ITERATIVE DECODING OF REDUNDANT RESIDUE NUMBER SYSTEM CODES

T.H. Liew, L-L. Yang, L. Hanzo

Dept. of Electr. and Comp. Sc., Univ. of Southampton, SO17 1BJ, UK.
Tel: +44-703-593 125, Fax: +44-703-593 045
thl97r@ecs.soton.ac.uk; http://www-mobile.ecs.soton.ac.uk

ABSTRACT

Turbo decoded Redundant Residue Number System (RRNS) codes are proposed and their performance is evaluated. An RRNS(n,k) code is a maximum-minimum distance block code, exhibiting identical distance properties to Reed-Solomon (RS) codes. Hence their error correction capability is given by $t=(n-k)/2$. We adapt the classic Chase algorithm in order to accept soft inputs and to provide soft outputs. Using the proposed soft input soft output (SISO) Chase algorithm, the turbo decoding of RRNS codes is contrived.

1. INTRODUCTION

Berrou *et. al.* demonstrated in their ground breaking paper [1] that the performance of turbo codes approaches the Shannon-bound. This novel form of coding scheme consists of two recursive systematic convolutional codes concatenated in parallel. At the receiver, both component decoders decode the received channel information, iteratively improving the performance upon exchanging information. In [1] the so-called maximum a-posteriori (MAP) algorithm was used to decode convolutional codes. The algorithm performs maximum-likelihood bit estimation, thus producing reliability information (soft output) for each received bit. However, the MAP algorithm exhibits excessive complexity, when applied to block codes, such as binary BCH codes, due to the excessive number of states in the trellis. In [2, 3], Pyndiah et. al. modified the classic Chase algorithm [4] so that it provided a soft output for each received bit. The resultant soft-in-soft-out (SISO) Chase algorithm offers a reduced complexity with a minute performance degradation.

Since their introduction, redundant residue number systems (RRNS) have been considered to constitute a promising way of supporting fast arithmetic operations [5]–[7]. 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 generating intermediate carry forward digits or internal delays [5]. Furthermore, RRNSs have been studied extensively for the fault-tolerant protection of arithmetic operations in digital filters as well as in general purpose computers [5]–[7]. A coding theoretical approach to error control coding invoking the RRNS has been developed in [6]. The concepts of Hamming weight, minimum distance, weight distribution, error detection capabilities and error correction capabilities were introduced. A computationally efficient procedure was described for example in [6], for correcting multiple errors. Recently, the Chase algorithm was applied in the context of RRNS codes [8] in order to contrive soft-decision detection and to exploit the soft channel outputs. Different bit mapping techniques were also proposed in [8], which resulted in systematic and non-systematic RRNS encoders.

In this paper, we combine the RRNS decoder proposed in [6] with the SISO Chase algorithm [2–4], in order to decode the soft channel outputs iteratively, as in turbo decoders. At the transmitter, the systematic RRNS encoder proposed in [8] will be used.

2. REDUNDANT RESIDUE NUMBER SYSTEM

A RRNS is defined in terms of an n-tuple of pairwise relative prime positive integers, m_1, m_2, ..., m_k, m_{k+1}, ..., m_n, referred to as moduli. The moduli m_1, m_2, ..., m_k are considered to be non-redundant moduli. The remaining $(n-k)$ moduli, m_{k+1}, m_{k+2}, ..., m_n, form the set of redundant moduli that support error detection and correction in the RRNS. The product of the non-redundant moduli represents the so-called dynamic range, M_k, of the RRNS, which is given by:

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

VTC 2000, TOKYO, JAPAN, 15-18 MAY, 2000

The financial support of the European Commission, Brussels, Belgium and that of the EPSRC, UK is gratefully acknowledged

The interval $[0, M_k - 1]$ is also often referred to as the *legitimate range*, while the interval $[M_k, M_n - 1]$ as 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:

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

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 mod m_j or $|X|_{m_j}$. The positive integer x_j is also termed the j-th residue digit of X.

Given the n-component residue vector, x_1, x_2, ..., x_n, we can reconstruct the integer X from the residues using a procedure known as the Chinese Remainder Theorem (CRT) [5], according to:

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

where $M_j = \frac{M_n}{m_j}$ and L_j is the so-called multiplicative inverse of M_j mod m_j, which is defined as $|L_j M_j|_{m_j} = 1$.

