Microelectronics Reliability 63 (2016) 314–318

Contents lists available at ScienceDirect

Microelectronics Reliability

journal homepage: www.elsevier.com/locate/mr

DMR+: An efficient alternative to TMR to protect registers in Xilinx FPGAs

P. Reviriego [a,*], M. Demirci [b], J. Tabero [c], A. Regadío [c], J.A. Maestro [a]

[a] Universidad Antonio de Nebrija, C/Pirineos, 55 E-28040 Madrid, Spain
[b] Aselsan, Mehmet Akif Ersoy Mahallesi 296, 16, 06370 Yenimahalle/Ankara, Turkey
[c] Instituto Nacional de Técnica Aeroespacial, E-28850 Torrejón de Ardoz, Spain

ARTICLE INFO

Article history:
Received 12 February 2016
Received in revised form 27 June 2016
Accepted 27 June 2016
Available online 29 June 2016

Keywords:
FPGA
Error correction
Fault tolerance

ABSTRACT

Registers are one of the circuit elements that can be affected by soft errors. To ensure that soft errors do not affect the system functionality, Triple Modular Redundancy (TMR) is commonly used to protect registers. TMR can effectively protect against errors affecting a single flip-flop and has a low overhead in terms of circuit delay. The main drawback of TMR is that it requires more than three times the original circuit area as the flip-flops are triplicated and additional voting logic is inserted. Another alternative is to protect registers using Error Correction Codes (ECCs), but those typically require a large circuit delay overhead and are not suitable for high speed implementations. In this paper, DMR+ an alternative to TMR to protect registers in FPGAs, is presented. The proposed scheme exploits the FPGA structure to achieve a reduction in the FPGA resources (LUTs and Flip-Flops) at the cost of a certain overhead in delay. DMR+ can correct all single bit errors like TMR but is more vulnerable to multiple bit errors. To evaluate the benefits, the DMR+ technique has been implemented and compared with TMR considering standalone registers and also some simple designs.

1. Introduction

Soft errors induced by radiation are a major issue for electronic systems [1]. In critical applications, techniques to mitigate the errors are used to ensure that they do not compromise system reliability [2]. These techniques can be applied at the technology, circuit or system level or combined to provide a cross layer solution. In all cases, protecting the circuit implies area and power overheads.

Field Programmable Gate Arrays (FPGAs) provide high performance and programmability making them attractive to implement electronic systems [3]. This, combined with the rising cost of development for Application Specific Integrated Circuits (ASICs), has made FPGA a popular choice for system implementation.

FPGAs can be implemented with different technologies. For example, SRAM can be used to store the configuration. This has a direct impact on their sensitivity as SRAM memories are affected by soft errors. This means that for SRAM based FPGAs there are two main types of soft errors: errors on the FPGA circuits elements and errors on the SRAM configuration memory [4]. Soft errors on the circuit elements can affect the logic gates, flip-flops and embedded memories. In these cases, the errors have the same effects as in ASICs. However, errors in the configuration SRAM can modify the circuit implemented on the FPGA. They can for example change a logic function or the interconnection of some blocks. These types of failures are specific to SRAM based FPGAs.

To evaluate the impact of errors on SRAM configuration, several fault injection methods and platforms have been proposed over the years [5–7]. These are needed as failures on configuration memory cannot be emulated in a traditional circuit simulator. The errors on configuration memory can be corrected by scrubbing its contents periodically [8,9]. Recently, some FPGA manufacturers like Xilinx have implemented a Soft Error Mitigation (SEM) IP core that can also be used to inject errors on the configuration SRAM [10]. The SEM IP core provides several implementations for the scrubbing of the configuration memory such that the designer can protect it in a design independent manner. This provides a standard method to deal with errors on the SRAM configuration memory.

