VOL. 11, NO. 10, MAY 2016 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

IMPLEMENTATION ON UTEMRISC MICROCONTROLLER WITH EMBEDDED FAULT-TOLERANCE

Mohd Hafiz Bin Sulaiman, Sani Irwan Md Salim and Masrullizam Mat Ibrahim
Department of Computer Engineering, Faculty of Electronics and Computer Engineering, Universiti Teknikal Malaysia Melaka, Melaka, Malaysia
E-Mail: m021320004@student.utem.edu.my

ABSTRACT

In the development of the microprocessor architecture, the focus is given more on the microprocessor's performance parameters such as speed, size, cost and power consumption, while less attention is paid to the reliability of data. With the advancement of semiconductor technology node, internal components of a microprocessor are also prone to soft error due to sensitivity to glitches and noise. This paper presents an internal implementation of the fault-tolerance design for a low-end microcontroller. The UTeMRISC Microcontroller is chosen for this research and the fault-tolerance is designed based on the error correction code (ECC). The design is focused on the implementation of Hamming Code and Single-Error-Correction Double-Error-Detection (SEC-DED) Code that are synthesizable in the Field Programmable Gate Array (FPGA). To evaluate the performance and functionality of the design, a number of pre-defined faults are injected into the Fault-Tolerant module at three different locations in the UTeMRISC Microcontroller architecture. Based on the experiment results, the embedded fault-tolerance design has produced acceptable error-recovery rate with the optimal operating frequency is peaked at 60MHz. The evaluation shows the promising results are obtained after comparison into error recovered and time latency. Overall, the integration of the fault-tolerance module in the microcontroller architecture offers a good starting point to create a reliable platform in the embedded system design.

Keywords: error correction code, fault-tolerant, field programmable gate array, UTeMRISC microcontroller.

INTRODUCTION

Microcontroller fundamentally is an integrated computer system that combines a processor core, memory and input/output peripheral in a single integrated circuit [1]. In designing the system, there are many important elements that need to be addressed and one of it is the reliability of the system such as fault-tolerance capability [2].

Fault-tolerant is a feature that enables the system to recover any existence of failure in the hardware or software error [3,4]. Commonly, the failures or errors are occurring from the hardware or software operation which are very hard to predict. The error would cause difficulty in obtaining back the correct data from the hardware or software without any fault-tolerance capability [5, 6]. This paper presents the implementation of fault-tolerance modules which are embedded in a microcontroller architecture called UTeMRISC. A customized fault injection method is also utilized to assess the reliability of the system.

A fault-tolerance feature in a microcontroller system is a crucial design aspect especially for industries in order to produce a robust and reliable system. Having a fault-tolerant capability would complement the system in term of data reliability. There are many techniques have been adopted by past researchers to incorporate elements of error correction to different microprocessor platform. For example, Hamming code has been tested on Microprocessor without Interlocked Pipeline (MIPS) platform [7, 8]. Fault injection tool is used in the DLX processor which also embedded with ECC to find the most erroneous components in the architecture [9-11]. Other ECC such as Single Error Correction, Double Error Detection (SEC-DED) is applied on the single event upset (SEU) tolerant 8051 microcontroller to gauge the reliability of the system [12]. The impact of the intermittent faults on a RISC microprocessor is also analyzed using simulation method in [13].

The rest of the paper is organized as follows. Section UTeMRISC Architecture reviews the UTeMRISC microcontroller architecture. Section Error Correction Code discuss an overview of the Error Correction Code. While Section Implementation of The ECC on The Microcontroller explains the design of the Fault-Tolerant modules. The comparison results between the two fault-tolerant modules designed in this paper are also presented in section Results and Analysis. Finally, the summary of this paper is stated in the final section.

UTEMRISC ARCHITECTURE

This section reviews the UTeMRISC microcontroller architecture. The UTeMRISC microcontroller is a soft-core 8-bit RISC processor with multiply-accumulate (MAC) capability through Application-specific instruction set processor (ASIP) design methodology [14]. The processor is originated from a baseline 8-bit RISC processor that consists of fundamental processor modules and 33 instructions. Modification of the internal registers and architectures of the base 8-bit processor is done with the objective to make it compatible to perform 16-bit multiply-accumulate operation as illustrated in Figure-1. A few modified instructions are set-up to enable MAC operation through assembly programming codes [15]. The main advantage of having a soft-core microcontroller is that it is easy to configure its architecture in accordance to specific application requirements.

