IET Computers & Digital Techniques

Research Article

IET Journals
The Institution of Engineering and Technology

Power-efficient reliable register file for aggressive-environment applications

ISSN 1751-8601
Received on 9th April 2018
Revised 1st February 2019
Accepted on 19th March 2019
doi: 10.1049/iet-cdt.2018.5047
www.ietdl.org

Ihsen Alouani[1] ✉, Hamzeh Ahangari[2], Ozcan Ozturk[2], Smail Niar[3]

[1]*IEMN-DOAE Laboratory, Polytechnic University Hauts-De-France, Famars, France*
[2]*CECS Department, Bilkent University, Ankara, Turkey*
[3]*LAMIH Laboratory, Polytechnic University Hauts-De-France, Valenciennes, France*
✉ *E-mail: ihsen.alouani@uphf.fr*

Abstract: In a context of increasing demands for on-board data processing, insuring reliability under reduced power budget is a serious design challenge for embedded system manufacturers. Particularly, embedded processors in aggressive environments need to be designed with error hardening as a primary goal, not an afterthought. As Register File (RF) is a critical element within the processor pipeline, enhancing RF reliability is mandatory to design fault immune computing systems. This study proposes *integer* and *floating point* RF reliability enhancement techniques. Specifically, the authors propose Adjacent Register Hardened RF, a new RF architecture that exploits the adjacent byte-level narrow-width values for hardening integer registers at runtime. Registers are paired together by special switches referred to as joiners and non-utilised bits of each register are exploited to enhance the reliability of its counterpart register. Moreover, they suggest sacrificing the least significant bits of the Mantissa to enhance the reliability of the floating point critical bits, namely, Exponent and Sign bits. The authors' results show that with a low power budget compared to state of the art techniques, they achieve better results under both normal and highly aggressive operating conditions.

1 Introduction

As sub-micron technology dimensions sharply decreased to a few nanometres range in commercialised integrated circuits, the sensitivity of electronic circuits increased drastically making embedded microprocessors more vulnerable to soft errors. Hence, designing dependable systems ready for deployment has become a critical task for chip designers. Especially, these systems have to keep operating reliably even with the presence of faults, to sustain the present growth-rate of device-count and clock-frequency with continuously growing reliability issues.

Random variation in the manufacturing process causes more production yield loss and increases the number of instabilities in chip die. Additionally, the sensitivity of chips is also intensified by voltage scaling since Vdd diminishes with feature size. It also decreases by dynamic voltage scaling (DVS), which is a widely used power reduction technique for dense and power-hungry circuits. As supply voltage dwindles, by technology or DVS, noise margin also decreases proportionally. Thereby, undesirable and accidental faults become more frequent.

Protecting the microprocessor's memory and sequential elements are critical because of its direct impact on the system's reliability and data correctness. Cache memory, register file (RF), flip-flops and latches are common sequential parts of microprocessor architecture, each of which requires its own suitable reliability enhancement. Although cache and RF are both based on SRAM memory technology, their prevalent reliability techniques differ, since their characteristics and applications differ. Architecture level error resilience techniques like Error Correcting Codes (ECCs) have been proposed and widely used [1]. The simplest form is the parity check method whose major weakness is its inability to correct the detected errors [2]. Another form of ECC used in memories is the SECDED (single error correction, double errors detection) [3]. The main shortage of the SECDED is its area overhead and the supplementary latency leading to performance loss. This is more pronounced in RFs, where performance is an essential requirement. Therefore, while effective for cache memories, generally ECC is not considered as a reliability solution in RFs since activity rate per address is higher than cache memories. Moreover, as RF is located in processors' critical path, preserving its performance is a priority. Registers in the integer and floating-point RF are used to hold source and destination operands for integer and floating-point computations, respectively. The floating-point operations differ from integer operations data-path and therefore a dedicated RF may be accordingly designed in the CPU micro-architecture such as.

Three main observations motivate us to do this work:

- The need for low power-budget, yet highly reliable, microprocessors for aggressive environments is still an open critical issue.
- The importance of RFs in processors' reliability.
- Although floating point operations are widely used in critical applications, to the best of our knowledge, there are no floating-point-dedicated works that target RF reliability.

We believe that proposing a suitable technique for RF reliability enhancement is a key challenge in reliable computing system design. The main contributions of this paper are summarised as follows:

- We extend our previous work [4] to a new fault tolerant *integer* RF based on exploiting vacant spaces in adjacent registers. We combine both architectural and circuit techniques to achieve more robustness in RF with reduced overheads, which targets processors in aggressive environments.
- Combining with our integer RF, a new *floating point* RF reliability enhancement technique is proposed. The idea is to take advantage of the precision tolerance of a range of applications to achieve low overhead protection. To the best of our knowledge, this is the first reliability enhancement technique considering precision-reliability tradeoffs and dedicated to floating point RF.

