Calculating FPGA System Error Rates and Comparing the Robustness of Two Upset Mitigation Techniques

RHBD vs. TMR

Gary M. Swift

Xilinx, Inc.

Larry D. Edmonds

Jet Propulsion Laboratory
California Institute of Technology

The work described in this presentation was conducted at the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

Overview

Introduction – Reducing errors via:

TMR = Triple Modular Redundancy (in spite of upsets)

RHBD = Rad-Hard by Design (by reducing upsets)

Calculating Upset and Error Rates

1. Ordinary (single-node) - SEFI example
2. TMR case - application example
3. Dual-node case - data-based example

Concluding Remarks

Three Options

1. Do nothing - live with intrinsic upset rate

   For rFPGA's not all config upsets yield errors

   However, nth error 'breaks' design ($n \approx 10$)

2. Upset mitigation - upsets ≠ errors

   Prevent a single upset from causing an error

   Prevent upset accumulation

3. Harden to upset - no upset = no error

Upset Mitigation

Redundancy -

- Extra information (bits) prevents all upsets from yielding system errors.

Scrubbing required –

- Accumulation of errors rapidly kills mitigation effectiveness.

Effective –

- Most spacecraft now fly large arrays of upset-soft memories with few or no errors.
- Typically, uncorrectable errors are detectable.

Upset Hardening – Two Basic Approaches

Both Approaches -

- Add circuit elements to basic storage cell
- Increase storage cell stability

Approach 1 - Increase “critical charge” to upset

- Add passive element(s) into cell feedback path.
- Cell size increase may be small, but it’s slower
- Standard upset rate calculation does work

Approach 2 - Require charge in two nodes

- Add geometrically separated active elements.
- Standard upset rate calculation doesn’t work

Three Options

1. Do nothing - live with intrinsic upset rate
   - For rFPGA's not all config upsets yield errors
   - However, nth error 'breaks' design ($n \approx 10$)

2. Upset mitigation - upsets ≠ errors
   - Prevent a single upset from causing an error
   - Prevent upset accumulation

3. Harden to upset - no upset = no error

Space Upset Rate Calculation

Involves three basic elements:

1. Upset susceptibility measurements
   - cross section vs. “effective” LET
2. Environment specification
   - integral flux vs. LET
3. Angular response model
   - RPP[†] chord length distribution
   - one adjustable parameter:
     - charge collection depth
     - (aka funnel length)

[†] RPP = rectangular parallel piped, i.e. a 3-D box

Simplifying concepts (or useful fictions)

Critical charge:

If a node collects more charge than the critical amount, then the cell upsets.

Effective LET:

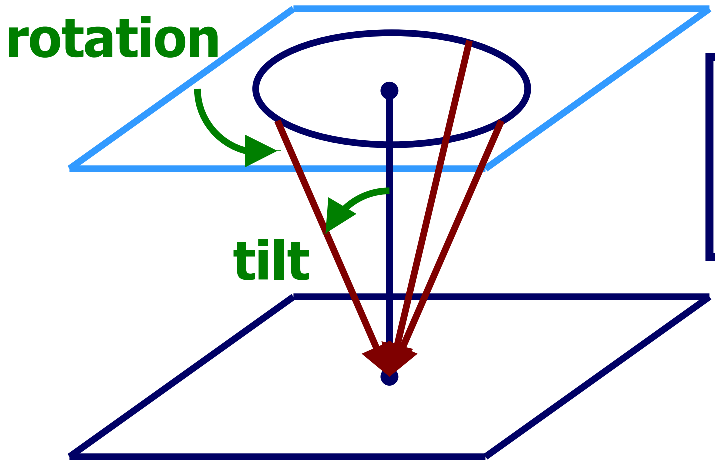
An ion's "effective" energy (or charge) deposition is related to the cosine of the tilt angle (off normal incidence) that it strikes.

