furthermore, the modified Chase algorithm is employed to accept soft inputs, as well as to provide soft outputs, assisting in the turbo decoding of RRNS codes by using the soft-input/soft-output Chase algorithm.**

***Index Terms*—Redundant residue number system (RRNS), residue number system (RNS), turbo detection.**

I. Introduction

In RECENT years, the so-called residue number system (RNS) arithmetic has attracted considerable attention for supporting fast arithmetic operations [1]–[8]. The arithmetic advantages accrue from the property that the RNS has the ability to add, subtract, or multiply in parallel, regardless of the size of the numbers involved, without having to generate intermediate carry forward digits, which would slow down the execution of operations [1]. By adding a number of redundant moduli, the so-called redundant RNS (RRNS) is obtained. RRNSs have been studied extensively for the fault-tolerant execution of arithmetic operations in digital filters and in general purpose computers [1]–[8]. A range of further novel applications in the context of RRNS-aided orthogonal frequency-division multiplexing (OFDM) and code-division multiple-access (CDMA) systems was proposed in [9] and [11]–[14].

However, the application of RRNSs in pure channel coding only received limited attention in the conference papers [16], [17]. A further substantial advantage of these RRNS codes accrues from [15], which implies that RRNS codes facilitated the generation of a whole family of different length, different rate codes based on simply concatenating or discarding the required number of redundant residues, without changing the decoding algorithm. The explicit advantage of this technique is that the redundant residues may be added or discarded at any point in a network, for example when the message enters a wireless channel, where typically stronger protection is needed or when it reenters the friendly wireline-based channel, respectively.

By contrast, the standard technique of generating RS codes of any required length is to shorten a longer so-called mother code. The disadvantage of this is that the shortened RS code's decoder always has to use the longer RS mother code's decoder, which is unnecessarily complex. Furthermore, in case of extremely low sub-1-V power supply voltages, a scenario anticipated in future mobile phones for example, not only the channel-contaminated transmitted signal, but also the received low-power signal is prone to internal signal processing errors, for example due to electromagnetic compatibility problems. When using RRNS-based channel codes, some redundant residues may be added for the sake of correcting internal processing errors. Finally, the arithmetic properties of the RRNS allow the decoding of a codeword from any of the residues, provided that the required minimum number of residues has been received, which facilitates the direct parallel processing based decoding of RRNS codes using systolic array based chips.

A coding theoretical approach to error control coding invoking the RRNS has been developed in [6], [7], and [15]. Also, the concepts of Hamming weight, minimum distance, weight distribution, error detection capabilities, and error correction capabilities are investigated. A computationally efficient decoding procedure relying on the so-called projection theory was described for example in [7] and [15] for correcting multiple errors. Recently, the Chase algorithm was applied in the context of RRNS codes [16] in order to perform soft decoding and to exploit the soft channel outputs. In [17], the Chase algorithm was extended for iterative decoding of turbo codes [18], where RRNS codes were employed as the component codes. However, there is no widely available journal paper on the turbo decoding of RRNS codes. Different methods of binary to RRNS symbol mapping schemes are proposed, which can generate both systematic and nonsystematic RRNS codes. Then, the analytical upper-bound performance of systematic RRNS codes is derived and investigated, when maximum-likelihood (ML) soft

Paper approved by S. G. Wilson, the Editor for Multicarrier Modulation of the IEEE Communications Society. Manuscript received June 14, 2001; revised May 11, 2005. This work was supported in part by the European Community, Brussels, Belgium, under the framework of the Phoenix and NEWCOM projects, and in part by the Engineering and Physical Sciences Research Council, Swindon, U.K.

The authors are with the School of Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, U.K. (e-mail: lh@ecs.soton.ac.uk).

Digital Object Identifier 10.1109/TCOMM.2006.876843

decoding is invoked. We then invoke the Chase algorithm [21] for near-ML soft decoding, in order to reduce the complexity of maximum likelihood decoding. The Chase algorithm is then appropriately modified [19] in order to accept soft inputs from the other constituent decoder as well as to provide soft outputs for the other decoder using soft-input/soft-output (SISO) decoding, as it is known in the classic parallel concatenated turbo decoder seen in [22, Fig. 7.3, p. 215]. Consequently, the turbo decoding of RRNS codes is reported for the first time in a journal paper.

The outline of the paper is as follows. A brief introduction to RRNS codes and to their properties is given in Section II. In Section III, we propose two methods of binary to residue mapping, which resulted in a so-called nonsystematic and systematic RRNS code, respectively. In Section IV, the upper bound performance of ML soft-decision RRNS decoding is investigated. In Section V, we apply the classic Chase algorithm in order to implement the reduced-complexity near-ML soft-decision decoding of RRNS codes. Later, we combine the RRNS decoder proposed in [7] with the SISO Chase algorithm [17], [19]–[21], in order to decode the soft channel outputs iteratively, as in turbo decoders. Our simulation results and discussions are given in Section VII. Finally, we conclude in Section VIII.

II. RRNS Code And Properties

A. RRNS

In order to render this paper self-contained, we first give an simple overview of the RRNS and its properties. An RRNS is defined in terms of an n-tuple of pairwise relative prime positive integers, $m_1, m_2, \ldots, m_k, m_{k+1}, \ldots, m_n$, which are referred to as moduli, where moduli $m_1, m_2, \ldots, m_k$ are considered to be the information-bearing nonredundant moduli, while the remaining $(n-k)$ moduli, $m_{k+1}, m_{k+2}, \ldots, m_n$, form the set of redundant moduli that facilitate error detection and correction in the RRNS. The product M_k of the nonredundant moduli represents the so-called dynamic range of the RRNS, which is given by

$$M_k = \prod_{j=1}^{k} m_j. \tag{1}$$

The interval $[0, M_k - 1]$ is also often referred to as the *legitimate range*, while the interval $[M_k, M_n - 1]$ is the *illegitimate range*, where $M_n = \prod_{j=1}^{n} m_j$.

Any positive integer X, where $0 \leq X < M_k$, can be represented by an n-tuple residue sequence given by

$$X \longleftrightarrow (x_1, x_2, \ldots, x_k, x_{k+1}, \ldots, x_n) \tag{2}$$

where the so-called residue x_j is the lowest positive integer remainder of the division X by m_j, which is designated as the residue of $[X \bmod m_j]$ or $|X|_{m_j}$. The positive integer x_j is also termed the jth residue digit of X. An RRNS code is then represented by k nonredundant residues $x_1, \ldots, x_k$ and $n-k$ redundant residues, $x_{k+1}, \ldots, x_n$. As shown in Fig. 1, this is similar to a shortend RS code having k data symbols, $s_1, \ldots, s_k$ and $n-k$ parity symbols, $s_{k+1}, \ldots, s_n$.

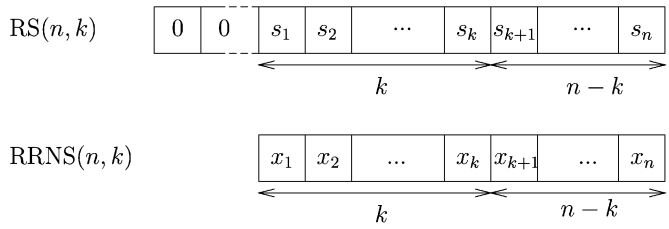


Fig. 1. Example of relationship between RS and RRNS codes.

Given the n-component residue vector, $x_1, x_2, \ldots, x_n$, where $0 \leq x_j < m_j$ for $j = 1, 2, \ldots, n$, the integer X can be constructed from the residues using a procedure known as the Chinese Remainder Theorem (CRT) [1], according to

$$X = \left[\sum_{j=1}^{n} M_j |x_j L_j|_{m_j} \right] \bmod M_n \tag{3}$$

where $M_j = (M_n/m_j)$ and L_j is the so-called multiplicative inverse of $[M_j \bmod m_j]$, which is defined as $|L_j M_j|_{m_j} = 1$.

