Radiation Effects to FPGAs: An Overview

George Tsiligiannis

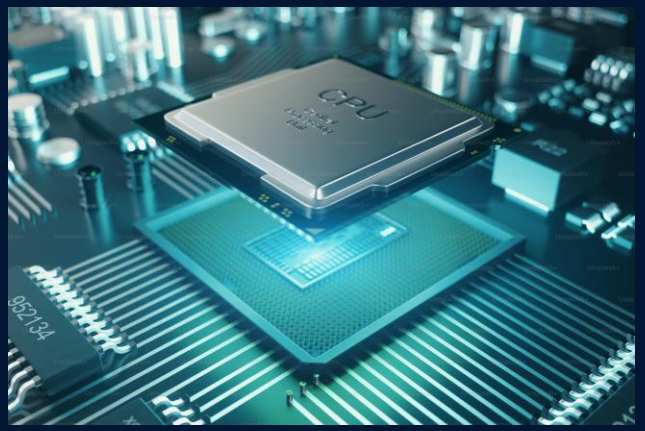
July 2026

Outline

- Introduction
- Radiation Effects to electronics
- FPGA – Introduction
- SEEs to FPGAs
- TID to FPGAs
- Application-Level Testing
- Fault Injection
- Conclusions

Introduction

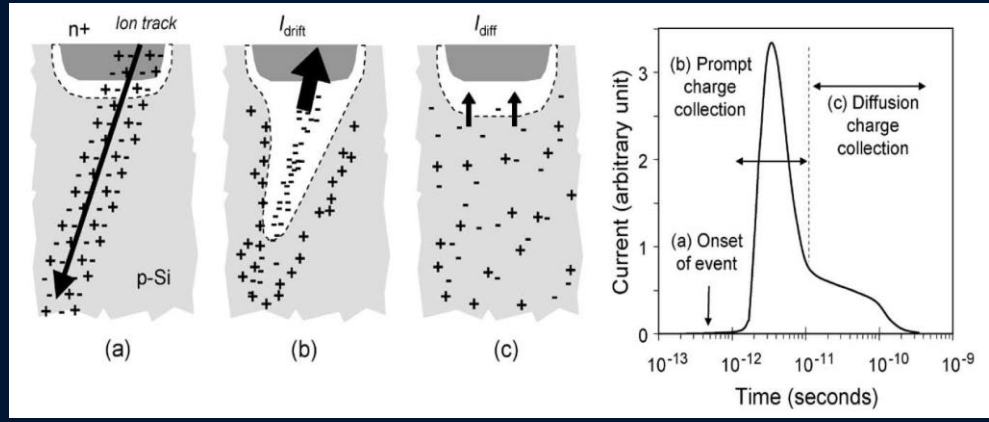
- Why FPGAs and not… CPUs or ASICs?
  - Pros: versatility, re-programmability, hardcoded IPs (ADC,DAC,MCU etc), short development cycles, cost, real time, time-to-market
  - Cons: versatility(???), hardcoded IPs, power consumption, pinout, thermal management, packaging, rigid development flow, proprietary tools, support
- What applications are we looking at that care about radiation effects?
  - Space, Avionics, Nuclear, Particle Accelerator, Safety, Critical
- What environments?
  - Mixed field, proton, heavy ion, electrons, neutrons
- So then how does that impact FPGAs?



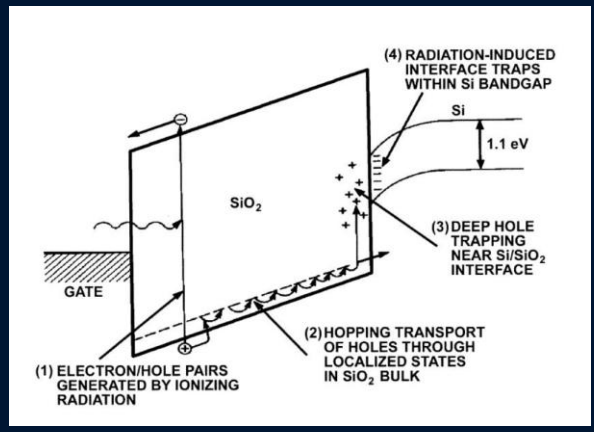
Courtesy of NASA

Radiation Effects to Electronics

- Particle Interactions with matter result to different types of effects: Single Events and Cumulative
- Single Events are divided in categories such as: SBU, MBU, MCU, SET, SEFI, SEL, SEB
- Cumulative Effects are also divided: TID, TNID
- SEEs can be categorized on two main types of interactions:
  - Direct ionization -> charge deposition from charged particle (HI, LE protons)
  - Indirect ionization -> charge deposition from products of nuclear inelastic interactions (Protons, Neutrons, electrons)
- Cumulative effects like TID are caused by the ionization induced by particles interacting with atoms in the material -> electron-hole pair generation -> transport to the oxide -> parametric drift (threshold voltage, transconductance and leakage)
- TNID effects are caused by the displacement of the nuclei resulting into parametric drift (typically bipolar)



