Soft Error Derating, or Architectural Vulnerability

Austin Lesea

October 25, 2011

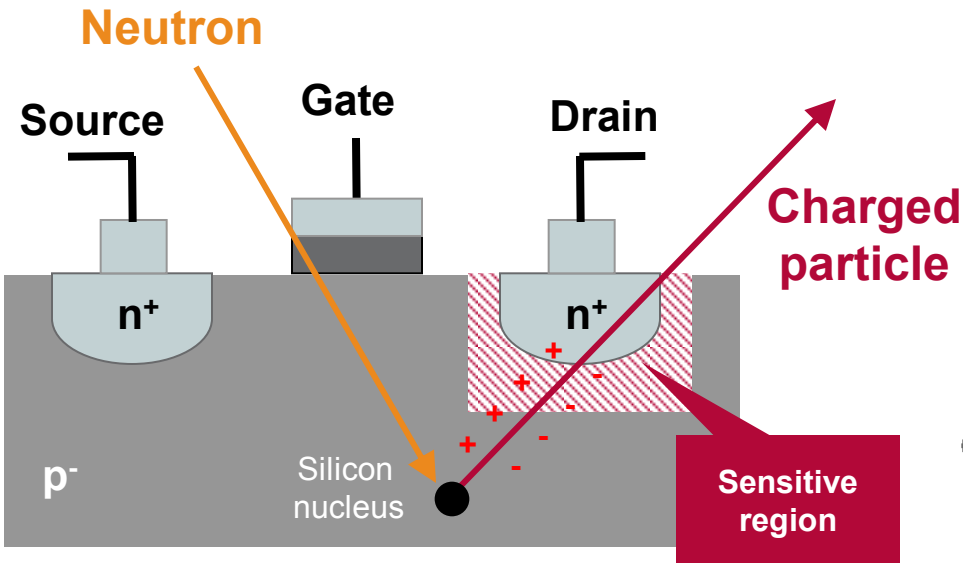
What, Why, How? [All Digital Integrated Circuits]

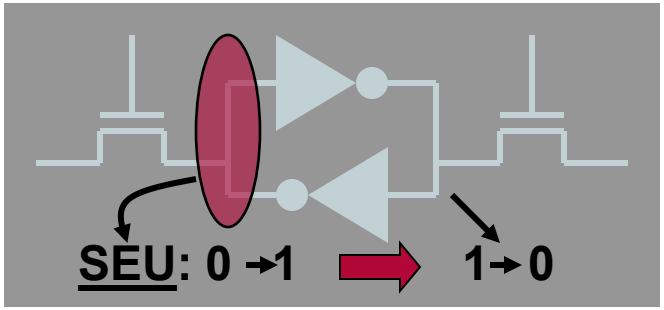
- **Neutrons from heavy ions strike the IC and cause secondary ions**
  - Secondary ions lead to secondary electrons
  - Electrons get collected and cause upsets and transients
  - Soft Errors cause digital integrated circuits to exhibit functional failures
  - Soft Errors „go away“ (if system is restarted, it operates normally, hence the “soft” nature of the failure)

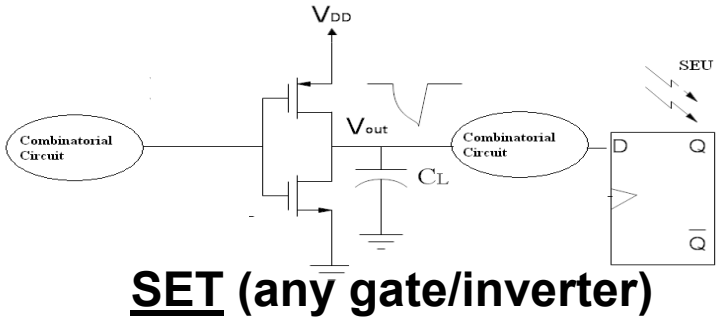
- **Knowing the raw soft error rate, how do you calculate the system failure rate?**
- **How can the system failure rate be estimated early in the design phase (to see if the design requirements are met)**

Soft Error Effect(s)

- **SEE = soft error effect**
  - SEU = soft error upset, a bit flips 0->1, 1->0
  - SET = single event transient, 1->0, 0->1 for a few hundred ps
  - MBU = multiple bit upset, as devices get smaller, more bits get hit
  - Latch-up = destruction due to short circuit from charged particle







