

CDMA Technology based Reliable Wireless Mobile Communication System on a Single Chip using Cellular Automata Concept

Subir Kumar Sarkar, Professor, Jadavpur University, sksarkar@etce.jdvu.ac.in
G. C. Manna, Research fellow, Jadavpur University, gcmanna@yahoo.com
S. S. Singh, Research fellow, Jadavpur University, sudh_69@yahoo.com
T. Datta, Assistant Professor, Institute of Engineering and Management, td_pms@rediffmail.com
Samir kumar Sarkar, Professor Jadavpur University, su-sarkar@hotmail.com
P. K. Naskar, Research fellow, Jadavpur University, pratju@radiffmail.com

Abstract

System designers of today, in view of cutting more and more costs are using integrated circuits, which are unique to their application. In the present work, the main intention is to improve the density of integration to a new height by integrating the Individual components of a typical mobile system into a semi-custom VLSI solution. Further, several error correction techniques are incorporated in the design to get better error free, reliable and secured communication. It is expected that the inclusion of one dimensional cellular automata technique will bring attractive features in the present design and will reduce error in communication thereby making it more reliable.

Key words: Cellular automata, mobile communication, encryption, pseudo noise, error correcting codes.

1. Introduction

There is no denying the fact that mobile communication has become an integral part of our lives today. Indeed the ultimate aim of anytime, anywhere and anybody communication will remain unfulfilled if it weren't for the birth and explosive growth of mobile communication systems [1-5]. As with other areas of cutting edge technology, mobile communication systems too, have undergone a sea change from when they first came in. In fact, with "integration" being the buzzword in today's electronics industry, gone are the days of bulky mobile sets. They are now cheaper, more reliable and more versatile than ever before. As had already been mentioned, we are seeing huge strides being made in mobile communication technology almost everyday. Mobile communication, encompassing the extremely popular cellular mobile telephone system to the Wireless Information Superhighway, has become much popular as they are virtually free from the operational limitations of conventional wired systems: Limited service capability and performance, and inefficient spectrum utilization. Mobile systems offer one significant advantage in that they provide mobility. This means that the receiver (hardware) can move around ranging from a limited area – say, within a building as in wireless LANs, to a much wider area – as in a cellular or satellite based system. Thus using the same receiver, one can communicate while on the move. Though mobile systems still suffer from several limitations like multipath fading, interference, dependence on weather condition for good reception, etc., with continual improvement in technology their harmful effects are on the way to being minimized or eliminated altogether [6,7]. It's a proven fact that without the concept of integration, mobile communication systems won't be having it so good. So in this work, we wish to improve the density of integration to a new height by integrating the individual components of a typical mobile system into a semi custom VLSI solution. Thus owing to the transmitter and receiver being integrated into a single chip, with the reduction in size comes a corresponding increase of speed as one of the main source of circuit delay viz. the interconnect delays get reduced as the RC - time constant decreases in value. Trends suggest a reduction in size, also brings a decrease in cost [8, 9].

Since inception of the concept of cellular automata (CA) by John Newman it has been recommended in favour of the potential usage of data security. The reasons behind the popularity of CA are due to their simplicity and enormous potential in modeling complex system. By developing apt set of laws into CA, it is possible to simulate a lot of complex behavior required in data security procedures. CA shows a promising advancement in Nanotechnology with the intention of providing new and simple concept and potentiality for real world applications in the near future.

In the present work, a CDMA technology based reliable wireless mobile communication system on a single chip using cellular automata concept is simulated using MATLAB. The simulated model is investigated using sample data and error possibilities are checked. The introduction of CA has provided both authentication and security. The scheme is designed from analytical study of the state transition behavior of non group CA. The application of CA for the realization of the simulated model makes it very appropriate for VLSI circuit implementation.

2. Concept of Cellular Automata (CA)

CA is an array of indistinguishable and identical automatically set automata or cells, which cooperate with each other. In encryption/decryption systems, as a rule a key stream generator is obligatory to endow with a key cycle [10-16]. Encryption can be realized by means of either block or stream ciphers. Figure 1 depicts a state of a particular one-dimensional (1D) CA. The essential features for CA comprise of

- **State:** This variable is either a number or a property, which confirms the difference between each cell.
- **Neighbourhood:** It represents the cells that exist around the specified cell. The specified cell interacts with its surrounding cells in every successive time interval. The present states of the neighbouring cell are taken into account for the next state of each cell [17].
- **Program:** It is the set of laws that are incorporated and the result is change in status of the state of the specified cell and the neighbouring cells.