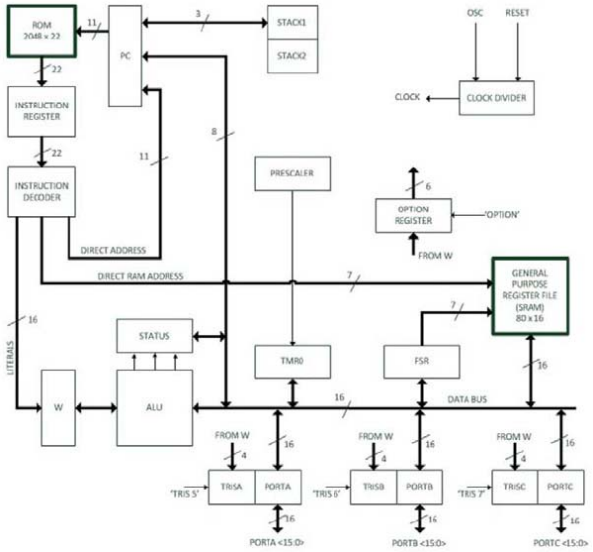

Figure-1. UTeMRISC microcontroller architecture.

ERROR CORRECTION CODE

This section describes an overview of error correction codes that is used in the fault-tolerant module for UTeMRISC microcontroller. There are two ECC chosen to be implemented, namely, Hamming Code and Single-Error-Correction Double-Error-Detection (SEC-DED) Code.

A. Hamming code

Hamming Code was proposed by Richard Hamming in 1950 [16]. It is also known as Single Error Detection Codes and it is a popular code and widely used for error control and its variation in the system. Moreover, this code is considered as the first class of the linear codes for error correction. Mathematically, for any positive integer, m, the Hamming code parameters are shown in Table-1.

Table-1. Parameters of hamming code.

Parameters	Equations
Code length, n	$n = 2^m - 1$
Number of information symbols, k	$k = 2^m - m - 1$
Number of parity-check symbols, m	$m = n - k$
Error-correcting capability, t	$t = [(d_{min} - 1)/2] = 1$

For many applications, a single error correcting code would be considered unsatisfactory, because it accepts all blocks received. Besides, Hamming code only detects single error and perform correction on that error. However, it offers simplicity in coding and has low latency [17].

B. SEC-DED code

Hsiao has proposed an optimal single error correction and double-error-detection (SEC-DED) in 1970 [18]. It is widely used for improving computer reliability. This code comes from extended hamming code whereby it converts from the Hamming code by adding an extra parity bit. For every positive integer, m, the SEC-DED code parameters are shown in Table-2.

Table-2. Parameters of hamming code.

Parameters	Equations
Code length, n	$n = 2^m - 1$
Number of information symbols, k	$k = 2^m - m - 1$
Number of parity-check symbols, m	$m = n - k$
Error-correcting capability, t	$d_{min} = 3$

A SEC-DED code seems safer and it is the level of correction and detection most often used in computer memories [17]. As its name implies, this code capable to detect two errors but can correct single error.

IMPLEMENTATION OF THE ECC ON THE MICROCONTROLLER

The selected ECC is embedded to the UTeMRISC Microcontroller which is implemented on top of a reconfigurable platform. This section shows the design of the Fault-tolerant modules involved in the integration process.

Figure-2 and Figure-3 illustrated the register-transfer level (RTL) schematic of the Fault-tolerant module design for Hamming code and SEC-DED code. These modules are designed and simulated using the Xilinx ISE Design Suite v13.2 software.

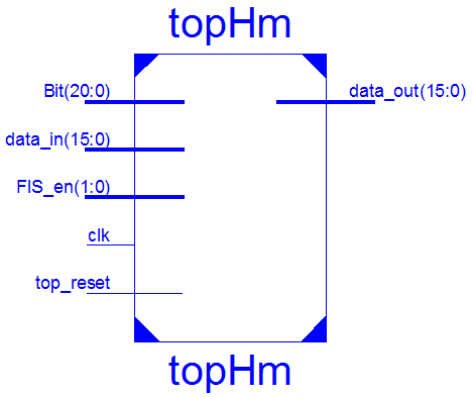

Figure-2. Fault-tolerant module for hamming code.