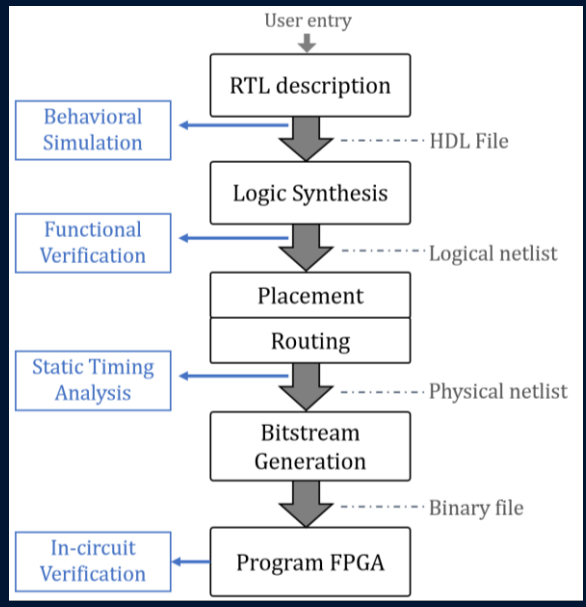
R. C. Baumann, "Radiation-induced soft errors in advanced semiconductor technologies," *IEEE Trans. Device Mater. Relib.*, vol. 5, no. 3, pp. 305–316, Sep. 2005, doi: 10.1109/TDMR.2005.853449.



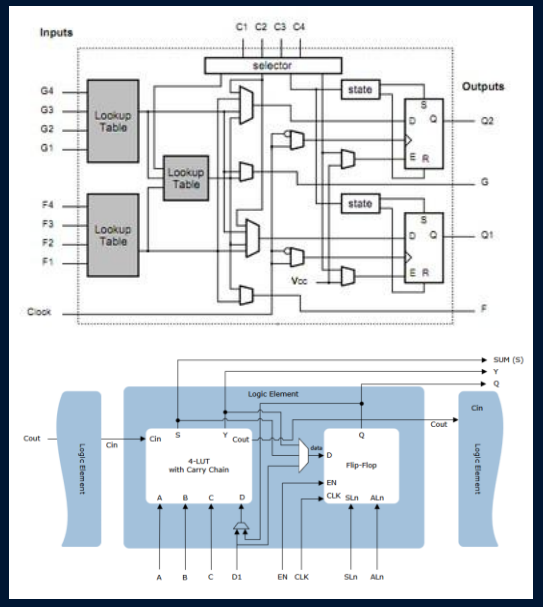
F. B. McLean and T. R. Oldham, "Basic mechanisms of radiation effects in electronic
materials and devices," Harry Diamond Laboratories Technical Report, vol. HDL-TR, p.
2129, 1987

FPGA – Introduction

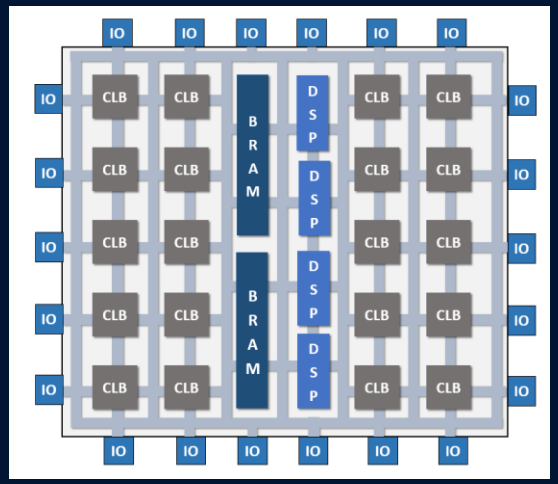
- Field Programmable Logic Arrays (FPGA) are highly configurable ASICs deploying large arrays of configurable logic, hardcoded IPs etc
- They have a similar flow to ASICs with some key differences:
  - No standard cells, instead, FPGA basic logic elements are called primitives (unique to each device family) and are used for the mapping from the synthesizer
    - Primitives for different categories: CLB(LUT, MUX, Carry, SRLC32E etc), Clock (BUFG, DPLL etc), I/O (IBUF, IBUFDS, PULLDOWN etc), Register (FDCE, ODDRE1 etc)
  - No memory compiler – block ram (inferred or instantiated)
  - Hardcoded IP blocks (PLL, Ethernet MAC, AES, ADC/DAC, DSP etc) – inferred or instantiated
- PnR maps those primitives to the FPGA floorplan and connects them using the existing routing resources
  - Switchboxes, Programmable Interconnection Points (PIP) and other hierarchical groups of logic resources are used to connect the primitives, hard blocks etc
  - Like in the PD flow, FPGA flow allows for floorplaning, regions etc
- Final bitstream creation which is used to write the configuration memory
- All those primitives are configured through the Configuration Memory (volatile or non-volatile)