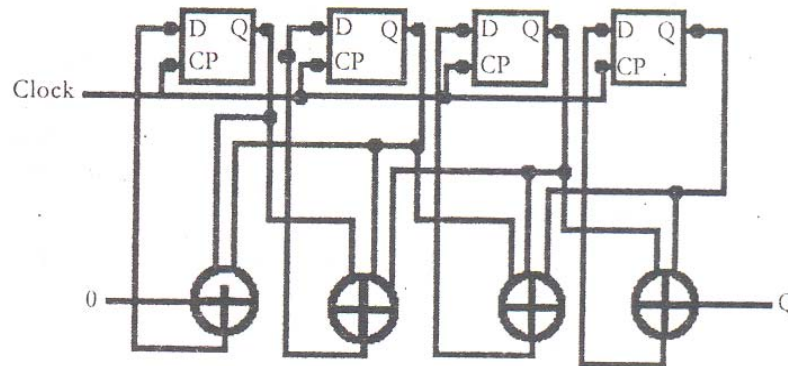


Fig. 1: State of particular one-dimensional CA

If we consider n-cell one-dimensional CA (shown in Fig. 1), the linear operator is ought to be a (nxn) matrix whose i^{th} row relates to the neighbourhood of the i^{th} cell. The matrix is symbolized as T and it is defined as the characteristic matrix of the n -cell CA. The i^{th} cell has a dependency on the j^{th} cell when the i^{th} row and j^{th} column of T have bits '1', while there exists no such dependency if T have bits '0'.

One remarkable feature of the characteristic matrix T is its essentiality for the formation of the next state of the cellular automata, which can be produced via a simple matrix multiplication. The accumulation involved in this process is modulo-2. Thus it can be rewritten as and hence

$$f_{t+1} = T * f_t$$

And hence

$$f_{t+p} = T^p * f_t$$

Let the characteristics matrix be

$$T = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}$$

Let the present state of the CA be

$$F = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}$$

Then the next state will be

$$T \times F = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \end{pmatrix}$$

The state transition graph for the above considered CA with the above characteristic matrix is given below (Fig. 2). It is in the shape of reversed tree:

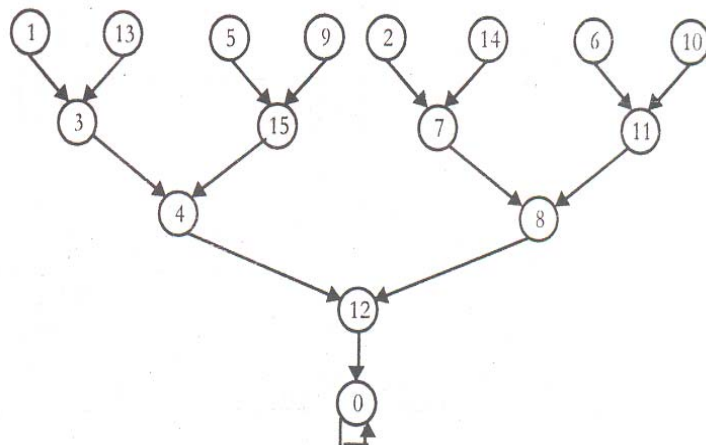


Fig.2 State transition graph of CA when $\det(T)=0$

CA can be either in group or non group, when every cells of CA lie in the same cycle of the state transition graph, then it is a group CA, otherwise it is a non-group CA. In a non-group CA, the $\det(T)=0$. It might be noted that for $\det(T)=1$ provides a group CA.

In case of linear one-dimensional CA, the subsequent state function for Rule 90 and Rule150 are set as:

Rule90: $q_i^{t+1} = q_{i-1}^t \oplus q_{i+1}^t$ and

Rule 150: $q_i^{t+1} = q_{i-1}^t \oplus q_i^t \oplus q_{i+1}^t$

In data coding the automata concept has been introduced with an intention of reducing error in the communication as it a known fact that automata can inherently provide [17].

3. Scheme of the work

The work involves integration of the various component parts of a typical mobile transmitter and receiver into a *single* chip. As is evident, the integration will no doubt increase system reliability, reduce size and will also cost less. The block diagram of a typical mobile transmitter is shown next (Fig.3):

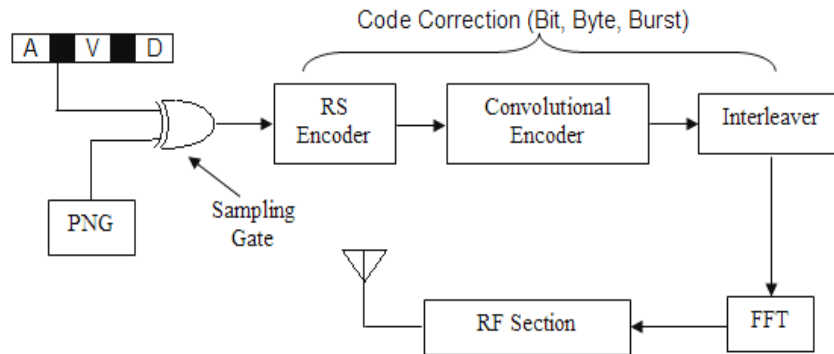


Fig. 3 Block diagram of a typical mobile transmitter

Before transmission, the data is spread (in frequency domain) by using a PNG (pseudo-noise generator). It may be noted that the data stream (which may consist of video, audio or other data sources) with appropriate guard times inserted between the components so that no interference occurs among them, is applied to a sampling gate (XOR). The XOR gate is used as a mod-2 summer. The PNG forms the other input to the XOR gate. For error correction several error correcting blocks are incorporated to get better communication. The block diagram of a typical mobile receiver is as shown below (Fig.4):