This paper is organised as follows: Section 2 presents related studies in the field, while Section 3 describes our architecture. Section 4 discusses floating point RF reliability enhancement. Section 5 gives our experimental results. Finally, conclusion and future work is explained in Section 6.

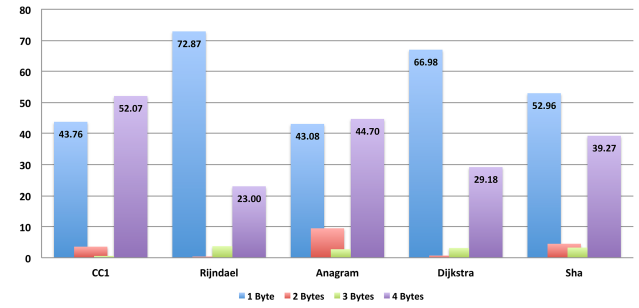

Fig. 1 *Percentage of appearance in the 32-bit RF with different effective lengths (in bytes)*

2 Related work

While our approach applies to any aggressive environment, we give the related work on potential application domains. One such domain that extensively focuses on reliability is space and aerospace. In the late 90 s, the European Space Agency (ESA) released a 32-bit microprocessor for embedded space-flight applications, called ERC32 [5] that is based on the SPARC V7 architecture. Later, ERC32 was followed by four generations of fault-tolerant LEON-FT processors used in several space projects, including the control computers of the International Space Station [6]. In LEON-FT architectures, parity bits are used to detect errors within the RF. In 2011, a reliable multicore processor was developed for NASA space missions [7] where memory is protected through Hamming Code based ECCs. The problem with ECCs is their high power overheads and time penalties.

In addition to ECCs, several RF reliability studies are based on information redundancy techniques. In some works, register duplication is proposed. For example, in [8], using register renaming unit, unused registers are detected and exploited to preserve redundant copies of other registers.

In-Register Duplication (IRD) is proposed in [9] in which, by an opportunistic idea, dummy sign bits of narrow-width register values are replaced with replication of meaningful bits during RF write operation. For a 64-bit RF, based on distribution of length of operands, extracted from benchmarks, registers are divided into three classes. Those by length of <32, between 32 and 34, and >34. For the first two classes which have dummy sign bits, IRD is applied. ALU detects such sign bits and replaces them with meaningful bits of data. Later, in read operation, replicated and original bits are bitwise compared to find a mismatch as an error indication. Additionally, two parity bits are embedded for each half. By means of both error detection mechanisms, similar to a 2D parity system, they added error detection/recovery for narrow width values in RF. Extra circuit for detecting effective length, applying replication and comparison are all collected in the execution stage with ALU. Duplication is done on the output of ALU, whereas communication path from ALU output to RF input is also protected against transient faults. Nevertheless, long operands are not protected by IRD. If applied to 32-bit RF, this disadvantage is more serious, because long operands are frequent.

In [10], the authors propose an extension to previous work, in which long operands in addition to short operands are also protected. In 32-bit RF, for long operands, values are replicated in other unused registers, similar to [8]. For avoiding the negative effect on performance, two stages are added to the pipeline for detecting efficient length first, and later performing sign extension in the read operation.

All of the works mentioned above are architectural level approaches based on information redundancy and explicit comparison operation. The main difference that our work proposes is that we combine a hardening technique with narrow width duplication. In addition to reducing error rates, by judicious replication in two paired registers, we protect long operands. Provided that a long operand is next to a short one, priority is given to long operand and replication of its more significant bits are done on dummy sign bits of the short operand.

In the context of emerging approximate computing applications, recent works [11] proposed to find a tradeoff between reliability and computing precision. But, to the best of our knowledge, this is the first work that proposes a reliability enhancement technique for both integer and floating point RFs.

3 Integer RF reliability Enhancement through Adjacent Register Hardening (ARH)

Our approach improves RF reliability by exploiting unused bits of integer numbers in adjacent registers. Although, any non-numeric data type like strings also benefit, our main focus is on integers since it is more effective for them. For any number in a given range of minimum to maximum possible values in 2's complement system, only one single sign bit is sufficient for correct representation of the number. The remaining sign bits are just multiple copies of the same sign bit and are redundant bits. Based on this observation, instead of preserving multiple redundant bits for sign, we suggest to exploit the redundant bits to enhance the reliability of adjacent registers. ARH is very efficient to protect highly critical data within an application using dummy bits of non-critical registers.

