Implementation and Evaluation of Countermeasures in a DIFT Mechanism against Fault Injection Attacks

William PENSEC, Vianney LAPÔTRE, Guy GOGNIAT
Lab-STICC, UMR 6285, Université Bretagne Sud, Lorient, France
firstname.lastname@univ-ubs.fr

Context

Internet of Things (IoT) devices have revolutionised data collection and analysis, yet their proximity raises concerns about physical attacks like Fault Injection Attacks (FIA). Numerous studies have highlighted critical system vulnerabilities to FIAs [1]. This poster presents an evaluation of different Hamming code [2] and SECDED (Single Error Correction Double Error Detection) implementation strategies in order to have a robust protection against FIA, taking into account constraints such as performance, area and efficiency.

D-RI5CY

We study the D-RI5CY [3] which introduces a Dynamic Information Flow Tracking (DIFT) mechanism to protect the processor against software attacks such as buffer overflows, SQL injections, etc. DIFT-related elements are represented in red.

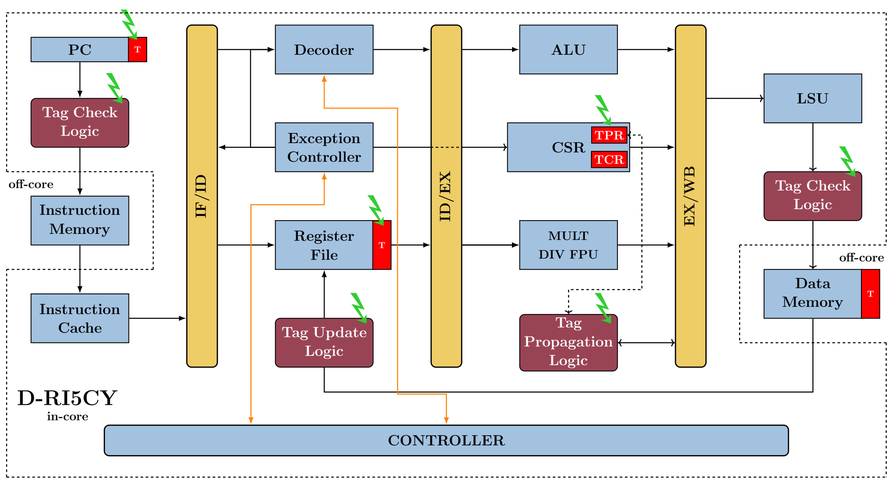

Figure 1: D-RI5CY processor architecture overview

Proposed approach

When multiple registers have to be protected, different implementation strategies can be considered to generate redundancies.

We propose 5 strategies:

- Minimisation of redundancy bits
- 1 group by pipeline stage
- 1 group by register
- 1 group by register and slicing of TPR and TCR registers
- 1 group for 2 bits of 2 different registers

We target 55 DIFT-related registers (127 bits) and all registers storing redundancy bits. This represents up to 133 registers (280 bits).

Focus on strategy 5

The main idea is to generate redundancy bits from input bits of different registers. Figure 2 presents this strategy.

Our rules for this strategy are:

- We associate 2 bits of 2 registers.
- Two bits in one register cannot be associated with the same two bits in another register, except in the rare case where no register is available (as in Figure 2 for R_1 and R_2).