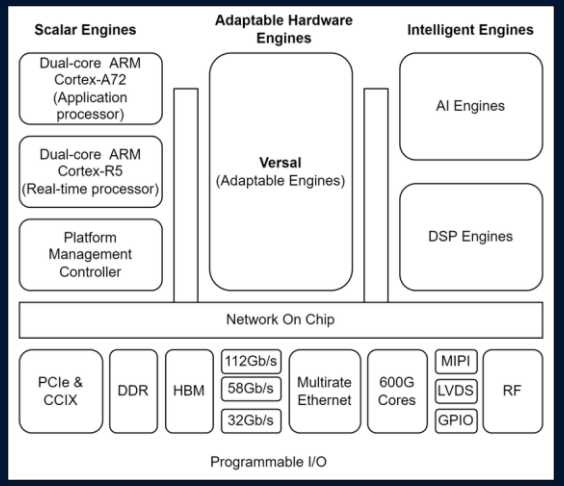
Radiation reliability analysis of FPGA-based systems: testing methodologies and analytical approaches, G. Bricas



Radiation reliability analysis of FPGA-based systems: testing methodologies and analytical approaches, G. Bricas



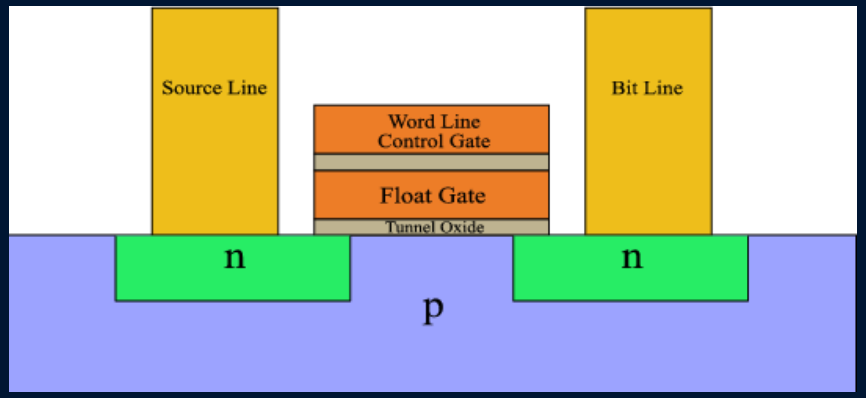
Radiation reliability analysis of FPGA-based systems: testing methodologies and analytical approaches, G. Bricas



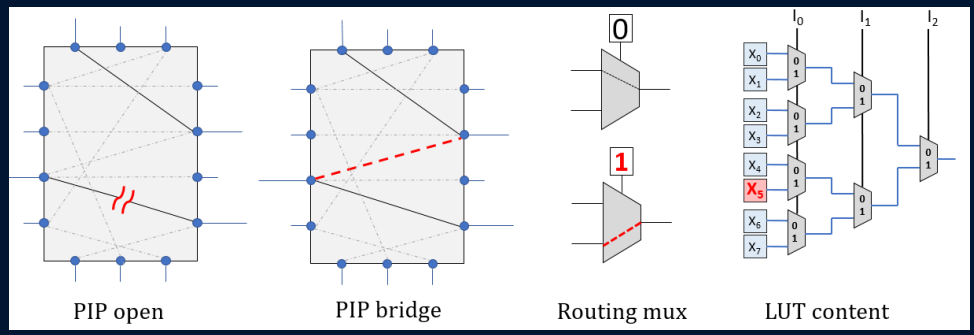
Electrical Tomography Hardware Systems for Real-Time Applications: a Review

