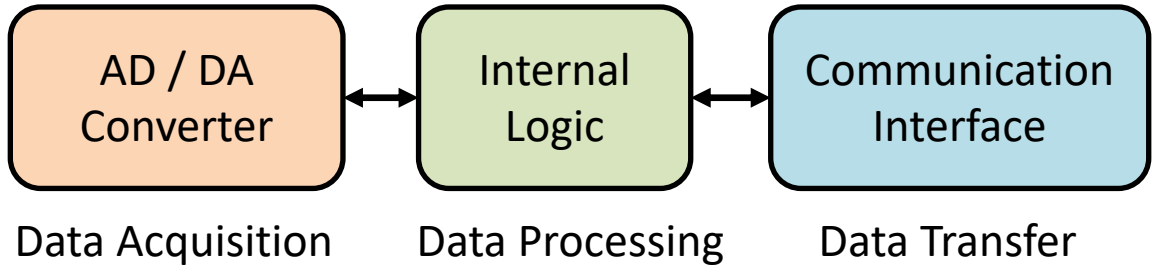
Fachhochschule
Dortmund
University of Applied Sciences and Arts

Fault Tolerance Evaluation of a RISC-V Microprocessor for HEP Applications

TWEPP 2023 – October 3rd, 2023

Introduction

- Many custom ASICs have a similar structure:

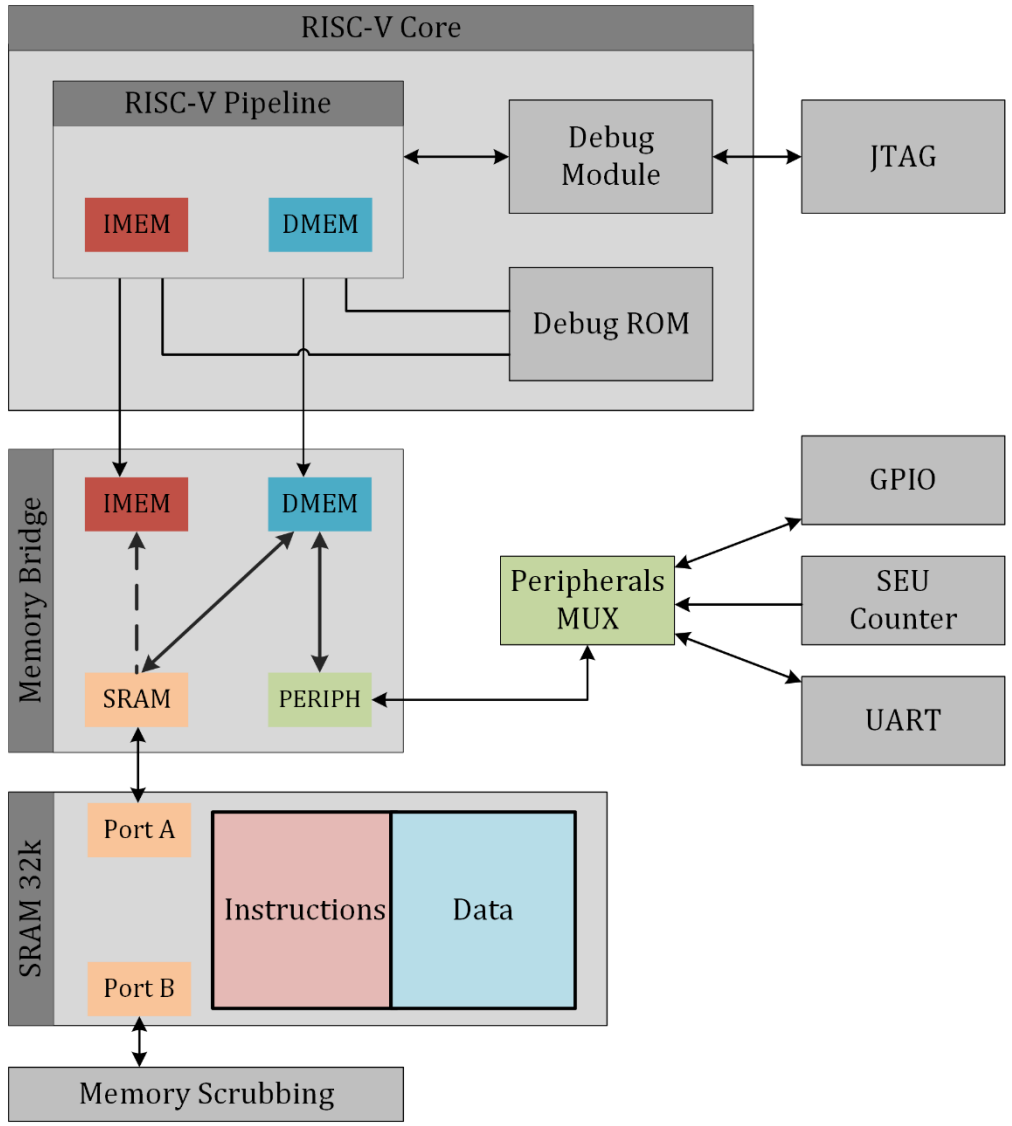


- Design and verification of a custom ASIC is complex and time-consuming
- Reuse of generic blocks possible (ADC, voltage regulators, etc.)
- Adaptation of internal logic difficult, custom to original application

- Internal data processing logic replaced by with RISC-V processing system
  - Adaptation to new application / Bugfixes via firmware updates
- Hybrid detector with RISC-V-based microprocessor SoC

STRV-R1 – Architecture

- RV32-IMC Core
  - 3 stage pipeline
  - Multiplication extension
  - 50 MHz @ 1.2V
  - Fully triplicated core
- SRAM shared between instruction & data
  - Flexible memory layout
  - IMEM & DMEM data bus can access whole SRAM address range
  - RISC-V pipeline stalls during load & store instructions to SRAM
  - load & store to peripherals simultaneously possible
- JTAG Interface
  - JTAG TAP & debug module
  - Non-volatile debug ROM with debug ISR

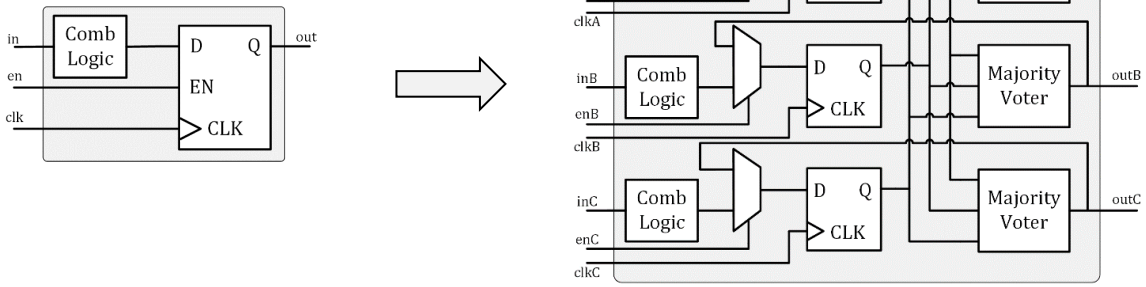


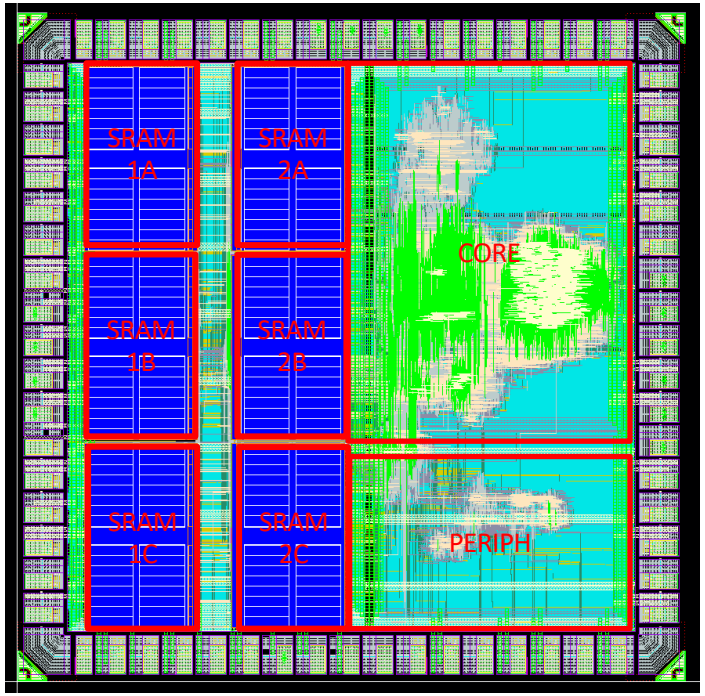
STRV-R1 – Implementation

- 2mm x 2mm in 65nm Technology

- TMR strategy in RISC-V Core:
  - Triplication of
    - All sequential elements
    - All combinational logic
  - Majority voter after every sequential element
  - Additional feedback path
  - Three separate clock-trees