This is even more pronounced for integer values with smaller magnitude. That is, they have more dummy bits which consequently provide more resources for the RF's reliability enhancement. Small numbers have less number of bits to be protected too. Considering this fact, a uniform distribution of large and small numbers will be beneficial for this approach. This is expected from a typical application, but the content of registers is unveiled at run time; hence the extent of reliability increase is application dependent. As can be seen from Fig. 1, numbers with one byte effective length represent more than half of the numbers stored in the RF on average for the tested benchmarks.

Our implementation consists of retrieving the data to store, implementing the technical solution to enhance reliability, and performing the different read/write access. Instead of merely high-level architectural solutions, in our approach, we benefit from a fast circuit level technique. As detailed in the next section, unlike redundancy-based approaches, ARH does not need explicit voting, since circuit-level hardening technique is used. By combining architectural and circuit-level techniques, we reach to a highly flexible reliability solution.

Since the system's higher layers like operating system or compiler are oblivious to the existence of a circuit-level mechanism, our approach does not impose any strict requirement on those higher layers. However, if additional system requirements exist, like criticality of a certain data, the compiler can consider this in register allocation. Dynamic run-time register renaming mechanism can also distribute registers in a better way based on effective length. Such improvements will be the next steps in our future work.

3.1 Circuit level reliability enhancement

To achieve adaptive reliability enhancement and cope with registers stored data state, the proposed approach relies on a configurable circuit level technique. In our proposed architecture, adjacent registers of RF are joined together by 7T/14T technique. Generally, each bit can be joined to any number of bits from any register by embedding multiple switches in between. Nevertheless, to avoid excessive area overhead and complexity, we optimise this idea by just allowing each bit to be joined into a single bit of a single specific register. Thus, registers are paired together bit by bit during RF design. For example, reg0-reg1,…, reg30-reg31. The joiner switches are embedded between these registers to join them when needed.

As shown in Fig. 2 left, 7T/14T [12] is a technique proposed to combine two SRAM cells in circuit level to achieve more reliability or better performance dynamically. According to this idea, two memory cells can be joined together upon request to store a single bit of data into two cells. Joining is done by means of activating two transistors which connect the internal nodes of two cells to each other. For biasing towards reliability, just one of the

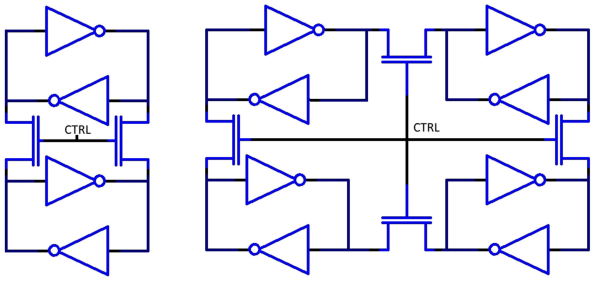

Fig. 2 *Left: 7T/14T memory cell with NMOS joiners [12] right: JSRAM cell with NMOS joiners [13]*

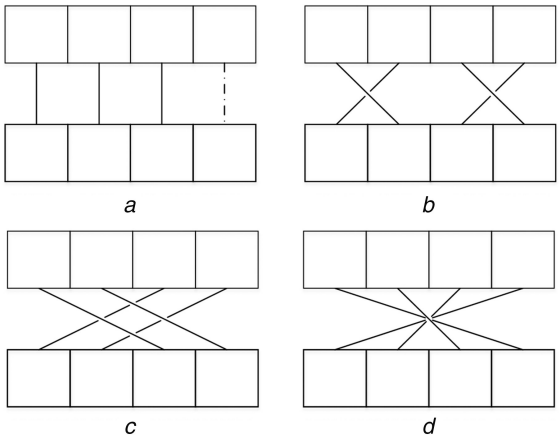

Fig. 3 *Some possible combinations for byte mapping*

wordline signals is used for read or write operation (it is called dependable mode). On the other hand, to get more performance both wordlines can be used simultaneously (high-performance mode). If neither cases are required, the joiners are not activated (CTRL = 'L', if the switches are NMOS), then the proposed structure works normally as two separate conventional 6T SRAM cells.

7T/14T structure has more reliability due to mutual support from two cells. If there is an instability in one cell, the other one provides more resistance against it. Additionally, in high-performance mode, it shows higher performance in terms of read operation speed as read current is provided by two cells. It also allows bigger dynamic voltage reduction because of increased static noise margin (SNM). Since in this work a combination of joined and normal bits may exist inside a single register (i.e. some bits are protected, some are not), we only consider the reliability enhancement benefit, not additional voltage reduction nor performance improvement benefits. However, our approach can be extended to exploit such benefits.