SEEs to FPGA – Configuration Memory

- Configuration memory of FPGAs is typically of three categories: SRAM, FLASH and AntiFUSE
  - SRAM memory can exhibit SBUs, MCU, MBUs, SEL etc
  - FLASH memory can exhibit upsets but typically for high LET particles (HI) and is rather rare
  - AntiFUSE memory cells are considered immune
- Bit flips on configuration memories can be catastrophic as it impacts the design implementation.
  - PIP, MUX or LUT reconfiguration on the CLBs can be catastrophic
- Testing of configuration memory is achieved by programming the FPGA with a bitstream (golden reference), irradiate, and then reading back the bitstream and comparing the results
  - For accurate results the vendor must contribute to this with tools that can accurately decode the bitstream (not all bytes of the bitstream correspond to configuration memory bytes) -> significant efforts have been put to do this without vendor tools
  - Using the Soft Error IP (SEM IP) for older families or the XilSEM for Versal devices (soft IP on fabric vs firmware on the Microblaze-based Platform Loader and Manager (PLM) -> Error detection and correction
- Mitigation
  - Both Xilinx and Altera scrub their memories in different ways to mitigate such effects.
    - May require non-volatile memory to store the bitstream for reconfiguration
    - High latency – few to tens of ms



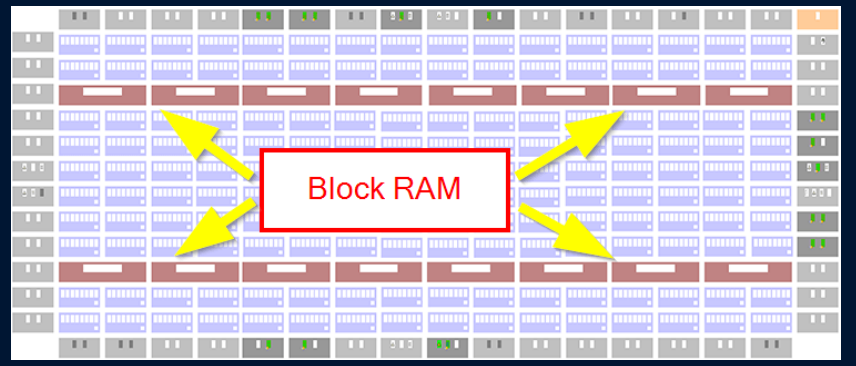
Basic scheme of a flash memory cell. Depending on the charge stored in the floating gate one bit SLC or multiple bits MLC can be saved. Courtesy of THOMAS WINDBACHER



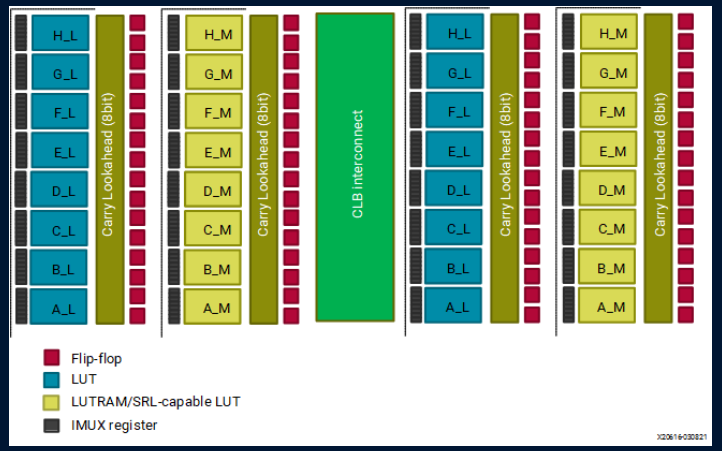
Radiation reliability analysis of FPGA-based systems: testing methodologies and analytical approaches, G. Bricas