SET (any gate/inverter)

XILINX

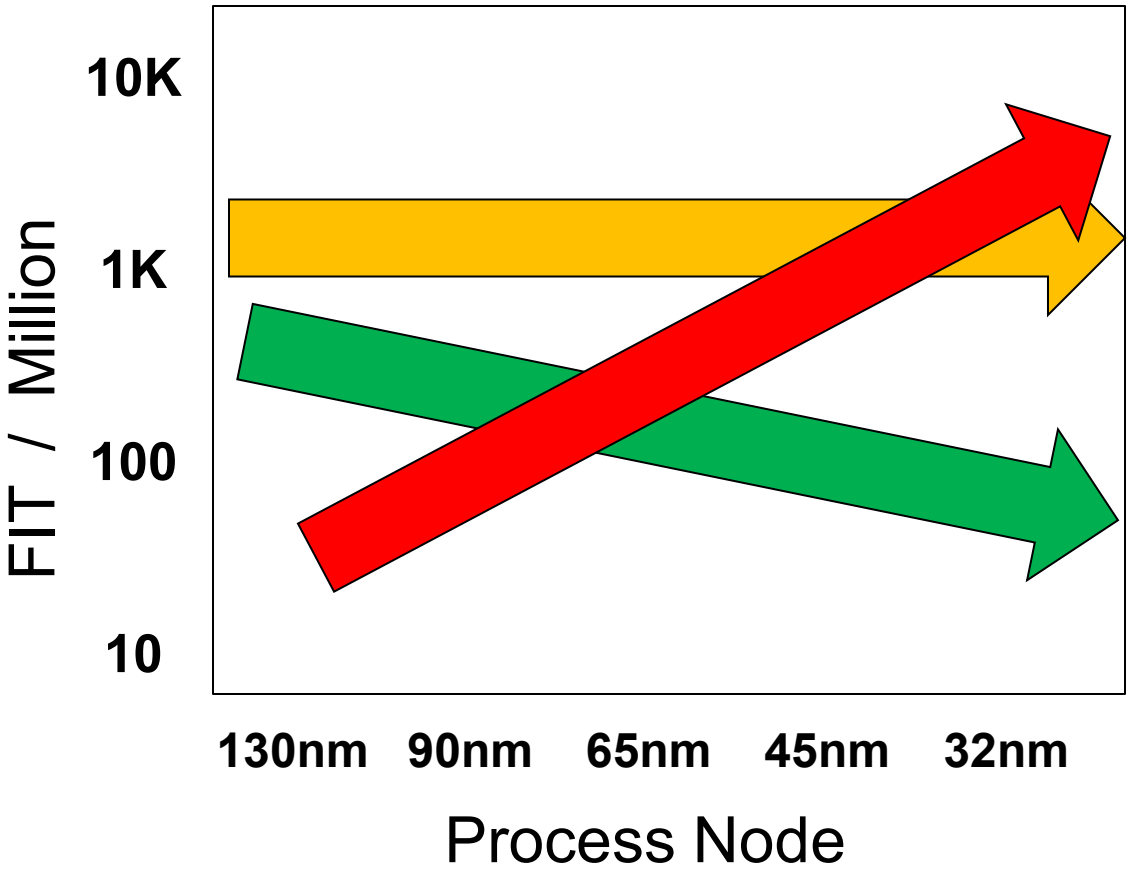
ug116: Raw Data (updated quarterly – 10/2011

Table 1-16: **Real Time Soft Error Rates**

Technology Node	Product Family	Neutron Cross-section per Bit(1)			FIT/Mb (Alpha Particle)(2)			FIT/Mb (Real Time Soft Error Rate)(3)		
		Configuration Memory	Block Ram	Error	Configuration Memory	Block RAM	Error(4)	Configuration Memory	Block RAM	Error(4)
250 nm	Virtex	9.90×10^{-15}	9.90×10^{-15}	±10%				160	160	±20%
180 nm	Virtex-E	1.12×10^{-14}	1.12×10^{-14}	±10%				181	181	±20%
150 nm	Virtex-II	2.56×10^{-14}	2.64×10^{-14}	±10%				405	478	±8%
130 nm	Virtex-II Pro	2.74×10^{-14}	3.91×10^{-14}	±10%				437	770	±8%
90 nm	Virtex-4	1.55×10^{-14}	2.74×10^{-14}	±10%				263	484	±11%
65 nm	Virtex-5	6.70×10^{-15}	3.96×10^{-14}	±10%				164	697	–13% +15%
40 nm	Virtex-6	1.26×10^{-14}	1.14×10^{-14}	±10%	40	100	–50% +100%	94	245	–26% +37%