The so-called mixed radix conversion (MRC) [1] can also be used to replace the CRT, representing the integer X in the form of $X = \sum_{i=1}^{n} a_i \prod_{j=1}^{i-1} m_j$, where $0 \leq a_i < m_i$ and $\prod_{j=1}^{0} m_j = 1$. In the MRC algorithm, the digits $a_1, a_2, \ldots, a_k$ are referred to as the mixed radix information digits, and $a_{k+1}, \ldots, a_n$ will be termed as the mixed radix parity digits.

B. Properties of RRNS Codes

The minimum distance $d_{\min}$ is a fundamental parameter associated with any error control code. In [6] and [7], Krishna *et al.* derived the necessary and sufficient conditions concerning the redundant moduli of an RRNS code in order to exhibit a minimum distance of $d_{\min}$. The minimum distance of an RRNS code is $d_{\min}$, if and only if the product of the redundant moduli satisfies the following relation [6], [7]:

$$\max \left\{ \prod_{i=1}^{d_{\min}} m_{j_i} \right\} > M_{n-k} \geq \max \left\{ \prod_{i=1}^{d_{\min}-1} m_{j_i} \right\} \tag{4}$$

where $M_{n-k} = \prod_{j=k+1}^{n} m_j$ represents the product of the redundant moduli of the code and m_{j_i} is any of the n moduli of the RRNS code, for $1 \leq j_i \leq n$. Similar to Reed–Solomon (RS) codes, the error-correcting capability t of an RRNS code is given by [6]

$$t = \left\lfloor \frac{d_{\min} - 1}{2} \right\rfloor. \tag{5}$$

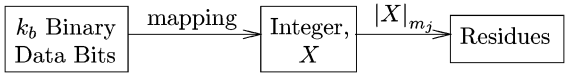


Fig. 2. Nonsystematic encoding procedures.

From (4), the smallest value of M_{n-k} for the RRNS codes to achieve the minimum distance of $d_{\min}$ is given by setting

$$M_{n-k} = \max\left\{\prod_{i=1}^{d_{\min}-1} m_{j_i}\right\}. \tag{6}$$

It can be seen from (6) that the left-hand side inequality of (4) is satisfied trivially. Equation (6) also suggests that an optimal RRNS code, which is associated with the minimum necessary redundant dynamic range of M_{n-k} for achieving a minimum distance of $d_{\min}$, has the largest $(d_{\min}-1)$ number of moduli as the redundant moduli, which results in

$$n = k + d_{\min} - 1. \tag{7}$$

Using the standard coding theoretical terminology, a class of RRNS codes that satisfies (7) is referred to as a maximum distance separable RRNS (MDS-RRNS) code.

III. RRNS Encoder

In Section II, we stated that an RRNS code is given by a set of residues with respect to a predefined set of moduli. Since the moduli and the residues assume positive integers, represented by an arbitrary number of binary bits, RRNS codes are nonbinary codes based on transmitting the residues conveying a number of bits. In this section, we propose two different methods for mapping the binary source bits to the nonbinary RRNS code symbols constituted by the residues, which result in a so-called nonsystematic or a systematic bit-to-RRNS-symbol mapping, respectively.

A. Nonsystematic Encoder

We summarized the nonsystematic encoding process in Fig. 2. The nonsystematic encoder accepts k_b number of binary data bits each time, where $k_b = \lfloor \log_2 M_k \rfloor$ and $\lfloor z \rfloor$ indicates the largest integer smaller than z, while M_k is the dynamic range as defined by (1). The data bits are then mapped to an integer X in the range of $[0, 2^{k_b}-1]$, as seen in Fig. 2. Note that the full dynamic range M_k of the RRNS might not be completely exploited, since typically we have $M_k \neq 2^{k_b}$. Using the moduli in the RRNS, the residues x_j are simply obtained by taking the modulus, as shown in Fig. 2.

Let $k_{b_j} = \lceil \log_2 m_j \rceil$ denote the required number of bits in order to represent the residue x_j in the binary form, where $\lceil z \rceil$ means the smallest integer larger than z and $j = 1, 2, \ldots, n$.

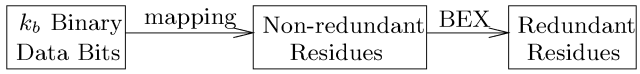


Fig. 3. Systematic encoding procedures.

Then, the code rate of the nonsystematic RRNS code can be expressed as

$$R_c = k_b / \sum_{j=1}^{n} k_{b_j}. \tag{8}$$

B. Systematic Encoder

Fig. 3 characterizes the systematic encoding process. Unlike in the nonsystematic encoder of Fig. 2, which maps all the data bits to be transmitted to an integer X, in the systematic encoder of Fig. 3 we propose to divide the bit sequence to be encoded into shorter groups of bits representing each nonbinary residue symbol separately by a number of bits. Again, since the moduli $m_j, j = 1, 2, \ldots, n$ are, in general, not an integer power of two, upon representing the corresponding residues of these moduli unambiguously with the aid of a given number of bits, the created legitimate range of residues is not fully exploited. This imperfect range match results in the reduction of the associated effective coding rate, without increasing the error correcting power of the code. An alternative is to actually use one bit more than necessary for covering the entire dynamic range of the residue symbols and hence map two different input databit patterns from the second and third column of Table I on to a given residue of the first column, while maintaining as high a Hamming distance between these two patterns as possible. An example of this would be mapping of $00\cdots00$ and $11\cdots11$ to the same residue in Table I. This policy allows us to maintain as high a coding rate as possible without reducing the coding performance, as we will demonstrate. According to this regime, each symbol is then represented by a nonredundant residue x_j according to

$$x_j = \begin{cases} X_j, & \text{if } X_j < m_j \\ 2^{k_{b_j}} - 1 - X_j, & \text{if } X_j \geq m_j \end{cases} \tag{9}$$

where the number of bits k_{b_j} of the jth symbol satisfies

$$\min_{k_{b_j}}\{2^{k_{b_j}}\} \geq m_j \tag{10}$$

and X_j is the integer value of the jth symbol represented by k_{b_j} data bits. Table I shows the associated mapping results and relationships in general terms. Explicitly, the proposed mapping rule implies that each of the residues in the range $[0, 2^{k_{b_j}} - m_j - 1]$ actually corresponds to two different binary represented integer messages, which is defined here as a "2→1 mapping." However, the residues in the range $[2^{k_{b_j}} - m_i, m_j - 1]$ only correspond to one binary represented integer message, hence this assignment is defined as a "1→1 mapping." Furthermore, according to Table I, the number of "2→1 mappings" is $(2^{k_{b_j}} -$

TABLE I
BINARY TO RESIDUE DIGIT MAPPING USING SYSTEMATIC RRNS MAPPING RULE

residue digits	binary represented integer	binary represented integer	
0	$000\cdots00$	$111\cdots11$	
1	$000\cdots01$	$111\cdots10$	$2^{k_{b_j}}-m_j$
2	$000\cdots10$	$111\cdots01$	$2\rightarrow1$
$\cdots$	$\cdots$	$\cdots$	mapping
$2^{k_{b_j}}-m_j-1$	$\cdots\cdots$	$\cdots\cdots$	
$2^{k_{b_j}}-m_j$	$\cdots\cdots$		$2m_j-2^{k_{b_j}}$
$\cdots$	$\cdots$		$1\rightarrow1$
m_j-1	$\cdots\cdots$		mapping

TABLE II
BINARY TO RESIDUE DIGIT MAPPING USING SYSTEMATIC RRNS MAPPING RULE AND MODULUS $m=5$

residue digits	binary represented integer	binary represented integer	mapping
0	000	111	
1	001	110	$2\rightarrow1$
2	010	101	
3	011		$1\rightarrow1$
4	100		

m_j), while the number of "1→1 mappings" is $(2m_j-2^{k_{b_j}})$. For example, modulus 5 is used as one of the moduli in an RRNS, we then have 3 "2→1 mapping" and 2 "1→1 mapping," which is shown in Table II. Based on the defined mapping rule, at the receiver, since the ambiguously represented integers exhibit the maximum possible Hamming distance separation of k_{b_j}, we can calculate the Euclidean distance of both integers from the received integer, in order to determine which was the more likely transmitted integer.