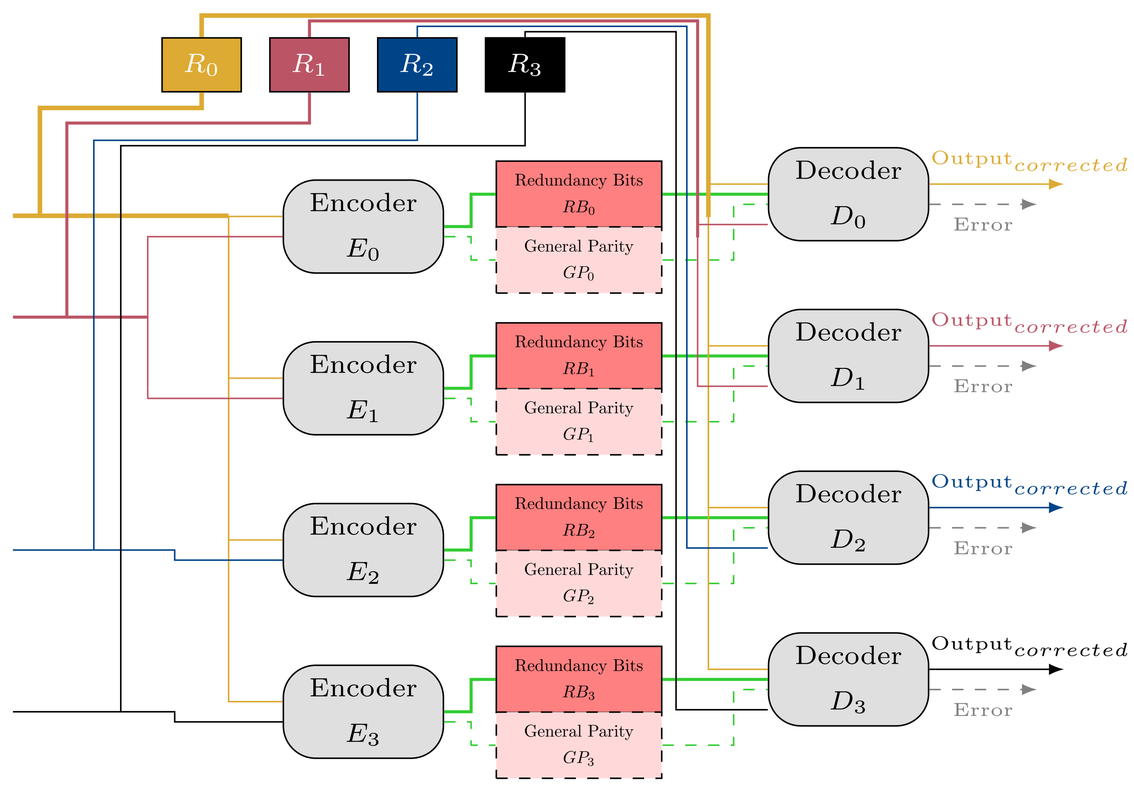

Figure 2: Proposed approach for the implementation 5

This strategy leads to the need to inject two faults into two associated bits in order to be bypassed, which reduces the possibility of successful attacks.

Experimental results

We conduct fault injection attacks in simulation using a tool, called FISSA [4]. We target two fault models : *single bit-flip in two registers at a given clock cycles* and *exhaustive multi-bits faults in two targets at a given clock cycle*. The second fault model consists by injecting up to 6 faults (max registers size) twice in specific register.

Table 1: Results for a buffer overflow attack with single bit-flip in two registers at a given clock cycles

	Silent	Delay	Single Error Correction	Double Error Detection	Success	Total
Baseline	45,097	1503	–	–	1406 (2.93%)	48,006
Hamming Code 1	0	575	67,829	–	452 (0.66%)	68,856
Hamming Code 2	0	297	72,867	–	312 (0.42%)	73,476
Hamming Code 3	0	263	108,326	–	281 (0.26%)	108,870
Hamming Code 4	0	57	155,112	–	99 (0.06%)	155,268
Hamming Code 5	0	55	173,367	–	98 (0.06%)	173,520

Table 2: Results for exhaustive multi-bits faults in two targets at a given clock cycle

	Silent	Delay	Single Error Correction	Double Error Detection	Success	Total
Baseline	67,072	926	–	–	450 (0.66%)	68,448
SECDED 1	0	1911	150,791	170,575	723 (0.22%)	324,000
SECDED 2	0	1186	170,805	184,761	584 (0.16%)	357,336
SECDED 3	0	1230	300,260	263,665	669 (0.12%)	565,824
SECDED 4	0	18	457,498	368,959	61 (0.01%)	826,536
SECDED 5	0	39	576,992	401,407	66 (0.01%)	978,504

Conclusion and Perspectives

In this work, we study different strategies to better protect a security mechanism, called DIFT, against Fault Injection Attacks. These strategies show some good results against our two fault models, even if there are still some successes due to the fact that our protections are not designed to support a multi-faults model. Our protections leads to an area overhead up to 8% and decrease the maximal frequency by less than 1.7%.

In future work, we plan to apply the proposed strategies to Error Correcting Codes (ECC) able to detect more than 2 bit-flips.

Bibliography

[1] S. Nashimoto *et al.*, “Bypassing Isolated Execution on RISC-V using Side-Channel-Assisted Fault-Injection and Its Countermeasure,” 2021. DOI: 10.46586/tches.v2022.i1.28-68.

[2] R. W. Hamming, “Error detecting and error correcting codes,” 1950. DOI: 10.1002/j.1538-7305.1950.tb00463.x.

[3] C. Palmiero *et al.*, “Design and implementation of a dynamic information flow tracking architecture to secure a RISC-V core for IoT applications,” in *High Performance Extreme Computing*, 2018. DOI: 10.1109/HPEC.2018.8547578.

[4] W. Pensec, *Fault Injection Simulation for Security Assessment*. [Online]. Available: https://github.com/WilliamPsc/FISSA.