1 FIT = 1 failure per billion hours
1000 FIT ~ 100 years MTBF
Device (raw) FIT = number of Mb X FIT/Mb

FIT Rate Swap: ASIC/ASSP vs. FPGAs



ASIC/ASSP SET (Gates/DFF)

SRAM SEU

Xilinx FPGA (CRAM) -no SET

Xilinx FPGA No latch-up

XILINX

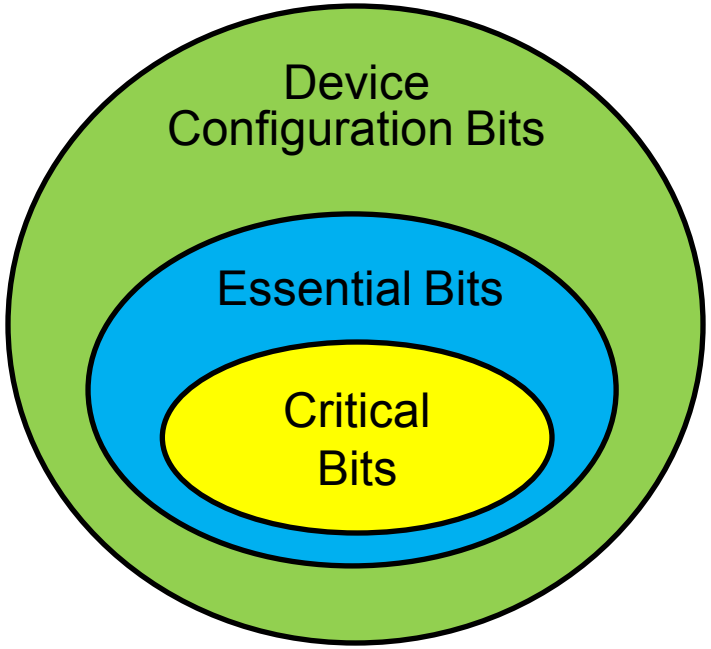
Reliability vs. Availability

- Mean Time to Repair after a failure determines availability
- If reliability is very poor, but repair is within milliseconds system may still meet its goals
- If reliability is excellent, but repair requires a technician on site, system may never achieve its goals
- You must know both reliability and availability

- ***Reliability is a good thing: But Availability is what the customer experiences!***

Soft Errors on FPGA Devices

- SEUs on SRAM-based FPGAs may impact functionality of programmed circuit [configuration bits]
- FPGA designs do not utilize all memory cells [used = essential bits]
- Not all upsets of used memory cells become functional failures
- Estimation flow uses design specific information to estimate susceptibility to SEUs [critical bits]
  - *Only the customer knows which actual bits are critical!*



Derating Factor

- **Percentage of upsets that cause functional failure**
  - Never more than 10% (1 in 10) for Xilinx FPGA devices in 12 years of papers and testing (BYU, UCLA, Vanderbilt, JPL, NASA….)
- **Bits may not be critical in „time," or not critical is „space"**
  - „Time" means they are not used „right now" or not being „looked at"
  - „Space" means they are not used (unreachable state, unused logic)

- ***Critical* is defined by the customer (test bench, beam test)**
- ***Essential* is defined by Xilinx schematics through bitgen**
- **Essential ~ 3-5X Critical (typical)**
- **Present in ASIC/ASSP (rarely known, or tested)**
- **Called "architectural vulnerability factor" by Intel for their uP's**
- **Architecture Vulnerability Factor == AVF**

Example: Essential, not critical, bits for FPGA

- **LUT INPUT 6 is a test enable, only used in manufacture test**
  - 32 bits are not critical in LUT
- **PULLUP on unused pin**
- **MSB of stack pointer, code never uses stack that deep**
- **Unused state in state machine**
- **LSB in arithmetic, faults may go unnoticed (noise)**

- **Quality level of Essential Bits : a bit that is critical, and marked non-essential shall occur ideally never, but until we can (figure out how to) prove it, at less than 100 FIT (once per 1,142 years)**

Why is AVF Needed?