The total number of data bits that the systematic encoder encodes each time becomes $k_b=\sum_{j=1}^{k}k_{b_j}$. Accordingly, as shown in Fig. 3, the data bit sequences to be encoded are mapped to the nonredundant residues directly. Then the so-called base extension (BEX) algorithm can be invoked [23] in order to compute the redundant residues from the known nonredundant ones [15]. Let $k_{b_j}=\lceil\log_2 m_j\rceil, j=k+1,k+2,\ldots,n$ represent the number of binary bits of the redundant symbol j representing the redundant residue x_j. Then, the code rate of the systematic RRNS code can be expressed as

$$R_c=\sum_{i=1}^{k}k_{b_i}/\sum_{j=1}^{n}k_{b_j}. \tag{11}$$

Note that since systematic RRNS codes are capable of exploiting the provided dynamic range of the corresponding RNS more efficiently than the nonsystematic RRNS codes, consequently, for a given set of moduli values and for a given number of nonredundant moduli, the code rate of systematic RRNS codes becomes higher than that of the nonsystematic RRNS codes.

The hard-decision RRNS decoder invoked in this paper was proposed in [7]. The multiple error-correction procedures in [7] are extensions of those in [4] and [8]. In [4] and [8], the algorithms proposed for locating a single residue digit error were based on the so-called modulus projection and MRC. However, the RRNS decoders of [4], [7], and [8] assumed that the output of the demodulator was hard-decision-based binary, and hence, the RRNS decoders previously proposed in [4], [7], and [8] were incapable of exploiting the soft outputs provided by the demodulator at the receiver. By contrast, in our forthcoming sections soft demodulator outputs are assumed and soft decoding using ML decision as well as SISO iterative decoding will be investigated. Let us first derive the upper bound of the systematic RRNS codeword decoding error probability by invoking ML decoding in the next section.

IV. ML DECODING: UPPER BOUND OF SYSTEMATIC RRNS CODEWORD DECODING ERROR PROBABILITY

Let $(m_1,m_2,\ldots,m_k,m_{k+1},\ldots,m_n)$ be the set of moduli used by the RRNS(n,k) code, where $(m_1,m_2,\ldots,m_k)$ are the information moduli and $(m_{k+1},\ldots,m_n)$ are the redundant moduli. As we discussed previously, the code rate of this RRNS(n,k) code is given by (11). According to the systematic mapping of Table I from the binary data bits to residues, the codevector space with $2^{(\sum_{i=1}^{k}K_i)}$ code vectors, where K_i is the number of bits representing residue x_i, can be divided into two subsets according to whether the codewords are subjected to "2→1 mapping," which can be expressed as:

1) $A=\{$codewords not having "$2\rightarrow1$ mapping"$\}$;
2) $B=\{$codewords having "$2\rightarrow1$ mapping"$\}$.

Consequently, according to Table I, the number of codewords in subsets A and B can be expressed as

$$\mathcal{N}(A) = \prod_{i=1}^{k} (2m_i - 2^{k_{b_i}}) \tag{12}$$

$$\mathcal{N}(B) = \prod_{i=1}^{k} 2^{k_{b_i}} - \mathcal{N}(A). \tag{13}$$

Hence, if independent identically distributed (i.i.d.) binary data bits are considered, the probability that the transmitted codeword is from the set A can be expressed as

$$P(A) = \frac{\mathcal{N}(A)}{\mathcal{N}(A)+\mathcal{N}(B)} = \prod_{i=1}^{k} \left(\frac{m_i}{2^{k_{b_i}-1}} - 1 \right) \tag{14}$$

and the probability that the transmitted codeword is from the set B is

$$P(B) = 1 - P(A) = 1 - \prod_{i=1}^{k} \left(\frac{m_i}{2^{k_{b_i}-1}} - 1 \right). \tag{15}$$

Let

$$X \longleftrightarrow (x_1, x_2, \ldots, x_k, x_{k+1}, \ldots, x_n) \tag{16}$$

be the transmitted codeword and x_k be the kth symbol. Then, if X is a codeword in subset A, since the codewords from subset A are not subjected to "2→1 mapping," the codeword decoding error probability satisfies

$$P_E(U, A) < \sum_{j=d_{\min}}^{n} \left(\sum_{Q\binom{n}{j}} \right) \sum_{v=j}^{\sum_{q=1}^{j} k_{b_{l_q}}} \mathcal{M}(j, v) P(j, v) \tag{17}$$

where $Q\binom{n}{j}$ represents that j out of n symbols of the codeword were in error, $\sum_{Q\binom{n}{j}}$ represents all possible selections of j symbols from the codeword, and $\mathcal{M}(j, v)$ represents the total number of events when v bit errors were encountered in j symbols, namely, in the l_1th, l_2th, …, l_jth symbols. $\mathcal{M}(j, v)$ can be expressed as in (18), shown at the bottom of the page. Since the l_1th, l_2th, …, l_jth symbols can be selected from the n codeword symbols in $\binom{n}{j}$ different ways, the total number of these possible different selections is expressed as the $\sum_{Q\binom{n}{j}}$ term in (17). Finally, $P(j, v)$ in (17) represents the probability of the event that the transmitted codeword has v bit errors in j number of symbols, which can be expressed as [24, p.440]

$$P(j, v) = Q(\sqrt{2\gamma_b R_c v}) \tag{19}$$

where γ_b represents the average signal-to-noise ratio (SNR) per bit. Apparently, $P(j, v)$ is independent of j.

If the transmitted codeword X is from the subset B, then the contributions to the upper-bound codeword error probability can be divided into two cases. First, it can be shown that the conditions resulting in 2→1 mapping errors are:

1) errors in the information part of the codeword;
2) errors where there exists at least one but at most k symbols in the information part of the codeword, in which all binary bits of a symbol are in error. The decoder cannot resolve these errors, due to the "2→1 mapping."

Hence, the codewords in the code space can be further divided into two cases according to whether the codewords obey the above conditions. Those obeying the above two conditions are said to belong to *Case 1*; otherwise, they belong to *Case 2*. For the transmitted codeword of (16), the codewords belonging to *Case 1* are

$$\left.\begin{matrix} \overline{x}_1 & x_2 & x_3 & \ldots & x_{k-1} & x_k & x_{k+1} & \ldots & x_n \\ x_1 & \overline{x}_2 & x_3 & \ldots & x_{k-1} & x_k & x_{k+1} & \ldots & x_n \\ \vdots & \vdots & \vdots & \ldots & \vdots & \vdots & \vdots & \ldots & \vdots \\ x_1 & x_2 & x_3 & \ldots & x_{k-1} & \overline{x}_k & x_{k+1} & \ldots & x_n \end{matrix}\right\} \binom{k}{1}$$

$$\left.\begin{matrix} \overline{x}_1 & \overline{x}_2 & x_3 & \ldots & x_{k-1} & x_k & x_{k+1} & \ldots & x_n \\ \overline{x}_1 & x_2 & \overline{x}_3 & \ldots & x_{k-1} & x_k & x_{k+1} & \ldots & x_n \\ \vdots & \vdots & \vdots & \ldots & \vdots & \vdots & \vdots & \ldots & \vdots \\ x_1 & x_2 & x_3 & \ldots & \overline{x}_{k-1} & \overline{x}_k & x_{k+1} & \ldots & x_n \end{matrix}\right\} \binom{k}{2}$$

$$\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots$$

$$\overline{x}_1\ \overline{x}_2\ \overline{x}_3\ \ldots\ \overline{x}_{k-1}\ \overline{x}_k\ x_{k+1}\ \ldots\ x_n \Big\} \binom{k}{k}. \tag{20}$$

