FPGA FAULT DETECTION USING ERROR CORRECTING CODES FOR SOFTWARE DEFINED RADIO

Motoko TANIGUCHI; Kazuyuki OKUIKE; Kenta UMEBAYASHI;
Hideki OCHIAI and Ryuji KOHNO
(Division of Electrical and Computer Engineering, Yokohama National University,
Yokohama, Kanagawa, JAPAN;
motoko-t [kuike, ume, ochiai, kohno]@kohnolab.dnj.ynu.ac.jp)

ABSTRACT

In Software Defined Radio (SDR) systems, establishing reliability through partial breakdown of the system is an important issue. Generally, fault diagnosis and detection may be finely performed before shipment of Field Programmable Gate Arrays (FPGAs). However, FPGAs can be also periodically tested in the SDR terminal after shipment for maintenance. In this kind of system, the reduction of the computational burden required for the test is desirable. In this paper, we propose a remote FPGA fault detection system that can detect and pinpoint faults with low computational cost at the mobile terminal by introducing regularity in the test pattern. Specifically, our scheme employs error correcting codes for this test pattern. It is shown that there is a trade-off between the computational cost and fault detection capability in the proposed scheme.

1. INTRODUCTION

Recently, SDR [1][2] has become notable as a wireless communication device that realizes more than one system on one piece of hardware. This is realized by reconfigurable wireless functions through software download. In the SDR systems, establishing reliability and robustness against system breakdown is an important issue. When conventional wireless terminals have faults, users have to bring them to the maker for repairing or replacing the hardware. However, the SDR terminal can be remotely maintained since its terminal system is remotely self-reconfigurable. This is an advantage for both users and manufacturers.

On the other hand, by changing the configuration of the hardware, programmable device also changes the composition of the SDR. Recently, FPGAs have been adopted as a primary component of SDR [3]. This is because the FPGA performs high speed signal processing and realizes low power consumption. We therefore focus on establishing reliability in these SDR terminals. One of the fault tolerance methods of the FPGA is fault detection. If an FPGA has a faulty part, the required logic functions can avoid the use of this part through software download. Therefore, as long as the number of fault parts is less than the threshold dictated by the system redundancy, the SDR system functionality can be recovered without manual repair. Fault detection based on a Built-In Self-Test (BIST) [5], which is a circuit capable of testing itself, realizes the remote maintenance of FPGA.

Fault detection methods of the FPGA have been studied in [4][5]. In [4], a method to test whether the FPGA performs discretional function correctly is proposed. This method was used for FPGAs that were not programmed before shipment. Generally, fault detection is carefully performed before shipment over many hours. This has the purpose of improving manufacture lines from the fault part and cause of any detected faults. However, when the FPGA is periodically tested within the SDR terminal after shipment for maintenance, the purpose is only detecting fault parts. In this case, the reduction of the computational burden is desirable since the terminal's equipment is limited.

Therefore, we propose a remote FPGA fault detection system for SDR system and an efficient scheme for detecting and pinpointing faults by introducing regularity in the test pattern. Our scheme employs error correcting codes (ECCs) for this regularity. Using the proposed system, computational burden can be reduced compared to the non-coded method. However, there is a trade-off between the reduction of the computational burden and degradation in the probability of aliasing (the probability of not detecting a fault correctly). This is because the proposed scheme has some undetectable fault patterns. Correspondingly, we show the trade-off and the basis of selecting error correcting code for individual FPGA of the SDR terminal.

This paper is organized as follows: In Sect. 2, the remote maintenance of the SDR terminal is presented, In Sect. 3, the structure of the FPGA and fault model are discussed. In Sect. 4, we propose a remote FPGA fault detection system. In Sect. 5, we evaluate the proposed scheme and system. Finally, conclusions and future research subjects are given in Sect. 6.

Table 1 Fault of the SDR terminal.

Faults class			Remote repair
Terminal	Software	Bug	Possibility
		Reversal of Memory	Possibility
	Hardware	Analog circuit	Impossibility
		Programmable device (FPGA, DSP, etc.)	Possibility

2. REMOTE MAINTENANCE OF THE SDR TERMINAL

If the SDR terminal has faults, it can communicate erroneously or interfere with other communication systems. Therefore, establishing reliability when the SDR terminal has faults is an important issue in the SDR systems.