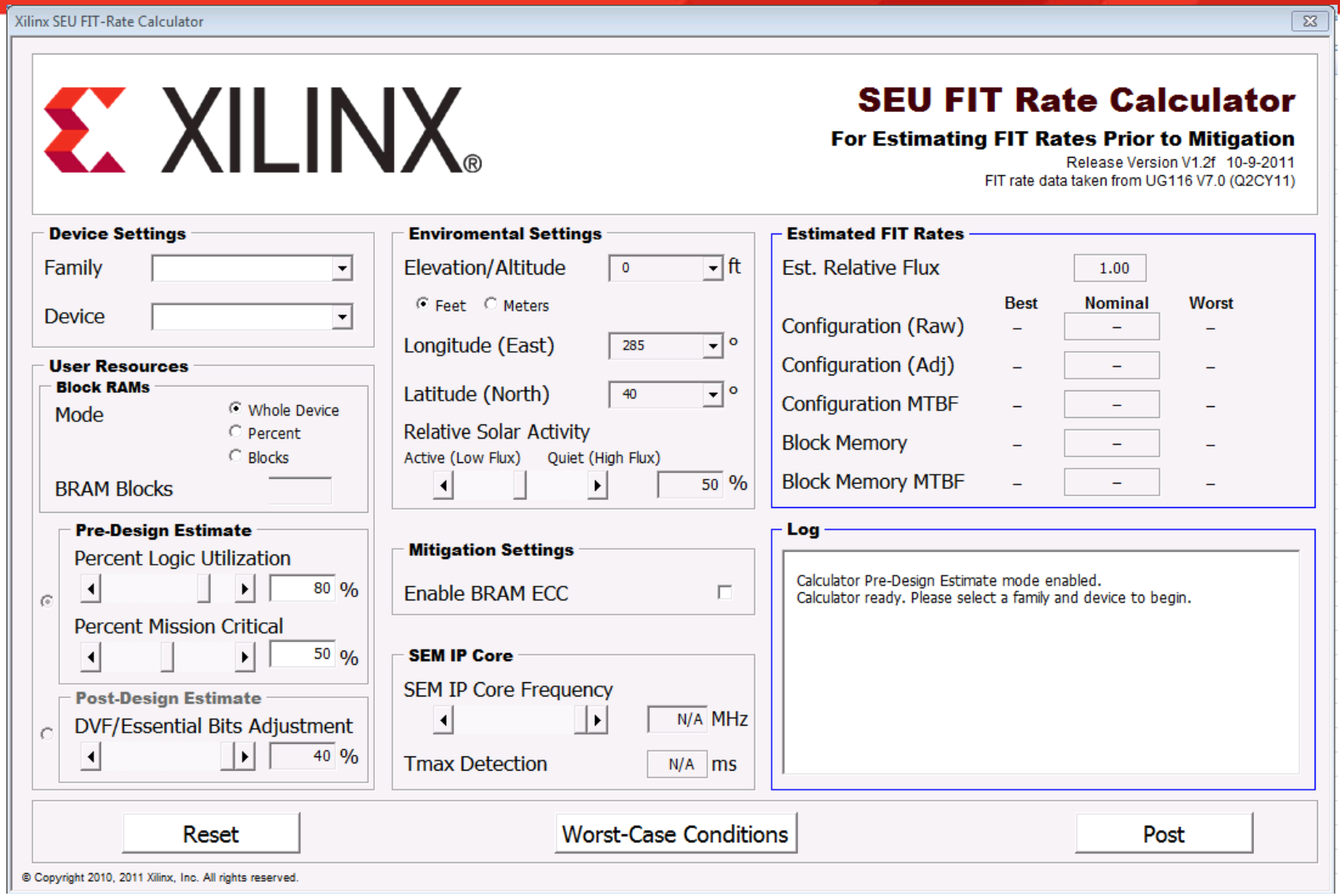
MTBF = AVF * MTBU

- **Calculate mean time between failures (MTBF), and estimate availability**
- **Calculate failure in time (FIT) rate**
  - Input into SEU FIT Rate Spreadsheet
- **Mitigation: the earlier, the better**
  - Identify RTL code early that may not meet requirements
  - Suggest improvements and enable *what if* scenarios
    - Recode, duplicate, triplicate, parity, ECC, etc.

