POLITECNICO DI TORINO

Faculty of Electonic Engineering

Master degree course in Electronic Engineering

Master Degree Thesis

Configurable fault tolerant Instruction Fetch stage for cv32e40p core

Based on cv32e40p core of OpenHW Group organization

Advisors
prof. Stefano Di Carlo

Correlatore:
prof. Alessandro Savino

Candidate:
Elia Ribaldone

Marzo 2021

To my family and Anna.

Passion is the most powerful weapon.

Acknowledgements

Abstract

The miniaturization of the microelectronic components together with the use of integrated circuits in more and more application leads to an increasing use of FT (Fault Folerant) architecture. This thesis investigate the use of FT techniques in a stage of the cv32e40p open source core. We used fault injection simulation to divide our stage in three blocks with increasing level of

Summary

A **FT (fault tolerant) system** continues to work properly even if some of the internal components are broken; this feature is necessary when a failure may cause damage to people, dangerous destruction, military upset or loss of data. FT systems are essential in aerospace, transport, medical and utility industries and they are usually composed by a power source, an hardware system and a software system, each of these parts are fault tolerant depending on application.

This work concerns the hardware system and in particular the chip architecture design. In this context the faults are **transient**, **intermittent** or **permanent** and they are generated by manufacturing defects, system degradation or particle strikes. **Manufacturing** defects generate permanent faults and they reduce the yield with an increase in the cost per piece since the affected chips are discarded during quality control process. **System degradation** produces permanent faults and it is a life-limiting phenomenon that brings the chip to wearout phase. Finally, **particle strikes** produce both transient or permanent faults. These problems rely on the application: system degradation generally depends on chip temperature, clock speed and the workload, otherwise particle strikes rely on sources of alpha particles or neutrons, which are generated by the cosmic rays or radioactive materials.

For these reasons an ideal fault tolerant system should be protected against transient faults caused by particle strikes and it should manage permanent faults in order to increase the yield and chip life. A complete protection against faults creates drawbacks in speed, area and power budgets. This is the reason why we create a configurable architecture where faults coverage can be changed according to the specific application and the project constrains.

In this Master Thesis the **Instruction Fetch of cv32e40p** core is converted in a **configurable fault tolerant stage** in order to reduce failures in fetching instructions. The core used was designed by the "OpenHW Group" organization and it can be integrated in PULPissimo platform in order to create a complete microcontroller architecture.

Before the design of the architecture we study the cv32e40p core and then we proceed with the creation of the simulation environment, building the script used for all simulations. These tools born in the cv32e40p *core-v-verif* repository with te purpose of automatize compilation of testbenches and their simulation using QuestaSim and Modelsim.

The most important feature of the tool is the **optimized fault injection method** used to simulate faults in the architecture. We use a worst case approach injecting faults only in sequential parts (FF and memory), in this way we consider that all faults injected in the combinatory path (for example after a particle strike) reach a FF and are sampled.

This is not always true since some bits can be logical masked in the following cases: if they are don't care bits when fault occurs, if they can be electrical masked due to attenuation before latch or if the fault don't have the time to reach a FF (latch-widows). All this masks can be clustered in AVF (Architecture Vulnerability Factor) which is 1 if every fault generate a failure, otherwise it is lower. Knowing this we can assert that our tool works in the worst-case scenario since the AVF of each combinatory path (excluding inputs) is consider equal to one.

Apart from this first considerations the tool optimizes simulation times also takes advantage of *vcdstim* feature in the case of single stage simulations. Indeed the whole core is initially simulated using a benchmark firmware, meanwhile the input and output data of a specific stage are saved into .vcd and .wlf files, finally a stage-specific simulation started using .vcd as input (*vcdstim* feature). In this way only one stage is simulated and we reduce working times. **Stage-specific simulation** can be repeated a specified number of times using fault injection and the output of the stage is finally compared with .wlf output file in order to find failures.

Using our tool we first simulate fault injection in the reference IF stage of cv32e40p core and we find a fault tolerance equal to 30%. Later we use simulation results to understand the fault masking for each signals and then we divide **IF stage** in **three main blocks**. *Each block have increasing level of intrinsic fault tolerance in original architecture* and knowing this we apply FT techniques to each block. Anyway during simulations and synthesis we can enable FT for one, two or all blocks, depending on these settings we can manage area, speed, power and FT trade-off.

In the design of architecture we use a FT technique that is able to detect and correct transient faults using **TMR** (Triple Modular Redundant), this methods works only if each of the three identical blocks compared don't have permanent faults, and if we assume to neglect multiple particle strikes. Although multiple strikes is improbable, permanent faults is already present after manufacturing process and increase during time, for this reason we implement a technique to detect and correct this faults using some backup stages. Summarizing *we use TMR to manage transient faults and additional logic to protect against permanent faults.* In this way if all FT architecture of the IF stage are enabled during syntesis the final result is an higher yeld, a longer life and a protection against noise and particle strikes. The last important feature of the design is the saving of *permanent faults information* in the **CSR** (Common and Status Registers), in this way we could restore permanent faults settings after a reboot.

Contents

List of Figures

List of Tables

Listings

Chapter 1

Introduction

1.1 General context

[7231157]

1.2 Objectives

1.3 Thesis structure

Chapter 2

Fault Tolerance

Chapter 3

Basic architecture

Chapter 4

Environment and programs

Chapter 5

Simulation

Chapter 6

Conclusion

Chapter 7

Future works

Chapter 8

Articles Summary

8.1 A Co-Design Approach for Fault-Tolerant Loop Execution on Coarse-Grained Reconfigurable Arrays

Article: [2]
Anno: 2015

This article create a hardware/software co-design configurable on reliability of application and on real time SER (Soft error rate). The techniques used are Coarse-Grained Reconfigurable Arrays (CGRAs), DMR and TMR.