Since conventional wireless terminal is composed by analog hardware circuits, in the case of fault, user must bring the terminal to the manufacture for fault detection and a hardware replacement. On the other hand, The SDR terminal can be reconfigured and the maintenance can be done remotely since reconfigurable devices compose it. Therefore, users are not needed to bring the SDR terminal to manufacturer when it has faults. This is a significant advantage for users and manufactures.

In the following, we define the Base Station as the entity that controls the remote maintenance of the SDR terminal, the fault detection, and the fault diagnosis as pinpoint fault parts.

Faults of the SDR terminal are classified as in Table 1. Bugs and reversals of memory cause software faults. As bugs are software glitch, the software debugging can repair the terminal remotely. Reversal of memory is caused by static electricity or alpha ray. This is repaired by the same software re-implement remotely.

Hardware faults are divided into analog circuits and programmable devices. Analog circuit faults need the hardware replacement. Therefore, these cannot be repaired remotely. On the other hand, programmable device such as FPGA can be repaired using itself without hardware replacement because it can be reconfigured using the configuration data that avoid the use of the fault part. Furthermore, fault detection based on BIST is realized by software. Therefore, FPGA fault can be repaired remotely.

3. FPGA AND FAULT MODEL

We aim to test for Configurable Logic Block (CLB) faults [4][5] of FPGA, especially, *Xilinx Virtex2* [6].

3.1. Structure of FPGA

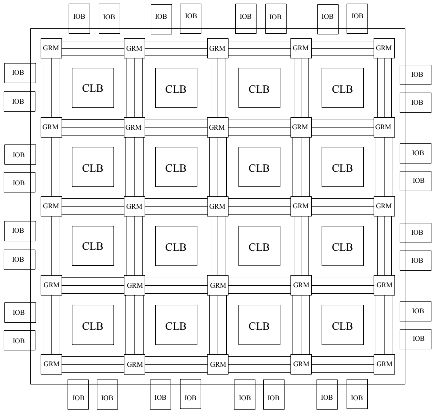


Fig.1 Structure of FPGA

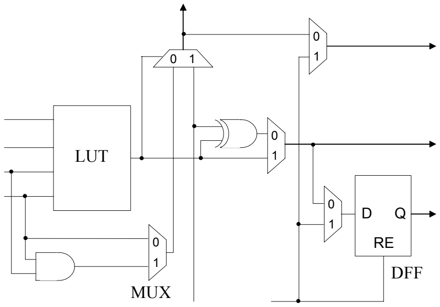


Fig. 2 Structure of LC.

Figure 1 shows the structure of an FPGA [6]. The CLBs provide the functional elements for constructing logic. The input/output blocks (IOBs) provide the interface between the package pins and the CLBs. CLBs are interconnected through a general routing matrix (GRM). The GRM comprises an array of routing switches located at intersections of horizontal and vertical routing channels.

3.2. Structure of CLB and Fault Model

Each *Virtex* CLB contains four logic cells (LCs) as shown in Figure 2. LC includes a 4-input look-up table (LUT), D-type flip-flop (DFF), and multiplexer (MUX). A LUT implements combinational logic as a 2^4 x 1 memory composed of synchronous RAM (SRAM). When one applies an input pattern to a LUT, the table selects a SRAM addressed by input pattern, and the output of cell provides the function's value. In implementing FPGA, one loads the memory with the bit pattern corresponding to the function's truth table. In the CLB, the connections among the input and output lines, the LUT, and the DFF are configured as multiplexer (MUX) controlled by SRAM.

In this paper, we assume the following faults [4][5].

(a) Stuck at fault on every signal line in the CLB, except the clock and reset line.

(b) Stuck at fault on every SRAM cell in the LUT.

(c) Functional faults for read-out decoder of the LUT, such as no selection, wrong selection, and multiple selections.
(d) Functional fault for the DFF, except the clock and reset function.
(e) Functional faults for MUXs.

Multiple faults, given by any combinations of the above faults in one CLB are also considered.