Vulnerability: Time & Space

- **Observations**
  - If a node toggles (a lot), it is (probably) important
  - Conversely, if a node does not toggle, it is not important
- **Random error injection**
  - Bits causing failures are (likely) LUT contents
  - Bits controlling interconnect cause failures less (if the bits get corrected)
  - Uncorrected interconnect control bits may cause failures (eventually)
  - Only interconnect which must be valid on every clock cycle breaks a design (e.g. cryptmon)
- **Finding and fixing upsets improves reliability by up to 30%**
  - MTTF increases by up to 1.3X on designs where not every wire is critical on every clock cycle

SEU Calculation Spreadsheet



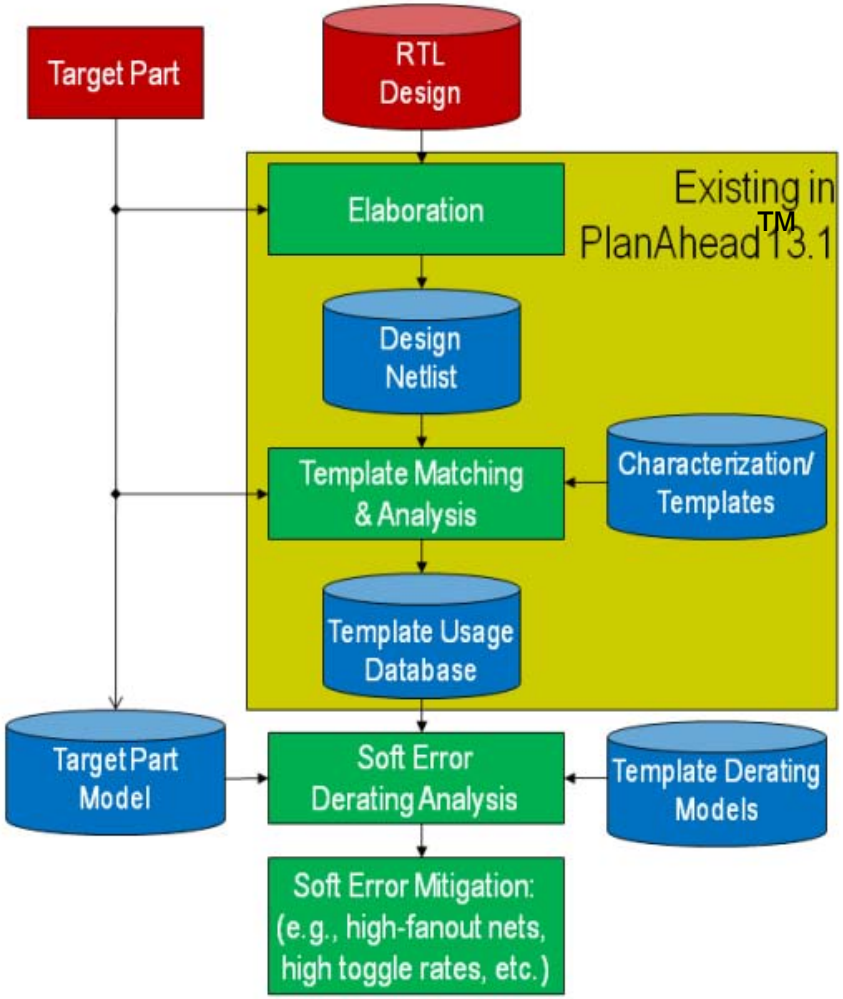
XILINX

Comparison with Prior Work

Intel Microprocessor*	Description	Xilinx FPGA
Identify total bits in system	What do we start with?	Identify part to find total resources and config bits
Define term: architecturally correct execution (ACE) bits	What's important? (upsets → failures)	Define term: critical bits
Analyze program instructions for ACE bits	What do we use?	Analyze logical hierarchy for HW resources (e.g., N_{LUT}) and estimate routing (N_{wire})
Analyze bandwidth (B_{ace}) and latency (L_{ace}) of ACE bits	How do we use it?	Analyze toggle rates (e.g., TR_{LUT}) of HW resources
$AVF = \frac{B_{ace} \times L_{ace}}{\text{total bits}}$	The Math	$AVF\,\alpha\,\frac{N_{wire}}{\text{total wires}} + \frac{N_{LUT} \times TR_{LUT}}{\text{total LUTs}}$

* S. Mukherjee, et. al., "A Systematic Methodology to Compute the Architectural Vulnerability Factors for a High Performance Microprocessor", *MICRO*, 2003.

