Self-Checking Decoder Using Hardware Redundancy

Khadija F.O.Algheitta[1], Amal J.Mahfoud[2], Ali H.Maamar[3], Khaled M.Alnegrat[4].
[1]Department of Computer Engineering, College of Electronic Technology Bani Walid, Libya, Khagitta@yahoo.com
[234]Department of Computer Engineering, College of Electronic Technology Bani Walid, Libya.

***Abstract*— A binary decoder is a combinational circuit that converts binary information from n input lines to maximum of 2n unique output lines. But there is no guaranty that the decoder always produce the correct output due to some types of faults. The characteristics of these types of faults render them undetectable by standard test strategies. The detection of intermittent faults requires the use of Concurrent Error Detection technique, which continuously monitors the operation of circuits and compares them with some known reference. Concurrent Error Detection is an important technique in the design of system in which dependability and data integrity are important. This can be achieved by incorporating some form of redundancy into the system. One method of implementing Concurrent Error Detection in Very Large Scale Integrated circuit is through the use of hardware redundancy. This paper investigates the use of hardware redundancy into unchecked system as a mean of incorporating Concurrent Error Detection into a self-checking binary decoder.**

Keywords—Binary decoder, concurrent error detection, self checking, hard ware redundancy.

I. INTRODUCTION

The advances of semiconductor technology have greatly increased the scale of integration that can be implemented in one chip. Unfortunately, as the scale of integration has increases, circuits become more susceptible to sources of temporary error (transient or intermittent fault). The characteristics of the intermittent faults, and increased use of complex Very Large Scale Integrated (VLIS) Circuits in "safety critical" applications, necessitate the use of a test strategy, which continuously monitors the operation of circuits and compares them with some known reference. This approach is usually referred to as *Concurrent Error Detection* (CED) technique, which continually monitor the operation of the circuit and compared it with some known reference. This is achieved by incorporating some form of redundancy into the system [1]. One of the most popular strategies for providing error detection properties is the hardware redundancy approach. This paper investigates the effectiveness of using hardware redundancy into unchecked system as a mean of incorporating CED into a self-checking decoder. Self-checking circuit can be defined as the ability to verify automatically whether there is any fault in the circuit (chips, boards, or assembled system). Thus, self-checking circuits allow on-line error detection, which means faults can be detected during the normal operation of the circuit [2].

II. DECODER

A binary decoder is a combinational circuit that converts an n-bit binary input code into m output lines such that each output line will be activated for only one of the possible combinations of the inputs. Decoders are used in many applications in digital systems such as address decoders which used to select memory address, seven segment decoders,..etc.. [3]. The block diagram of typical decoder is shown in figure (1). The output wave form of the typical binary decoder is also shown in figure (2).

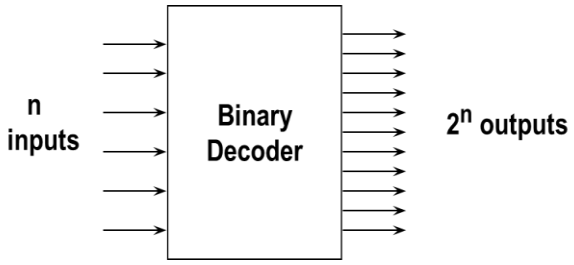

Fig. 1 Block diagram of a typical decoder.

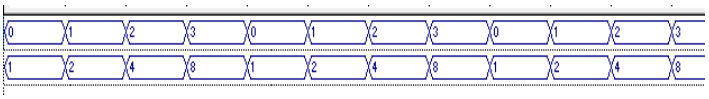

Fig. 2. Output waveform of a typical decoder.

III. HARD WARE REDUNDANCY

Hardware redundancy approaches consist of the replication of the normal blocks of the device for concurrent execution of parallel function, then the results from both blocks are compared step by step, as shown in Fig. (3). The simplest error-detecting code is duplication. A copy of functional logic is used as checker. The functional logic and its copy (checker) receive the same input at the same time and execute the same operation, and an equality checker is used to compare the outputs of the duplicate functional logic [4].