FPGA faults are detected by analysis of the outputs corresponding to test inputs. The detection is limited by certain logic functions caused by faults. In this paper, the proposed scheme is based on [4][5], and signal pass condition of *Xilinx Virtex2* is based on [5][6].

4. PROPOSED SYSTEM

In FPGA fault detection of a remote SDR terminal, the avoidance of severe computational burden on the terminal is required because the terminal's dimensions are limited. Therefore, we propose an efficient scheme for fault detection and a remote fault detection system of FPGA for SDR system.

In the proposed system, we assume that the SDR terminal has no fault that way cause disconnection or erroneous communication with base station.

4.1. FPGA Hierarchical Fault Detection Using Error Correcting Code

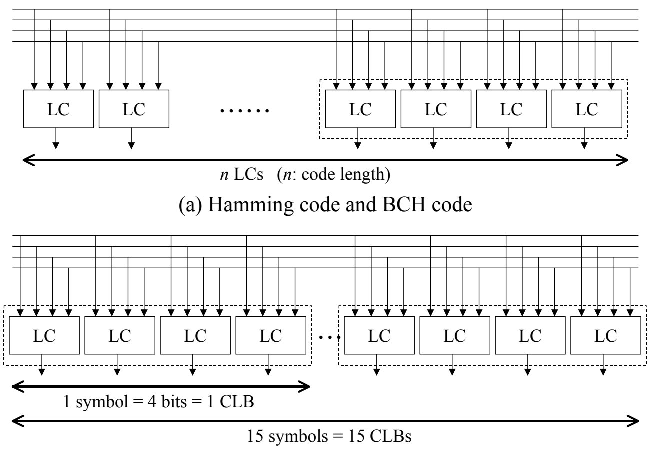
The proposed scheme reduces computational burden for testing the FPGA of an SDR terminal.

Generally, FPGA fault detection is based on the analysis of the outputs corresponding to test inputs. FPGA faults can be detected because outputs are reversed when FPGA has faults.

The proposed scheme detects faults based on the regularity which is beforehand given to the output to the test inputs. In this scheme, the terminal only detected the faults and do not pinpoint them. Accordingly, since the test areas are arranged hierarchically, the computational burden is considerably reduced. If faults are detected upper level or FPGA has no fault, test can be completed at upper level.

Our scheme employs ECCs for that regularity. This scheme uses decoding error detection in order to detect faults in output analysis. Consequently, a fault locating an error correction equals locating. ECC adds redundant symbols as check symbols to the information symbols so that errors can be corrected or detected. We use them for neither error detecting nor correcting but fault detecting and pinpointing. Also, since the check symbols of ECC can be used for fault detection, they are not redundant.

In the proposed scheme, we use cyclic codes such as the Hamming code, BCH codes, and RS codes. With these codes, the error can be efficiently detected using simple shift-registers and logic elements. Figure 3 shows a way to output these codes in the proposed scheme. Here, we assume that LC has only one output. Other outputs are assumed to be the same and therefore this supposition is not effective for evaluation of the proposed scheme.



(a) Hamming code and BCH code

(b) GF(2^4) RS code

Fig.3 The way to output ECCs.

Table 2 The test levels and computing items.

Test level	Output code		Computing item
	Hamming code & BCH code	GF(2^4) RS code	
1	Module Implemented Part		Plurality of level 2
2	Code length pieces of LCs	15 CLBs	Syndrome
3	1 LC	1 CLB	Error locator
4	--	1 LC	Error evaluator

Using the proposed scheme, FPGA test is realized hierarchically as shown in Table 2. Therefore, if FPGA has no fault at the level 1, test can finish without carrying the analysis to the last level.

4.2. Remote FPGA Fault Detection System of the SDR Terminal

The proposed system can be used in any application of an SDR composed with FPGAs. FPGA faults are repaired remotely by fault detection based on BIST and software download both controlled by the base station, because in general, base station can handle higher computational load than the mobile terminals. Therefore, in the terminal, operation with little complexity is desirable. Figure 4 shows the protocol of the proposed system, which is summarized as follows;

1. The base station requests the FPGA data about fault rate and fault part to the SDR terminal.
2. The SDR terminal sends the FPGA data of itself to the base station.