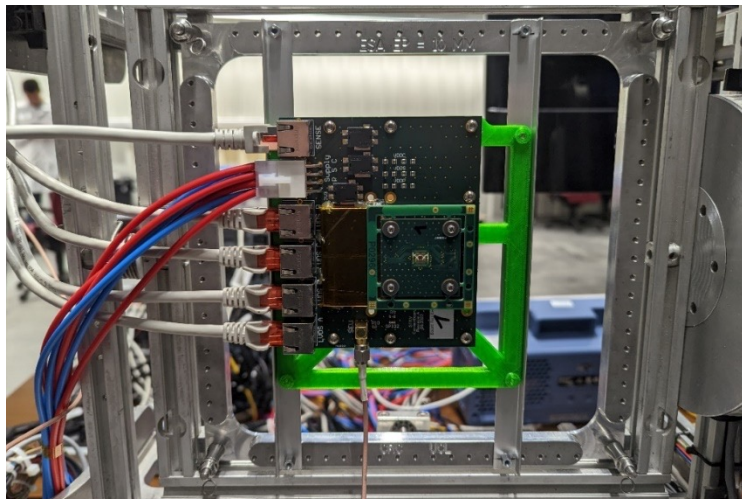
- TMR SRAM strategy:
  - 3 dual-port SRAM instances
  - Majority voter in datapath to core
  - Scrubbing on second SRAM port
  - 3x 32Kbyte
  - Divided into two 16-bit wide SRAM cells
  - Scrubbing time limit 320µs @ 50MHz

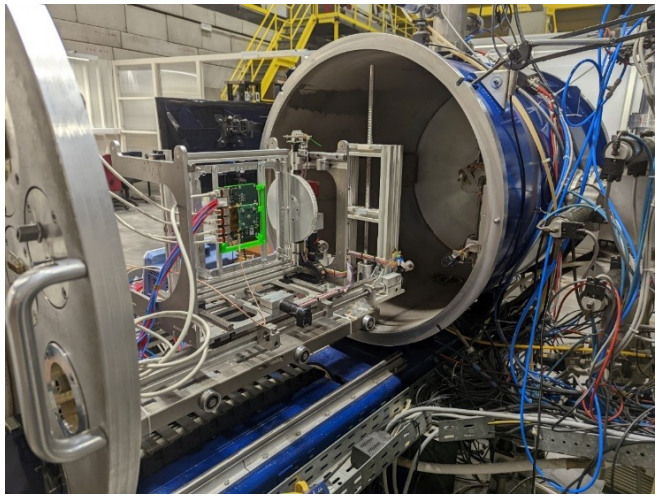


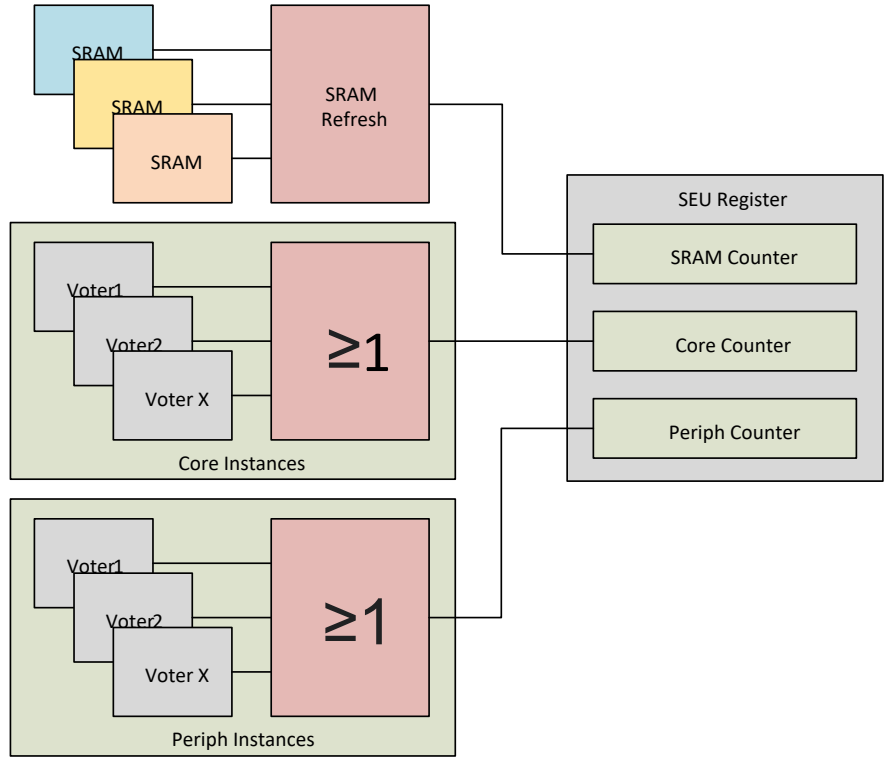


STRV-R1 – SEU Detection

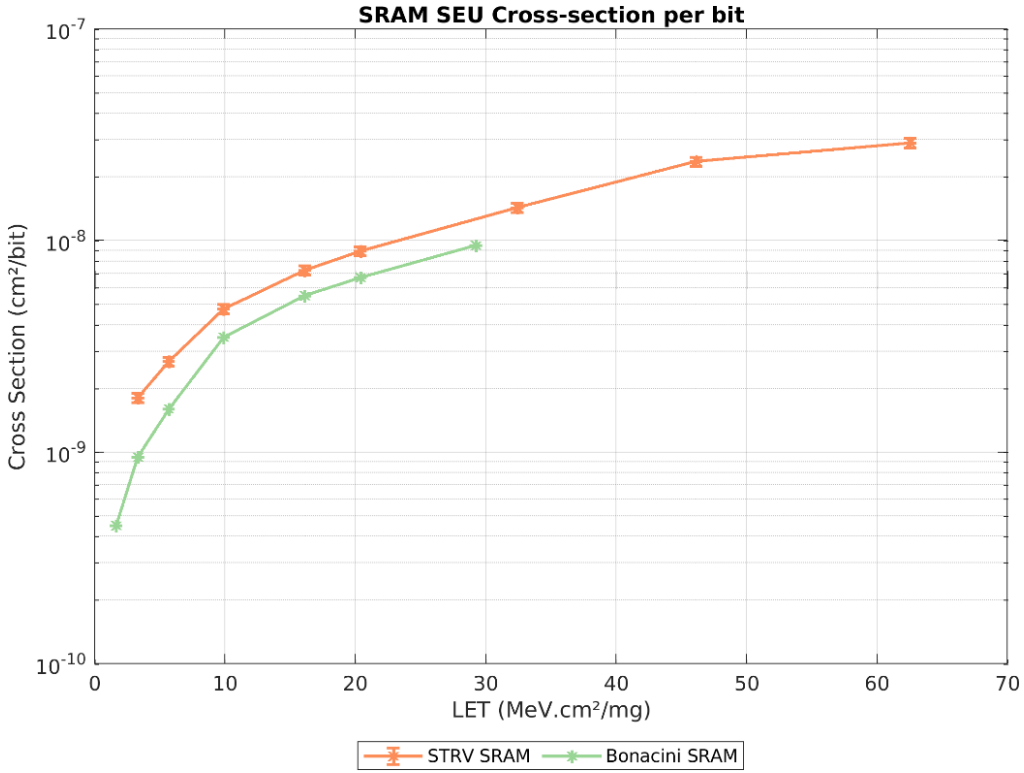
- Detection of occurred SEUs during irradiation
- Externally via test system and internally via integrated counters
  - 32Bit counter accessible via memory mapped registers
  - RISC-V core:
    - Output of majority voters
    - Routed through or-gate tree
  - Detection in the SRAM:
    - During data access by RISC-V core
    - Continuous data scrubbing on secondary SRAM port