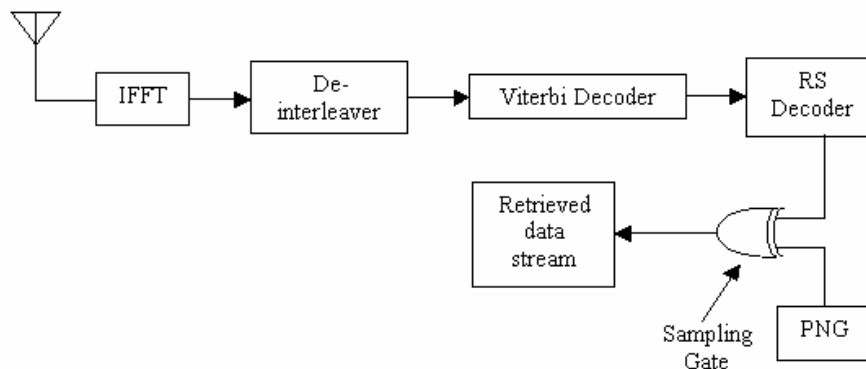


Fig. 4 Block diagram of a typical mobile receiver

In the receiver part, the corresponding IFFT, decoder blocks have been incorporated. The final output from the decoder block is again fed to a sampling gate with the same PNG as had been used in the transmitter, forming the other input to the gate. Error correcting schemes are also considered (as shown in Fig. 4) so that reliable communication is established. Synthesizable VHDL models exist for all the blocks shown [18,19]. In this work, we have implemented the simplest architecture of the blocks with the understanding that their complexity will be increased when needed. For synthesis, the parameters chosen are as: Device Family: *Spartan3*, Device: *xc3s400* and the Package: *pq208*.

3.1. PNG

The PN (pseudo-noise) code sequence consists of ones and zeros (called chips). The chip rate is set to be higher than the input data rate. For generating the PN sequence we would generally employ a feedback shift register configuration, which can be shown to generate a maximal length (ml) sequence, also called a pseudo-noise sequence.

To minimize jamming and/or casual eavesdropping, the PN sequence should be as long as possible; the longer the sequence, the more difficult it is for the unauthorized listener to detect the correct one. With CDMA systems, where each user has a unique PN code, the receiver should be able to reject other interfering SS signals and/or prevent false correlation [20]. Unfortunately, sequences generated from a single shift register, don't have good cross-correlation properties. So in this work we have employed Gold Codes generated by the mod-2 combination of two or more registers. Here we have employed two 10-bit feedback shift registers. The preferred pair used is given by: $[-10,8,5,1]$ $[10,7,6,4,2,1]$ giving the auto/cross-correlation ratio as 0.064 [21].

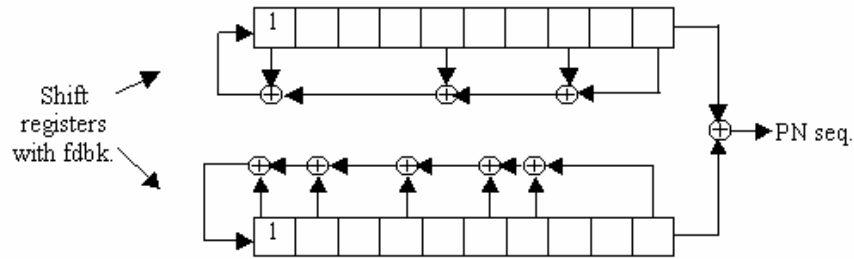


Fig. 5 PN sequence generator

Using the above configuration (Fig. 5), it has been found that we can produce up to 1023 ($2^{10}-1$) unique output sequences [20].