Block Diagram: How it FIT's (p.i.)



XILINX

RTL Proof-of-Concept

- **New method derived from RTL resource estimator**
  - Space: Percent of resources used on target part
  - Time: Estimated toggle rates of all structures and *wires*

XILINX

Fitting into the Xilinx Roadmap

Xilinx SEU Mitigation Capabilities and Roadmap

- 4 levels of protection for different customer needs:

- 1. Silicon and circuit improvements to reduce SEU FIT rates
  - Solutions: Process improvements and improved cell design, Hardened CRAM
- 2. Rapidly detect and correct errors to minimize impact
  - Solutions: Error correction for CRAMs, average ~25-30ms for most devices
- 3. Improved error classification to improve availability
  - Solutions: Essential Bits, AVF
- 4. Prevent SEU from impacting operation
  - Solutions: Triple Module Redundancy (TMR Tool)

Mentor Precision Hi-Rel Synthesis Tool

AVF Reporting Status

- **Hidden in ISE® 13.2 (i.e., report_power –seu seu_report.txt)**
- **Code exercised and report generated *only* if option enabled**
- **Supports:**
  - Spartan®-*6;* Virtex®-*6;* Virtex®-*5;* Kintex™-7; Virtex®-*7 (today)*
  - Artix™-7 (soon)
- **Contact your Xilinx or Distribution FAE for support**

- **Visible in ISE® 13.3**

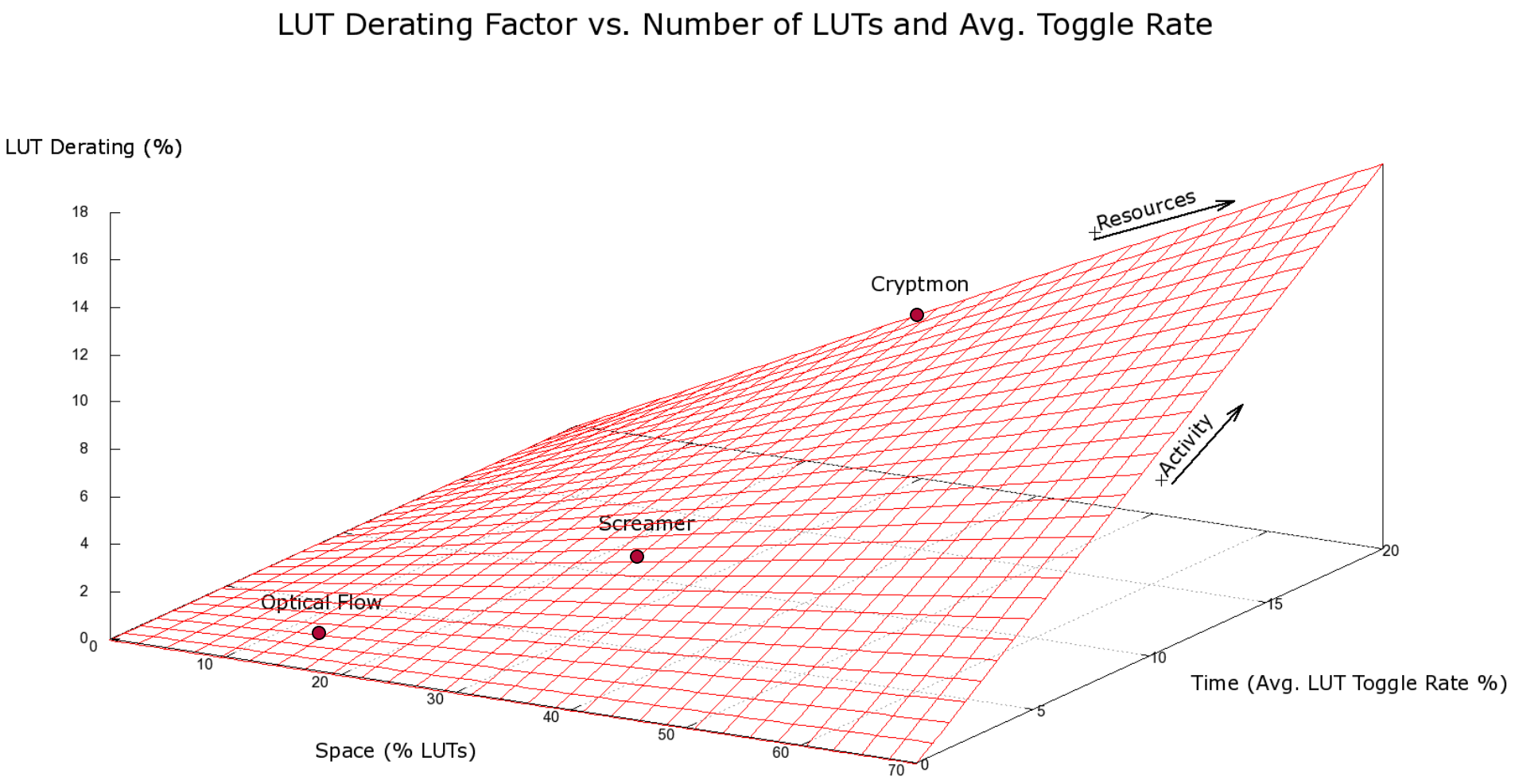
Essential Bits (Bitgen)