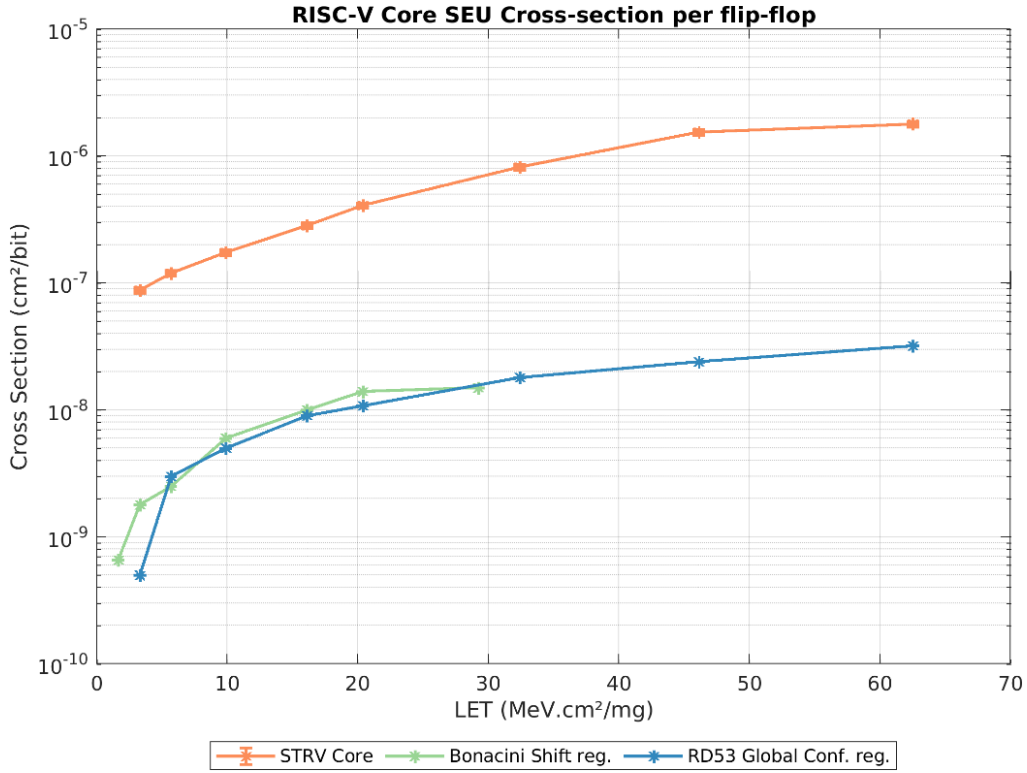


STRV-R1 – Heavy-Ion Irradiation



SEU Cross-Section of SRAM macros

- Good agreement with previously published 65nm technology characterization



SEU Cross-Section of sequential elements

- Larger cross section compared to published 65nm technology characterization
- Likely caused by different architecture / additional combinational logic

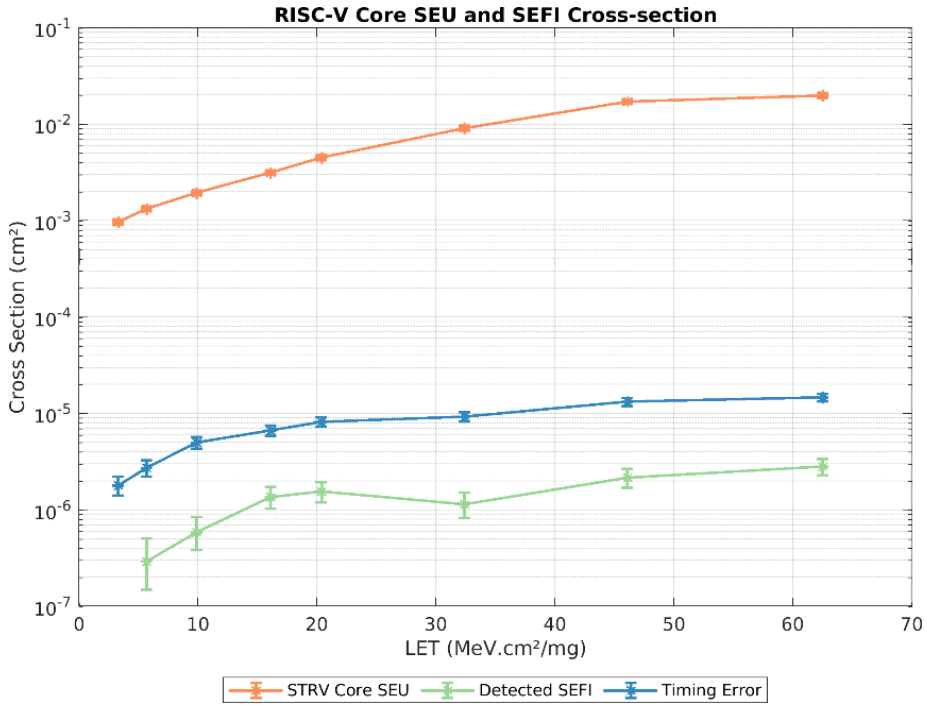
STRV-R1 – Heavy-Ion Irradiation SEFI

- Despite the SEE mitigation techniques SEFIs o
  - SEFIs observed during heavy-ion Irradiation
  - Average improvement over SEU cross-section
    - At low LETs (<16 MeV.cm²/mg): 2800x
    - At high LETs (>32 MeV.cm²/mg): 7700x

- Estimated SEFI rate in HL-HLC environment
  - SEE particle flux 1×10^9 p/cm²/s
  - 2.2 Chip level SEFI per hour

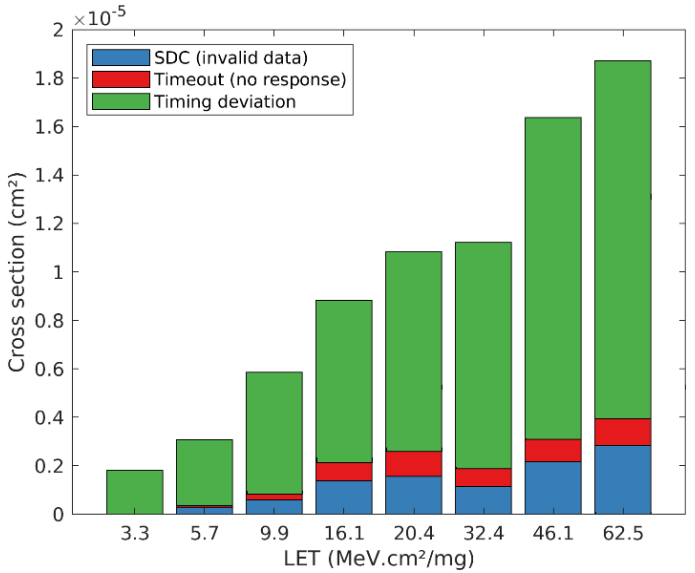
Cross-section	L_0 $[\frac{MeVcm^2}{mg}]$	$\sigma_{HI\infty}$ $[cm^2]$	$L_{0.25}$ $[\frac{MeVcm^2}{mg}]$	$\sigma_{p\infty}$ $[cm^2]$
SEU	< 1.0	4.27×10^{-2}	18.87	2.66×10^{-9}
SEFI	< 5.7	2.95×10^{-6}	10.32	6.15×10^{-13}
Timing	< 3.3	2.86×10^{-5}	19.95	1.58×10^{-12}

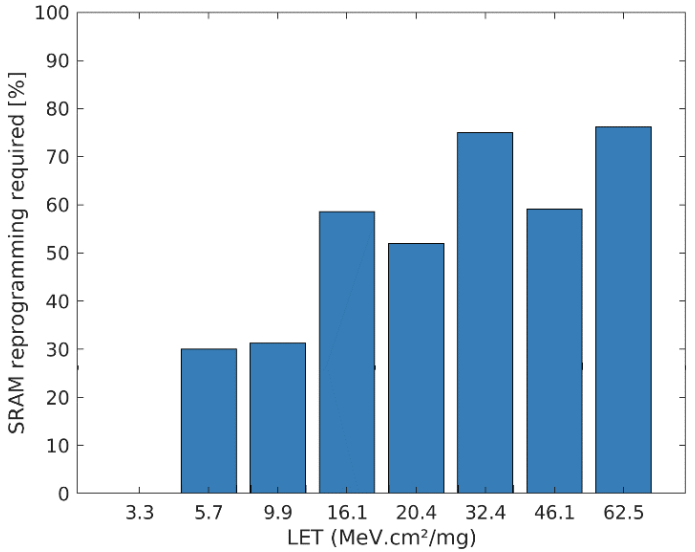
Cross-section	$\sigma_{p\infty}$ $[cm^2]$	[a] Φ $[\frac{Hz}{cm^2}]$	Event rate $[Hz]$	Events / h $[\frac{1}{h}]$
SEU	2.66×10^{-9}	1×10^9	2.66	8.14×10^3
SEFI	6.15×10^{-13}	1×10^9	6.15×10^{-4}	2.21