3.2. RS Encoder

Here R S encoder is automatically delt with and is simulated based on the analytical model. The Reed-Solomon codes are a subclass of cyclic codes, being a special case of Bose-Chaudhuri-Hocquenghem (BCH) codes [16]. This code can be easily implemented using linear feedback shift registers (LFSRs). This cyclic error correcting code takes a fixed length input (k symbols) and produces a fixed length check code ($n-k$ symbols), n being the total number of symbols in the codeword [22]. A particular cyclic code can be represented by a polynomial divisor, called the generator polynomial. For an (n,k) code, the generator polynomial has the form:

$$G(X) = 1 + \sum_{i=1}^{n-k-1} A_i X^i + X^{n-k} \quad (1)$$

We thus have,

$$X^{n-k}D(X)/G(X) = Q(X) + C(X)/G(X) \quad (2)$$

where $D(X)$ is the data polynomial, $C(X)$ is the remainder polynomial and it being concatenated properly with $D(X)$ produces the transmitted block,

$$T(X) = X^{n-k}D(X) + C(X) \tag{3}$$

All the summations are, in fact, mod-2 summations, even within the symbols. RS codes work with M-ary modulation systems. Here data are processed in chunk of m bits ($M = 2^m$) called symbols. A popular value of m is 8 [23]. In the implementation of RS encoder, we have used an 8-ary alphabet, i.e. a 3-bit encoder. The RS coding scheme used is the well-known (7,4) coding scheme. Hence in terms of symbols we have $d_{\min}=3$ symbols. However, in terms of bits the above mentioned RS code gives us a distance of $(2^3)^3 = 512$ bits.

For this code, the symbols are from GF(8). We take the generator polynomial to be $p(X)=1+X+X^3$. Its one of the primitive polynomials for GF(8). Using this polynomial, the GF(8) alphabets are generated. If we consider α to be a primitive element in GF(2^m), the generator polynomial of a primitive t-error correcting RS code of length 2^m-1 is:

$$G(X) = (X + \alpha)(X + \alpha^2) \dots (X + \alpha^{2^t}) \tag{4}$$

$$= g_0 + g_1X + g_2X^2 + \dots + g_{2^t-1}X^{2^t-1} + X^{2^t} \tag{5}$$

From the above expression its apparent that $G(X)$ has $\alpha, \alpha^2, \dots, \alpha^{2^t}$ as its roots and has coefficients from GF(2^m). The code generated using $G(X)$ is an $(n, n-2t)$ cyclic code which consists of those polynomials of degree $n-1$ or less with coefficients from GF(2^m) that are multiples of $G(X)$. In this case we have $n-k = 2t = 3$ and so

$$G(X) = (X + \alpha)(X + \alpha^2)(X + \alpha^3) \tag{6}$$

$$= \alpha^6 + \alpha X + \alpha^6 X^2 + X^3 \tag{7}$$

As RS code is a systematic code, we have the appearance of the transmitted codeword as: $X = (k | n-k)$, where $k =$ message bits, $n-k =$ check bits. The code polynomial is generated (as in convolutional code style) by long division of $M(X)$ by $G(X)$ the generator polynomial. A shift register is used to aid in the division process. At the end of each block of message bits, the shift register is reset to contain the symbol "000." A schematic diagram of the encoder is shown next (Fig. 6):

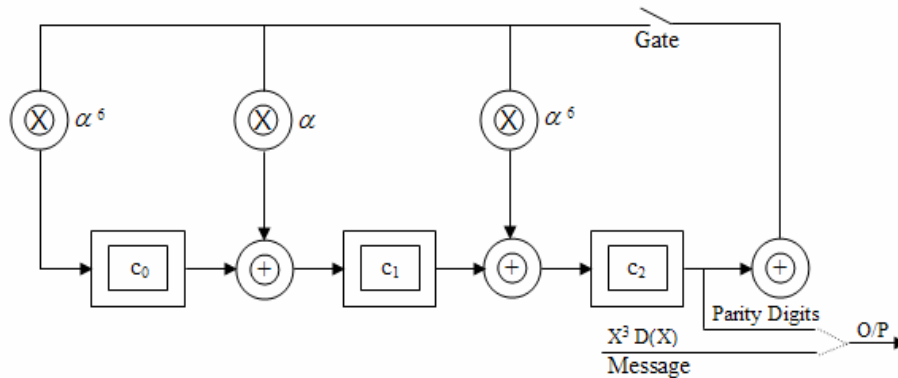
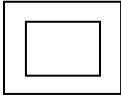


Fig. 6 RS encoder

where the symbol:-

\oplus ----- denotes an adder that adds two elements from GF(8)

\otimes_{g_i} ----- denotes a multiplier that multiplies a symbol from GF(8) with a fixed symbol g_i from the same field



----- denotes a storage for elements from GF(8)

Also to pre-multiply the data polynomial with X^{n-k} , it's shifted into the above circuit from the front end. As soon as the complete message has entered the circuit (and also the communication channel), the $n-k$ parity check symbols are in the register. These are now sent one by one into the channel starting from c_0 . The incoming of message symbols are temporarily stopped during this manoeuvre. The effectiveness of such encoder is not individually checked and hence individual plot showing its efficiency is not included here. However, plots are added at the end (Fig. 11 & Fig. 12) so represent the effects of all the error correction encoder.