The total number of these codewords is $2^k - 1$. Hence, the number of codewords belonging to *Case 2*, except for the transmitted codeword, is $(\prod_{i=1}^{k} 2^{k_{b_i}} - \prod_{i=1}^{k}(2m_i - 2^{k_{b_i}}) - 2^k)$. The contribution from *Case 1*-type codeword errors to the upper bound can be expressed as

$$P_E(U, B1) < \sum_{i=1}^{k} \sum_{Q\binom{k}{i}} Q\left(\sqrt{2\gamma_b R_c \left(\sum_{j=1}^{i} k_{b_{l_i}} \right)} \right). \tag{21}$$

$$\mathcal{M}(j, v) = \underbrace{\sum_{i_1=1}^{\min\{k_{b_{l_1}}, v-j+1\}} \sum_{i_2=1}^{\min\{k_{b_{l_2}}, v-i_1-j+2\}} \cdots \sum_{i_q=1}^{\min\{k_{b_{l_q}}, v-\sum_{p=1}^{q-1} i_p - j+q\}} \cdots \sum_{i_j=1}^{v-\sum_{q=1}^{j-1} i_q}}_{\sum_{q=1}^{j} i_q = v}$$

$$\cdot \binom{k_{b_{l_1}}}{i_1} \binom{k_{b_{l_2}}}{i_2} \cdots \binom{k_{b_{l_q}}}{i_q} \cdots \binom{k_{b_{l_j}}}{i_j} \tag{18}$$

For the *Case 2*-type codeword errors, the contributions to the upper bound can be expressed as

$$P_E(U,B2)=\sum_{j=d_{\min}}^{n}\left(\sum_{Q\binom{n}{j}}\right)\sum_{v=j}^{\sum_{q=1}^{j}\overline{k}_{b_{l_q}}}\overline{\mathcal{M}}(j,v)P(j,v) \tag{22}$$

where we have (23), shown at the bottom of the page, where

$$\overline{k}_{b_{l_q}}=\begin{cases}k_{b_{l_q}}-1, & \text{if } l_q\le k,\\ k_{b_{l_q}}, & \text{if } l_q>k.\end{cases} \tag{24}$$

As before, the probability $P(j,v)$ was given by (10).

Finally, the codeword decoding error probability of $P_E(U)$ can be expressed as

$$P_E(U)<P(A)\cdot P_E(U,A)+P(B)\cdot[P_E(U,B1)+P_E(U,B2)]. \tag{25}$$

The performance loss due to the "2→1 mapping" errors is given by

$$P_{\text{loss}}(U)=P(B)\cdot P_E(U,B1). \tag{26}$$

Note that due to the "2→1 mapping"-induced unidentified errors, the system's performance is limited not only by the minimum distance of the RRNS codes, as indicated in (17) and (22), but also by the residue symbols having relatively short binary representations, as suggested by (21). We note further that the most interesting parameter in the context of ML soft-decision decoding of systematic RRNS codes is the ratio of the codeword decoding error probability due to "2→1 mapping" errors to that due to the erroneous decoding of the RRNS code itself, which is expressed as

$$\rho=\frac{P(B)\cdot P_E(U,B1)}{P(A)\cdot P_E(U,A)+P(B)\cdot P_E(U,B2)}. \tag{27}$$

In order to maintain a good performance, RRNS codes have to satisfy the condition of $\rho\ll 1$. Otherwise, even if powerful RRNS codes are employed, the system's performance will be limited by the relatively high probability of "2→1 mapping" errors.

Let us assume that the "2→1 mapping" errors are contributed mostly by single symbol errors. Then, $P_E(U,B1)$ can be approximated by

$$P_E(U,B1)\approx\sum_{i=1}^{k}Q\left(\sqrt{2\gamma_b R_c k_{b_i}}\right). \tag{28}$$

By referring to (22) and assuming furthermore that most decoding errors have $d_{\min}$ number of erroneous symbols as well as that each erroneous symbol has a single-bit error, then

$$\mathcal{M}(j,v)=\overline{\mathcal{M}}(d_{\min},d_{\min})=\prod_{i=1}^{d_{\min}}k_{b_{l_i}}\le\max\left\{\prod_{i=1}^{d_{\min}}k_{b_{l_i}}\right\}. \tag{29}$$

Consequently, (17) and (22) can be approximated as

$$P_E(U,A)\approx P_E(U,B2)\approx\binom{n}{d_{\min}}\max\left\{\prod_{i=1}^{d_{\min}}k_{b_{l_i}}\right\}P(d_{\min}) \tag{30}$$

where

$$P(d_{\min})=Q(\sqrt{2\gamma_b R_c d_{\min}}). \tag{31}$$

Finally, upon substituting (14), (15), (28), and (30) into (27), it can be shown that ρ can be approximated by

$$\rho\approx\frac{\left[1-\prod_{i=1}^{k}\left(\frac{m_i}{2^{k_{b_i}}-1}-1\right)\right]\cdot\sum_{i=1}^{k}Q\left(\sqrt{2\gamma_b R_c k_{b_i}}\right)}{\binom{n}{d_{\min}}\max\left\{\prod_{i=1}^{d_{\min}}k_{b_{l_i}}\right\}P(d_{\min})}. \tag{32}$$

In summary, in this section, we have derived the upper bound of the codeword decoding error probability for systematic RRNS codes involving ML decision decoding and investigated the performance loss due to "2→1 mapping" errors. However, since the associated decoding complexity increases exponentially with k, simplified soft-decoding algorithms are required for the soft-decoding of RRNS codes for $k>6$. Furthermore, due to the performance limitations of the "2→1 mapping," novel decoding approaches are preferred in order to mitigate or remove the effect of the "2→1 mapping" errors. Hence, in the next section, the reduced complexity but suboptimum Chase algorithm [21] is invoked for near-ML decoding of RRNS codes. Furthermore, a novel iterative RRNS decoding algorithm is proposed, which can substantially reduce the effects of the "2→1 mapping" errors.

$$\overline{\mathcal{M}}(j,v)=\underbrace{\sum_{i_1=1}^{\min\left\{\overline{k}_{b_{l_1}},v-j+1\right\}}\sum_{i_2=1}^{\min\left\{\overline{k}_{b_{l_2}},v-i_1-j+2\right\}}\cdots\sum_{i_q=1}^{\min\left\{\overline{k}_{b_{l_q}},v-\sum_{p=1}^{q-1}i_p-j+q\right\}}\cdots\sum_{i_j=1}^{v-\sum_{q=1}^{j-1}i_q}}_{\sum_{q=1}^{j}i_q=v}\cdot\binom{k_{b_{l_1}}}{i_1}\binom{k_{b_{l_2}}}{i_2}\cdots\binom{k_{b_{l_q}}}{i_q}\cdots\binom{k_{b_{l_j}}}{i_j} \tag{23}$$

V. SISO RRNS Decoder

In the following two sections, all explanations and derivations are based on binary representations, since every bit is decided separately.

A. Soft-Input Decoding Using Chase Algorithm

We consider the transmission of the block coded binary symbols $\{-1, +1\}$ using binary phase-shift keying (BPSK) modulation over an additive white Gaussian noise (AWGN) channel. At the receiver, the demodulator provides the continuous-valued soft-decision-based received signal samples $\underline{y}$ for the RRNS decoder. A ML decoder is capable of finding the codeword that satisfies

$$\min_j \left\{\text{weight}\left(|\underline{y} - \underline{x}_j|^2\right)\right\} \tag{33}$$

where $x_{ji} \in \{-1, +1\}$ are the transmitted legitimate binary coded symbols, and the range of j encompasses all possible legitimate codewords. The decision rule given by (33) is optimum, but the associated computational complexity increases exponentially with the length k of the information part of an (n, k) codeword, and its evaluation becomes prohibitive for block codes with $k > 6$. As a reduced-complexity alternative, the Chase III algorithm [21] was proposed for near-ML decoding of block codes. The algorithm is suboptimum, but it offers substantially reduced complexity in comparison to ML decoding.