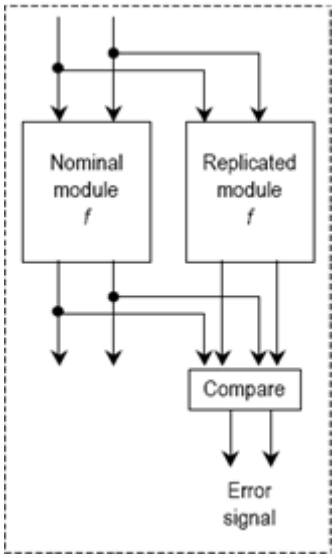

Fig.3 Hardware redundancy

Received July 2019, Received in received from October 2019, Accepted November 2019, and Published July 2020

IV. SELF CHECKING CIRCUIT

It is the ability to verify automatically if there is any fault in circuit. Self-checking circuits allow faults to be detected during the normal operation (on-line). Self-checking circuits must satisfy: self-testing and fault secure. A self-checking circuit, shown in figure (4), consists of a functional circuit (F), which produces encoded output vectors, and a checker (C), which checks the vectors to determine if an error has occurred. The checker has the ability to give an error indication even when a fault occurs in the checker itself [5] [6].

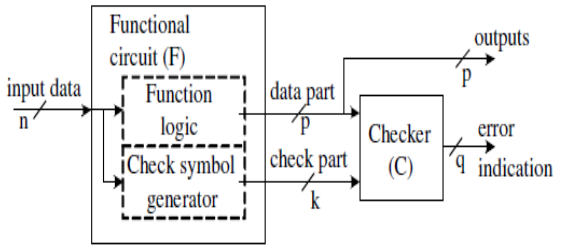

Fig.4 General structure of a self-checking circuit

V. TWO RAIL CHECKER

Two-rail checker unit (TRC) is used to compare two complementary code words. The checker determines whether the output of the functional circuit is a valid or invalid code word. Two-rail checker unit has two groups of inputs: (x1,x2,…..xn) and (y1,y2,…..yn) [7]. It also has two outputs: f and g. The signals observed on the outputs should always be complementary. Consider a two rail checker with n=2, as shown in Figure (5), the two input groups are (x1,x2) and (y1,y2). In a non-error situation where (y1=x1') and (y2=x2'), the result of this is(f=g'). In a situation where due
to a fault where (y1=x1) or (y2=x2), this will then produce (f=g), that means a non-code word output thus causes an error indication [8] [9].

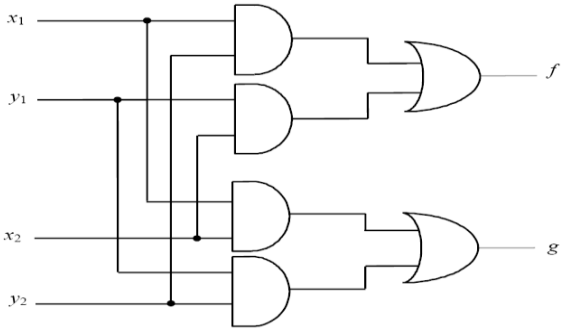

Fig.5 Two rail checker with 2 pairs of inputs

VI. SELF CHECKING HARD WARE

The hardware of the checker circuit depends on the type of the redundancy to be used, whether it is a hardware redundancy, information redundancy, or time redundancy. In this paper, a hardware redundancy will be used in the design of self-checking decoder. Figure (6) shows a block diagram of a self-checking decoder. The extra hardware which is used for error detection consists of a special register designed to generate the same function as the normal binary decoder and checkers. The special register is used instead of duplicating the normal decoder, to prevent the common mode error that might be occurring if the two blocks are designed with the same logic. The special register designed in such way that it generates the same wave form as the normal decoder. Figure (7) shows the wave form of the special register which acts as a normal binary decoder.