The so-called Mixed Radix Conversion (MRC) [5] 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, ..., a_k are referred to as the mixed radix information digits, and a_{k+1}, ..., a_n will be termed as the mixed radix parity digits.

3. RRNS CODES

The minimum distance d_{min} is a fundamental parameter associated with any error control code. In [6], Krishna et. al. derived the necessary and sufficient conditions concerning the redundant moduli in order for an RRNS code 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 redundant moduli satisfies the following relation [6]:

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

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

$$t = \left\lfloor \frac{d_{min} - 1}{2} \right\rfloor \ , \qquad (5)$$

where $\lfloor \bullet \rfloor$ means the largest integer not exceeding $\bullet$.

From Equation 4, the smallest positive value of M_{n-k} for a minimum distance d_{min} is obtained by setting $M_{n-k} = \max \left\{ \prod_{i=1}^{d_{min}-1} m_{j_i} \right\}$ This shows that the left hand side inequality of Equation 4 is satisfied trivially. It also shows that an optimal RRNS, which is associated with the minimum necessary redundant dynamic range of M_{n-k} required for achieving a minimum distance of d_{min} has the largest modulus of $d_{min} - 1$ as the redundant modulus. Therefore, we can write that:

$$d_{min} - 1 \ = \ n - k \qquad (6)$$

Using the standard coding theoretical terminology, we will refer to an RRNS that satisfies Equation 6 as a maximum distance separable RRNS (MDS-RRNS) code.

The RRNS decoder invoked in this paper was proposed in [6]. The multiple error correction procedures in [6] are extensions of those in [7]. In [7], the algorithms for locating a single residue digit error are based on the properties of the so-called modulus projection and Mixed Radix Conversion (MRC), where the proposed RRNS decoder assumed that the protected signal was discrete. Here, we propose turbo RRNS codes by combining the SISO Chase algorithm [2–4] with the hard decision based RRNS decoder of [6].

4. SOFT INPUT SOFT OUTPUT RRNS DECODER

In this section, we derive the SISO Chase algorithm [2] in the context of the proposed RRNS decoder. However, the derivation outlined below is different from [2]. The so-called Log Likelihood Ratio (LLR) of each decoded bit u_k - given that the received sequence is $\underline{y}$ - can be expressed as:

$$L(u_k|\underline{y}) = \ln \frac{P(u_k = +1|\underline{y})}{P(u_k = -1|\underline{y})} \ , \qquad (7)$$

where k is a bit position in a codeword. Since the probability of $u_k = +1$ is equal to the summation of all the probabilities of all codewords $\underline{x}_i$, which have $u_k = +1$, we can rewrite the numerator of Equation 7 as follows:

$$P(u_k = +1|\underline{y}) = \sum_{\underline{x}_i \in \alpha^{+k}} P(\underline{x}_i|\underline{y}) \ , \qquad (8)$$

where α^{+k} is the set of codewords $\underline{x}_i$ such that $u_k = +1$. By applying Bayes' rule, we can rewrite Equation 8 as:

$$P(u_k = +1|\underline{y}) = \sum_{\underline{x}_i \in \alpha^{+k}} \frac{P(\underline{y}|\underline{x}_i)P(\underline{x}_i)}{P(\underline{y})} \ . \qquad (9)$$

Similarly, we can rewrite the denominator of Equation 7 as:

$$P(u_k = -1|\underline{y}) = \sum_{\underline{x}_i \in \alpha^{-k}} \frac{P(\underline{y}|\underline{x}_i)P(\underline{x}_i)}{P(\underline{y})} \ , \qquad (10)$$

where α^{-k} is the set of codewords $\underline{x}_i$ such that $u_k = -1$.