The Chase III algorithm can be summarized in the flow chart shown in Fig. 4. At the demodulator, the continuous-valued received soft-decision samples $\underline{y}$ are demodulated, yielding the bit sequence $\underline{z}$ and the associated soft-decision-based confidence values $|\underline{y}|$ are fed to the Chase III Algorithm. The associated demodulated bit sequence $\underline{z}$ is perturbed by a set of test patterns TP, which is a binary sequence that contains binary ones in the bit positions that are to be tentatively inverted. By adding this test pattern, modulo two, to the received bit sequence a new bit sequence $\underline{z}'$ is obtained, where

$$\underline{z}' = \underline{z} \oplus TP \tag{34}$$

which is also shown in Fig. 4. As a result of using different test patterns, the perturbed demodulated bit sequences $\underline{z}'$ fall within the decoding sphere of a number of different valid codewords, for example in that of $\underline{c}^1 \ldots \underline{c}^4$, as shown in Fig. 5. In the figure, r represents the maximum Hamming distance of the perturbed demodulated bit sequence $\underline{z}'$ from the original demodulated bit sequence $\underline{z}$. If we increase r, which can be achieved by increasing the number of TPs used resulting in an increased number of perturbed bit positions, the perturbed demodulated bit sequence $\underline{z}'$ will fall within the decoding sphere of a higher number of valid codewords. In order to maintain a low complexity, only a limited number of l bit positions associated with the least reliable l number of bit confidence values $|\underline{y}|$ is perturbed; hence, the number of test patterns TP involved is 2^l.

If the perturbed demodulated bit sequence $\underline{z}'$ falls within the decoding sphere of a valid codeword, it is hard-decision RRNS decoded, allowing us to identify what the associated channel-induced error pattern $\underline{e}'$ was, which may be an all-zero or a nonzero tuple. The actual error pattern $\underline{e}$ associated with the demodulated binary bit sequence $\underline{z}$ is given by

$$\underline{e} = \underline{e}' \oplus TP \tag{35}$$

which may or may not be different from the original test pattern TP, depending on whether or not the perturbed demodulated bit sequence $\underline{z}'$ falls into the decoding sphere of a valid codeword. However, only those specific perturbed demodulated bit

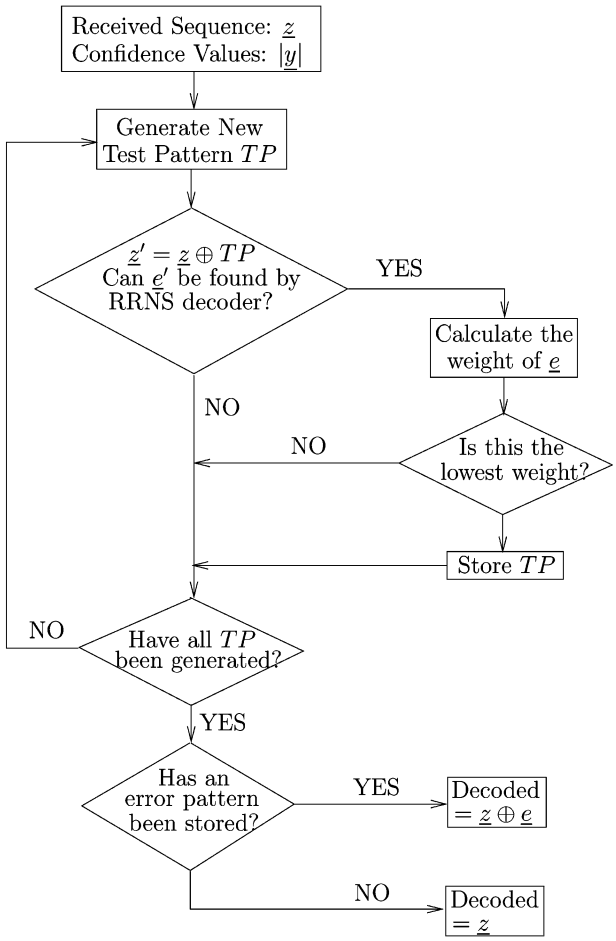


Fig. 4. Flow chart of Chase III algorithm.

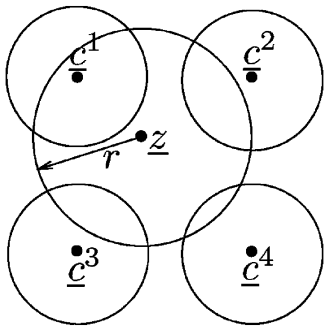


Fig. 5. Simple illustration of Chase III algorithm.

sequences $\underline{z}'$ that fall into the decoding sphere of a valid codeword are considered for decoding, while those which fall in the decoding space of Fig. 5 outside the legitimate decoding spheres are ignored. In this case, we are concerned with finding the most likely error pattern $\underline{e}$ and inverting the corresponding bit positions in the received demodulated sequence, which does not necessarily correspond to a legitimate codeword before decoding. In order to achieve this, we define the minimum "analog weight" associated with an error pattern, where the analog weight $W(\underline{e})$ of an error sequence $\underline{e}$ is defined as

$$W(\underline{e}) = \sum_{i=1}^{n_b} e_i |y_i| \tag{36}$$

and n_b is the number of coded bits. The minimum analog-weight error pattern identifies explicitly the most likely transmitted codeword, namely $\underline{z} \oplus \underline{e}$.

As shown in Fig. 4, the current test pattern TP will be stored, if the associated analog weight $W(\underline{e})$ is found to be lower than the previously registered analog weights. The above procedure will be repeated for the maximum number of test patterns, namely, 2^l, which is tolerable in complexity terms. Upon completing the loop shown in Fig. 4, the memory is checked in order to determine, whether any nonzero error pattern has been stored, and, if so, the corrected decoded sequence will be $\underline{z} \oplus \underline{e}$. Otherwise, the RRNS$(n, k)$ decoded bit sequence is the same as the demodulated bit sequence $\underline{z}$.

B. Soft Output Derivations

Using the Chase algorithm [16], [21], we can find a surviving codeword $\underline{x}$ which generates x_k on the basis of finding the RRNS codeword $\underline{x}$ having the minimum Euclidean distance from continuous-valued received signal vector $\underline{y}$. The algorithm can be readily extended to store another competing (or discarded) codeword $\hat{\underline{x}}$, which decodes to $\hat{x}_k \neq x_k$ and has the minimum Euclidean distance as compared with the other codewords, which decode to the same $\hat{x}_k \neq x_k$. Given the surviving and discarded codewords, we approximate the soft output as [19], [20]

$$L(x_k|\underline{y}) \approx x_k \left[\frac{|\underline{y}' - \hat{\underline{x}}|^2 - |\underline{y}' - \underline{x}|^2}{4} \right] \tag{37}$$

and

$$\underline{y}' = L_c \underline{y} + L(\underline{x}) \tag{38}$$

where L_c is the channel reliability value and $L(\underline{x})$ is the *a priori* information. This expression can be interpreted physically as the difference between the Euclidean distances of the surviving codeword and the discarded competing codeword, which would result in $\hat{x}_k \neq x_k$. In order to increase the algorithm's performance, we have to increase the number of least reliable bit positions l, which are perturbed in the Chase algorithm, and hence, also the number of test patterns or codeword perturbations TP. It is clear that the probability of finding the most likely transmitted codeword $\underline{x}$ and the discarded competing codeword $\hat{\underline{x}}$, which decodes to $\hat{x}_k \neq x_k$, increases with l. However, the complexity of the decoder increases exponentially with l; hence, we must find a tradeoff between complexity and performance. This also implies that in some cases, we shall not be able to find a discarded codeword $\hat{\underline{x}}$ which decodes to $\hat{x}_k \neq x_k$, given the l test positions. If a discarded codeword $\hat{\underline{x}}$ associated with $\hat{x}_k \neq x_k$ is not found, we have to find another method of approximating the soft output. In this case, Pyndiah [19], [20] suggested that the soft output can be approximated as