8.2 A dependence graph-based approach to the design of algorithm-based fault tolerant systems

Article: [4]
Anno: 1990

This article propose a two stage ABFT (Algorithm-based fault tolerance) system for computation intensive applications[4] ABFT is a method invented in 1984 in this article "Algorithm-Based Fault Tolerance for Matrix Operations" [1], the basic method encode data at high level and then the algorithm work on this data producing an encoded output. The computation is distributed along many unit to enhance fault tolerance, the method was first applied to matrix operation which are the basis of many intensive calculation. ABFT method can detect and correct errors in many matrix operation, to do this many processors are needed [1] . ABFT is a low cost CED (Concurrent error detection) scheme and fault location scheme [4]. ABFT is applied to: Multiplication, triangularization, fft, sorting, mesh array and hypercubes (in a hypercube processor each processor is the corner of a cube and can communicate only with n other corner). This paper present a method to synthesize ABFT system based on graph-theoretic model and the use of dependence graph. The basic idea of the graph theoretic model is to use a checksum in the matrix for each processor, at the end of the computation the results are compared and a fault processor can be found.

8.3 A Methodology for Alleviating the Performance Degradation of TMR Solutions

Article: [3]

Anno: 2010

Reliability degradation has increasing importance in faster and complex architecture due to sub 65nm technologies. In FPGAs the faults are more dangerous since can change design not just user data. TMR introduced by Xilinx ensures fault masking but increase chip area and power consumption, there is also a delay degradation due to redundancy that is catastrophic for critical application. This paper provide a method with reasonable balance between fault masking (F.M.) and performance/power overhead. Recent works use TMR in most sensitive subcircuits. In this letter, we introduce a software supported frame-work that provides a tradeoff between the desired level of fault masking and the performance/power degradation due to hardware redundancy. This is done by removing TMR from non sensitive subcircuits. The methods consist in the application of TMR to all processor, then the core is P&R (Place and route) to FPGA and finally the temperature distribution is analyzed. Since temperature distribution give guidelines about where it is most likely faults to occur!! since the failure probability for a device region increases with temperature. So it is possible to eliminate redundancy from parts of the design that operate under low temperatures without affecting practically fault masking.Given the affordable level of errors for a design, our methodology guarantees to find the maximum redundancy that can be selectively removed from noncritical for failure regions of the device.

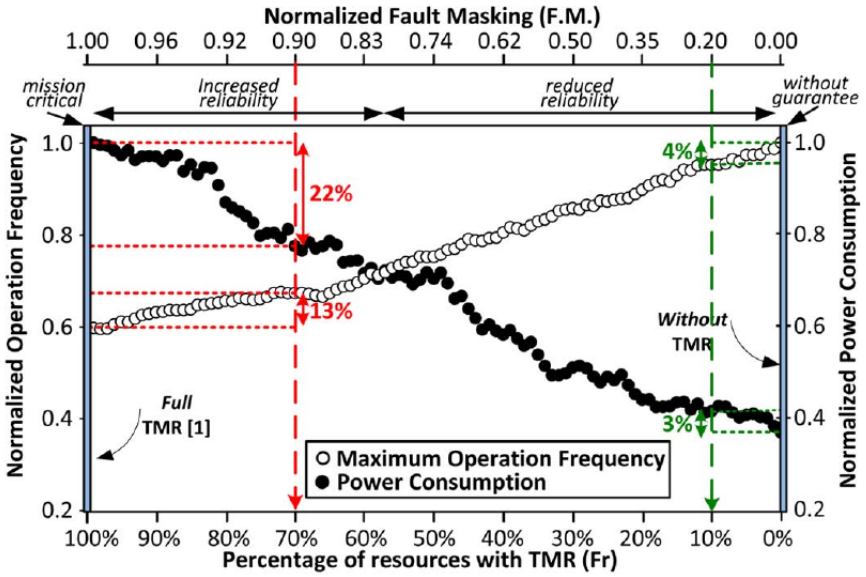

Figure 8.1: DIfferent fault masking exploration

Appendix A

An appendix

Bibliography

[1] Kuang-Hua Huang and J. A. Abraham. "Algorithm-Based Fault Tolerance for Matrix Operations". In: *IEEE Transactions on Computers* C-33.6 (1984), pp. 518–528. DOI: 10.1109/TC.1984.1676475.

[2] V. Lari et al. "A co-design approach for fault-tolerant loop execution on Coarse-Grained Reconfigurable Arrays". In: *2015 NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*. 2015, pp. 1–8. DOI: 10.1109/AHS.2015.7231157.

[3] K. Siozios and D. Soudris. "A Methodology for Alleviating the Performance Degradation of TMR Solutions". In: *IEEE Embedded Systems Letters* 2.4 (2010), pp. 111–114. DOI: 10.1109/LES.2010.2083632.

[4] B. Vinnakota and N. K. Jha. "A dependence graph-based approach to the design of algorithm-based fault tolerant systems". In: *[1990] Digest of Papers. Fault-Tolerant Computing: 20th International Symposium*. 1990, pp. 122–129. DOI: 10.1109/FTCS.1990.89347.