Substituting Equation 9 and 10 into Equation 7, and assuming that all codewords are equally probable, we arrive at:

$$L(u_k|\underline{y}) = \ln \frac{\sum_{\underline{x}_i \in \alpha^{+k}} P(\underline{y}|\underline{x}_i) P(\underline{x}_i)}{\sum_{\underline{x}_i \in \alpha^{-k}} P(\underline{y}|\underline{x}_i) P(\underline{x}_i)} . \quad (11)$$

Let us assume that the transmitted bit x_k has been sent over an AWGN channel using BPSK modulation. Then, the probability density function of the received symbol y_k conditioned on the transmitted bit x_k can be expressed by [9]

$$P(\underline{y}|\underline{x}) = \left(\frac{1}{\sigma\sqrt{2\pi}}\right)^n exp^{-\frac{E_B}{2\sigma^2}|\underline{y}-a\underline{x}|^2} , \quad (12)$$

where n is the number of coded bits, σ^2 is the noise variance, E_B is the energy per bit and a is the fading amplitude (=1 for a non-fading AWGN channel). Since the probability of a specific codeword, $P(\underline{x})$, is equal to the product of all probabilities of its constituent coded bits x_j, $j = 1, 2, ..., n$, we can then write

$$P(\underline{x}) = C^n exp^{\underline{x}\frac{L(\underline{x})}{2}} , \quad (13)$$

where $L(\underline{x})$ is defined as the LLR of codeword $\underline{x}$, while C^n is a constant which will be cancelled out in Equation 11.

Using Equation 12 and 13, we can rewrite Equation 11 as:

$$L(u_k|\underline{y}) = \ln \frac{\sum_{\underline{x}_i \in \alpha^{+k}} exp^{-\frac{E_B}{2\sigma^2}|\underline{y}-a\underline{x}_i|^2} exp^{\underline{x}_i \frac{L(\underline{x}_i)}{2}}}{\sum_{\underline{x}_i \in \alpha^{-k}} exp^{-\frac{E_B}{2\sigma^2}|\underline{y}-a\underline{x}_i|^2} exp^{\underline{x}_i \frac{L(\underline{x}_i)}{2}}} . \quad (14)$$

Let $\underline{x}^{+k} \in \alpha^{+k}$ and $\underline{x}^{-k} \in \alpha^{-k}$ be the codewords, which are at minimum Euclidean distance from the received sequence $\underline{y}$. Then, upon using the approximation [10], $\ln\left(\sum_j exp^{A_j}\right) \approx \max_j(A_j)$, where $\max_j(A_j)$ denotes the maximum value of A_j and assuming that there were no transmission errors and hence the decoded bit sequence $\underline{u}$ is identical to the transmitted codedword, i.e. $\underline{u} = \underline{x}$, then we have $L(\underline{x}) = L(\underline{u})$. Hence we can approximate Equation 14 as:

$$\begin{aligned} L(u_k|\underline{y}) &\approx -\frac{E_B}{2\sigma^2}|\underline{y} - a\underline{x}^{+k}|^2 + \frac{1}{2}\underline{x}^{+k}L(\underline{u}) \\ &+ \frac{E_B}{2\sigma^2}|\underline{y} - a\underline{x}^{-k}|^2 - \frac{1}{2}\underline{x}^{-k}L(\underline{u}) . \quad (15) \end{aligned}$$

Since $|\underline{y} - a\underline{x}^{\pm k}|^2$ is the Euclidean distance between the received sequences $\underline{y}$ and the codewords $\underline{x}^{\pm k}$, we can write:

$$|\underline{y} - a\underline{x}^{\pm k}|^2 = \sum_{j=1}^{n} \left(y_j - a x_j^{\pm k}\right)^2 . \quad (16)$$

Upon substituting Equation 16 into Equation 15, we arrive at:

$$\begin{aligned} L(u_k|\underline{y}) &\approx \frac{E_b}{2\sigma^2}\sum_{j=1}^{n}\left\{y_j^2 - 2ay_jx_j^{-k} + \left(ax_j^{-k}\right)^2\right\} \\ &-\frac{E_b}{2\sigma^2}\sum_{j=1}^{n}\left\{y_j^2 - 2ay_jx_j^{+k} + \left(ax_j^{+k}\right)^2\right\} \\ &+\sum_{j=1}^{n}\frac{L(u_j)}{2}\left(x_j^{+k} - x_j^{-k}\right) . \quad (17) \end{aligned}$$