SEEs to FPGA – Fabric Memory

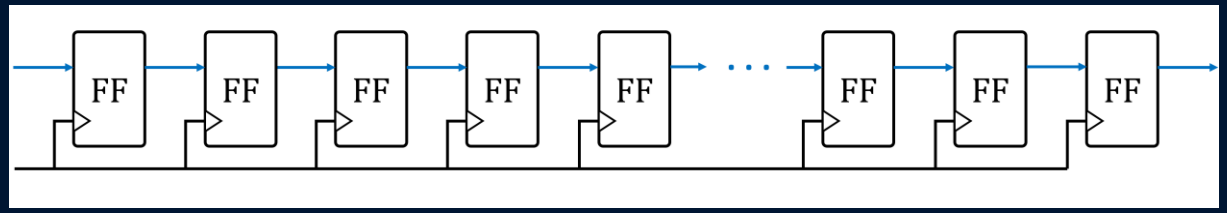
- Block RAMs (BRAM) are SRAM blocks that are distributed across the die for use by the design
- Distributed RAM (Xilinx): LUTs can be configured as RAM or Shift Registers (SRL) to increase internal memory size
  - SEB,MCU,MBU etc can appear in the BRAM
- Fabric Flip Flops
  - Lots of different primitives, each one should have its own dedicated structure
    - FFs without reset, with reset, with/without set pins, etc
- how do we test them?
  - BRAM can be tested either by using BIST structures inside the FPGA, or by using an external device and wiring directly the BRAM I/Os to the FPGA I/Os
  - Static/Dynamic testing
  - FFs are tested with flip flop chains -> floorplan? Interconnection? Can we read statically/passively the flip flops?
    - Combined testing using logic in between (LUTs implementing a function, muxes, carry logic etc)
- Mitigation
  - ECC for memories
  - TMR or DMR for FFs and alike schemes of mitigation. Beware! It doesn't work always!



Courtesy of Vhdl Wiz



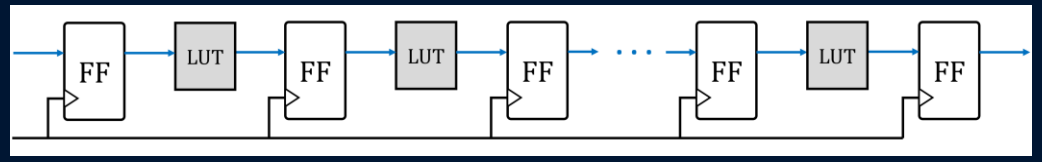
Courtesy of Xilinx



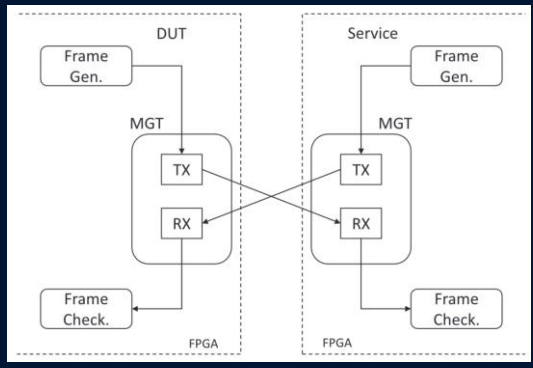
Radiation reliability analysis of FPGA-based systems: testing methodologies and analytical approaches, G. Bricas

SEEs to FPGA – SET, SEFI, SEL

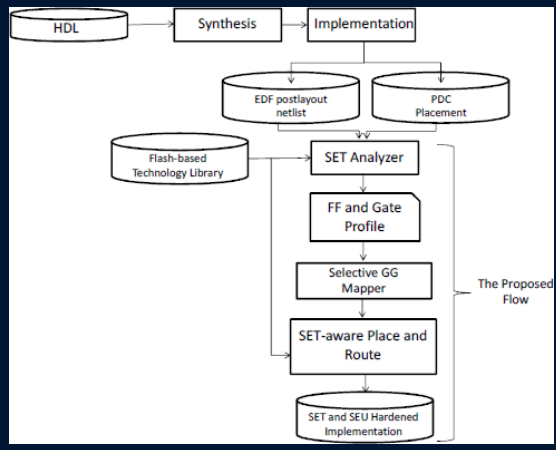
- LUTs, Multiplexers, PIPs, Carry Logic can exhibit transients, which if propagated can be captured by FFs
- Global Routes are a separate chapter of their own as they have their independent routing logic with buffers etc (reset,clock)
  - SETs on Global Routes will most probably impact multitude of FFs leading to what is observed at system level as SEFI
- MicroSELs can demonstrate increased power consumption, not always detected by our tests
- Buffers which are typically used to reach timing can act either as enhancers or filters of SETs
- DSPs, ADC/DACs are logic blocks with multiple functions, registers and loops which can exhibit SETs, SEUs and other effects. They need to be managed individually
- How do we test them?
  - Dedicated chains that instantiate those primitives ideally with external testing structures
  - BIST for DSP, ADC/DACs with dedicated stimuli for dynamic testing and capturing of events against golden references
- Mitigation?
  - SET filtering has been a deep research field -> it has its pros and cons
  - Redundancy schemes at local or system level (distributed TMR, Global TMR, TMR at logic level etc) -> some of them are hitting timing really hard (besides area!) so they are not a universal solution
  - Some times, SETs cannot be mitigated. The system must be designed in a way to recover



