Automatic compiler-guided reliability improvement of embedded processors under proton irradiation

Alejandro Serrano-Cases[1], Yolanda Morilla[2], Pedro Martín-Holgado[2], Sergio Cuenca-Asensi[1] and Antonio Martínez-Álvarez[1]
[1] *Dept. of Computer Technology, Univ. of Alicante, Spain*
[2] *National Centre for Accelerators (Univ. of Sevilla, CSIC, JA), Spain*

Universitat d'Alacant
Universidad de Alicante

Centro Nacional de Aceleradores

Introduction

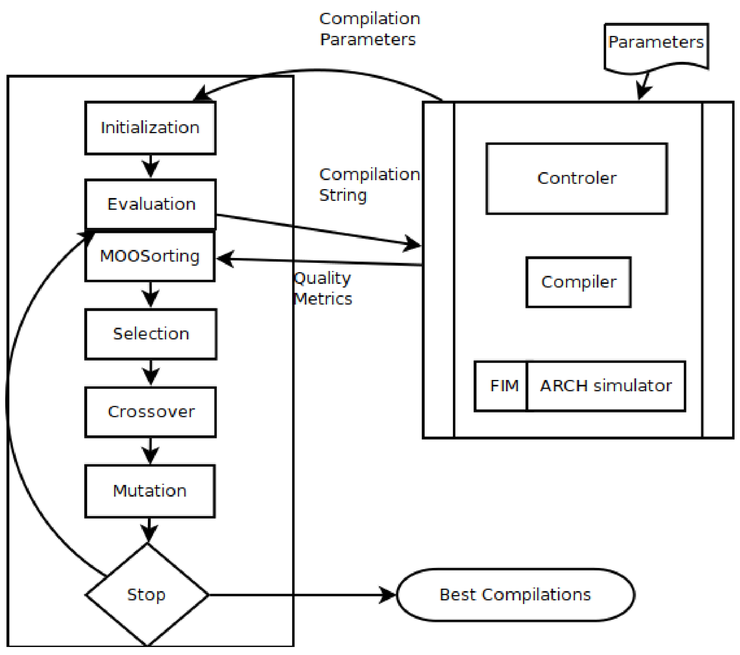
- COTS processors are not designed to cope with the harmful effects of radiation and, because of their nature, traditional hardware redundant techniques can not be applied to their structural components. A potential software method for modifying the reliability without instrumenting the code is operating the way programs are built (compiled). Modern compilers, such as GNU-GCC offer a set of optimizations which are intended to reduce the code size or the execution. However, they do not offer any predefined optimization associated with reliability improvements.
- The goal of this research is twofold: Demonstrate that the fault tolerance of embedded software can be improved just by selecting the appropriated compilation parameters and optimizations. And propose an **automatic exploration strategy** for tunning the compilation process **focused on softerror mitigation**.

Compiler-guided Hardening

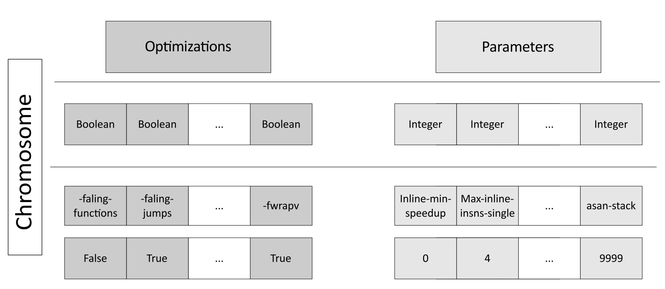
Our proposal combines two algorithms:

- ***Genetic Algorithm*** (GA) provides an efficient exploration of the solutions space. An individual is defined by its genes (an specific combination of compiler optimizations and parameters). GA makes use of crossover and mutation to combine two individuals in a new one sharing the genes of both parents.
- ***Multi-Objective-Optimization algorithm*** (MOO) ranks the individuals of a given generation taking into account different and opposed objectives that affect reliability. Non-dominate solutions offer the best trade-offs (**Pareto Optimal Front**).

⇒ MOOGA iterates over several generations offering all the individuals in the Pareto Front.



⇒ Chromosome definition.