In the rest of the paper, it is assumed that scrubbing is used to detect and correct errors on the SRAM configuration memory. In that case, the design has still to provide protection against errors on the FPGA circuit elements. Combinational logic is mostly implemented with lookup tables that rely on configuration bits. Therefore they are protected by the scrubbing mechanisms. There are a few logic gates in the FPGA but errors on the logic gates are likely to be masked before they propagate to flip-flops. This masking can occur at the electrical, temporal or logic levels [11]. On the other hand, errors on a flip-flop or memory cell can have a permanent effect unless the value is overwritten as part of circuit operation. In addition, they are the vast majority of the circuit elements (excluding the configuration SRAM) on the FPGA. This makes the protection of memories and flip-flops more critical. Finally, it is worth

* Corresponding author.
E-mail addresses: previrie@nebrija.es (P. Reviriego), mdemirci@aselsan.com.tr (M. Demirci), taberogj@inta.es (J. Tabero), regadioca@inta.es (A. Regadío), jmaestro@nebrija.es (J.A. Maestro).

http://dx.doi.org/10.1016/j.microrel.2016.06.019
0026-2714/

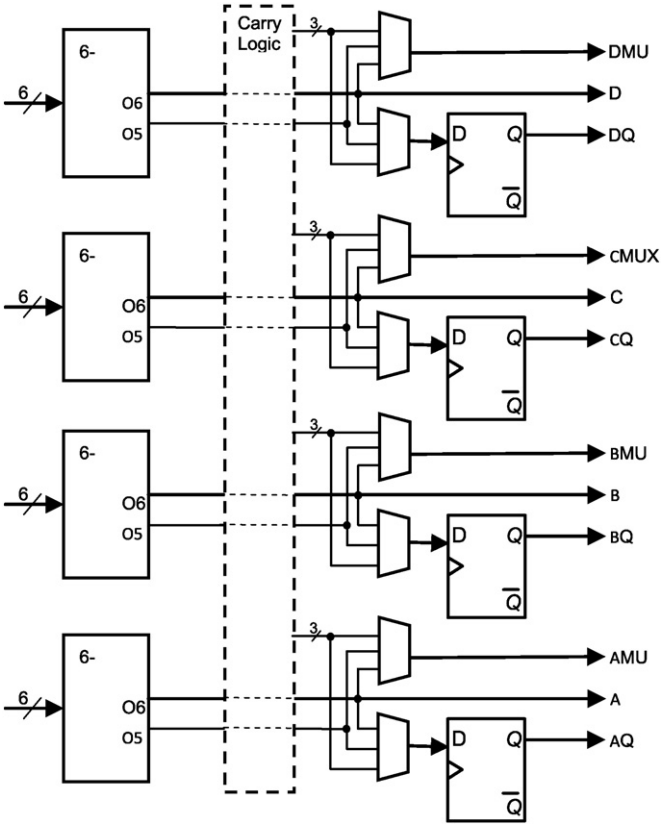

Fig. 1. Block diagram of a Xilinx Virtex-5 FPGA slice.

noting that although FPGAs may suffer other types of effects such as Single Event Latch ups, the protection against those is outside the scope of the paper.

Memories are present on most FPGAs as block RAMs (BRAMs) that can be configured with different word sizes. Additionally, the FPGA is also in many designs connected to external memory devices. To protect memories, Error Correction Codes (ECCs) are commonly used [12]. In particular, Single Error Correction Double Error Detection (SEC-DED) codes are implemented in many FPGAs [13]. More advanced codes that can correct multiple bit errors such as Orthogonal Latin Square codes have also been implemented and optimized for SRAM based FPGAs [14]. In a memory, the ECC encoder and decoder are shared by all the memory words, making the implementation efficient.

There are several alternatives to protect flip-flops. For ASIC implementations, one option is to use radiation hardened flip-flops [15].

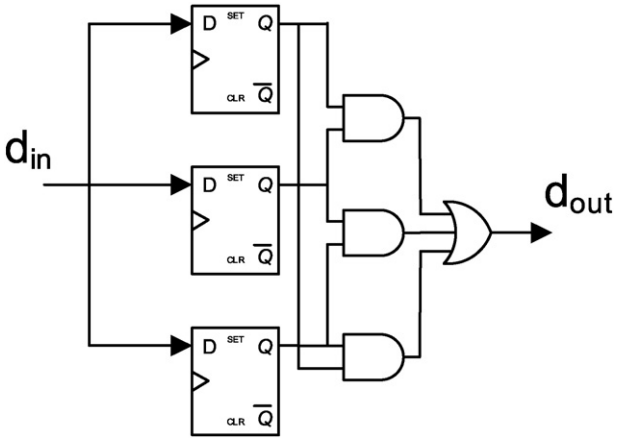

Fig. 2. Register protected with TMR.