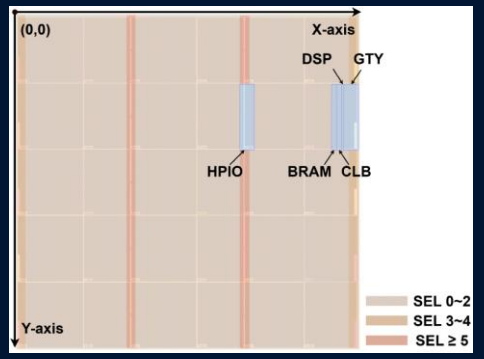
Radiation reliability analysis of FPGA-based systems: testing methodologies and analytical approaches, G. Bricas



Transceivers test structure. Two mirrored FPGA are used to transmit and received data frame between themselves “Evaluating Xilinx 7 Series GTX Transceivers for Use in High Energy Physics Experiments Through Proton Irradiation”



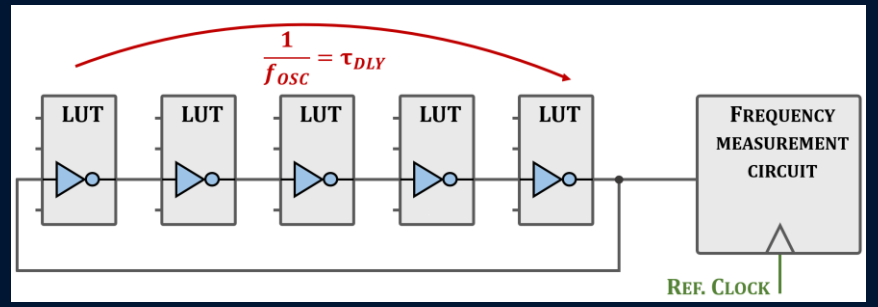
L. Sterpone and B. Du, "Analysis and mitigation of single event effects on flash-based FPGAS,"



M. Shen et al., "Single-Event Latch-Up Characteristics of 16-nm FinFET SRAM-Based Field-Programmable Gate Array Under Single-Photon Absorption Laser Testing,"

TID to FPGA

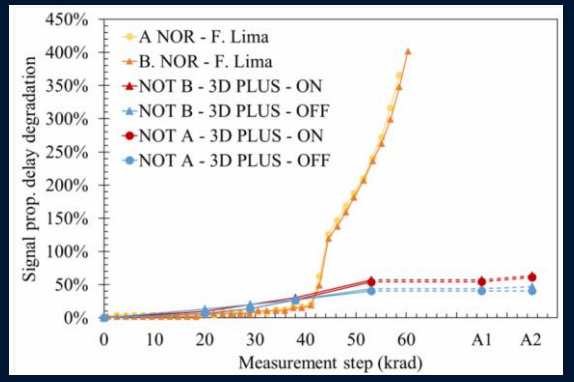
- FLASH-based FPGAs -> TID permanently biases FLASH cells towards either 1 or 0 state
  - Charge pump also gets broken leading to earlier stage failure regarding reprogramability
  - It takes just one essential bit to get broken for the entire design to brake
- Parametric drift of propagation delay (increase or decrease!)
  - Propagation delay also impacts power consumption of the FPGA
- IOs also have their own complexity as they implement small circuitry that is also susceptible to TID as a result of their ESD protection mechanism
- Power on Reset
- ADC/DACs
- How do we test them?
  - Reference with flash fpga tests
  - Chains of DSP LUT and Carry logic from one side of the FPGA to the other captured with some sort of circuitry to measure propagation delay
- Mitigation
  - Temperature annealing, design margins!!!



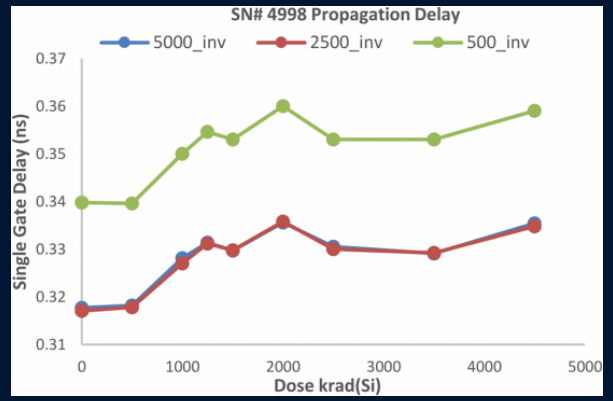
Radiation reliability analysis of FPGA-based systems: testing methodologies and analytical approaches, G. Bricas