Conclusions

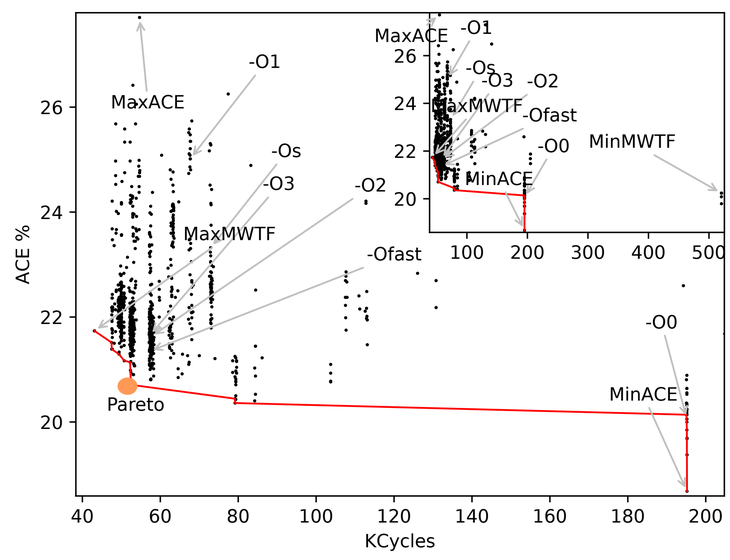
Modern compilers can be tune to improve code reliability by means of optimizing simultaneously fault coverage, execution time and memory size. Our automatic strategy offers an efficient method to find the versions with the best tradeoffs among them. Simulation results matches the behavior of applications under proton irradiation.

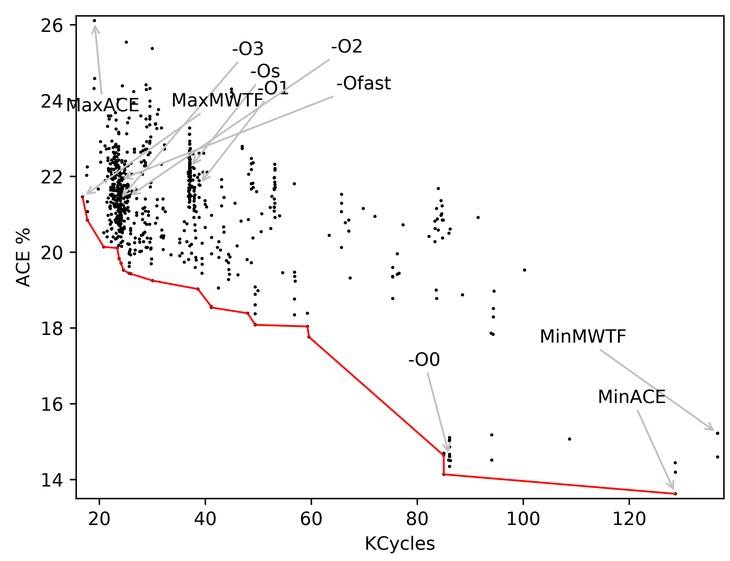
Experimental Setup

- Device Under Test: **ARM cortex-A9** architecture, embedded in a 28nm CMOS device *Xilinx Zynq XC7010 System on Chip*. Algorithms from *BeeBs* Benchmarks: Matrix Multiplication, QuickSort, NDES, Dijkstra and BubbleSort.
- Compiler: **GNU-GCC** from Linaro project, version *arm-eabi-gcc v7.2-2017.11* (Board Support Package provided by Xilinx).

MOOGA Simulation

Previous to the irradiation, an optimization stage using MOOGA was performed for each application to elicit their behavior in terms of **three objectives**: fault coverage (percentage of ACE faults, i.e. SDC+HANG), performance (execution time in Kcycles) and memory footprint (KiB).





⇒ Overall results. BubbleSort (left): 9% improvement on fault coverage whereas its performance improved about $12x$. NDES (right): 12% of fault coverage improvement and performance variation closed $5x$.
⇒ Interesting individuals: Pareto Front (red), default optimization flags (Ox), fault coverage boundaries (MaxACE, MinACE), Mean Work to Failure boundaries (MaxMWTF, MinMWTF).