3.3. Convolutional Encoder

Convolutional codes have a structure that effectively extends over the entire transmitted bit stream, rather than being limited to the codeword blocks [24]. For encoding, the message bits (m_k 's) are first loaded serially into a shift register with tap gains (g_x 's). The message bits in the register are combined by mod-2 addition to form the encoded bit

$$X_j = m_{j-1} \oplus \dots \oplus m_{j-1}g_1 \oplus m_j g_0 \tag{8}$$

$$= \sum_{i=0}^l m_{j-i} g_i \pmod{2}$$

where m_j is the j^{th} message bit.

The name convolutional encoding comes from the fact that the above equation, in its summation form, has the appearance of a binary convolution, analogous to the well-known convolution integral [21]. The implementation of convolutional encoding is carried out using a (2,1,6) encoder. As is evident, the encoder can be expressed by two generator polynomials with

$$G_1(X) = X^6 + X^4 + X^3 + X + 1 \tag{9}$$

and

$$G_2(X) = X^6 + X^5 + X^4 + X^3 + 1 \tag{10}$$

The schematic diagram of the (2,1,6) encoder is given next (Fig.7):

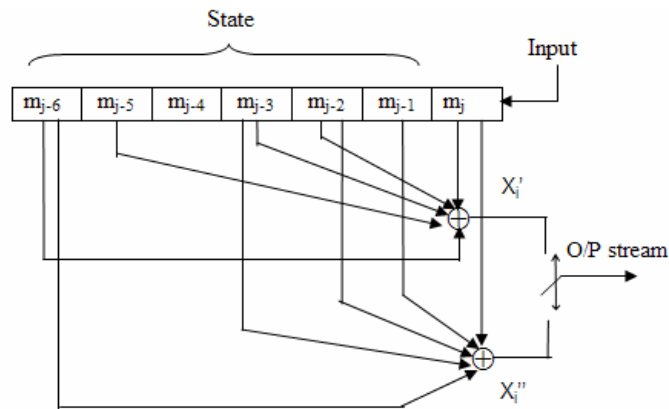


Fig. 7 Convolutional encoder

Here as is evident,

$$X_j' = m_j \oplus m_{j-2} \oplus m_{j-3} \oplus m_{j-5} \oplus m_{j-6}$$

and

$$X_j'' = m_j \oplus m_{j-1} \oplus m_{j-3} \oplus m_{j-6}$$

3.4. Interleaver

Interleaving is accomplished by reading and writing data from the memory in different orders. We have implemented the block interleaving technique here. In this case the data to be transmitted are stored in a rectangular array in which each row consists of n bits, equal to block size. Data are then read out one column at a time. The result is that the k message bits and their corresponding $n-k$ check bits, which form a single n bit block, are spread out and interspaced with bits from other blocks. Here we have set the depth of the interleaver i.e. the number of rows to be four. The codeword length is left to be *generic* to enable arbitrary length codewords being interleaved. For ease of simulation we have set the length to six. The interleaver actually has two blocks (buffers – 4×6 each). Thus when one block stores the incoming bits row-wise, bits are read out of the other block column-wise. This is done to eliminate bit loss when data read out is being performed.

A schematic diagram of the interleaver is shown next (Fig.8):

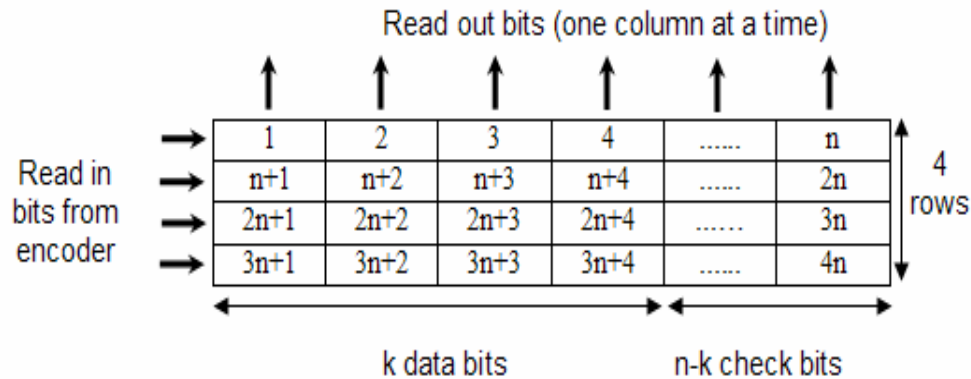


Fig. 8 A block interleaver

3.5. De-interleaver

As with the decoding blocks to follow, the received bit stream must first be de-interleaved to obtain the stream to be fed to the following convolutional decoder. A schematic diagram of the de-interleaver is as shown next (Fig.9):

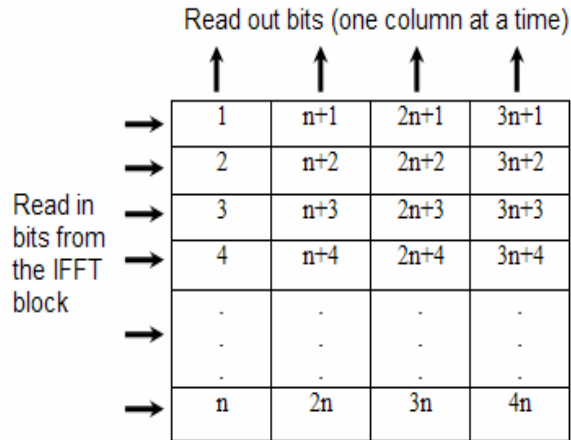


Fig. 9 A block de-interleaver

The de-interleaving operation in a sense is simply a matrix transform operation. It can easily be seen if one compares the schematic diagrams of the interleaver and the de-interleaver.