JSRAM cell [13] is an extension of the 7T/14T cell to combine four cells in a ring fashion to achieve full immunity against single bit errors by providing an auto-correction mechanism (Fig. 2 right). It is also capable of tolerating multiple bit upsets (MBUs). Since the RF reliability enhancement in our current work is based on a statistical study and is dependent on the stored data within registers, using 7T/14T scheme is more suitable.

3.2 Architecture level organisation

To enhance the RF error resiliency, we take advantage of the reconfigurable aspect of the 7T/14T cell. Instead of relying on ECCs or extra memory space for reliability enhancement, we use an opportunistic approach that exploits unused bits within the stored data. At bit-level, two paired registers may have up to 32 distinct control signals (named CTRL in Fig. 2) for having bit-level granularity to join individual bits separately (each bit to its counterpart bit). However, this granularity level is not suitable due to its high area overhead. To optimise the reconfiguration circuitry as well as the additional bit cells, we opted for a byte-level width

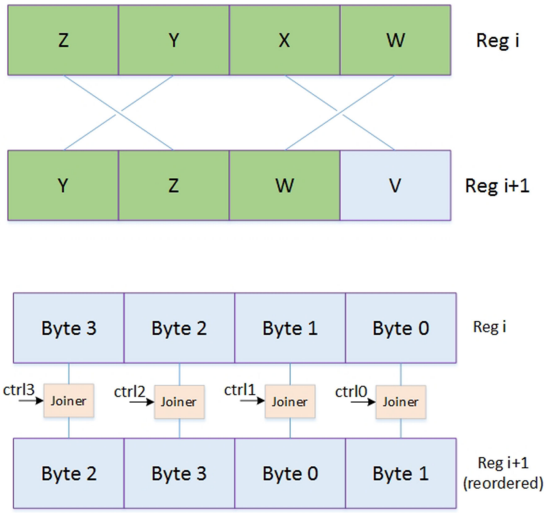

Fig. 4 *Top: Three bytes in reg-i are replicated in sign bits of reg_i + 1 using combination b in Fig. 2. Byte V is not replicated. Bottom: routing by byte reordering*

granularity. Accordingly, the idle bytes are used to harden registers against errors.

Considering byte level granularity, a clever one-to-one mapping between bytes of two registers is required to exploit the empty bits efficiently. Dummy sign bits are on the left-hand side (MSB side), while real data bits are on the other side. Thus, the first obvious paradigm of mapping is in a crossed way, byte-0 of one register to byte-3 of the paired register, byte-2 to byte-1 and so on. Faults in more valuable bits of an integer lead to absolute numerical error. Therefore, alternative mappings are also worth to be investigated (Fig. 3).

While the best mapping is application dependent, we extracted the distribution of operand length for our benchmarks as shown in Fig. 1. Operands with lengths one and four bytes are the dominant ones. Then paired registers of length one-one, one-four, and four-four are more frequent. This means byte mapping has to be biased toward protecting one-one and one-four combinations (four-four cannot be protected anyway). Accordingly, combination (a) in Fig. 3 is not a good option, because it cannot protect one-one case. Among the rest, (b) is more preferable because more valuable bytes are protected in the case of one-four combination. Hence, by limiting ourselves to at most four groups of byte-to-byte joiners, we take mapping of Fig. 3*b* as most efficient one which leads to better RF error resiliency. For a 32-bit RF, four control signals are required for any of the aforementioned mappings.

For example in the case of mapping in Fig. 3*b*, if 'ZYXW' and 'V' hexadecimal values are stored in reg-i and reg-i + 1, respectively, dummy sign bits are filled with redundant values as illustrated in Fig. 4 (top). For having easier routing, bytes can be reordered in one of the registers. In Fig. 4: bottom, reg-i + 1 is reordered. But this requires two multiplexers in input and output of RF to reorder during write and recover proper order during read (Fig. 5).

One superiority of our work in comparison with IRD is that, by pairing registers, ARH can protect long operands. For example, in Fig. 4, reg-i takes four bytes and three of these bytes are protected by reg-i + 1 which takes only one byte. However, in IRD, long operands that represent larger integers are not protected.

Below, we describe the mechanism for basic write/read operations.

3.2.1 Write access: The mechanism behind the write operation is critical to achieve efficiency. During a write operation, only meaningful bytes are written, while dummy sign bits should not be written. Respective bytes in the register are left intact because they may be keeping redundant data of the paired register. This can be satisfied by having byte selectable write enables. Besides this, while those meaningful bytes are being written and if their