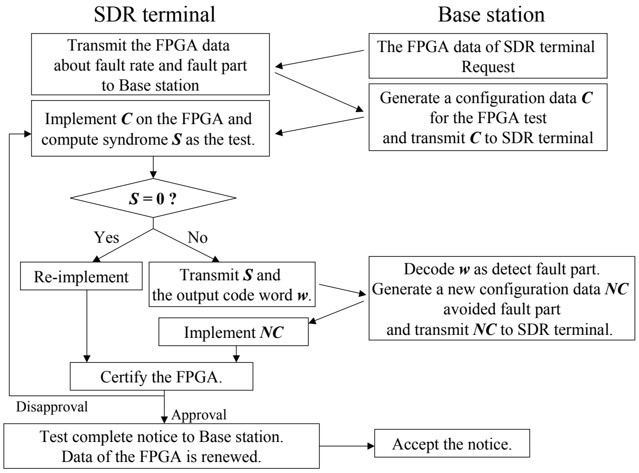

SDR terminal
Base station
Transmit the FPGA data about fault rate and fault part to Base station
The FPGA data of SDR terminal Request
Generate a configuration data C for the FPGA test and transmit C to SDR terminal
Implement C on the FPGA and compute syndrome S as the test.
S = 0 ?
Yes
No
Re-implement
Transmit S and the output code word w.
Decode w as detect fault part. Generate a new configuration data NC avoided fault part and transmit NC to SDR terminal.
Implement NC
Certify the FPGA.
Disapproval
Approval
Test complete notice to Base station. Data of the FPGA is renewed.
Accept the notice.

Fig. 4 Protocol of the proposed system.

3. The base station generates a configuration data C for test based on the terminal FPGA data, and sends it to the SDR terminal.
4. The terminal downloads C and performs minimum for fault detection whether the FPGA has faults or not. In the proposed scheme, syndrome S at level 1 should be only computed on FPGA. When it is ascertained that FPGA has fault at level 1, which means $S \neq 0$, the terminal sends S and output to the base station. If FPGA has no fault, which means $S = 0$, the SDR terminal re-implements the same configuration data.
5. If FPGA has any fault, the base station receives S and output, and analyzes them for pinpointing faults locate, and generates a new configuration data NC which avoids the fault part.
6. The SDR terminal downloads NC and implements it on FPGA. If FPGA's performance certified, the SDR terminal sends the complete notice to the base station and renews data of FPGA. If not, FPGA should be re-tested.

5. EVALUATION OF THE PROPOSED SYSTEM

In the proposed scheme, we can select EEC for the test. In this section, we evaluate the proposed scheme and present the basis for the selection. We compare the proposed scheme to the non-coded method that detects faults by comparing every output bit if a given output to the correct sequence held in the memory. In this method, each fault is pinpointed in a bit by bit basis.

5.1 Classification of Codes by FPGA Fault Situation

The base station estimates the FPGA fault situation from the average rate of faults that had been sent from the terminal.

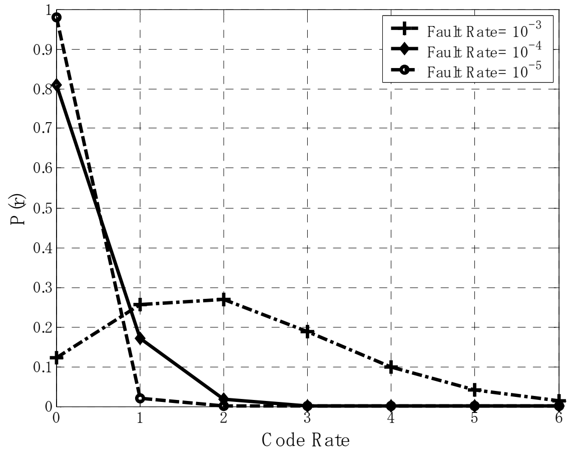


Fig. 5 The probability of r faults in 2100 LCs.

Table 3 The classification of codes by FPGA fault pattern.