Since $x^{\pm k} \in \{-1, +1\}$, $\left(x_j^{-k}\right)^2 = \left(x_j^{+k}\right)^2$, and upon introducing $Lc = \frac{2E_b}{\sigma^2}a$, we can simplify Equation 17 to:

$$L(u_k|\underline{y}) \approx L_c y_k + L(u_k) + \sum_{\substack{j=1 \\ i\neq k}}^{n} e_j \left[L_c y_j + L(u_j)\right] , \quad (18)$$

where

$$e_j = \begin{cases} 0 & \text{if } x_j^{+k} = x_j^{-k} \\ 1 & \text{if } x_j^{+k} \neq x_j^{-k} \end{cases} . \quad (19)$$

Let us define the extrinsic information [2] as:

$$L_e(u_k) = \sum_{\substack{j=1 \\ i\neq k}}^{n} e_j \left[L_c y_j + L(u_j)\right] , \quad (20)$$

which allows us to approximate the soft output as:

$$L(u_k|\underline{y}) \approx L_c y_k + L(u_k) + L_e(u_k) , \quad (21)$$

constituted by the summation of the soft channel output $L_c y_k$, the a-priori (intrinsic) information $L(u_k)$ and the extrinsic information $L_e(u_k)$.

5. ALGORITHM IMPLEMENTATION

In the previous section we have shown in Equation 15 that in order to approximate the soft output $L(u_k|\underline{y})$, two codewords $\underline{x}^{+k}$ and $\underline{x}^{-k}$ which are nearest to $L_c\underline{y} + L(\underline{u})$ have to be found. Using the Chase Algorithm [4, 8], we can find a surviving codeword $\underline{x}$, which generates x_k on the basis of finding the codeword $\underline{x}$ having the minimum Euclidean distance from $L_c\underline{y} + L(\underline{u})$. The algorithm can be readily extended to finding 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 to the other codewords, which decode to $\hat{x}_k \neq x_k$. Given the surviving and discarded codewords, we approximate the soft output as:

$$L(x_k|\underline{y}) \approx x_k \left[\frac{|\underline{y}' - \hat{\underline{x}}|^2 - |\underline{y}' - \underline{x}|^2}{4}\right] , \quad (22)$$

where $\underline{y}' = L_c\underline{y} + L(\underline{u})$. This expression is related physically to the difference between the Euclidean distances of the surviving codeword and discarded codeword from the received codeword. In order to find the most likely surviving codeword $\underline{x}$, we have to consider a higher number of least reliable bit positions l in the Chase Algorithm [4] and invoke a higher number of test patterns (TP) or codeword perturbations, since the probability of finding the most likely codeword $\underline{x}$ increases with l. However, the complexity of the decoder increases exponentially with l and hence we must find a tradeoff between complexity and performance. This also implies that in some cases we shall not to be able to find a perturbed codeword $\underline{\hat{x}}$, which decodes to $\hat{x}_k \neq x_k$, given the l test positions used to perturb the codewords. If such a discarded codeword $\underline{\hat{x}}$ is not found, we have to find another method of approximating the soft output. Pyndiah [2, 3] suggested that the soft output can be approximated as:

$$L(u_k|\underline{y}) \approx y'_k + \beta \times L_c x_k \ , \quad (23)$$

where $y'_k = L_c y_k + L(u_k)$ and β is a reliability factor, which increases with the iteration index and that can be optimized by simulation. This rough approximation of the soft output is justified by the fact that if no discarded codewords $\underline{\hat{x}}$ were found by the Chase Algorithm which decode to $\hat{x}_k \neq x_k$, then the discarded codewords $\underline{\hat{x}}$ which decode to $\hat{x}_k \neq x_k$ are probably far from $\underline{y}'$ in terms of the Euclidean distance. Since the discarded codewords $\underline{\hat{x}}$ are far from $\underline{y}'$, then the probability that decision u_k is correct is relatively high and the reliability of u_k, $L(u_k)$, is also high.

Actually, 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 sequence $\underline{y}$ and the a-priori information $L(\underline{u})$. The surviving path $\underline{y}$ 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 $\underline{\hat{s}}$, which decodes to $\hat{x}_k \neq x_k$. Similarly, Equation 22 identifies the codewords having the *minimum Euclidean distance difference* and evaluates the weight difference between the surviving codeword $\underline{x}$ and the discarded codeword $\underline{\hat{x}}$.