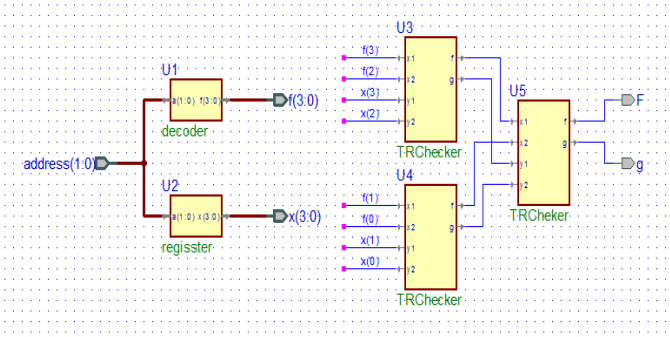

Fig.6 Self-checking decoder

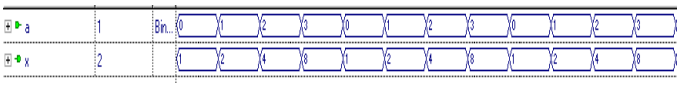

Fig.7 Output waveform of the special register.

VII. CHECKER CIRCUIT

The checker circuit consists of 3-TRCs, connected as shown in Fig. (6). The output of the normal decoder and the output of the special register are both fed to the checker. If the two outputs are matched, then no error occurred in the decoder or in the special register. In case of the outputs are not matched, then an error is detected. The error can be from the decoder or from the special register. The only case where the error cannot be detected when both decoder and special register, experienced the same type of error (common mode error). But as two circuits designed with different logic, the common mode error cannot occur.
Fig. (8) shows the wave forms of the decoder and the special register when no error detected.

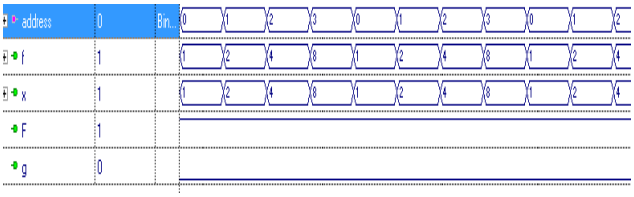

Fig.8 . The wave forms of the output of the normal decoder and the special register

Received July 2019, Received in received from October 2019, Accepted November 2019, and Published July2020

VIII. CONCLUSION

The self-checking decoder presented in this paper was designed and simulated using HDL language. Self-checking was achieved by using hardware redundancy. As the decoder always changes its state, the new state is checked before it can be accepted as: true state. Incorporated Concurrent Error Detection technique (CED) is not free. The cost of (CED) is time and hardware. The time delay is the delay of the two of TRCs. The extra hardware is the hardware of the special register and three two real checkers.

REFERENCES

[1] [1] Maamar, M., and Russell, G., " A 32-bit RISC processor with concurrent error detection., Proc. 24th Euromicro Conference, August 1998, Sweden, PP. 461-467.

[2] [2] M. Morris Mano and Charles R. Kime "Logic and Computer Design Fundamentals" by Prentice Hall, 3rd edition, chapter 7.p331.

[3] [3] M.MORRIS MANO, "Digital Design", 2002, 1991, 1984 by Prentice Hall, Upper Saddle River, New Jersey 07458, Chapter 6, P234.

[4] [4] Dhiraj K. Pradhan, "Fault Tolerant Computer System Design", Prentice Hall PTR, Upper Saddle River, New Jersey 07458, 1995, PP.1-22.

[5] [5] Russell, G.; Maamar, A.H., "Check bit prediction scheme using Dong's code for concurrent error detection in VLSI processors," Computers and Digital Techniques, IEE Proceedings - , vol.147, no.6, pp.467-471, Nov 2000.

[6] [6] M. A Marouf, A. D. friedman, "Design of self-checking checker using Berger code", Proc. Int. Symp. Fault tolerant computing, 1978, PP. 179-194.

[7] [7] Miron Abramovici, Melvin A.Breuer, and Arthur D.Friedman, "Digital Systems Testing and Testable Design",1990,ISBN 0-7803-1062-4, Chapter 13:SELF-CHECKING DESIGN, pp.569-587.

[8] [8] Huda Abugharsa, and Ali Maamar," Self Checking Systolic LIFO Stack",7th WSEAS Int. Conf. on Instrumentation, Measurement, Circuits and Systems (IMCAS '08), Hangzhou, China, April 6-8,2008.

[9] [9] Amal J. Mahfoud, Khadija F. Algheitta, Ali H. Maamar, "Design of a self-checking down counter", Libyan International Conference on Electrical Engineering and Technologies, Tripoli-Libya, 4-6 2018.

Received July 2019, Received in received from October 2019, Accepted November 2019, and Published July2020