		LC fault # in one code word			
		1	2	3	4
CLB #	1	Hamming (15,11) BCH RS S_4EC	Hamming (15,7) BCH RS S_4EC	(15,5) BCH RS S_4EC	RS S_4EC
	2	--	Hamming (15,7) BCH RS D_4EC	(15,5) BCH RS D_4EC	RS D_4EC

A Poisson distribution can describe LC faults. The reason being that very law ratio faults break because the number of LC is large. If the average rate of faults is λ then, the probability of r faults in n LCs is given by

$$P(r) = \frac{(n\lambda)^r e^{-n\lambda}}{r!} \tag{1}$$

The average rate of fault is given by the rate that an item which has been working breaks down within a specified time. This reduces in the early period of service and is constant until the item is redundant. Here, we assume that LC faults are not correlated with each other. The probability of faults in 2100 LCs given by Eq. (1) is shown in Figure 5.

It is assumed that an ECC is output from near LCs since the test area should be laid out so as to make it easier to avoid the fault part. In the proposed scheme, the LC is the minimum item in which faults can be detected. The fault pattern is therefore defined as the number of LC faults in one code word and the number of CLB dispersed LC faults as shown in Table 3. According to this table, ECC used in the test can be selected by checking the circuit area, probability of aliasing, and the average information quantity as basis.

5.2 Probability of Aliasing

In the proposed scheme, errors caused by faults within the error correcting capability of ECC can all be detected. The

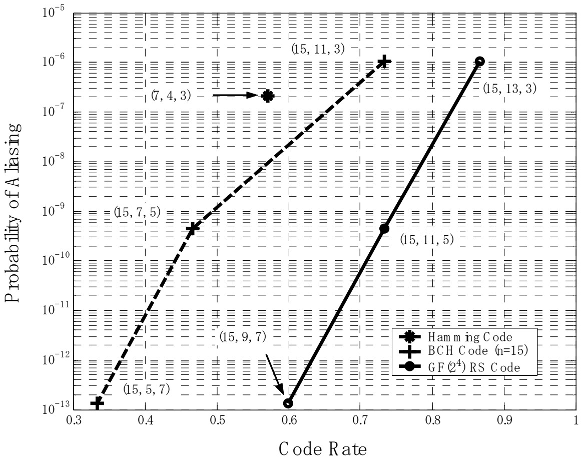


Fig. 6 Probability of aliasing.

probability of aliasing, denoted P_{al}, is the probability that the test cannot detect the fault correctly and is represented as the sum of the following:

(a) The probability of not detecting a fault when the output is included in the error correcting sphere of a different code, denoted P_e.

(b) The probability of incorrectly detecting a faulty part because the output is included in the error detecting sphere, denoted P_d.

Additionally, using the probability of a correct detection, denoted P_c, P_{al} can be represented as follows.

$$P_{al} = P_e + P_d = 1 - P_c \tag{2}$$

P_c equals the probability of correct decoding. When the FPGA fault rate is λ, in the case of a $(n, k, d_{\min})$ code, which can correct t errors, P_c becomes [7]

$$P(\lambda) = \sum_{i=0}^{t} {}_nC_i \lambda^i (1-\lambda)^{n-i} \tag{3}$$

P_c for a RS code over GF(2^m) is given by Eq.(3) using the probability of a symbol error, denoted λ_s, as follows.

$$\lambda_s = 1 - (1-\lambda)^m \tag{4}$$

Figure 6 shows P_{al} in the proposed scheme when $\lambda = 10^{-4}$. P_{al} increases as the code rate increases. This is because the patterns that have a number of faults exceeding t at the same output cannot be detected. These patterns increase as the code rate increase. The non-coded method's P_{al} is 0 in all tests since each fault is pinpointed in a bit by bit comparison with the correct sequence.

5.3 Check Circuit Area

In the remote fault detection system, avoidance of severe computational burden on the terminal is required. Here, we compare the check circuit area versus computational burden. It is good for the terminal that the check circuit is small. In the proposed system, the syndrome is computed on the terminal. The cyclic codes use simple shift-register based on their representation using polynomials for syndrome generation [7][8]. Figure 7 shows the syndrome generator for a binary cyclic (n, k) code with generator polynomial given by

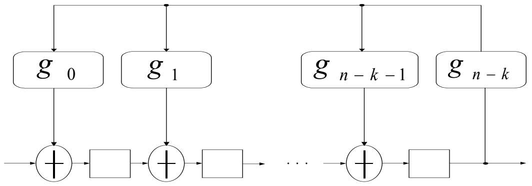


Fig. 7 Syndrome generator for a binary cyclic code.