STRV-R1 – Heavy-Ion Irradiation SEFI

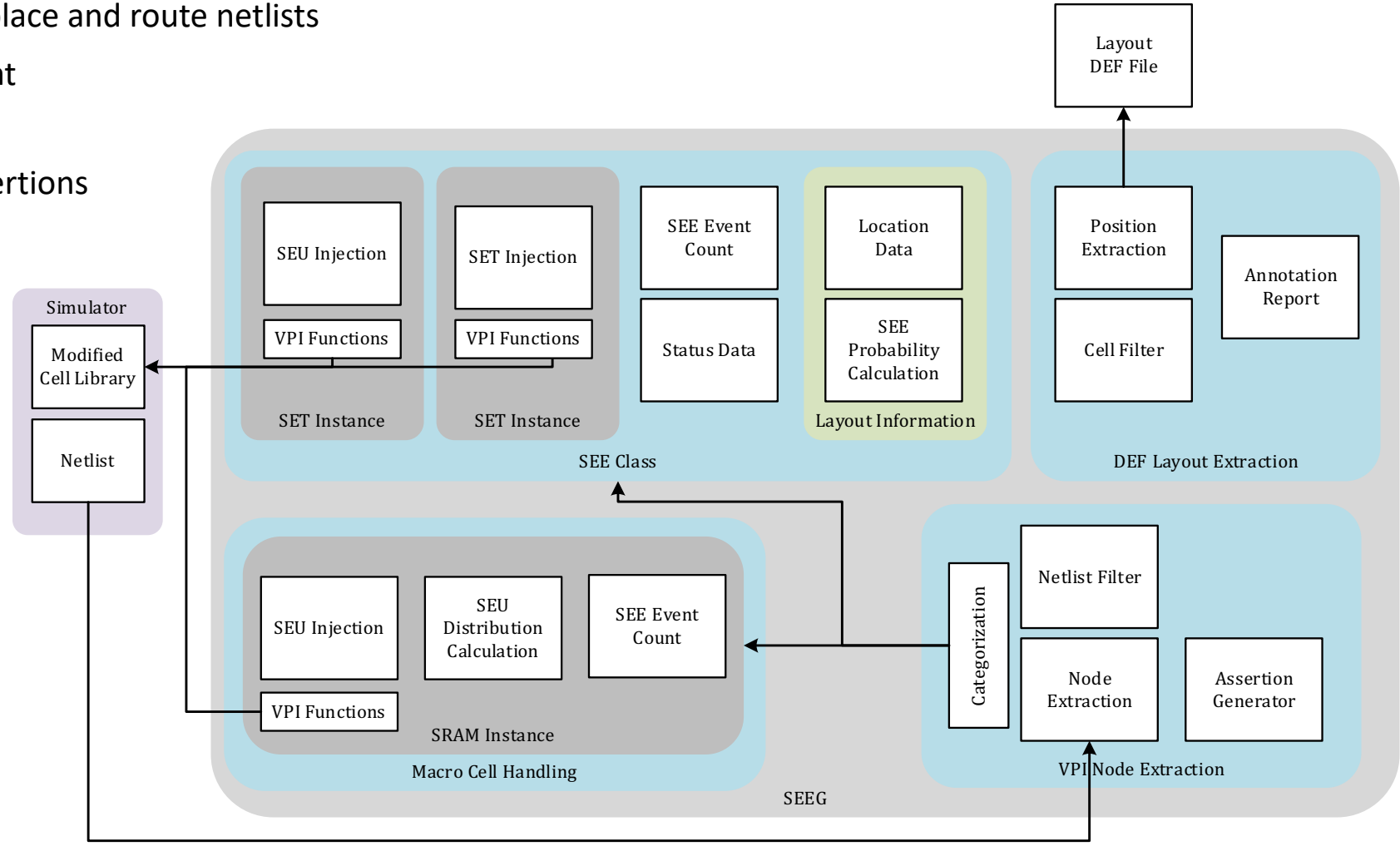
- Observed types of SEFIs during Irradiation:
  - **Silent Data Corruption (SDC)**:
    - Application cycle completes normally
    - Values calculated by DUT deviate from expected values
  - **Timing Deviation**:
    - Application cycle completes normally
    - No indication of an error
    - Calculated data correct
    - At least one clock cycle deviation
  - **Timeout**:
    - DUT does no longer responds to test system
    - Reset required
- SEFIs that cannot be recovered by resetting of the RISC-V core:
  - Data or instructions in the SRAM corrupted
  - Reprogramming of the SRAM required
- Reprogramming rate:
  - For low LET (<16 MeV.cm²/mg): Reprogramming required in 30% of SEFIs
  - For higher LET (>16 MeV.cm²/mg): Reprogramming required for >50% of SEFIs





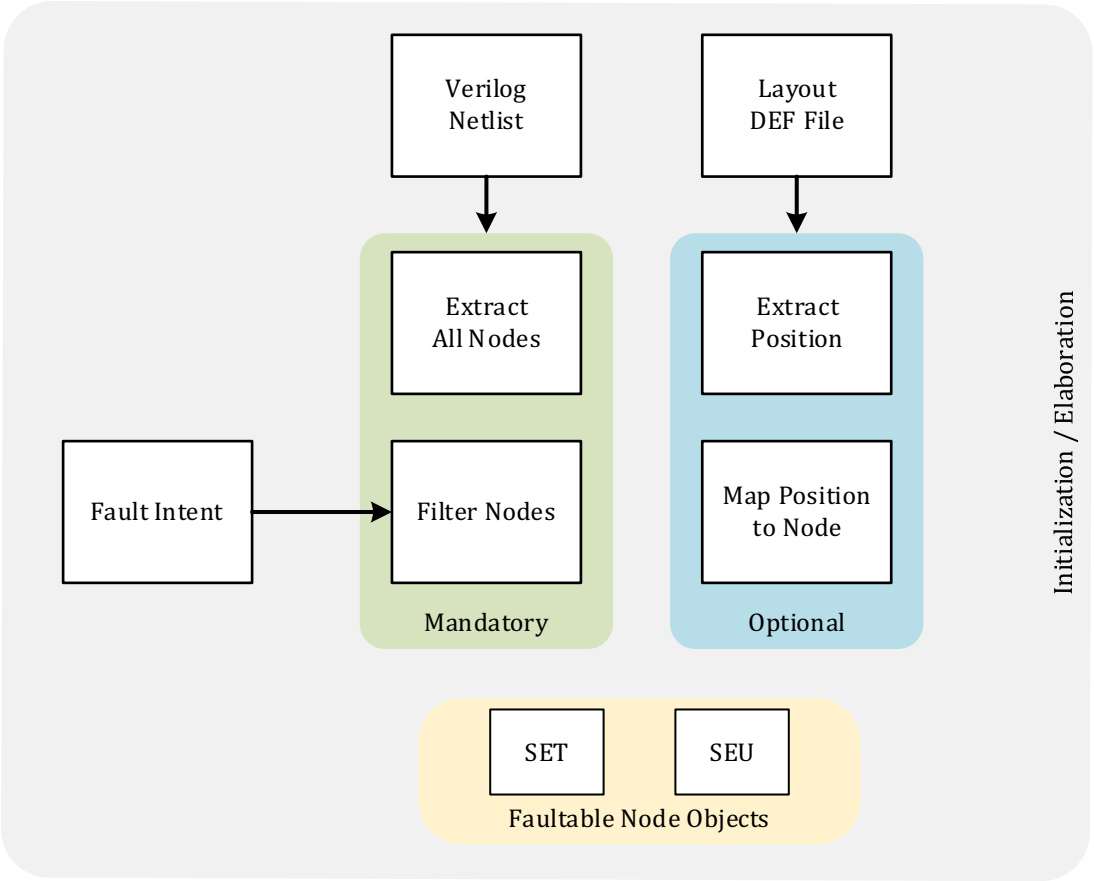
SEE-Injection Simulation Framework

- Designed to replicate real-world impact of SEE
- Intended for simulations with synthesis or place and route netlists
- Ability to incorporate physical cell placement information into the design
- Automatic generation of SystemVerilog assertions
- No design or netlist modification required
  - modification of cell library required
- VPI Functions used to communicate with simulator