Device Family	Feature Size (nm)	Total Dose (krad(Si))
Virtex [34]	220	100
Virtex-II [34]	150/120	200
Virtex-4 [34]	90	300
Virtex-5 [34]	65	340
Virtex-6 [34]	40	380
Kintex [35]	25	446
UltraScale [36]	20	100
UltraScale+ [37]	25	200

D. M. Hiemstra, "A Review of Xilinx Field Programmable Gate Array Radiation Effects Performance Part II,



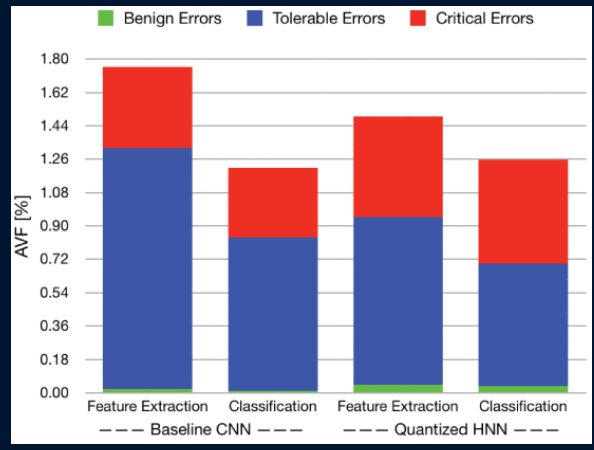
A. L. Bosser *et al.*, "Review of TID Effects Reported in ProASIC3 and ProASIC3L FPGAs for 3D PLUS Camera Heads," . Comparison of signal propagation delay increase through A and B inverter chains with data from F. Lima-Kastensmidt et al.



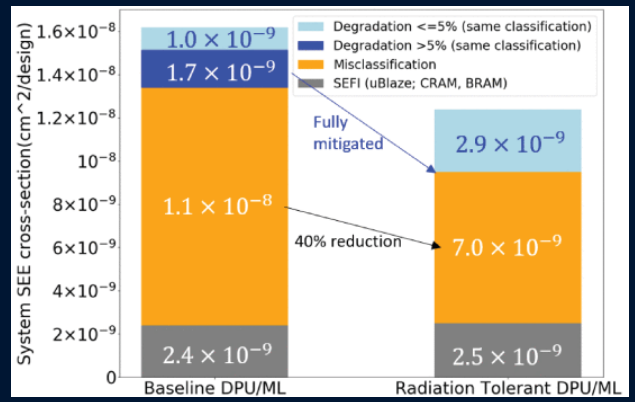
S. Vartanian, G. R. Allen and D. Thorbourn, "SRAM-Based FPGA: High Dose Test Methods Using Evaluation Boards,"

Application-Level Testing

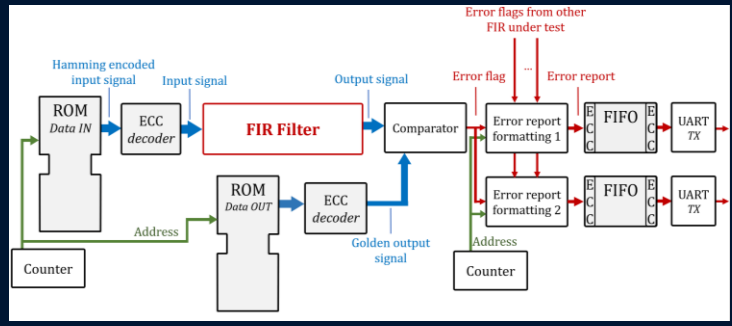
- Application-level testing has its own value but it cannot substitute the FPGA qualification process we described
- SEE System-level testing can be useful but also challenging for many reasons:
  - Compare mitigation methodologies and system response (PPA tradeoffs analysis)
  - Understand how the system recovers during scrubbing and reconfiguration in case of CRAM loss -> recovery time, boot process etc
  - Collect failure patterns (it can never be exhausting…)
  - Accelerated testing can be very challenging especially for SRAM-based FPGAs
    - SEMIP and the like cannot handle high fluxes -> moreover such fluxes are not realistic… beam degraders for example may be used
    - Most failures are a result of CRAM bit flips and not related to the design itself -> of course, if we come up with a design using less resources, we will see less failures ;)
- DSP applications, AI accelerators, cryptographic cores, soft processor IP (RISCV etc)
  - MPSoC applications combining MCU+FPGA Fabric make testing much more complex -> only a tiny fraction of what actually happens is observable/understandable
- Monitoring and reporting of subsystems can be of great help!
- Benchmarking structures -> the middle ground…