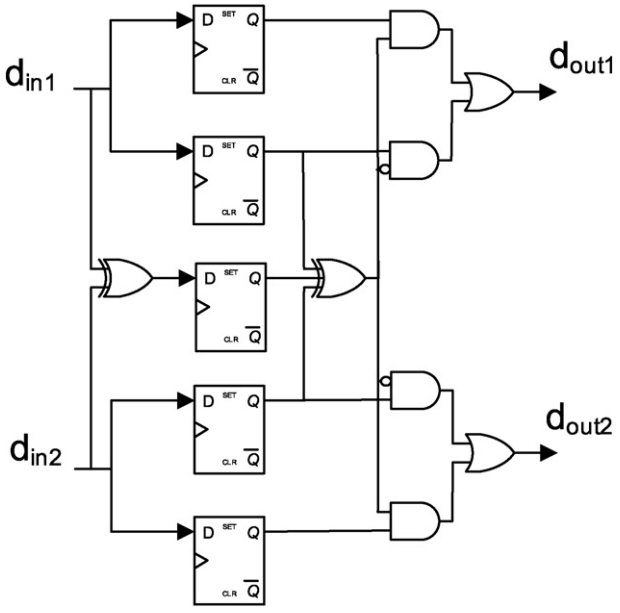

Fig. 3. A pair of registers protected with DMR+.

Those are designed at the circuit level to prevent errors from occurring. However, this option cannot be used for FPGAs as the users cannot modify the circuit implementation. In FPGAs, flip-flops are typically protected with Triple Modular Redundancy (TMR). In fact, the use of TMR to protect SRAM based FPGAs has been widely studied [16–20]. For example, [17] proposed the use of selective TMR to protect only the critical parts of the design. The effectiveness of TMR against error accumulation [18] and multiple cell upsets has also been considered [19,20]. TMR uses three copies of the design and a voter that selects the correct value for the final output [21]. This requires a large penalty in terms of circuit area but can protect against errors with low delay. Flip-flops can also be protected with ECCs, but in this case, firstly the flip flops have to be grouped in blocks something that is not always possible. Secondly, an encoder and decoder is needed for each register, thus requiring a larger overhead. Finally, the ECC decoder typically adds much more delay than the TMR voter. This in turn can reduce the maximum operating frequency of the circuit. Recently, Single Error Correction (SEC) codes that can be decoded with low delay have been presented [22]. However, even these codes require a much larger delay than a TMR voter.

In this paper, DMR+, a scheme to protect flip-flops in Xilinx FPGAs is presented. The objective is to achieve a decoding delay comparable to that of TMR and a reduced usage of FPGA resources. To do so, the scheme exploits the structure of the LUTs in Xilinx FPGAs Spartan-6, Virtex5, 6 & 7, Artix-7 and Kintex-7. The proposed technique has been implemented for standalone flip-flops, flip-flops used in some simple blocks and finally in two practical designs. In all cases, the results show that DMR+ achieves a similar or moderately lower speed than TMR but with a reduced usage of FPGA resources (both LUTs and flip-flops). Therefore, DMR+ can be an interesting option to protect flip-flops in Xilinx FPGAs when the configuration SRAM is already protected with scrubbing.

The rest of this paper is organized as follows. Section 2 presents a brief overview of Xilinx FPGAs. The proposed DMR+ technique is described in Section 3 and evaluated in Section 4. Finally, the conclusions of this work are summarized in Section 5.

Table 1
Cost of a protected flip-flop.

	TMR	DMR+	DMR+ savings
LUTs	1	0.5	50.0%
Flip-flops	3	2.5	16.7%
XOR gates	0	0.5	N.A.