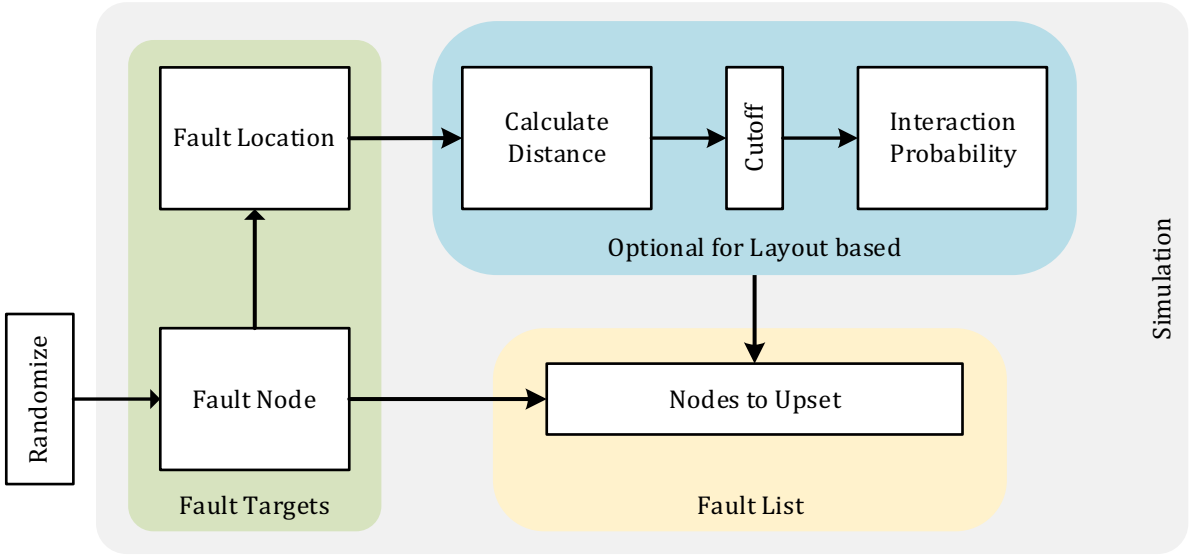
SEE-Injection Signal Selection

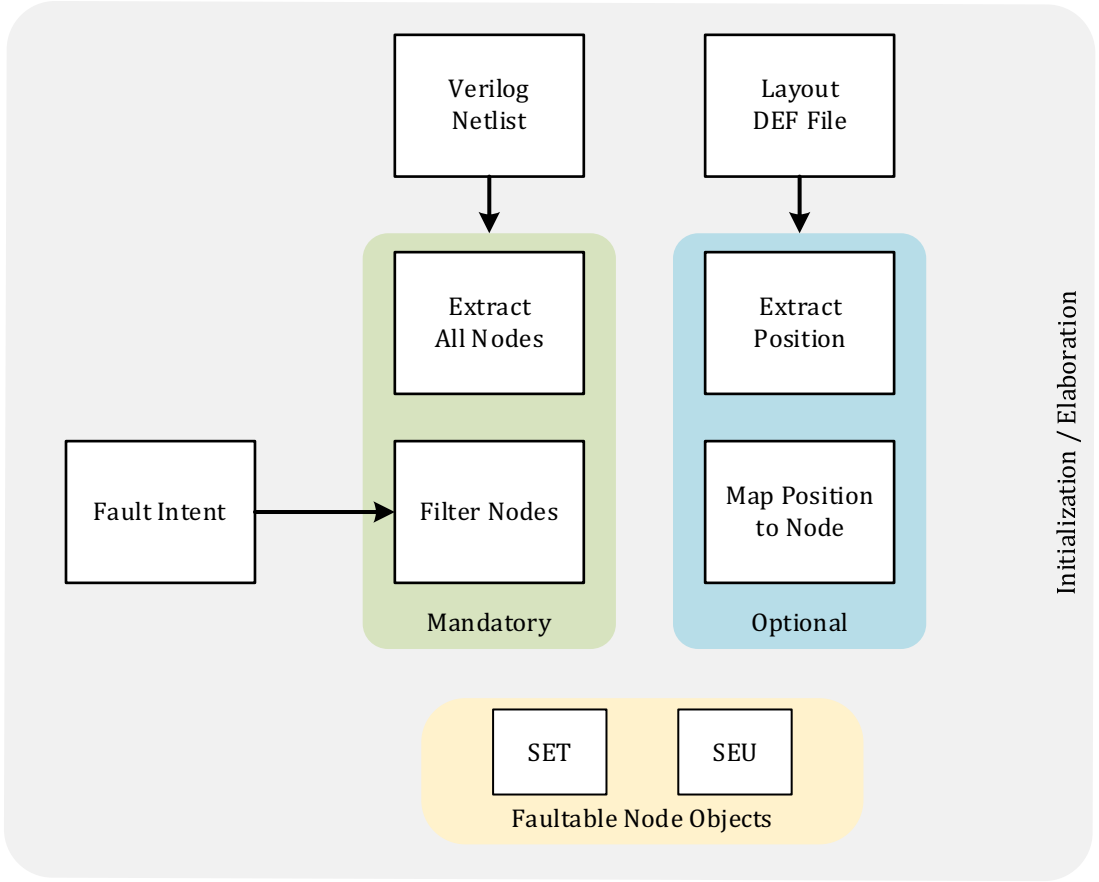
- Randomization
- Reproducibility and random stability
  - Framework uses PRNG with one-time seed provided by simulator
- Fault intent specification
  - Scope to be covered by injection (top level of injection)
  - Type of fault to inject (SET / SEU / Macro specific)
- Filtering options
  - Nodes to be injected on
  - Netlist exclusions (string manipulation)
  - Cell type selection (with DEF mapping)



SEE-Injection Layout Information

- Addition to randomized selection from netlist
- Layout Information from DEF
  - Positions mapped to faultable node objects
  - Distance from faulted node to other nodes calculated
  - Interaction probability determines secondary SEEs
  - Additional nodes upset



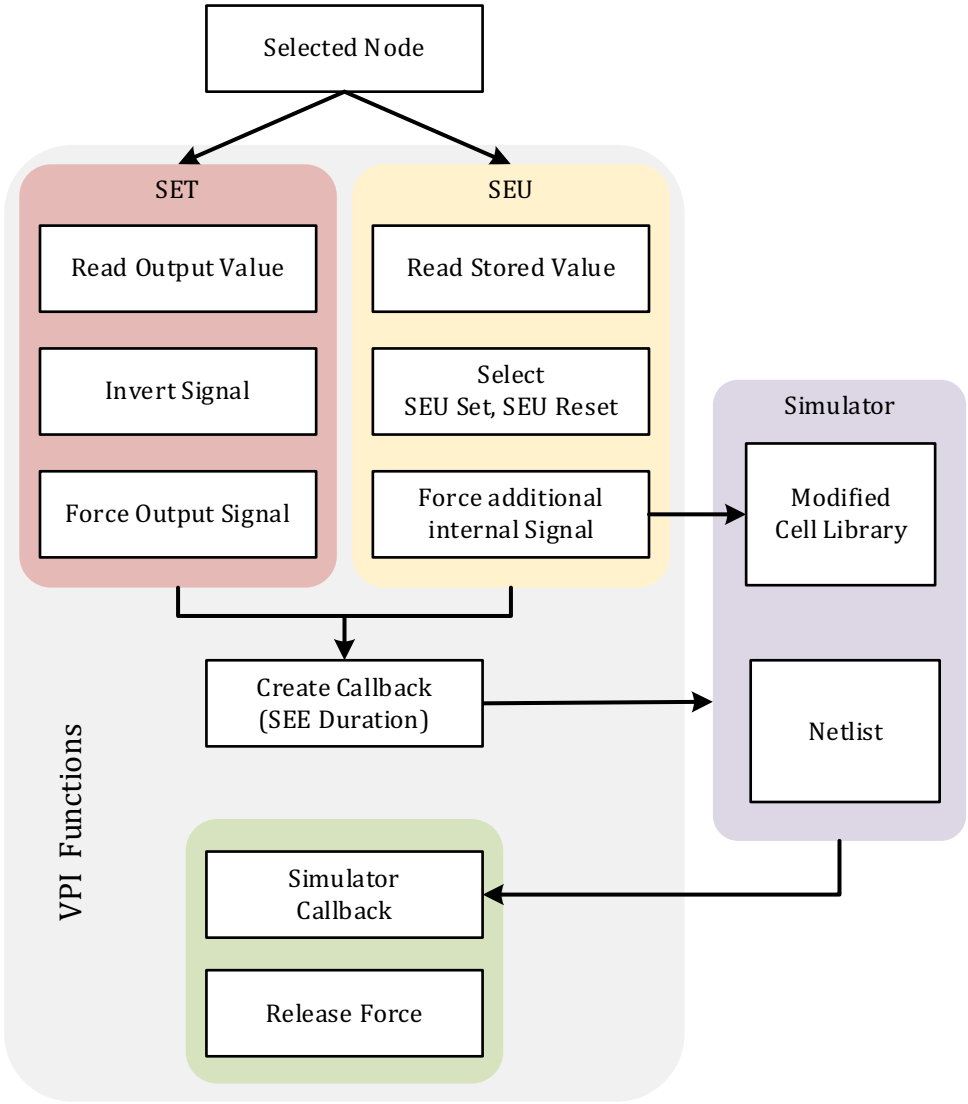


Runtime SEU | SET Modelling

- SET are less meaningful in RTL
  - Synthesis and place & route netlist used
- SEU Injection requires instrumentation of the STD cell library
  - Added internal signal to invert the stored value

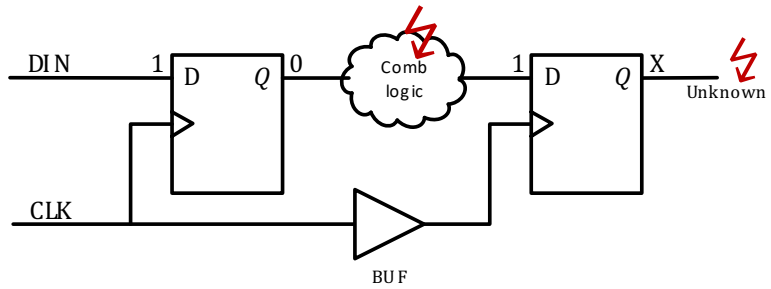
- Select (randomized) node and SEE duration
- Read state of selected node from simulator using VPI functions
- Invert net state using VPI set value function with force flag
- Create a callback for the SEE duration
- Simulator continues for the given amount of time
- Callback from Simulator when time elapsed
- Release the net using VPI function

