computing@tanet.edu.te.ua
www.tanet.edu.te.ua/computing

ISSN 1727-6209
International Scientific Journal of Computing

POSSIBILITIES OF INCREASING THE RELIABILITY BY METHODS OF SOFTWARE AND TIME REDUNDANCY

Peter Široký, René Harťanský, Ján Petrilák

Alexander Dubcek university of Trencin, Faculty of mechatronics, Dpt. of Mechatronics, Studentska 1, 911 50 Trencin, Slovak Republic
E-mail: siroky@tnuni.sk, rene@yhman.tnuni.sk, petrilak@tnuni.sk

Abstract: *This paper includes some possibilities how to solve problems with the improvement of systems reliability on the field where traditional methods of EMC are not sufficient. Two basic types of methods will be introduced. First, methods of software redundancy and second methods of time redundancy.*

Keywords: *computing, error, failure, two rail logic, software, hardware, decentralized systems.*

1 INTRODUCTION

The methods of EMC are concerned with the radiated and conducted emissions on systems, predominantly in hardware platform. In our article we would like to introduce some methods of elimination of the influences of emissions on transmission or computed data by using the software method. In most cases it is possible to detect errors or failures on existing systems via using these methods. These are always suitable to improve the communication between decentralized systems. The methods which can do that are methods of software and time redundancy.

The methods of software and time redundancy possess an advantage of possibilities to use them without additional modification of hardware platform. However in some cases the increase of reliability is possible only with changing the hardware platform, or changing the whole conception of design. The major point of using software and time redundancy methods lay on microprocessors systems and system computational technique where there is a high demand for specificity or radiated emissions susceptibility or conducted emissions susceptibility.

2 SOFTWARE REDUNDANCY METHOD

The basic idea of this method is in using insured code on communication between decentralized systems or can be used in processing with signals thus being able of easy detection of a hardware failure. From all possible codes we would like to present only the most preferred ones. Often the codes with constant of Haming's distance (e.g. Hd = 2 or more) are used, which can detect possible errors on transmitted data. In case of high value Hd the correction is possible. For detection and repair the following equation can be set:

$$Hd \geq 1+2c+d$$

where c - represents a number of bits to be corrected
d - represents number of errors to be detected

For example in case of Hd=2, using 3 bits, we obtain only four cases from eight possible.

0 0 0
0 1 1
1 0 1
1 1 0

However when only one error is detected, correction is impossible (Figure 1).

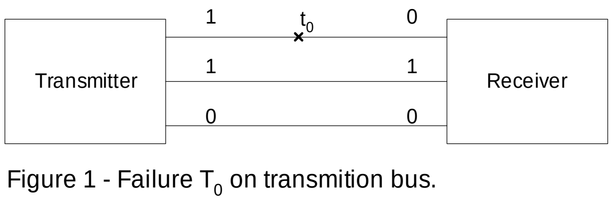

Figure 1 - Failure T_0 on transmition bus.

In Hd=3, using 3 bits, only two possibilities remain.

0 0 0
1 1 1

in this case it is possible to detect two errors and one error can be corrected.

Another method is using a code with parity in cases where there are more possibilities of parity applications:

a) Word parity

One of the easiest ways is simply adding the parity bit at the end of the word (Figure 2).

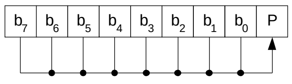

Figure 2 - Word parity.

b) Cross parity

This method provides much better word security than the one described previously. (Figure 3)

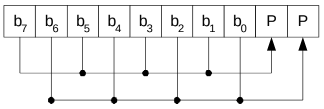

Figure 3 - Cross parity.

c) Parity overlapping

This method offers a possibility of correcting errors however the redundancy is larger (Figure 4)

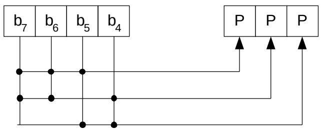

Figure 4 - Parity overlaping.

d) other

Seldom weight codes M of N (codes with constant number of "1") are used. These codes consist of original word along with the same word in negation form. Main disadvantage of these codes is 100% redundancy and ability of detecting only one error (that same can be done with parity codes with far lower redundancy).