3.6. Convolutional Decoder

Of the three generic methods for decoding convolutional codes, we have chosen to go with the maximum-likelihood decoding approach. Thus, we have employed the Viterbi decoding algorithm which, although requires extensive hardware usage for computation and storage, achieve the optimum performance as it finds the decoded word as having the minimum Hamming distance from the received sequence. As with the maximum-likelihood case, the decoder employed examines an entire received sequence and finds a valid path that has the minimum Hamming distance from the received sequence. In other words, the received sequence is decoded as the surviving path with the smallest metric. Also, we have assumed that 0's and 1's have the same transmission probability. As the encoder employed is a (2,1,6) encoder, our Viterbi decoder has to deal with $2^{kl}=64$ surviving paths requiring that much maximum memory at each stage of comparison. Also, when two arriving paths at a particular state have the same equal running metric value, we have arbitrarily chosen one path. The storage (memory) requirement may be reduced significantly by relying on the metric divergence effect [21,25].

3.7. RS Decoder

As the RS encoder formed the first of our coder blocks, the RS decoder forms the last of our decoder blocks. To decode an RS code vector we have to follow four steps. They are as follows:-

1. The syndrome $S=(s_1,s_2,\dots,s_{2t})$ is computed from the received code polynomial $r(X)$.
2. From the syndrome components s_1,s_2 etc., we compute the error location polynomial $\sigma(X)$.
3. The error location numbers $\beta_1, \beta_2,\dots,\beta_v$ is to be found from the roots of $\sigma(X)$, where v is the number of locations in $r(X)$ where an error has occurred. Obviously for proper correction, $v \leq t$.
4. As the RS code is non-binary in nature, calculation of the error values is required. Completion of step 4 will give us the error values and the error locations in the received code vector $r(X)$ [23].

Thus, let the received vector be $r(X)=r_0+r_1X+\dots+r_{n-1}X^{n-1}$, and the error pattern added by the noisy channel is

$$e(X)=e_0+e_1X+\dots+e_{n-1}X^{n-1} \tag{11}$$

where e_i 's are symbols from GF(8) [12]. If the error vector $e(X)$ contains v errors at locations $X^{j_1}, X^{j_2}, \dots, X^{j_v}$, where $0 \leq j_1 \leq j_2 < \dots < j_v \leq n-1$, then

$$e(X) = e_{j_1} X^{j_1} + e_{j_2} X^{j_2} + \dots + e_{j_v} X^{j_v} \quad (12)$$

Hence to determine $e(X)$, we need to know the error locations X^{j_i} 's and the error values e_j 's. So we now compute the $2t$ syndrome components S_1, \dots, S_{2t} using the received vector components $r_0, r_1, r_2, \dots, r_{n-1}$

Following step 2 mentioned before, we have

$$\begin{aligned} \sigma(X) &= (1 + \beta_1 X)(1 + \beta_2 X) \dots (1 + \beta_v X) \\ &= 1 + \sigma_1 X + \dots + \sigma_v X^v \end{aligned} \quad (13)$$

We have used here Berlekamp's iterative algorithm to find out the error location polynomial $\alpha(X)$. Substituting $1, \alpha, \alpha^2, \dots, \alpha^{n-1}$ into $\alpha(X)$ we finally get the error location numbers, which are in fact inverse of the roots of $\sigma(X)$. Next we compute the polynomial

$$Z(X) = 1 + (\sigma_1 + \sigma)X + \dots + (\sigma_v + \sigma_1 \sigma_2 \dots + \sigma)X^v \quad (14)$$

The last step in decoding requires that we find the error values at location β_i using the formula:

$$e_{j_i} = \frac{Z(\beta_i^{-1})}{\prod_{\substack{i=1 \\ i \neq 1}}^v (1 + \beta_i \beta_i^{-1})} \quad (15)$$

It may be mentioned that the algorithm's use yields the error pattern with the smallest number of errors, or in other words, yields the most probable error pattern caused by the channel noise. Finally after obtaining $e(X)$, we do $r(X) + e(X)$ to yield the corrected code vector. In our realization of the RS decoder, we have used both sequential circuit models and high level behavioural models. Their usage is governed by the ease with which their functionality can be coded into the decoder model.