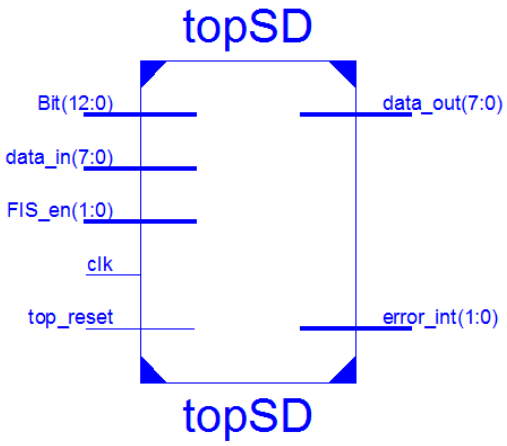

Figure-3. Fault-tolerant module for SEC-DED code.

The Fault-tolerant module contains three blocks register namely encoder, decoder and fault injection signal (FIS). The input data are loaded into the Fault-tolerant module through "data_in" pin. Then, the encoder block encodes the data that is configured according to the Hamming code or SEC-DED code algorithm. FIS block handles the fault injection process to generate an error to the original data. There are three types of error that are injected to the Fault-tolerant modules, namely Stuck-at-0, Stuck-at-1 and Double Bit flip [19]. Then, the decoder block is responsible to correct any error on the received data and decode the data back to its original state.

The Fault-tolerant modules are implemented at three locations in UTeMRISC Microcontroller which are the crucial location; Program counter (PC) at FT1, Programmable read-only memory (PROM) at FT2 and Multiplexer (MUX) at FT3 as illustrated in Figure-4. Table-3 shows the configuration of the encoder and decoder for the Hamming code and the SEC-DED code at three different components of the UTeMRISC microcontroller.

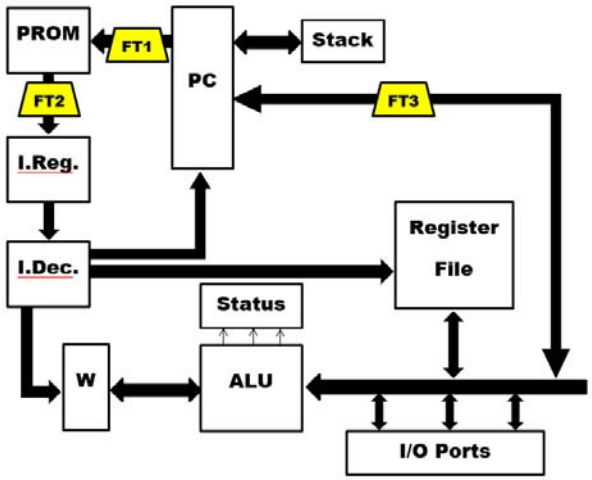

Figure-4. The fault-tolerant module implemented at three location in UTeMRISC microcontroller.

Table-3. Encoder and decoder configuration.

Component	DataBus (Bit)	Codeword	
		Hamming	SEC-DED
PC	11	(15,11)	(16,11)
PROM	12	(17,12)	(18,12)
MUX	8	(12,8)	(13,8)

Table-4. FPGA Resource Usage for Device Utilization Summary.

Types of code	Slices register	Slices input LUTs	Slices flip flops	Max freq. (MHz)
Hamming	16	78	16	402.253
SEC-DED	53	92	12	205.128

The synthesis report of the device utilization for the two Fault-tolerant modules is summarized in Table-4. The SEC-DED code module used 53 slices register and 92 slices input LUTs. This is higher than the Hamming Code module which used 16 slices register and 78 input LUTs. However, SEC-DED code module only used 12 slices Flip Flops while Hamming code module used 16 slices Flip Flops. Essentially, these two Fault-tolerant modules have low resource usage and provide simplicity in design. In term of speed, the maximum frequency of the Hamming code module is almost two times of the maximum frequency achieved by the SEC-DED code module.