Original code	Weight code 3 of 6
0 0 0 0	0 0 0 0 1 1 1 1
0 0 0 1	0 0 0 1 1 1 1 0
0 0 1 0	0 0 1 0 1 1 0 1
0 0 1 1	0 0 1 1 1 1 0 0

The Izocodes N of M are codes with constant numbers of "1" and comparing to weight codes they can reliably detect only one error, or more one-sided errors (e.g. transition from 0 to 1)

Original code	Izocodes 2 of 6
0 0 0 0	0 0 0 1 1
0 0 0 1	1 1 0 0 0
0 0 1 0	1 0 1 0 0
0 0 1 1	0 1 1 0 0

Double codes hide very interesting possibilities. The basic idea of these codes is in repeated transmission. After the original code has been transmitted, the second transmission is done but in a form of negation. These codes do not need to enlarge the existing data buses because they are transmitted twice within the some number of bits. The hardware changes are not necessary. Another advantages of double codes are their possibilities of failure detection on data buses (e.g. short circuit of any data wires on ground or power supply). However, the detection of short circuit between wires will still be a problem since it can not be detected by this method. This advantage decreases the transmission speed because of 100% redundancy.

Last but not least among the methods of improving the reliability we would like to present are the methods based on check sum.

Well known are:

a) – single precision (Figure 5). The principle of this method is to add check sum at the end of the original data. After receiving this a new check sum is created. These check sums are compared and consequently recognition of an error is possible on received data. If checksums are identical data is correct.

	Original data with overflow flag	Data ready for transmition without flag
	0 1 1 0	0 1 1 0
	0 1 0 1	0 1 0 1
	<u>1 0 0 0</u>	<u>1 0 0 0</u>
check sum	10 0 1 1	0 0 1 1

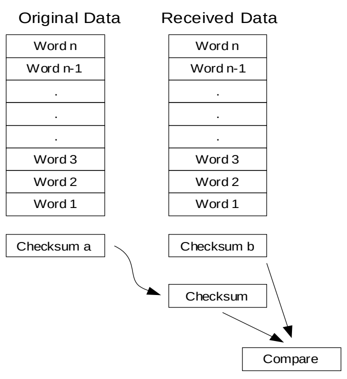

Figure 5 - Single precision.

The problem can appear because ignored the overflow flag (bit) is.

	Transmitted data	Received data
	0 1 1 0	1 1 1 0
	0 1 0 1	1 1 0 1
check sum	1 0 0 0	1 0 1 1
check sum after transmit		1 0 1 1

It can be seen that the check sum is equal to opposite continuous failure (Figure 6).

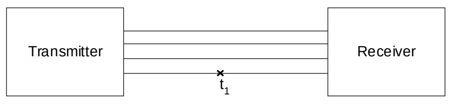

Figure 6 - Failure T_1 on transmition bus.

b) – double precision

By this method the problem with overflow flag will be eliminated. The only difference comparing to a single precision method is that the double precision method uses the whole overflow flag. The increase of redundancy is negligible in case of insuring larger number of bytes. In following table the increase of redundancy is presented (compare to none insured transmit with various number of bytes).

Non insured transmission	Single precision	Double precision
4 Byes	20%	33%
6 Bytes	14%	25%
8 Bytes	11%	20%
10 Bytes	9%	17%

		Orig. data with flag	Data ready for transmission with flag
		0 1 1 0	0 1 1 0
		0 1 0 1	0 1 0 1
		1 0 0 0	1 0 0 0
Check sum	0 0 0 1	0 0 1 1	0 0 1 1
			0 0 0 1

		Transmitted data		Received data
		0 1 1 0		1 1 1 0
		0 1 0 1		1 1 0 1
Check sum	0 0 0 1	0 0 1 1	0 0 1 0	0 0 1 1
Check sum after transmit			1 0 1 0	1 0 1 1

Comparing the check sums, the error will be detected (Figure 6).

c) – Honeywell

This method divides bytes into two columns (Figure 7) and the check sum will be added. The transmission is conducted with original number of data wires (similar to a method of single or double precision). Added redundancy is identical to the method of double precision.

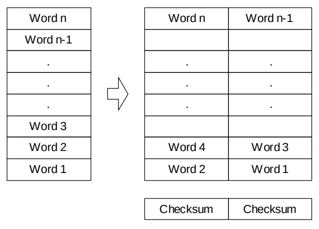

Figure 7 - Honeywell method.

Original data		data after modification	
0 0 0 1		0 0 0 1	0 1 1 1
0 1 1 1		0 0 0 0	0 1 1 0
0 0 0 0	check		
0 1 1 0	sum	0 0 0 1	1 1 0 1

Transmitted data	Received data	Data after modification	
0 0 0 1	1 0 0 1	1 0 0 1	1 1 1 1
0 1 1 1	1 1 1 1	1 0 0 0	1 1 1 0
0 0 0 0	1 0 0 0	0 0 1 0	1 1 0 1
0 1 1 0	1 1 1 0		
0 0 0 1	1 0 0 1	Received check	
1 1 0 1	1 1 0 1	sum	
		1 0 0 1	1 1 0 1

Error on data bus is detected again (Figure 6).