- SEE duration in SEUs: Time the upset is actively forced
  - Upset is help until next valid sequential activity



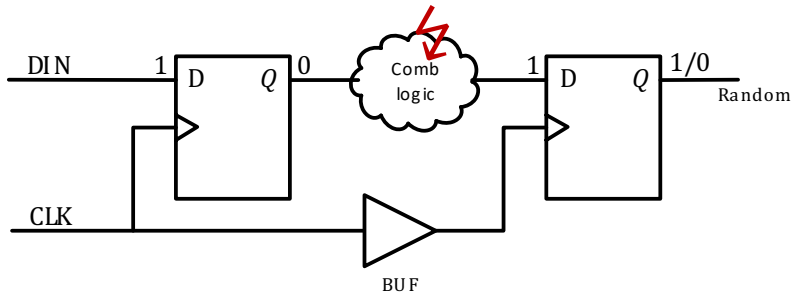
Standard Cell Library Instrumentation

- Timing of SEE independent of clock (randomized)
- SET in the combinational logic or clock-tree
  - → Timing violations possible in sequential logic
  - – Setup, Hold, Width violations
- Typical standard cell models set sequential output to X (unknown)
- Propagation through netlist according to simulator settings

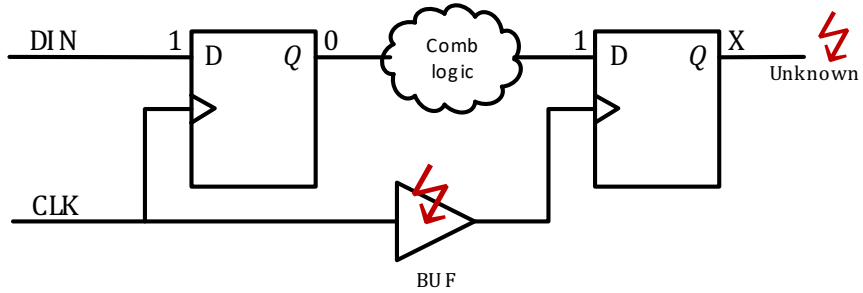


SET in comb. logic (setup / hold violation)

- Modified standard cell library to replicate real-world behavior
  - – Randomized valid output propagated to next cells



SET in comb. logic (setup / hold violation), output randomized

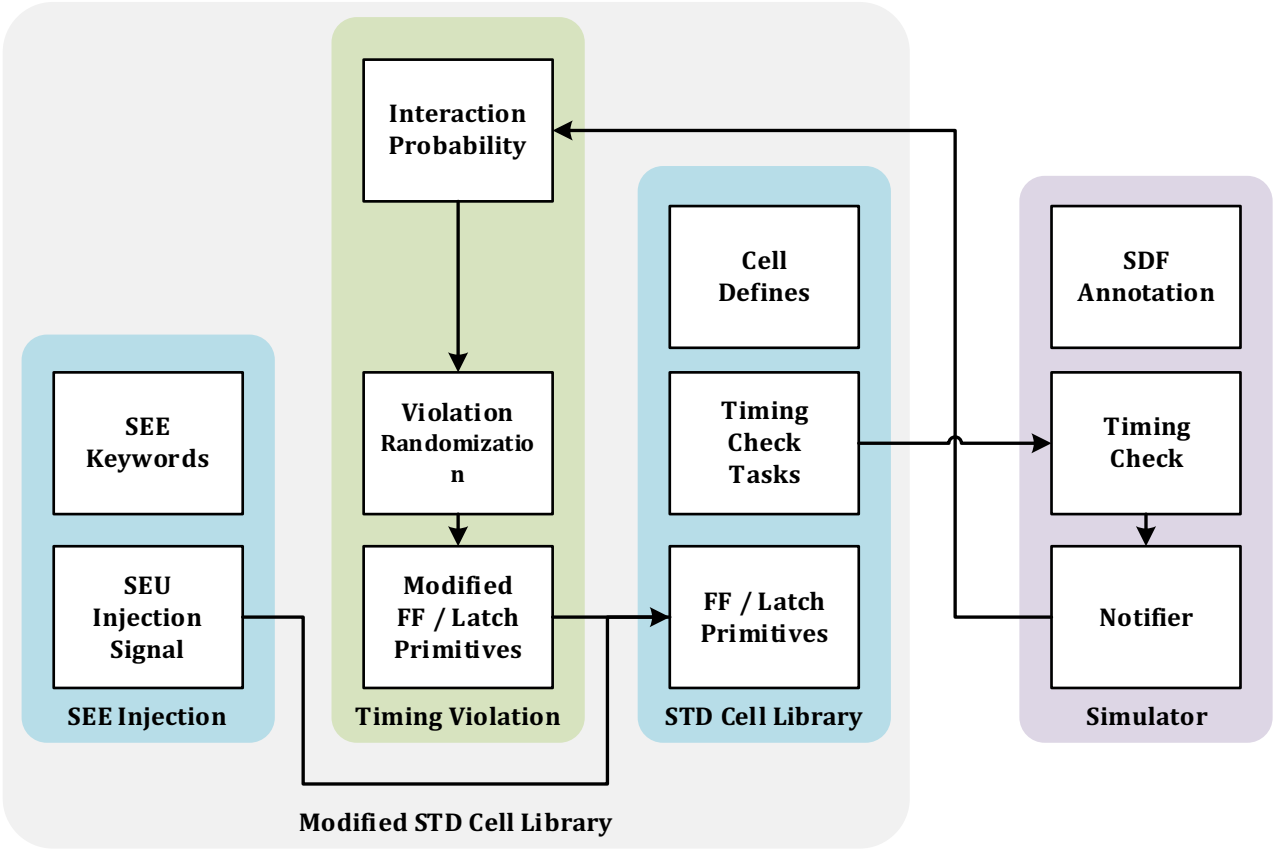


SET in clock-tree (setup / hold / width violation)

Standard Cell Library Instrumentation

- Timing violation propagation instrumentation:
  - Replicate real-world behavior of cell
  - Separate probability calculation for
    - Setup / Hold
    - Width (clock)
  - Randomized output
  - Modified primitives required

- SEE Injection instrumentation:
  - Introduction of a keyword
    - Detected by framework node extraction step
  - SEU: Additional signal to invert the stored value
  - Original STD cell primitives can be reused

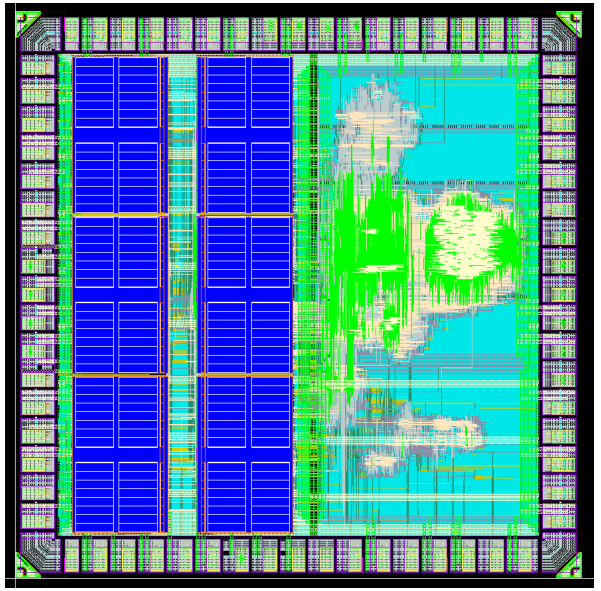


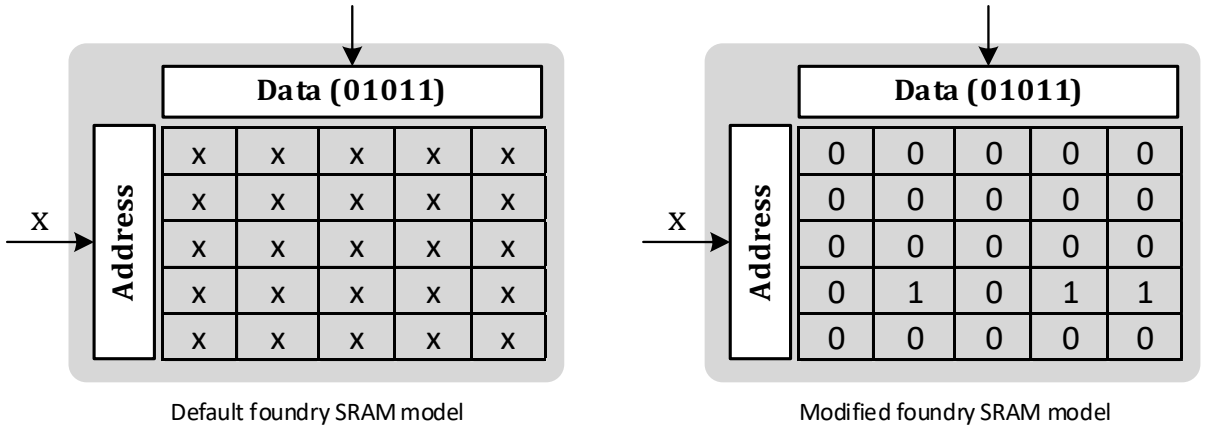
SRAM Macro Cell Instrumentation

- SRAM macros handled differently than standard cells
  - Depending on SRAM cells used, location information not available
  - Interleaving architecture, the bits in a data word are not physically adjacent
  - Multiple-bit upset (MBU) distribution can be used
    - Randomized distribution over multiple bits & multiple words