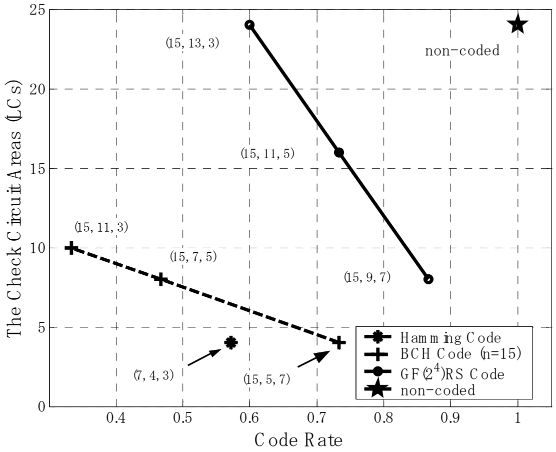


Fig. 8 The check circuit areas (The number of LCs)

$$G(x) = g_{n-k}x^{n-k} + g_{n-k-1}x^{n-k-1} + \Lambda + g_1 x + g_0 \tag{5}$$

In the case of a RS code over *GF*(2^m), a field element of *GF*(2^m) can be represented by *m* bits and can be transferred within the circuit in parallel.

In the non-coded method, the check circuit consists of a comparator and memory for the correct sequence and output. In both methods, the test time on the FPGA depends on the area under testing.

Figure 8 shows the check circuit areas in the proposed scheme when implemented on the FPGA, *Xilinx Virtex2*. In the proposed scheme, the check circuit area is smaller than the non-coded method, because in the proposed scheme, the check circuit does not need memory for the correct sequence. However, the check circuit area becomes larger area as the code rate decrease. In the case of RS code, $(15,9,7)$ RS code is the same as the non-coded method.

5.4 The Average of The Transmit Information Quantity

In the proposed system, when faults are detected the SDR terminal sends both the syndrome and the output to the base station. Assuming that the SDR terminal sends this whenever a fault is detected, the average information quantity required is computed as follows.

N is assumed to be the number of LCs under test using the code with the code length n, error correcting capacity t, and estimated that they have R faults. Here, the information quantity sent to the base station by the terminal is $s+n$ per fault occurrence, where s is the number of syndrome bits. The number of transmission is determined from the number of codes that dispersed faults, where the number of dispersed patterns is denoted as u. Using a_{ur} as the number of the code that has r faults ($r \le t$), disperse patterns are represented as follows.

$$\begin{bmatrix} a_{11} & \Lambda & & a_{1r} \\ \mathrm{M} & \mathrm{O} & & \mathrm{M} \\ a_{i1} & a_{ik} & & a_{ir} \\ \mathrm{M} & & \mathrm{O} & \mathrm{M} \\ a_{u1} & \Lambda & & a_{ur} \end{bmatrix} \tag{8}$$

Note that Matrix (8) satisfies Eq. (9).

$$\sum_{k=1}^{r} r \cdot a_{ik} \le R \quad (i = 1,2,\Lambda\ ,u) \tag{9}$$

The number of transmissions is given by

$$q(u) = \sum_{k=1}^{r} a_{uk} \tag{10}$$

Then the average of the number transmission when N LCs have r faults is:

$$T(i) = \sum_{i=1}^{u} q(i) \cdot \frac{{}_{N/n}C_{q(i)} \cdot \prod_{k=1}^{r} ({}_{n}C_{k})^{a_{ir}}}{{}_{N}C_{r}} \tag{11}$$

LC faults can be described by a Poisson distribution given by Eq. (1). Based on the above, the average information quantity C is:

$$C(r) = \sum_{j=1}^{R} (s+n) \cdot T(i) \cdot \frac{P(j)}{\sum_{l=1}^{R} P(l)} \tag{12}$$

Figure 9 shows the numerical results of Eq. (12), when N=2100. The average information quantity required increase as the code rate decreases. In the non-coded method, the SDR terminal may send to the base station only information of fault part.