F. Libano, "Understanding the Impact of Quantization, Accuracy, and Radiation on the Reliability of Convolutional Neural Networks on FPGAs,"



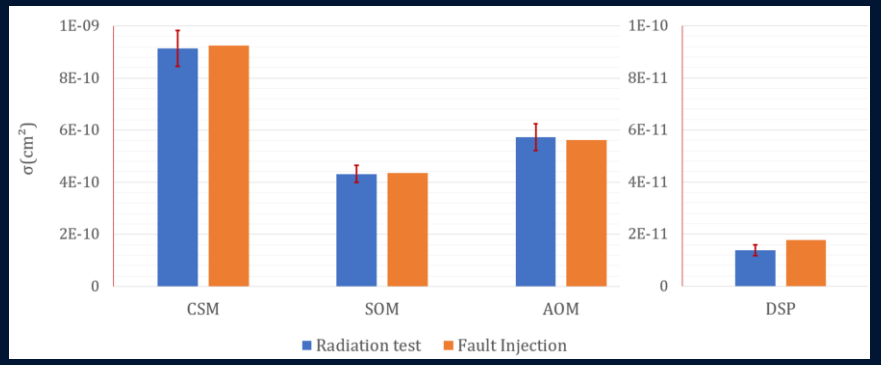
P. Maillard *et al.*, "Radiation-Tolerant Deep Learning Processor Unit (DPU)-Based Platform Using Xilinx 20-nm Kintex UltraScale FPGA,"



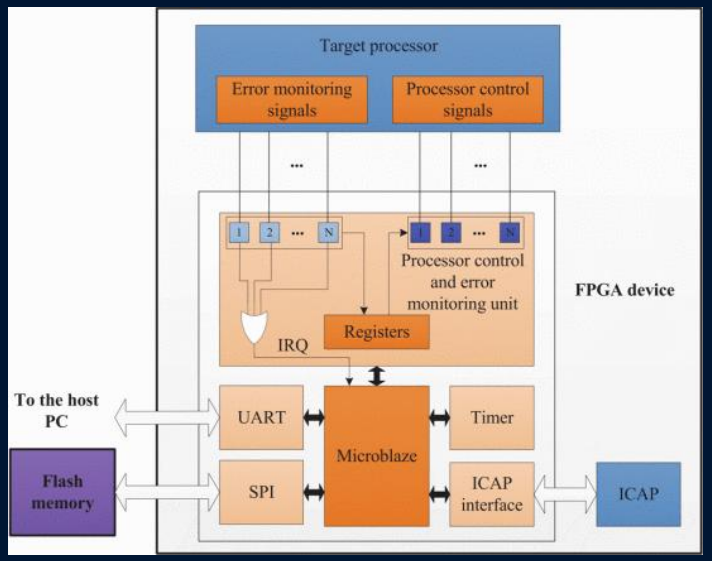
Radiation reliability analysis of FPGA-based systems: testing methodologies and analytical approaches, G. Bricas

Fault injection

- Fault injection is the process of injecting faults to specific parts of the design without using particles
  - Laser: laser beam targeting specific parts of the design trying to inject faults and assess system performance -> difficult to access, difficult to assess specific parts of the design
  - Simulation: RTL simulation tools, either on the design, or on the synthesized netlist fault injection through muxes and net values forcing. -> too long to implement
  - Emulation:
    - fault injection by writing the essential bits from configuration memory in real time (ICAP)
    - RTL structures that get activated in real time (muxes FF inputs)
- Does it make sense?
  - Laser testing can be useful but floorplan information is critical
  - Simulation based fault injection is a good learning tool but cannot scale well for large designs – only small critical parts are interesting
  - Emulation results can approximate very well radiation campaigns



Carry Save Multilpier, Speed Optimized Multiplier, Area Optimized Multiplier



A. Sari and M. Psarakis, "A fault injection platform for the analysis of soft error effects in FPGA soft processors,"

Conclusions

- FPGAs are very complex to test
  - A lot of fundamental blocks that need evaluation
  - Multi-year projects that require deep expertise
  - Difficult to explain and observe events
  - Lack of statistically significant events makes things hard to evaluate
- Each building block must be approached as a different device
- SEE and TID evaluation require exhaustive testing that can require hundreds of beam hours
- Application testing is useful but we can’t rely on that only
- Fault injection is a strong tool that allows evaluation of a design and mitigation strategy
- Benchmarking may reduce testing time and complexity

THANK YOU!!!!