- Typical foundry HDL SRAM models assume worst case
  - Read operations are generally not critical to the internal state
  - Write operation to unknown address invalidates entire memory

- Foundry SRAM models modified to replicate real-world behavior
- Timing violation handling
  - Control signals: Assume random operation
  - Address: Assume single randomized address
  - Data input: Store randomized word





Default foundry SRAM model

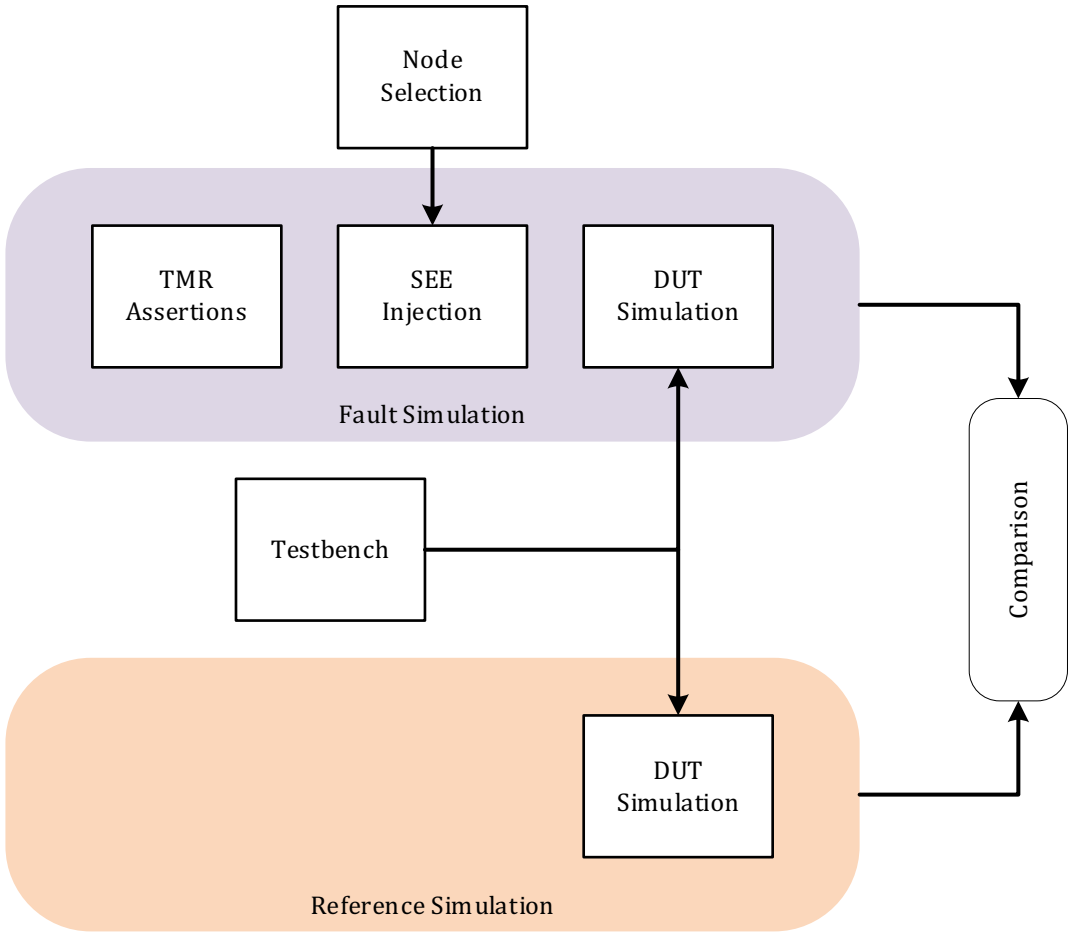
Modified foundry SRAM model

Runtime | Verification | Assertions

- Verify that triplication is implemented correctly
  - Correction of SEUs within one clock cycle for fully triplicated nodes
- TMR assertions for full TMR
  - regA.seu |=> ##1 (regA.Q == regB.Q == regC.Q)
  - regB.seu |=> ##1 (regA.Q == regB.Q == regC.Q)
  - RegC.seu |=> ##1 (regA.Q == regB.Q == regC.Q)
- TMR assertion can be automatically generated by framework

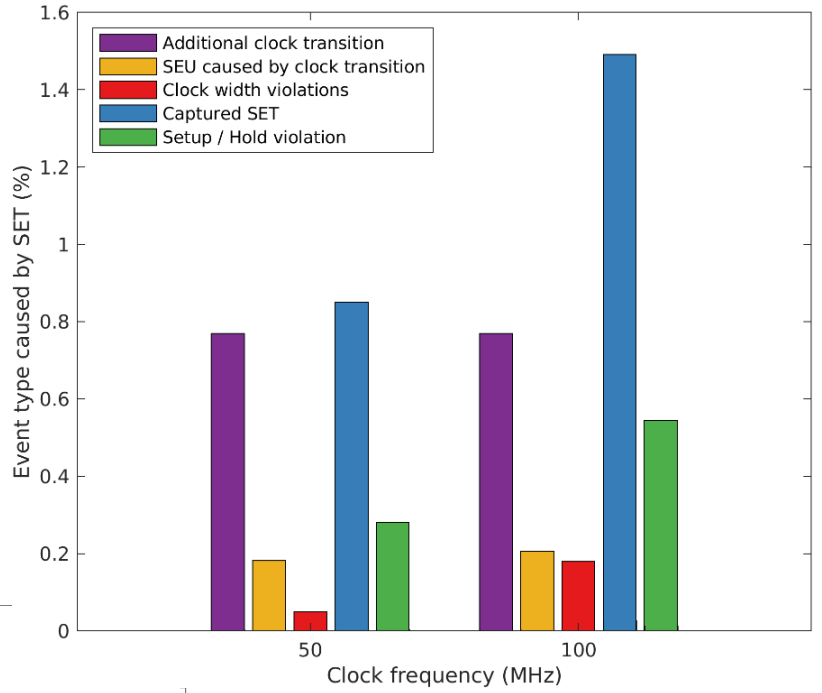
- Fault simulation with reference simulation without fault injections
  - Differences in majority voted data indicate potential SEFI

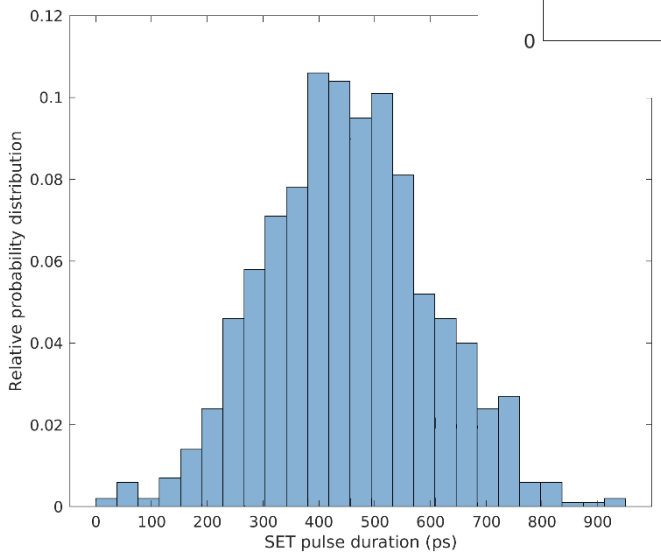
- Limitations of direct comparison with reference simulation
  - Not all differences lead to an error on the CPU (SEFI)
- Checksum of the RISC-V Core register set, status register, etc.
  - Compare state changes between checksums
  - Valid state changes provided by golden reference