RESULTS AND ANALYSIS

This section presents the result and analysis for three locations of both the Hamming code module and SEC-DED code module. In the experimental setup, the modules are tested about 50 times using the Acer Aspire E1-522, PC with AMD Quad-Core Processor A6-5200, (2.0 GHz), 4GB DDR3 RAM and Windows 7 OS. Logic analyzer Tektronix TLA-5202 is used to analyze both Fault-tolerant modules. The results of the experiment are observed based on the number of error recovered, and time latency from both Fault-tolerant module at three locations inside the UTeMRISC microcontroller architecture. Thirty faults have been injected for each of the FT1, FT2, and FT3 modules consisted of ten Stuck-at-0 error, ten Stuck-at-1 error and ten Double Bit Flip error. The percentages of error recovered defined as Equation. (1): $N_{Corrected\ Data}$ is the number of data that have been corrected. $N_{Total\ Data}$ is all the data that have been passed through in fault-tolerant.

$$P_{Error\ Recovered} = \frac{N_{CorrectedData}}{N_{TotalData}} \times 100 \quad (1)$$

Table-5 shows the corresponding result Hamming code module and SEC-DED code module at three locations. The percentage of the error recovered from the Hamming code module is in the range 63% to 67%. While, the percentage of the error recovered from the

SEC-DED code module is in the range 64% to 67%. From the overall results, the most reliable and highest module is SEC-DED code module at the location FT2 with 66.67% error recovered.

Table-5. Error recovered results.

Fault-tolerant modules	Error Recovered (%)	
	Hamming	SEC-DED
FT1	63.78	64.52
FT2	66.51	66.67
FT3	66.35	66.38

In this experiment, both of the Fault-tolerant modules have successfully recovered the single error. However, time latency occurred during the error correction process. Table-6 shows the time latency for both Fault-tolerant modules with different clock speed. The clock speed is set up in sequence from 50, 60, 70 and 80 Mega Hertz (MHz) by using a digital clock manager (DCM).

Table-6. Time latency of both fault-tolerant module with different clock speed.

Clock speed (MHz)	Time Latency (ns)					
	Hamming			SEC-DED		
	FT1	FT2	FT3	FT1	FT2	FT3
50	2.40	2.20	3.33	4.20	3.73	3.87
60	1.67	1.93	3.20	2.67	2.87	3.33
70	2.93	3.87	3.87	3.73	3.20	3.60
80	3.33	4.33	4.60	3.80	3.47	3.80

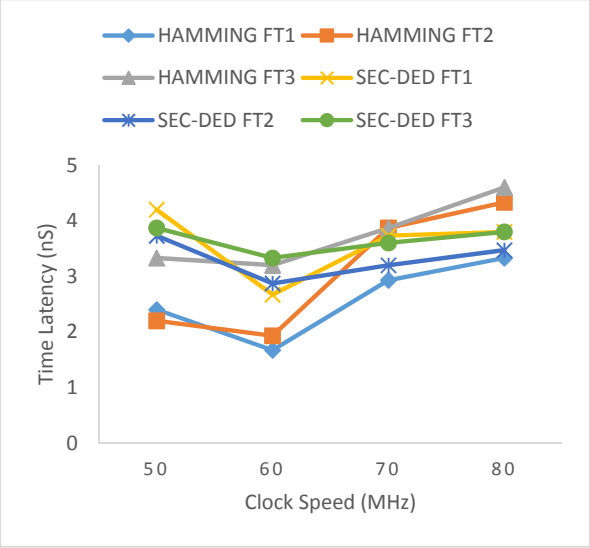

Figure-5. The graph of time latency against the clock speed.

The time latency of both model according to clock speed is demonstrated in Figure-5. The graph shows the time latency for each fault-tolerant modules with respect to their different locations in the UTeMRISC microcontroller. For Hamming code module, the minimum time latency at FT1 location with clock speed 60MHz. Similar to the SEC-DED code module, the minimum time latency is FT1 location with clock speed at 60MHz. Both of the Fault-tolerant modules in the three locations show that at 60MHz clock speed, the time latency is the lowest compare to other clock speed. This means that the optimum clock speed for the implementation of UTeMRISC microcontroller with embedded fault tolerance is at 60MHz.

CONCLUSIONS

This paper presented the Fault-tolerant modules which are implemented in the UTeMRISC microcontroller. Two types of ECC-based Fault-tolerant modules are designed and implemented in the UTeMRISC microcontroller. Based on the experimental and analytical work, the SEC-DED code module works acceptably well for the UTeMRISC Microcontroller. The result shows the highest percentage of error recovered is achieved. From a practical point of view, the Hamming code module provides the lower usage of resources and higher maximum frequency. Nevertheless, SEC-DED code capability must be highlighted because it is more powerful in comparison to the Hamming code in terms of error recovery and error detection. The optimum clock speed for both Fault-tolerant modules are at 60 MHz. These two types of error correction code are suitable for the low-end microcontroller with the advantages of lower latency and small resource requirement. For the future work, others type of error correction code that is suitable with low-end microcontroller will be designed together with its fault injection algorithm.