It was also proposed by Pyndiah [2] that a weighting factor α should be introduced in Equation 21, as follows:

$$L(u_k|\underline{y}) \approx L_c y_k + \alpha L(u_k) + L_e(u_k) \ . \quad (24)$$

The weighting factor α takes into account that the standard deviation of the received sequence $\underline{y}$ from its expected value and that of the a-priori information $L(\underline{y})$ are different [1, 2]. The standard deviation of the extrinsic information $L_e(u_k)$ is comparatively high in the first few decoding iterations and decreases during future iterations. This scaling factor α is also used to reduce the effect of the extrinsic information in the decoder in the first decoding steps, when the BER is relatively high. The value of α is small in the initial stage of decoding and it increases as the BER tends to zero.

The parameters α in Equation 24 and β in Equation 23 can be determined experimentally, in order to achieve an optimum performance. Both α and β were given in [2], which are reproduced in Table 1, where the decoding index j in Table 1 is the index of the decoding iterations, which increase by one after each component decoder.

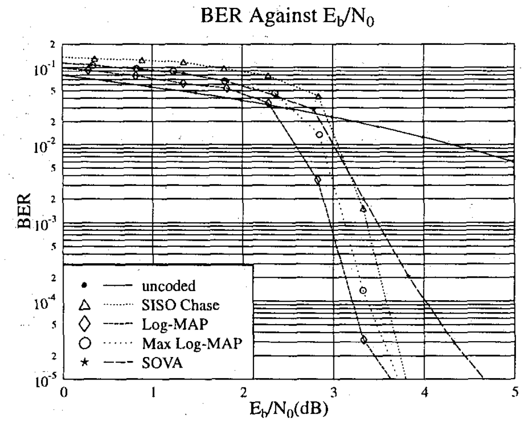

Figure 1: Performance comparison between different decoding algorithms using the turbo BCH(63,57) code over AWGN channels. There were six decoding iterations and a 57×57 block interleaver was used. For the SISO Chase algorithm, $\alpha(j)$ and $\beta(j)$ were specified in Table 1 and $l = 4$.

	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
$\beta(j)$	0.2	0.4	0.6	0.8	1.0	1.0	1.0	1.0

Table 1: The weighting factors α and reliability factors β for different decoding number j.

6. SIMULATION RESULTS

In this section, we compare the performance of the SISO Chase algorithm with other well known algorithms - such as the Log-MAP, Max Log-MAP and SOVA - in the context of binary BCH turbo codes. BCH codes were favoured, since a Viterbi decoder can be invoked for their decoding and hence the Log-MAP, Max Log-MAP and SOVA can be applied as benchmarkers. Figure 1 shows our performance comparison of the different decoding algorithms using the turbo BCH(63,57) code over AWGN channels. There were six decoding iterations and a 57×57 block interleaver

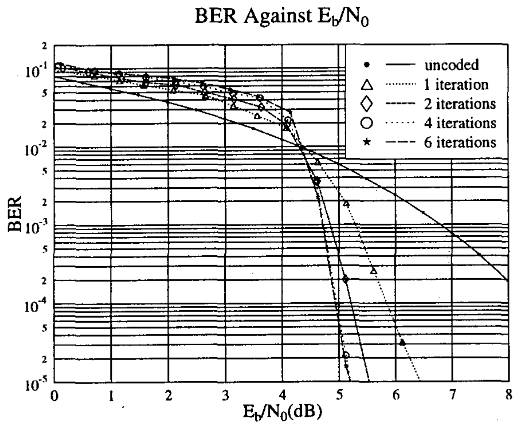

Figure 2: BER performance of the rate $R = 0.87$ RRNS(28,26) turbo code using 8-bit residues and a 26×26 symbol block interleaver as well as the $\alpha(j)$ and $\beta(j)$ values shown in Table 1 upon employing BPSK over AWGN channels.

was used. For the SISO Chase algorithm, $\alpha(j)$ and $\beta(j)$ were specified in Table 1 and we used $l = 4$ perturbed bit positions, resulting in a total of $2^4 = 16$ TPs. Since the Log-MAP decoding algorithm constitutes the optimum technique, its BER performance is the best in Figure 1. The Max Log-MAP decoding algorithm gives a slight degradation of 0.1 dB at BER= 10^{-5} compared to the Log-MAP decoding algorithm. With the optimum values of $\alpha(j)$ and $\beta(j)$ given by Pyndiah [2] (shown in Table 1), the SISO Chase algorithm gives a slight degradation of 0.2 and 0.1 dB at BER= 10^{-5} compared to the Log-MAP and Max Log-MAP decoding algorithms, respectively. As compared to the SOVA, the SISO Chase algorithm seems to perform better, having a 0.8 dB E_n/N_0 advantage at a BER of 10^{-5}.