STRV-R1 SEU Contributing Sources

- Apart from direct hits, data in sequential elements can be modified by:
- SETs in clock buffers / inverter of the clock tree
  - Depending on the level in the clock tree, large number of leafs affected
  - Additional clock pulses inserted
- Additional clock pulses can be masked by inactive / static data path
  - Static data paths are common in general purpose circuits such as RISC-V cores
- Clock pulse timing width violation in sequential logic
  - Sequential element may not store new state
  - Reduced impact compared to SET in clock signals
- Capture of SET in data path
  - Masked by combinational logic and application-specific state
  - Setup-Hold violations can mask the impact of SETs
- Simulation constraints to simulate additional contributing SEU sources:
  - Dhrystone benchmark executed by RISC-V core
  - SETs evenly distributed over a clock cycle
  - Shown randomized distribution of SET pulse duration used
- Effective SEU rate increased with higher clock frequency
  - Critical for high performance RISC-V ASIC designs



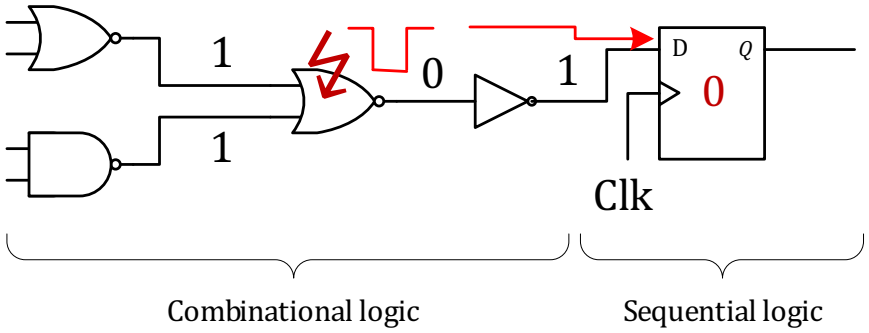


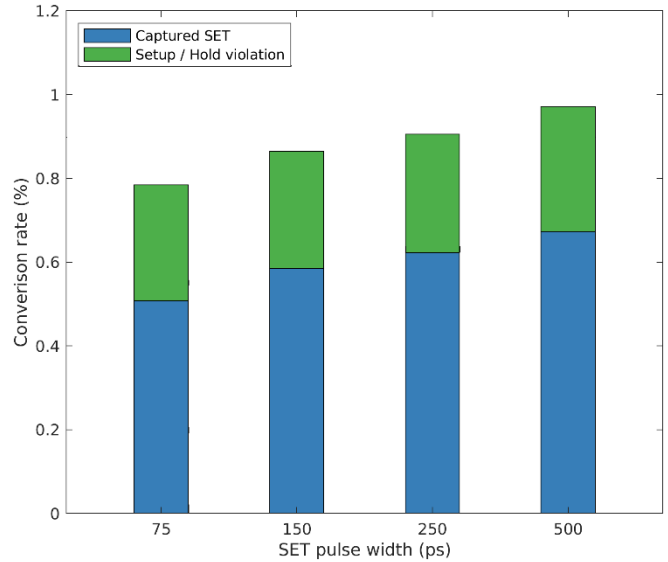
SET Capture in Sequential Logic

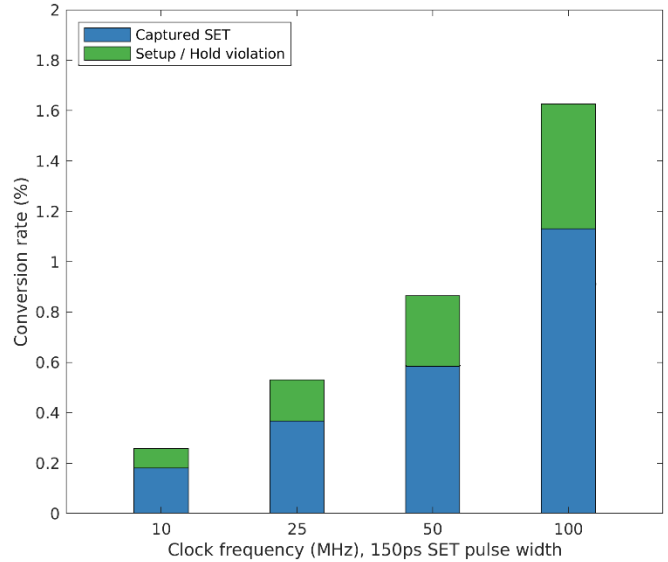
- Single Event Transients captured by endpoint sequential Logic
- Cone of logic as input to sequential Logic
  - Dissipation during propagation through design
  - Elongation during propagation through design
  - Masking via other combinational logic

- Application-specific designs contain a significant number of masked data paths
  - SET capture rate in specific test structure is higher

- Simulation constraints for SETs in data paths:
  - Different application software executed (masked path variation)
  - SETs evenly distributed across clock cycle

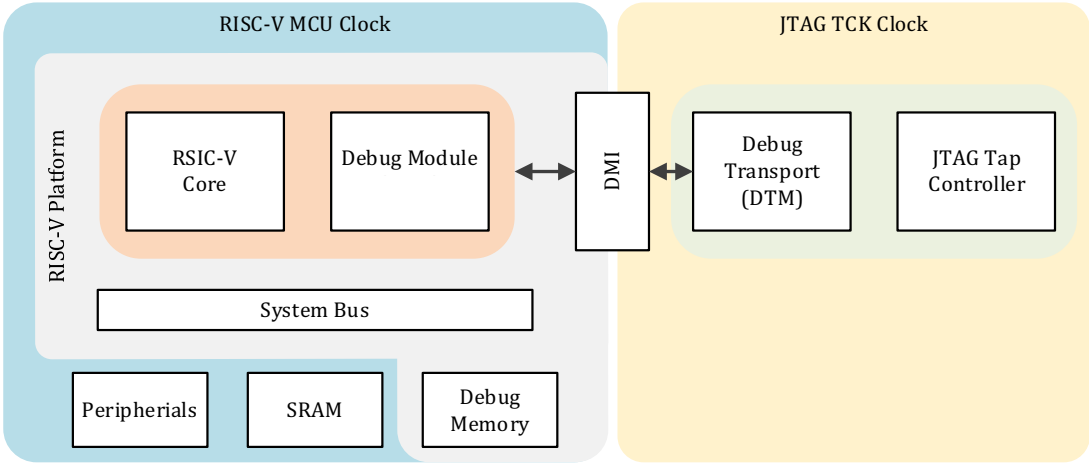






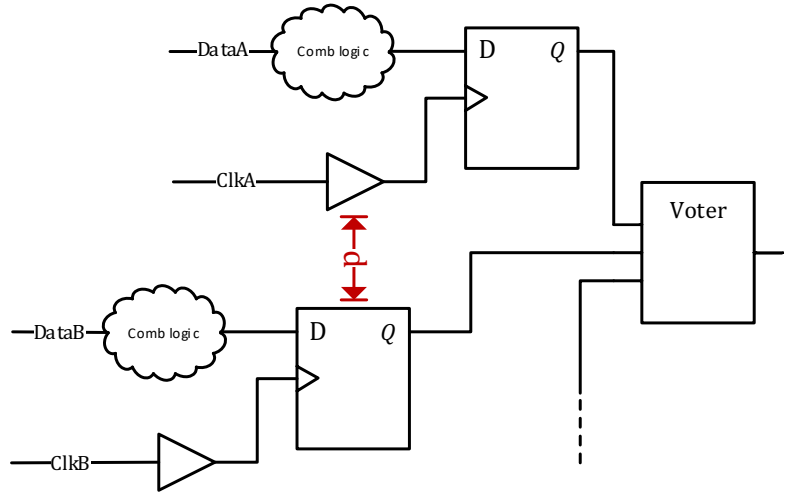
STRV-R1 SEFI Sources

- Clock domain crossing
- RISC-V Example:
  - JTAG Interface debug module
  - Debug module part of core clock domain
  - DTM driven by externel JTAG clock
  - Risk of SEU accumulation in section without active clock



- Dynamic SEE behaviour of SRAM macro cells

- Phyisical constraints:
- Clock-tree spacing (CTS)
  - successive ECO placement and routing steps
- Clock-tree spacing between flip-flops and clock buffer
  - Distance from Clock buffer of TMR group A to Flip-Flop of group B
  - Timing constraints place clock buffer and start / endpoints in the same area
  - Distance to combinational logic has less impact
  - Masked data paths, SET capture rate



Summary | Conclusion

- **Heavy-Ion irradiation results**
  - Effective SEU cross-section is larger than in test-structures for sequential elements
  - TMR protection scheme in RISC-V core achieves up to 8000x improvement
    - SEFI cross-section directly compared to the SEU cross-section
  - Additional soft-error mitigation required to achieve an acceptable residual risk at 1 GHz / cm2 particle flux

- **SEE-Injection simulation framework has been developed**
  - Designed to replicate the real-world impact of SEE
  - Intended for simulations using synthesis or place and route netlists
  - Ability to incorporate physical placement information
    - Simulation of multiple concurrent SEEs