4. Results and Discussion

The works involve the Matlab based simulation of the several functional blocks as represented in Fig. 3 and Fig. 4. Synthesizable models are developed for the entire code block in the transmitter as well as the Gold Code based PNG. The models have been coded using delta delays in all the signal assignments. This has been done because of the non-availability of a suitable hardware prototype-testing platform at the time the models were being developed. To test how the model as a whole behaves in the presence of noise, we have used the additive noise model to see the efficacy of our code correcting techniques. We assume only the channel to be noisy and disregard any effect of noise caused by other system components (in the transmitter and receiver). The model used here in the present work is as shown in Fig.10.

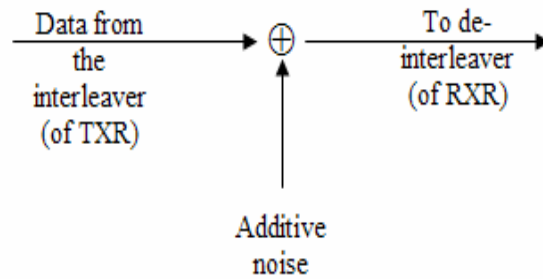


Fig. 10 Additive noise model

To introduce the additive noise into the system and subsequently measure the BER, we have used the popular MATLAB software. To simulate the noise we have used the *randn* function, which produces a suitable matrix with normally distributed random entries [25]. Also to vary the amount of noise, a scale factor has been used which when multiplied with the random entries gives us some control over the amount of noise (error) introduced. Errors in the received data obtained by the proposed simulated model of wireless mobile communication systems are recorded and reflected in the curves of Fig. 11 and Fig. 12. Three message streams are considered here for the present study; one of length 300 bits, another 600bits and the last 800 bits.

The resulting BER vs. E_b/N_0 plot is shown next (Fig.11):

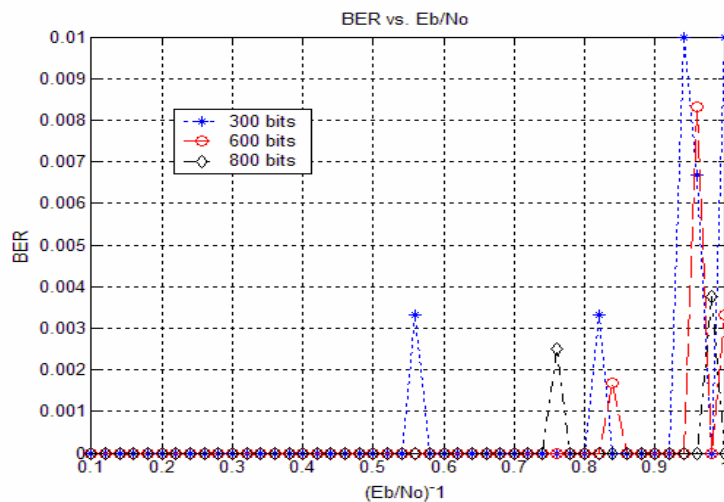


Fig. 11 BER vs. E_b/N_0 plot

From the plot we see the existence of noiseless windows. However the number of bit errors doesn't monotonically increase with increase in the value of the scale factor. This is because although this factor is increased linearly, its product with normally distributed random values doesn't monotonically increase above the noise "threshold" to produce errors in the transmitted bit stream. Also we can notice that although the number of transmitted bits has been increased the BER has decreased in some places. This is because due to the error correction incorporated into the system, numbers of bits in error have remained practically constant although the numbers of bits transmitted have increased. This in fact decreased the proportion of errors in the bit stream, which the BER reflects. Although the number of bits transmitted during performance testing is quite small yet we see that the proportion of error (BER) is comparable to various existing systems.

The next performance plot shows the number of bits actually in error. The plot is as shown (Fig.12):

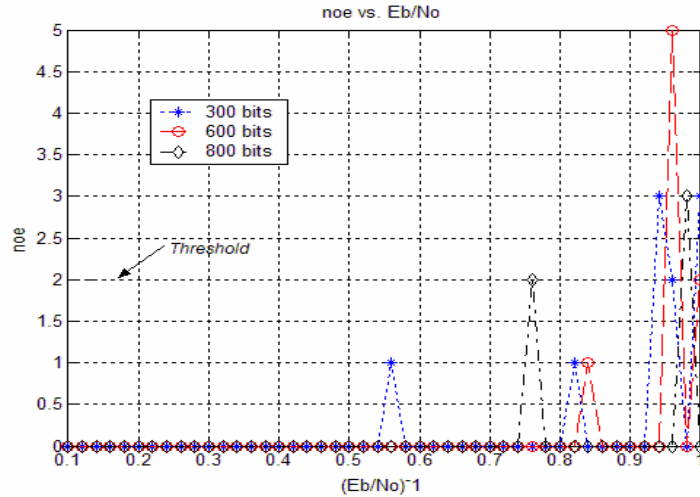


Fig. 12 noe vs. E_b/N_o plot

The plot gives us important knowledge, as we can actually know under what amount of noise how many bits are erroneous, i.e. when our code correction technique begins to fail. Here we have arbitrarily set our error threshold to 2, i.e. when there are more than two bits in error in the decoded message we consider the performance of our system to be wholly unsatisfactory. The plot shows that on the whole we can safely transmit till the noise factor exceeds 0.9. Only then the transmission becomes unreliable as our coding techniques begin to fail.