ACKNOWLEDGEMENTS

The authors would like to thank the Ministry of Higher Education Malaysia and Universiti Teknikal Malaysia Melaka for the financial support given through the research short-grant number PJP/2013/FKEKK (47B) /S01274.

REFERENCES

[1] Y. Nakamura and K. Hiraki. 2002. Highly Fault-tolerant FPGA Processor by Degrading Strategy. Pacific Rim International Symposium on Dependable Computing. 75-78.

[2] J.P. Shen, M. H. Lipasti. 2013. Modern Processor Design: Fundamentals of Superscalar Processors. Waveland Press.

[3] A. Avizienis. 1975. Fault-Tolerance and Fault-Intolerance: Complementary approaches to reliable

computing. New York, NY, USA: ACM Press. 458-464.

[4] Avizienis. 1971. Fault-tolerant Computing: An Overview, IEEE Computer. 4(1). 5-8.

[5] C. Constantinescu. 1999. Using Physical and Simulated Fault Injection to Evaluate Error Detection Mechanisms. Pacific Rim International Symposium on Dependable Computing. 186-192.

[6] J. Aidemark, J. Vinter, P. Folkesson and J. Karlsson. 2001. GOOFI: Generic Object-Oriented Fault Injection Tool. International Conference on Dependable Systems and Networks. 83-88.

[7] Sarjoughian, Hessam, Yu Chen, and Kevin Burger. 2008. A Component-Based Visual Simulator for MIPS32 Processors. Frontiers in Education Conference, IEEE 38. 9- 14.

[8] H. Kanbara. 2009. Dependable Embedded Processor Core for Higher Reliability. Consumer Electronics, ISCE'09. IEEE International Symposium. 13. 819-822.

[9] D.M.B. Ancajas, A.P. Ballesil, J.R.E. Hizon, E.A. Opelinia, J.A.P. Reyes, A.G.L. Sepillo, W.A. Sumalia, W.M. Tan. 2007. Dual Core Capability of a 32-bit DLX Microprocessor. IEEE Region 10 Conference TENCON. 1-4.

[10] John Edrian, H. Aguilar. 2007. DLX Gold: Design and Implementation of a DLX Microprocessor with Single Precision Floating-Point Operations. IEEE Region 10 Conference TENCON. 1-4.

[11] H. R. Zarandi, S. G. Miremadi, A. Ejlali. 2003. Dependability Analysis Using a Fault Injection Tool Based on Synthesizability of HDL Models. Proceedings of the International Symposium on Defect and Fault Tolerance in VLSI systems. 485-492.

[12] Lima, F., Luigi Carro, Raoul Velazco, and R. Reis. 2002. Injecting Multiple Upsets in a SEU tolerant 8051 Micro-controller. In On-Line Testing Workshop, IEEE International, IEEE Computer Society. 194-194.

[13] Gil-Tomás, Daniel, Joaquín Gracia-Morán, J-Carlos Baraza-Calvo, Luis-J. Saiz-Adalid and Pedro-J. Gil-Vicente. 2012. Studying the effects of intermittent faults on a microcontroller. Microelectronics Reliability. 52(11). 2837-2846.

[14] A. J. Salim., S. I. M. Salim, N. R. Samsudin, and Y. Soo. 2013. Instruction Set Extension through Partial Customization of Low-End Risc Processor. Australian Journal of Basic and Applied Sciences. 7(6). 678-687.

[15] J. Salim, N. R. Samsudin, S. I. M. Salim and Y. Soo. 2012. Multiply Accumulate Instruction Set Extension in a Soft-Core RISC Processor. IEEE International Conference on Semiconductor Electronics, ICSE2012. 512-516.

[16] T. Rao and E. Fujiwara. 1989. Error Control Coding for Computer Systems. Prentice-Hall Inc.

[17] J. S. Chitode. 2007. Information Coding Technique. Technical Publications.

[18] Costello, D. and Shu Lin. 2004. Error control coding. Pearson Higher Education.

[19] Eghbal, Ashkan, Hamid R. Zarandi, and Pooria M. Yaghini. 2009. Fault tolerance assessment of PIC microcontroller based on fault injection. In Test Workshop, LATW09, IEEE. 10. 1-6.