$$L(x_k|\underline{y}) \approx y'_k + \beta \times L_c x_k \tag{39}$$

where $y'_k = L_c y_k + L(x_k)$ and β is a reliability factor, which increases with the iteration index and can be optimized by simulation. This rough approximation of the soft output is justified by the fact that if no discarded codewords $\hat{\underline{x}}$ were found by the Chase algorithm, which decode to $\hat{x}_k \neq x_k$, then these are probably far from $\underline{y}'$ in terms of the Euclidean distance. When the discarded codewords $\hat{\underline{x}}$ are far from $\underline{y}'$, then the probability that the decision x_k is correct is relatively high and the reliability of x_k, $L(x_k)$, is also high. Through simulations, we found out that the reliability factors β should be fixed to unity for maintaining the best performance of turbo RRNS codes. Hence, (39) is reduced to

$$L(x_k|\underline{y}) \approx y'_k + L_c x_k. \tag{40}$$

Here, we note that there is a similarity between this algorithm and the soft output Viterbi Algorithm (SOVA). In the SOVA, the surviving path $\underline{s}$ is decided on the basis of the received continuous-valued soft-decision sequence $\underline{y}$ and the *a priori* information $L(\underline{x})$. The surviving path $\underline{s}$ determines the surviving codeword $\underline{x}$ in this case. Then, the soft output of the SOVA is proportional to the minimum *path metric difference* between the surviving path $\underline{s}$, which decodes to x_k, and a discarded path $\hat{\underline{s}}$, which decodes to $\hat{x}_k \neq x_k$. Similarly, (37) identifies the codewords $\underline{x}$ and $\hat{\underline{x}}$, having the *minimum Euclidean distance difference* from the received soft-decision value sequence, where the Euclidean distances $|\underline{y}' - \hat{\underline{x}}|$ and $|\underline{y}' - \underline{x}|$ represent the distances between the received soft-decision value sequence $\underline{y}'$ and the codewords $\underline{x}$ and $\hat{\underline{x}}$, respectively. Finally, the weight difference associated with the above two Euclidean distances, namely with $|\underline{y}' - \hat{\underline{x}}|$ and $|\underline{y}' - \underline{x}|$, is calculated, in order to identify those survivor selection steps which are associated with a low difference, i.e., low confidence.

It was also proposed by Pyndiah [19] that a weighting factor α should be introduced in (38), as

$$\underline{y}' = L_c \underline{y} + \alpha L(\underline{x}). \tag{41}$$

The weighting factor α takes into account that the standard deviation of the continuous-valued sampled received sequence $\underline{y}$ from its expected value and that of the *a priori* information $L(\underline{x})$ are different [18], [19]. The standard deviation of the extrinsic information is comparatively high in the first few decoding steps and decreases during future iterations. This scaling factor α is

TABLE III
WEIGHTING FACTORS α FOR DIFFERENT DECODING INDEX j

	Decoding index j							
	1	2	3	4	5	6	7	8
$\alpha(j)$	0.0	0.2	0.3	0.5	0.7	0.9	1.0	1.0

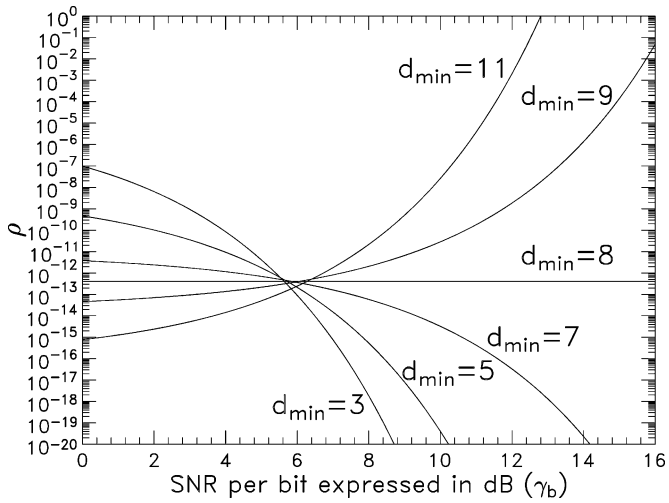


Fig. 6. Ratio ρ of error probability due to "2→1 mapping" to that due to error-correction capability of RRNS codes versus SNR. γ_b performance computed from (32) when BPSK modulation over AWGN channels and ML decision decoding are considered.

used to reduce the effect of the extrinsic information in the decoder during the first decoding steps, when the bit-error rate (BER) is relatively high. The value of α is small in the initial stages of decoding, and it increases as the BER tends to zero.

The parameter α in (41) can be determined experimentally, in order to achieve an optimum performance. Values of α were given in [19], which are reproduced in Table III. The decoding index j in Table III is the index of the decoding steps, which is increased by one after invoking each component decoder.

VI. RESULTS AND DISCUSSIONS

In this section, the performance of RRNS codes using eight-bit residue symbols over GF(256) has been evaluated both numerically and by simulations. The moduli employed were 131, 137, 139, 149, 151, 157, 163, 167, 173, 179, 181, 191, 193, 197, 199, 211, 217, 223, 227, 229, 233, 239, 241, 247, 251, 253, 255, and 256. For most situations considering the RRNS(28,24) code, the largest moduli, namely 251, 253, 255, and 256, were the redundant moduli and the others were the nonredundant moduli. According to (4), it can be shown that the minimum distance $d_{\min}$ of this code is equal to five. Therefore, the error correction capability is $t = 2$ from (5). Besides, the RRNS(28,24) code also satisfies (7); hence, it is a maximum distance separable code.

Fig. 6 shows the ratio ρ of the error probability due to "2→1 mapping" errors to that due to the limited error-correction capability of the RRNS codes employed, when BPSK modulation and ML decision decoding are considered over AWGN channels. The results were computed from (32) for the codes RRNS(28,26), RRNS(28,24), RRNS(28,22), RRNS(28,21),

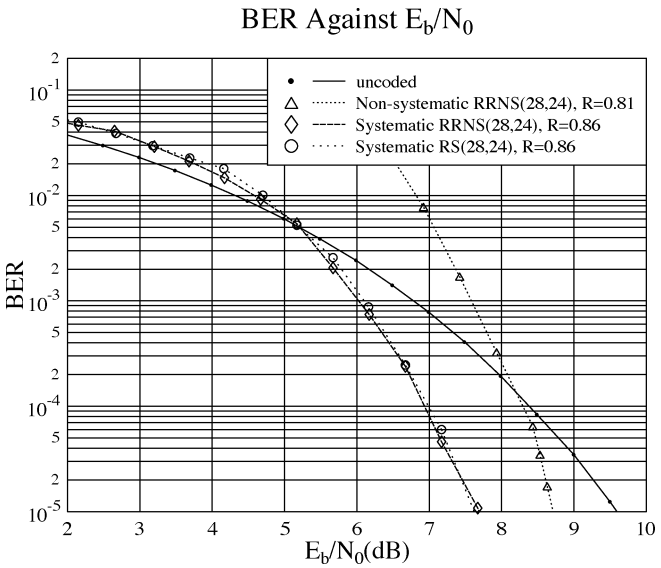


Fig. 7. Performance comparison between nonsystematic and systematic RRNS hard decoding using BPSK modulation over AWGN channels. Performance of RS(28,24) RS code over GF(256) is also included.

RRNS(28,20), and RRNS(28,18) having minimum distances of $d_{\min} = 3, 5, 7, 8, 9, 11$, respectively. From the results, we observe that if $d_{\min} < 8$, which is identical to the number of bits per symbol, the value of ρ decreases upon increasing the SNR per bit, γ_b. However, if $d_{\min} > 8$, the value of ρ increases upon increasing the SNR per bit. Furthermore, if $d_{\min} = 8 = k_{b_i}$, it can be shown with the aid of (32) that ρ is constant, as shown in the figure. The results of this figure imply that in systematic RRNS codes the system performance is determined not only by the error-correction capability of the RRNS codes concerned, but also by the associated "2→1 mapping" errors. For example, when $\gamma_b > 13$ dB, the decoding errors of the RRNS(28,18) code associated with $d_{\min} = 11$ are mainly contributed by the "2→1 mapping" errors, since $\rho > 1$; hence, the system performance cannot be further enhanced by simply increasing the minimum distance of $d_{\min}$ to values higher than 11. Moreover, according to (32), the value of ρ is determined not only by the number of bits per symbol and the minimum distance of the RRNS code, but also by the probability of $P(B)$, as suggested by (15). It can be readily shown that if $m_i = 2^{k_{b_i}}$, then $P(B) = 0$, which means that no "2→1 mapping" exists. Hence, if RRNS codes having a minimum distance higher than the number of bits per symbol is required and ML decoding is employed, an important design criterion is to keep $P(B)$ as low as possible. This design criterion, in turn, implies that any nonredundant modulus should take values as close to an integer power of two as possible.