5. Conclusion

In conclusion, in the present work, an attempt is made to simulate (using MATLAB based simulation technique) a CDMA technology based wireless mobile communication system incorporating all relevant error correcting schemes for reliable communication establishment. Cellular Automata concept is considered for providing stringent security as well as authentication. The present work has huge potential to provide a cheap and efficient alternative to the existing multi-chip solutions available in the market. The simulation results (using both ModelSim and MATLAB) have been encouraging. However, the work still needs further modification and refinement of the existing models of the blocks. Only then the semi-custom VLSI solution can become a reality. The addition of automata concept has significantly improved the performance of the system so far error and security are concerned. Our result will be best judged when system with our concept will appear in practice.

Acknowledgement

Subir Kumar Sarkar thankfully acknowledges the financial support obtained from DRDO, Govt. of India (ERIP/ER/0503561/M/01/905 dt.01/08/2006).

References

1. Belcher, F. H. "Advanced Mobile Phone Service," IEEE Transactions on Vehicular Technology, vol. VT-29, May 1980, pp. 238-244.
2. Steele, R. "The Evolution of Personal Communication," IEEE Personal Communications, April 1994.
3. Haykin, S. "Communication Systems," Wiley, 2001.
4. Rappaport, T. S. "Wireless Communication: Principles and Practice," Prentice-Hall, 1996.
5. Stallings, W. "Wireless Communications and Networking," PHI, 2003.
6. Lee, W. C. Y. "Mobile Cellular Telecommunications - Analog and Digital Systems," McGrawHill, 1995.

7. Lee, W. C. Y. "Mobile Communication Engineering," McGraw-Hill Book Co., 1982.
8. Schilling et al, "Electronic Circuits - Discrete and Integrated," McGraw-Hill Book Co., 1989.
9. Sze, S. M. "VLSI Technology," McGraw-Hill, 1983.
10. Rivest, R. L. Shamir, A. and Adleman, L. "A Method for Obtaining Digital Signatures and Public Key Cryptosystems", Communication of the ACM, '01 21, 1978, pp 120-126.
11. Bach, E. and Shallit, J. "Algorithmic Number Theory", The MIT Press, 1996.
12. Stallings, W. "Cryptography and Network Security-Principles and Practice"
13. Stinson, D. R. "Cryptography: Theory and Practice," CRC Press, 1995.
14. Ashith. M. B. "1024-Bit/2048-Bit RSA Implementation on 32-Bit Processor for Public Key Cryptography," IETE Technical Review, Vol 19. no 4, 2002, PP 203-205
15. Kumar, P. and Sinha, A. "Survey of Intrusion Detection Systems and Security Audit analysis," International Conference on Quality, Reliability and information Technology, December 21-23, 2000, New Delhi.
16. Roesch, M. S. "Lightweight Intrusion Detection for Networks", Proceeding of the 1999 USENIX LISA conference, <http://www.snorl.org>, November 1999.
17. Sarkar, S. K., Karmakar, T. Kumar A., Sharma K, Pradhan, P. C. and Puttamadappa C, "Journal of Institution of Engineers, vol 87, pp-1-7, May 2006.
18. Ashenden, P. J. "The Designer's Guide to VHDL", Morgan Kaufmann Publishers, 2002.
19. Bhasker, J. "A VHDL Primer", Pearson Education, 2004.
20. Salamasi and Gilhousen, K. S. "On the System Design Aspects of Code Division Multiple Access Applied to Digital Cellular and Personal Communication Networks," IEEE VTC 91' Con. Rec. May 19-22, 1991, pp. 5762.
21. Carlson, A. et al, "Communication Systems An Introduction to Signals and Noise in Electrical Communication," Mc Graw Hill, 2002.
22. Peterson, W. W. and Brown, D.T. "Cyclic Codes for Error Detection," Proc. IRE, vol. 49, pp.228-235.
23. Lin Shu and Costello, D. J. Jr. "Error control coding fundamentals and applications" Prentice Hall Inc. 1985.
24. Johannesson, R. and Zigangirov, K. Sh. "Fundamentals of Convolutional Coding," IEEE Press, NY, 1999.
25. Harada and Prasad, "Simulation and Software Radio for Mobile Communication", Artech House, 2002, pp. 45-256.