Another way is using the cycle codes, the arithmetical codes etc.

3 METHODS OF TIME REDUNDANCY

The basic idea is to do any operation e.g. numerical or the data transfer repeatedly - then compare the results. The advantage of this method is that it is able to detect short random errors or serious permanent failures. These methods mostly do not require any hardware platform changes. Software changes will usually do. The disadvantage is the length of the computing time.

4 THE STRATEGY OF TRANSIENT ERRORS DETECTION

The principle is presented in Figure 8. Every operation is done repeatedly. The interpretation of results can be various. In one case it can be used only as an error indicator, in other case it may contain the corrected result (after voting).

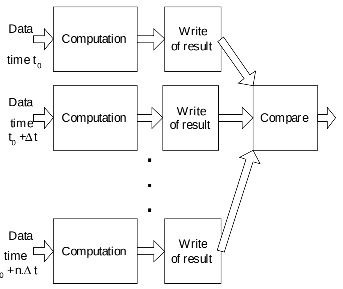

Figure 8 - Strategy of transient errors detection.

5 THE STRATEGY OF PERMANENT FAILURE DETECTION

In this method the calculation is again done repeatedly (in most cases two times: - second step in calculation is realized with inverted data). This is a software solution of Two Rail Logic – in hardware field (Figure 9).

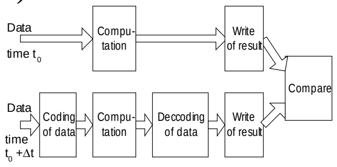

Figure 9 - Strategy of permanent failure detection.

To fulfill the previous, the following equation has to be valid:

$$f(x) = \overline{f}(\overline{x})$$

in which

$f_d(x)$ is the dual function of $f(x)$ function (if $f_d(x) = \overline{f}(\overline{x_1}, \overline{x_2}, ...\overline{x_n}))$

If $f_{vd}(x)$ is the function with own duality and if $f_{vd}(x) = x_{n+1}.f(x) + \overline{x_{n+1}}.f_d(x)$, then complementarily inputs generate complementarily outputs, the outputs are identical by Two Rail Logic.

Recomputing with Shifted Operands –RESO:

1. Step – computing with original operands (Figure 10)

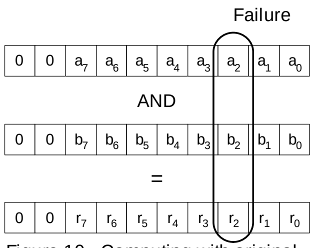
Failure
0 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0
AND
0 0 b_7 b_6 b_5 b_4 b_3 b_2 b_1 b_0
=
0 0 r_7 r_6 r_5 r_4 r_3 r_2 r_1 r_0

Figure 10 - Computing with original operands.

2. Step – recomputing with one times shifted operands (Figure 11)

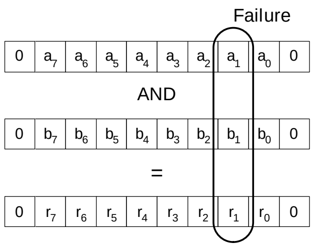
Failure
0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0 0
AND
0 b_7 b_6 b_5 b_4 b_3 b_2 b_1 b_0 0
=
0 r_7 r_6 r_5 r_4 r_3 r_2 r_1 r_0 0

Figure 11 - Recomputing with shifted operands.

3. Step – recomputing with two times shifted operands (Figure 12)

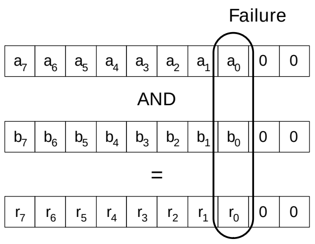
Failure
a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0 0 0
AND
b_7 b_6 b_5 b_4 b_3 b_2 b_1 b_0 0 0
=
r_7 r_6 r_5 r_4 r_3 r_2 r_1 r_0 0 0

Figure 12 - Recomputing with two times shifted operands.

Result after shifting

After voting we obtain correct result (Figure 13).