Fig. 7 shows our performance comparison between nonsystematic and systematic RRNS encoders, which were described in Section IV. Due to their different mapping methods, the bit-based, rather than residue-based, code rate for the systematic encoder is $R = 0.86$, as compared with $R = 0.81$ for the nonsystematic encoder. The performance of the systematic encoder is about 1.0 dB better than that of the nonsystematic encoder. The figure also shows the performance of the systematic RS(28,24) code over GF(256) in comparison to the system-

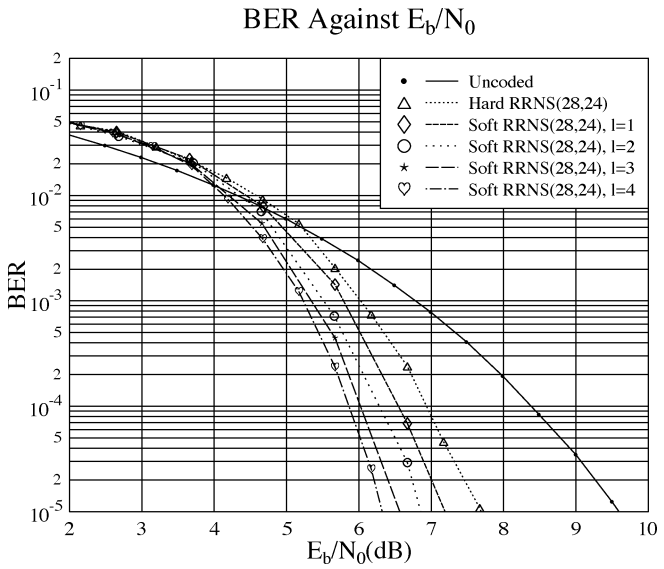


Fig. 8. Performance of systematic RRNS soft decoding for different number of test positions l using BPSK modulation over AWGN channels. Performance of systematic RRNS hard-decision decoding is also shown for comparison.

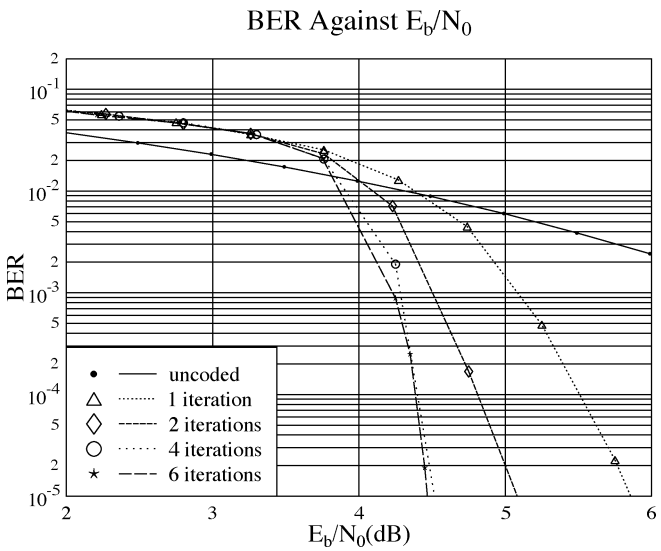


Fig. 9. Performance comparison between different numbers of iterations using eight-bit/residue RRNS(28,24) turbo code, $R = 0.75, 24 \times 24$ residue symbol block interleaver, where $\alpha(j)$ is shown in Table III over AWGN channels.

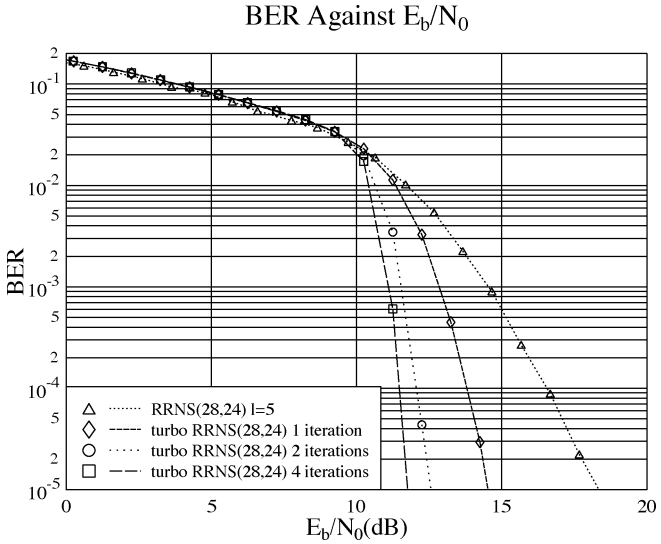


Fig. 10. Performance comparison between different numbers of iterations using eight-bit/residue RRNS(28,24) turbo code, $R = 0.75, 24 \times 24$ residue symbol block interleaver, where $\alpha(j)$ is shown in Table III over uncorrelated Rayleigh fading channels. Performance of soft decoding RRNS(28,24) with $l = 5$ is also shown.

atic RRNS(28,24) code. It can be seen that the performance of the systematic RS(28,24) code and the systematic RRNS(28,24) code is similar.

Fig. 8 portrays the associated performance curves of systematic RRNS soft decoding for different number of perturbed test positions l. At a BER of 10^{-5}, there is a coding gain of 2.3 dB for $l = 1$. Upon increasing the number of test positions l, the larger the subset of tentatively decoded words, the more likely that it contains the transmitted codeword; hence, the better the coding gain is. However, the improvement becomes smaller, as the number of perturbed test positions l increases. Furthermore, the complexity of the algorithm increases exponentially, since the number of test patterns TP is equal to 2^l. The performance of systematic RRNS hard decoding is also shown in Fig. 8 for comparison. For $l = 4$, i.e., for 16 test patterns, the coding gain of RRNS soft decoding is about 3.2 dB at a BER $= 10^{-5}$.

Fig. 9 characterizes a turbo RRNS code's performance for different numbers of iterations over AWGN channels. Specifically, the turbo component codes were based on the RRNS(28,24) code using eight-bit residues and the code rate was 0.75, since the parity bits of both encoders were transmitted. The values of $\alpha(j)$ used are shown in Table III, and the turbo interleaver was a 24×24 residue symbol block interleaver. It can be seen from the figure that at a BER of 10^{-5}, the performance of the turbo RRNS(28,24) code using two iterations is about 0.8 dB better than after the first iteration. However, the performance of the turbo RRNS(28,24) code does not improve significantly after four iterations.

In Fig. 10, we investigate the performance of the turbo RRNS(28,24) code over uncorrelated Rayleigh fading channels, implying the presence of independent complex fading values for each RRNS code symbol, which implicitly assumes the employment of a long symbol interleaver. As a comparison, the performance of the soft-decoded RRNS(28,24) code using $l = 5$ perturbed test positions is also shown in the figure. Since only $l = 4$ test positions were considered in each component decoder in the turbo code, the complexity of the turbo RRNS(28,24) code using one iteration is about the same as that of the soft-decoded RRNS(28,24) code. However, as shown in Fig. 10, the performance of the turbo RRNS(28,24) code using one iteration is about 4 dB better than that of the soft-decoded RRNS(28,24) code. Again, the performance of the turbo RRNS(28,24) code does not improve significantly after four iterations.

VII. Conclusion

In conclusion, we have analyzed a novel family of nonbinary error control codes, namely RRNS codes. Their performance is similar to that of RS codes. However, the flexibility of RRNS codes facilitated the generation of a whole family of

different-strength, different-rate codes based on simply concatenating or discarding a number of redundant residues, without having to shorten long RRNS codes, as in RS codes. Besides, the arithmetic properties inherited from the RNS enable parallel-processing-based decoding of RRNS codes. Novel bit-mapping techniques were proposed, which resulted in nonsystematic and systematic RRNS codes. The Chase III algorithm was then used to implement the soft-decision decoding of RRNS codes. The algorithm was further modified in order to create the SISO Chase algorithm for the iterative decoding of RRNS turbo codes.