Radiation Results

The testing campaign was carried out at the beginning of 2018 at the National Centre for Accelerators.
⇒ 15.2 MeV protons, beam uniformity >90%, fluence accuracy 10%

		Version	Cache	Flux $p/cm^2 \cdot s$	Fluence p/cm^2	Cycles	σ_{SDC}	σ_{Hang}	σ_{Total}	MWTF
BubbleSort	OCM	MaxACE	✓	$7,80 \cdot 10^{8}$	$3,36 \cdot 10^{12}$	44409	$3,33 \cdot 10^{-11}$	$4,17 \cdot 10^{-12}$	$3,75 \cdot 10^{-11}$	$3,96 \cdot 10^{14}$
		-O0	✓	$7,40 \cdot 10^{8}$	$4,87 \cdot 10^{12}$	220842	$1,56 \cdot 10^{-11}$	$8,21 \cdot 10^{-12}$	$2,38 \cdot 10^{-11}$	$1,26 \cdot 10^{14}$
		MaxMWTF	✓	$8,30 \cdot 10^{8}$	$3,13 \cdot 10^{12}$	38412	$2,53 \cdot 10^{-11}$	$1,15 \cdot 10^{-12}$	$3,68 \cdot 10^{-11}$	$4,67 \cdot 10^{14}$
		MinMWTF	✓	$1,00 \cdot 10^{9}$	$2,39 \cdot 10^{12}$	389806	$2,89 \cdot 10^{-11}$	$1,89 \cdot 10^{-11}$	$4,78 \cdot 10^{-11}$	$3,54 \cdot 10^{13}$
		Pareto	✓	$1,10 \cdot 10^{9}$	$3,37 \cdot 10^{12}$	39635	$1,93 \cdot 10^{-11}$	$1,13 \cdot 10^{-11}$	$3,06 \cdot 10^{-11}$	$5,45 \cdot 10^{14}$
	DDR	-O3	✗	$1,10 \cdot 10^{9}$	$8,64 \cdot 10^{12}$	646595	$1,74 \cdot 10^{-12}$	$5,09 \cdot 10^{-12}$	$6,82 \cdot 10^{-12}$	$1,50 \cdot 10^{14}$
		MaxACE	✗	$2,50 \cdot 10^{9}$	$7,44 \cdot 10^{12}$	666005	$4,30 \cdot 10^{-12}$	$9,14 \cdot 10^{-12}$	$1,34 \cdot 10^{-11}$	$7,38 \cdot 10^{13}$
		MinMWTF	✗	$2,50 \cdot 10^{9}$	$1,04 \cdot 10^{13}$	7495395	$1,35 \cdot 10^{-12}$	$8,49 \cdot 10^{-12}$	$9,84 \cdot 10^{-12}$	$8,95 \cdot 10^{12}$
NDES	OCM	MinACE	✓	$1,20 \cdot 10^{9}$	$1,82 \cdot 10^{12}$	127210	$3,96 \cdot 10^{-11}$	$1,92 \cdot 10^{-11}$	$5,88 \cdot 10^{-11}$	$8,82 \cdot 10^{13}$
		MaxMWTF	✓	$1,10 \cdot 10^{9}$	$9,36 \cdot 10^{11}$	10658	$9,73 \cdot 10^{-11}$	$1,60 \cdot 10^{-11}$	$1,13 \cdot 10^{-10}$	$5,47 \cdot 10^{14}$
		-O0	✓	$9,70 \cdot 10^{8}$	$2,17 \cdot 10^{12}$	93639	$3,68 \cdot 10^{-11}$	$1,52 \cdot 10^{-11}$	$5,20 \cdot 10^{-11}$	$1,36 \cdot 10^{14}$
	DDR	-O0	✓	$9,90 \cdot 10^{8}$	$1,60 \cdot 10^{12}$	93954	$5,12 \cdot 10^{-11}$	$1,81 \cdot 10^{-11}$	$6,93 \cdot 10^{-11}$	$1,01 \cdot 10^{14}$

⇒ Correlation between the simulated and irradiated results can be seen highlighted (red means worst reliability, green means best reliability)
⇒ BubbleSort evaluated without using any on-chip memories (OCM or cache) shows also a good correlation.

Acknowledgement

This work was funded by the Spanish Ministry of Economy and Competitiveness and the European Regional Development Fund Refs: *ESP2015-68245-C4-3-P and C4-4-P.*