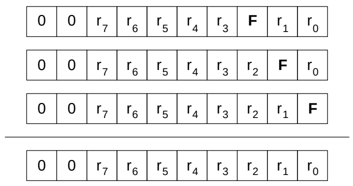
0 0 r_7 r_6 r_5 r_4 r_3 **F** r_1 r_0
0 0 r_7 r_6 r_5 r_4 r_3 r_2 **F** r_0
0 0 r_7 r_6 r_5 r_4 r_3 r_2 r_1 **F**
0 0 r_7 r_6 r_5 r_4 r_3 r_2 r_1 r_0

Figure 13 - Result after voiting.

Recomputing with Swapped Operands –RESWO: (Figure 14).

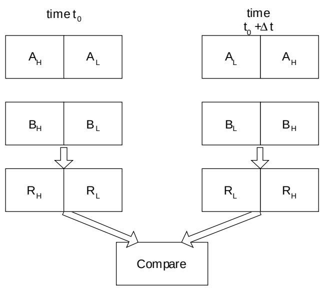

Figure 14 - Recomputing with swapped operands

Recomputing with Duplication with Comparison – REDWC:

1. Step

The computation unit will be tested first with the lower half of data then the results will be compared and overflow flags will be also compared (Figure 15).

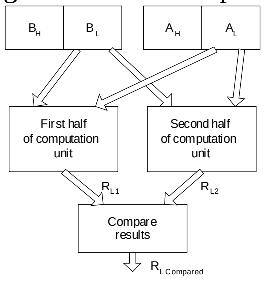

Figure 15 - Computing with the lower half of data.

2. Step

Then the computation unit will be tested with the higher half of data then the results will be compared and overflow flags will be compared (Figure 16)

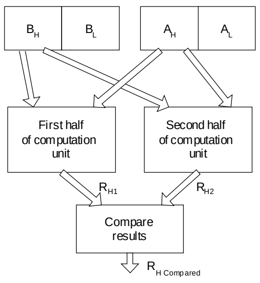

Figure 16 - Computing with the higher half of data.

6 CONCLUSION

In our article we wanted to introduce interesting possibilities of improving the reliability of existing systems, or software possibilities in the field of errors or failures detection. These methods can be easily applied e.g. in insuring reliability controlling. Also they can be used when evaluating results in decentralized systems in surrounding with radiated or conducted emissions. (In such case usual resources of EMC are not sufficient for preventing the intersection of emissions into transmission channel or system itself).

REFERENCES

[1] *Z. Manna: "Mathematical Theory of Computation", New York, Mc Graw Hill 1974*

[2] *G.J. Myers: "Software Reliability: Principles and Practices", New York, John Wiley & Sons 1976*

[3] *F.F Sellers, M.Y. Hsiao, L.W. Bearnson: "Analyzing Errors with the Boolean difference", IEEE Trans. on Computers*

[4] *D. Maga, W. Demski: "The Accuracy of FEM - FDM Magnetic Field Solution Based Torque Computation", 7th international IGTE Symposium on Numerical Field Calculation in Electrical Engineering, pp. 395-398, 23.-25.9. 1996, Graz, Austria*

[5] *D. Maga: "Elektromechanika 2 – návody na laboratórne cvičenia", Trenčianska Univerzita, 2002, ISBN 80-88914-59-0 – VŠ skriptum*

[6] *Julényová, A.: Computer Models, UNINFOS 2001, medzinárodná konferencia, 2001, Nitra.*

__Peter ŠIROKÝ__ (MSc.) was born in Trenčin on 26.03.1973. He has received his MSc. (1997) on Faculty of Information and Security Systems in University of Žilina. Recently he is head of power laboratories of Alexander Dubček University of Trenčín. His research interests are Fault tolerance systems, electric system design, electrical machines.

René HARŤANSKÝ *(MSc., PhD.) was born in Banská Bystrica on May 13th, 1969. He received the MSc. degree in 1992 and then PhD. degree in electrical engineering from the Faculty of Electrical Engineering of the Slovak Technical University (FEE STU). He is currently the Assistant professor at the Department of Mechatronics TnU of Alexander Dubcek. His research interest are the computer modelling EMC phenomena, antennas and propagation.*

Ján Petrilák *(Doc., Ing., CSc.) was born in Kremna on 19 June 1938. graduated from the Faculty of Electrical and Machine Engineering, the University of Transport, Žilina, in 1969. He received his CSc. degree in 1978. In 1989 he became a full time lecturer at the Department of Theoretical and Applied Electrotechnics of Faculty of Electrical Engineering of the University of Žilina. Since 1993 he is Associate Professor Theoretical Electrotechnics. His research interests include application of theoretical electronics and electric measurements on electric machines.*