As another application of the proposed SISO Chase decoder, Figure 2 shows the performance of the rate $R = 0.87$ RRNS(28,26) turbo code using 8-bit residues and a 26×26 symbol block interleaver as well as the $\alpha(j)$ and $\beta(j)$ values shown in Table 1 upon employing BPSK over AWGN channels. The code rate became $R = 0.87$, since the parity symbols generated by both the upper and lower turbo encoder were transmitted without puncturing. As the number of iterations performed by turbo decoder increased, the performance improved, although the improvements after 4 iterations became insignificant.

7. CONCLUSION

In conclusion, we have invoked and modified the SISO Chase algorithm for the iterative decoding of RRNS turbo codes, which constitute a class of powerful maximum-minimum distance codes. Due to their non-binary nature RRNS codes are attractive burst-error correcting codes, especially in the context of M-ary systems, such as for example the second-generation Pan-American mobile radio system known as IS-95, which employs 6-bit symbols conveyed by one of a set of 64 Walsh-Hadamard codes. The proposed turbo decoding algorithm offered a turbo iteration gain in excess of 1 dB in the context of the RRNS code investigated and its most attractive application domain is the decoding of high-rate RRNS codes, where a relatively low number of turbo iterations and a moderate number of perturbation test patterns provides a good performance. Our future work will incorporate the proposed RRNS turbo codec in interactive wireless speech and video systems.

8. REFERENCES

[1] C. Berrou, A. Glavieux and P. Thitimajshima, "Near Shannon Limit Error-Correcting Coding and Decoding : Turbo Codes," *Proceedings, IEEE International Confrence on Communications*, pp. 1064-1070, 1993.

[2] R. M. Pyndiah, "Near-Optimum Decoding of Product Codes: Block Turbo Codes," *IEEE Transactions on Communications*, pp. 1003-1010, August 1998.

[3] O. Aitsab and R. Pyndiah, "Performance of Reed-Solomon Block Turbo Code," *GLOBECOM '96*, pp. 121-125, November 1996.

[4] D. Chase, "A Class of Algorithms for Decoding Block Codes With Channel Measurement Information," *IEEE Transactions of Information Theory*, vol. 18, No 1, pp. 170-182, January 1972.

[5] N. S. Szabo and R. I. Tanaka, "Residue Arithmetic and Its Applications to Computer Technology," *McGraw-Hill Book Company*, New York, 1967.

[6] H. Krishna, K-Y. Lin and J.-D. Sun, "A Coding Theory Approach to Error Control in Redundant Residue Number Systems - Part I and II: Theory and Single Error Correction," *IEEE Transactions on Circuits and Systems-II: Analog and Digital Signal Processing*, vol. 39, No 1, pp. 8-34, January 1992.

[7] M.H. Etzel and W. K. Jenkins, "Redundant Residue Number Systems for Error Detection and Correction in Digital Filters," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. ASSP-28, No 5, pp. 538-544, October 1980.

[8] T. H. Liew, L-L. Yang and L. Hanzo, "Soft-Decision Redundant Residue Number System Based Error Correction Coding," *Proceedings of VTC '99 Fall*, Vol. 5, pp.2546-2550, September, 1999; also http://www-mobile.ecs.soton.ac.uk/

[9] J.G. Proakis, "Digital Communications," *McGraw-Hill, Inc.*

[10] J. Erfanian, S. Pasupathy and G. Gulak, "Reduced Complexity Symbol Detectors with Parrallel Structures for ISI Channels," *IEEE Transactions on Communications*, vol. 42, No 2, pp. 1661-1671, February 1994.