RPP charge collection volume:

All charge deposited in RPP goes to node, while all charge outside does not.

Inherently, this is a "single node" calculation

Although a cell may contain multiple charge collection nodes capable of upsetting the cell, the charge collected is only dependent on the "tilt" angle and not the rotational orientation:



Several ion trajectories all with same tilt angle, but various rotation angles.

Results for XQR2V6000 in GEO

Configuration upsets:

Less than five per day

SEFIs:

About one per century

Three Options

1. Do nothing - live with intrinsic upset rate

    For rFPGA's not all config upsets yield errors

    However, nth error 'breaks' design (n≈10)

2. Upset mitigation   -   upsets ≠ errors

    Prevent a single upset from causing an error

    Prevent upset accumulation

3. Harden to upset - no upset = no error

Limits of Upset Mitigation

Common sense says -

At some point, upsets will occur too rapidly and the mitigation will be "overwhelmed."

In fact, Edmonds approx. equation says –

There's not really a "cliff."

The relationships are known; the error rate:

(1) increases with the square of upset rate

(2) decreases linearly with faster scrub rates

(3) is directly proportional to EDAC word size†

† EDAC word size = data bits + check bits ; EDAC=error detection and correction

Edmonds TMR Equation

Approximation when r (upset rate) is small:

$$R \approx 3M\,T_C\,(\mathcal{M}_2\,r)^2$$

System Error Rate

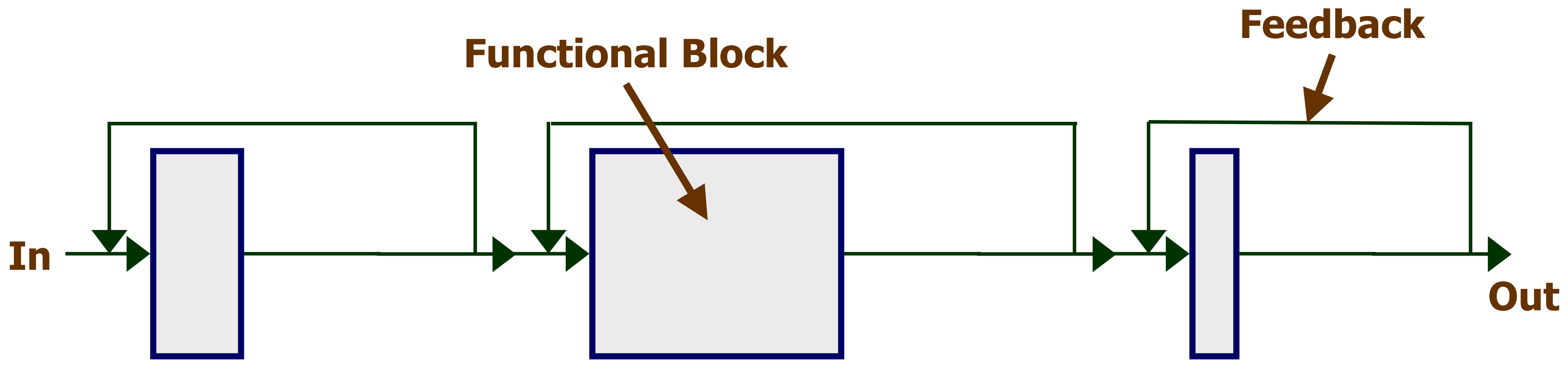
Total Modules

Scrub Time

Underlying Upset Rate

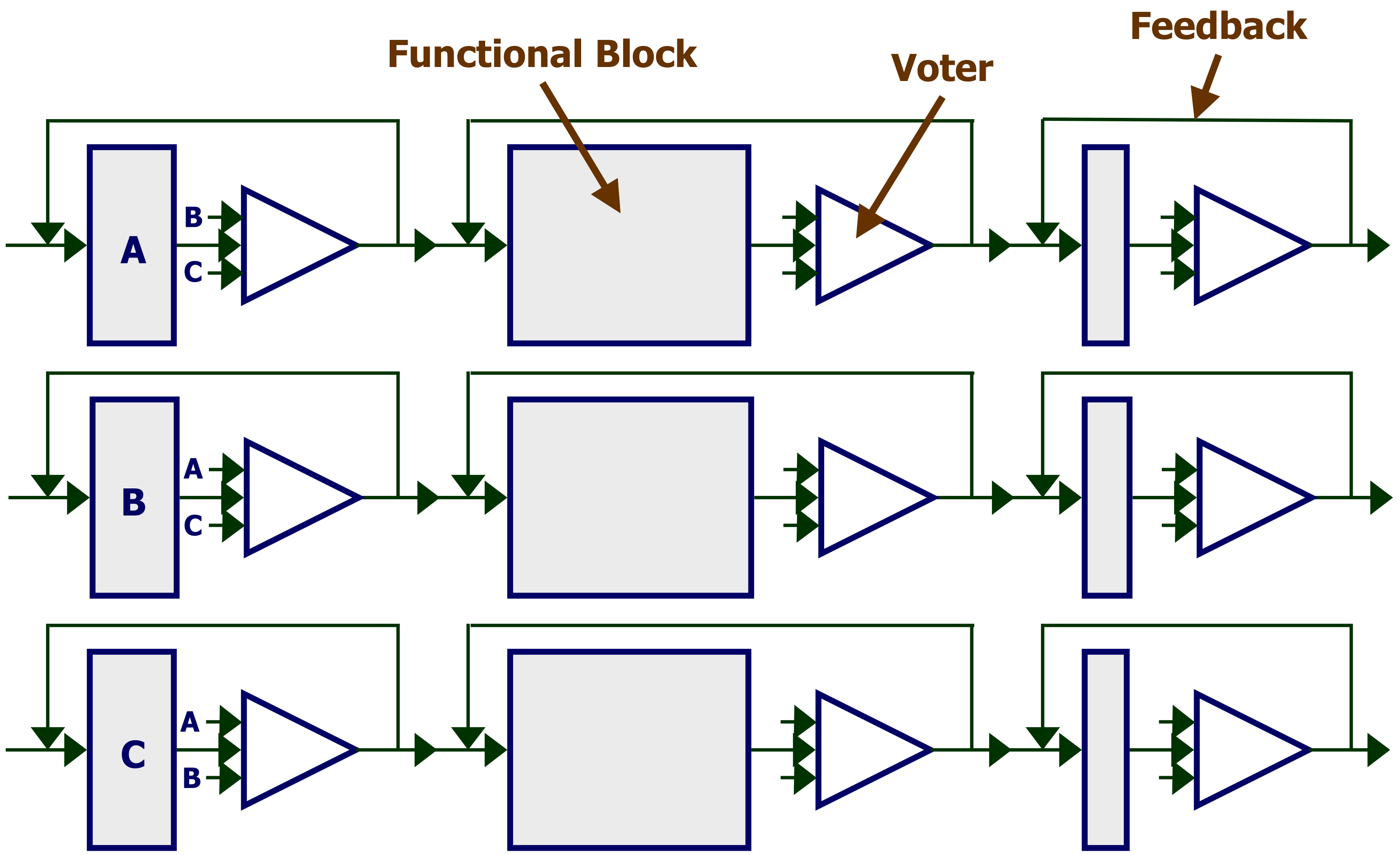
“Fitting” Parameter
is
root mean square module size