5.5 Selection of Error Correcting Code for Fault Detection

ECC used in the test can be selected by check circuit area, probability of aliasing, and the average information quantity from Table 3. FPGA faults can be estimated from the fault rate which is known before the test. The code, which has the lowest probability of aliasing and computational burden on the terminal under the fault situation estimated, should be selected. If such code is selected, the code is selected based on an optimal basis, probability of aliasing, check circuit area or the average information quantity.

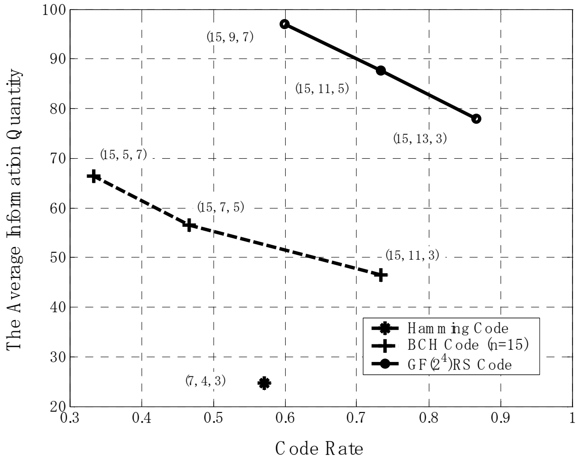


Fig. 9 The average information quantity.

6. CONCLUSIONS

In this paper, we proposed the remote FPGA fault detection system with error correcting codes. We showed that the proposed system has the trade-off between the computational burden and probability of aliasing. Therefore, it is shown there is the trade-off and the basis of selecting error correcting code.

Our future work includes how to test the terminal fault which may cause erroneous communication.

7. REFERENCES

[1] J. Mitola, "Software Radio," *IEEE Commun. Mag.*, pp. 24-38, May 1995.

[2] J. Mitola, "Software Radio Architecture: A Mathematical Perspective," *IEEE Journal on Selected Areas in Communications*, Vol. 17, No. 4, pp. 514-538, April 1999.

[3] M. Cummings, S. Haruyama, "FPGA in the Software Radio," *IEEE Commun. Mag.*, pp. 108-112, Feb. 1999.

[4] T. Inoue, S. Miyazaki, H. Fujiwara, "Universal Fault Diagnosis for Lookup Table FPGAs," *Design & Test Computers, IEEE*, Vol. 15, No. 1, pp. 39-44, Jan.-March 1998.

[5] N. Itazaki, F. Matsuki, Y. Matsumoto, K. Kinoshita, "Built-In Self-Test for multiple CLB faults of a LUT FPGA," *Test Symposium, ATS'98, Proceedings, Seventh Asian, IEEE*, pp. 272-277, 2-4 Dec. 1998.

[6] Xilinx Inc., *Virtex 2.5V Field programmable Gate Arrays*, DS003-2 (v2.6), www.xilinx.com, 19 July 2001.

[7] Robert H. Morelos-Zaragoza, *The Art of Error Correcting Coding*, John Wiley & Sons, Ltd, 2002.

[8] Richard E. Blahut, *Algebraic Codes for Data Transmission*, Cambridge University Press, 2003.

Copyright Transfer Agreement: The following Copyright Transfer Agreement must be included on the cover sheet for the paper (either email or fax)—not on the paper itself.

“The authors represent that the work is original and they are the author or authors of the work, except for material quoted and referenced as text passages. Authors acknowledge that they are willing to transfer the copyright of the abstract and the completed paper to the SDR Forum for purposes of publication in the SDR Forum Conference Proceedings, on associated CD ROMS, on SDR Forum Web pages, and compilations and derivative works related to this conference, should the paper be accepted for the conference. Authors are permitted to reproduce their work, and to reuse material in whole or in part from their work; for derivative works, however, such authors may not grant third party requests for reprints or republishing.”

Government employees whose work is not subject to copyright should so certify. For work performed under a U.S. Government contract, the U.S. Government has royalty-free permission to reproduce the author's work for official U.S. Government purposes.