REFERENCES

[1] N. Szabo and R. Tanaka, *Residue Arithmetic and Its Applications to Computer Technology*. New York: McGraw-Hill, 1967.

[2] R. Watson and C. Hastings, "Self-checked computation using residue arithmetic," *Proc. IEEE*, vol. 54, no. 12, pp. 1920–1931, Dec. 1966.

[3] D. Mandelbaum, "Error correction in residue arithmetic," *IEEE Trans. Comput.*, vol. C-21, no. 6, pp. 1538–543, Jun. 1972.

[4] F. Barsi and P. Maestrini, "Error correcting properties of redundant residue number systems," *IEEE Trans. Comput.*, vol. C-22, no. 3, pp. 307–315, Mar. 1973.

[5] S. Yau and Y. Liu, "Error correction in redundant residue number systems," *IEEE Trans. Comput.*, vol. C-22, no. 1, pp. 5–11, Jan. 1984.

[6] H. Krishna, K. Lin, and J. Sun, "A coding theory approach to error control in redundant residue number systems—Part I: Theory and single error correction," *IEEE Trans. Circuits Syst.*, vol. 39, no. 1, pp. 8–17, Jan. 1992.

[7] J. Sun and H. Krishna, "A coding theory approach to error control in redundant residue number systems—Part II: Multiple error detection and correction," *IEEE Trans. Circuits Syst.*, vol. 39, no. 1, pp. 18–34, Jan. 1992.

[8] M. Etzel and W. Jenkins, "Redundant residue number systems for error detection and correction in digital filters," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. ASSP-28, no. 5, pp. 538–544, Oct. 1980.

[9] L. Yang and L. Hanzo, "Performance of residue number system based DS-CDMA over multipath fading channels using orthogonal sequences," *Eur. Trans. Telecommun.*, vol. 9, no. 6, pp. 525–536, Nov.–Dec. 1998.

[10] ——, "Residue number system arithmetic assisted M-ary modulation," *IEEE Commun. Lett.*, vol. 3, no. 2, pp. 28–30, Feb. 1999.

[11] L.-L. Yang and L. Hanzo, "Residue number system arithmetic based orthogonal signaling schemes for AWGN and Rayleigh channels— Part I," *IEEE Trans. Veh. Technol.*, vol. 51, no. 6, pp. 1534–1546, Nov. 2002.

[12] ——, "Residue number system arithmetic based orthogonal signaling schemes for AWGN and Rayleigh channels—Part II," *IEEE Trans. Veh. Technol.*, vol. 51, no. 6, pp. 1547–1559, Nov. 2002.

[13] L. Hanzo, M. Münster, B. J. Choi, and T. Keller, *OFDM and MC-CDMA for Broadband Multi-User Communications, WLANs and Broadcasting*. New York: Wiley–IEEE, Jul. 2003.

[14] L. Hanzo, L.-L. Yang, E.-L. Kuan, and K. Yen, *Single- and Multi-Carrier DS-CDMA: Multi-User Detection, Space-Time Spreading, Synchronisation, Standards and Networking*. New York: IEEE–Wiley, Aug. 2003.

[15] L. Yang and L. Hanzo, *Coding theory and performance of redundant residue number system codes.* [Online]. Available: http://www-mobile.ecs.soton.ac.uk/

[16] T. Liew, L. Yang, and L. Hanzo, "Soft-decision redundant residue number system based error correction coding," in *Proc. Veh. Technol. Conf.*, Sep. 1999, pp. 2546–2550.

[17] ——, "Turbo decoded redundant residue number system codes," in *Proc. Veh. Technol. Conf.*, Tokyo, Japan, 2000, pp. 576–580.

[18] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo codes," in *Proc. IEEE Int. Conf. Commun.*, 1993, pp. 1064–1070.

[19] R. Pyndiah, "Near-optimum decoding of product codes: Block turbo codes," *IEEE Trans. Commun.*, vol. 46, no. 8, pp. 1003–1010, Aug. 1998.

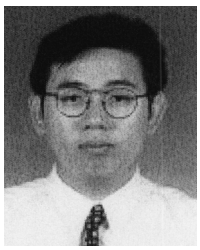
[20] O. Aitsab and R. Pyndiah, "Performance of Reed–Solomon block turbo code," in *Proc. GLOBECOM*, Nov. 1996, pp. 121–125.

[21] D. Chase, "A class of algorithms for decoding block codes with channel measurement information," *IEEE Trans. Inf. Theory*, vol. IT-18, no. 1, pp. 170–182, Jan. 1972.

[22] L. Hanzo, T. H. Liew, and B. L. Yeap, *Turbo Coding, Turbo Equalisation and Space-Time Coding*. New York: Wiley, Aug. 2002.

[23] F. Taylor, "Residue arithmetic: A tutorial with examples," *IEEE Comput. Mag.*, pp. 50–62, May 1984.

[24] J. Proakis, *Digital Communications*, 3rd ed. New York: McGraw-Hill, 1995.

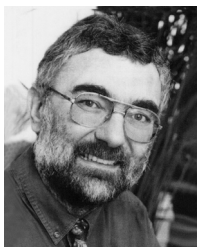


T. H. Liew received the B.Eng. and Ph.D. degrees in electronics engineering from the University of Southampton, Southampton, U.K.

His research interests include coding and modulation for wireless channels, space-time coding, and adaptive transceivers. He has published his research results widely.

Lie-Liang Yang (SM'02) received the M.Eng. and Ph.D. degrees in communications and electronics from Northern Jiaotong University, Beijing, China, in 1991 and 1997, respectively, and the B.Eng. degree in communications engineering from Shanghai TieDao University, Shanghai, China, in 1988.

Since December 1997, he has been with the Communications Research Group, Department of Electronics and Computer Science, University of Southampton, Southampton, U.K., where he held various research posts as a Visiting Postdoctoral Research Fellow, Research Fellow, and Senior Research Fellow. Currently, he holds an academic post as a Lecturer. From June 1997 to December 1997, he was a Visiting Scientist of the IREE, Academy of Sciences of the Czech Republic. He has been involved in a number of projects funded by the National Sciences Foundations of China, the Grant Agency of the Czech Republic, the Engineering and Physical Sciences Research Council (EPSRC) of U.K., and the European Union. His research interests include data network and security, intelligent wireless networking, error-control coding, modulation and demodulation, spread-spectrum communications and multiuser detection, pseudonoise code synchronisation, smart antennas, adaptive wireless systems, as well as wideband, broadband, and ultra-wideband code-division multiple-access for advanced wireless mobile communication systems. He has published over 90 papers in various journals and conference proceedings.



Lajos Hanzo (M'87–F'03) received the M.S. degree in electronics, in 1976, and Ph.D. degree, in 1983. In 2004, he was awarded the D.Sc. degree by the University of Southampton, Southampton, U.K.

During his 30-year career in telecommunications, he has held various research and academic posts in Hungary, Germany, and the UK. Since 1986, he has been with the School of Electronics and Computer Science, University of Southampton, U.K., where he holds the Chair in Telecommunications. He has co-authored 11 Wiley/IEEE Press books totalling about 9000 pages on mobile radio communications, has published in excess of 600 research papers, and acted as TPC Chair of numerous IEEE conferences. He has presented keynote lectures and been awarded a number of distinctions. Currently, he is managing an academic research team, working on a range of research projects in the field of wireless multimedia communications.

Dr. Hanzo is a Fellow of the Royal Academy of Engineering and is also an IEEE Distinguished Lecturer of both the Communications Society and the Vehicular Technology Society. He is a a Fellow of the IEE and a Governor of the IEEE Vehicular Technology Society, as well as an Executive Board member of the Pan-European NEWCOM consortium.