- New Bitgen feature: fully released & supported
- Provides file of bits that are absolutely used by the customer design
- Don‟tknow if the design breaks when they flip, however
- Limited correlation with critical bits
  - ~30% or ~3X (worst case) the derating factor
- If the design “must not take bad action” action may be gated by “no essential bits have flipped”

SEU Monitor IP Core

http://www.xilinx.com/products/intellectual-property/SEM.htm

LUT Derating vs. Space & Time



XILINX

Sample Report Summary (summary @end)

Soft Error De-rating Summary
==========

Target part: xc5vlx110ff1760-1
Total Estimated Resources: LUTs: 22772, Flops: 564, DSPs: 0
Total Available Resources: LUTs: 69120, Flops: 69120, DSPs: 64
Overall Average Toggle Rate: 18.71%
LUT Average Toggle Rate: 18.65%
De-rating factor from interconnect usage: 1.76%
De-rating factor from LUT usage: 7.13%
This design has an estimated de-rating factor of 8.90%

Some (verified) AVF

- Fully unrolled triple DES: 10% (cryptmon)
- 400 24-bit counters @ 200 MHz: 5% (screamer)
- MicroBlaze Object Avoidance: 2.5% (optical flow)
- Customer Line Card: x.x% (under NDA)

- More than XXX designs run through tools, results checked for (tool) errors (regression suite)

Verified by customers

- To date, only a few customers have provided confirming data for their designs (requires they test their systems)
- Beam testing results (or bench tests) rely on the test bench to catch errors
- Difficult to catch errors (Quality of Test Bench)

- Historically, beam testing of functional systems is rare, difficult, and expensive
- Those test results we have seen (under NDA) are in line with these predictions
- Bench Testing with SEU Monitor IP is less costly
  - In use by a number of customers today!

Method to Improve Testing, Verify Results

- Duplicate the design
- Use random Error Injection feature of SEM IP on one copy of design
- Compare all outputs
- Wait perhaps as much a 5 seconds after each error
- If no difference, repair bit, choose next random bit
- If an error, collect statistics, reprogram, inject next error

- Tends to over-estimate faults (all differences not customer visible)
- Factor of 1.3 not present (errors are uncorrected ASAP)

Summary

- **To find mean time between failures from soft errors:**
  - Need device raw FIT/Mb rates (ug116.pdf @xilinx.com)
  - Need the Device Vulnerability Factor (estimated by PlanAhead™)
  - Use of SEU Calculation Spreadsheet
  - May be verified by use of SEM IP core (random error injection)
  - Or, verified by beam tests

- **AVF estimated by counting resources, finding toggle rates**
  - Formula used based on empirical data gathered from years of testing
  - Accuracy of results presently better than +/- 20% of actual AVF found by beam testing (more verification work is needed)