Single-String Design



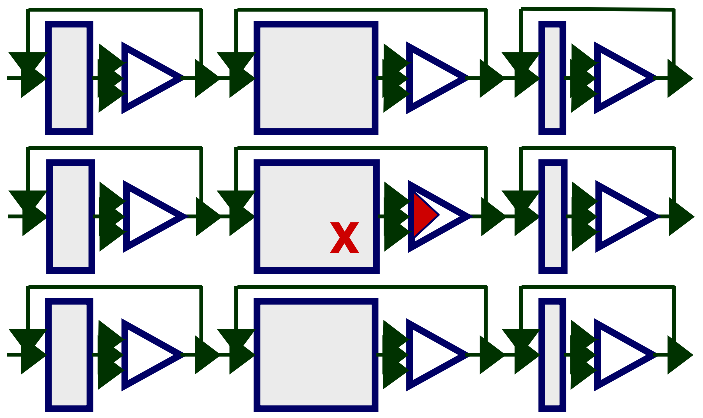
Conceptually, a design is a string of logic blocks (sequential or combinational) bounded by feedback loops.

TMR Design

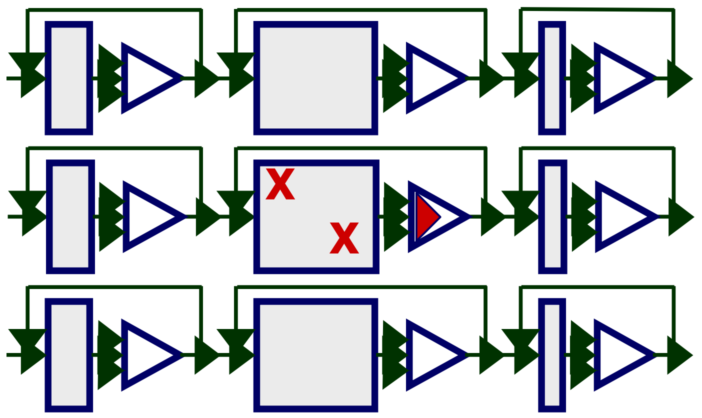


Feedback from the voters corrects state errors inside blocks

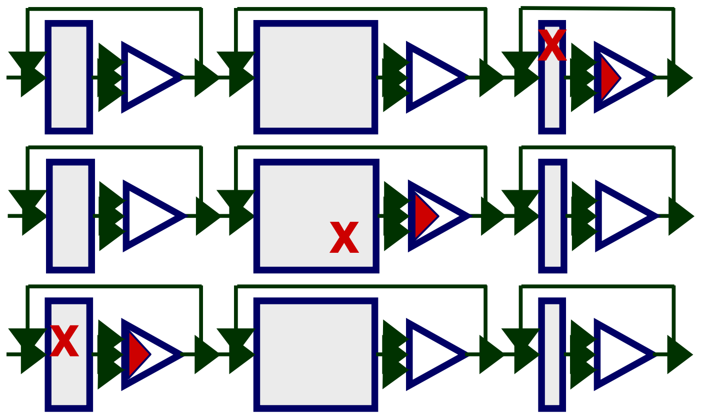
TMR prevents almost all errors



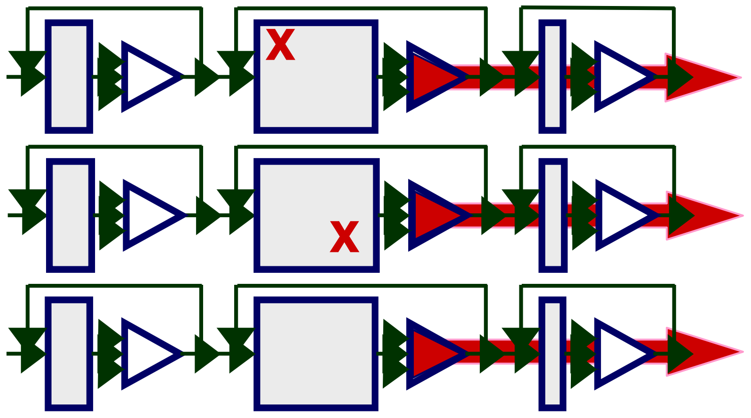
Single upsets cannot cause errors



Multiple upsets but no error

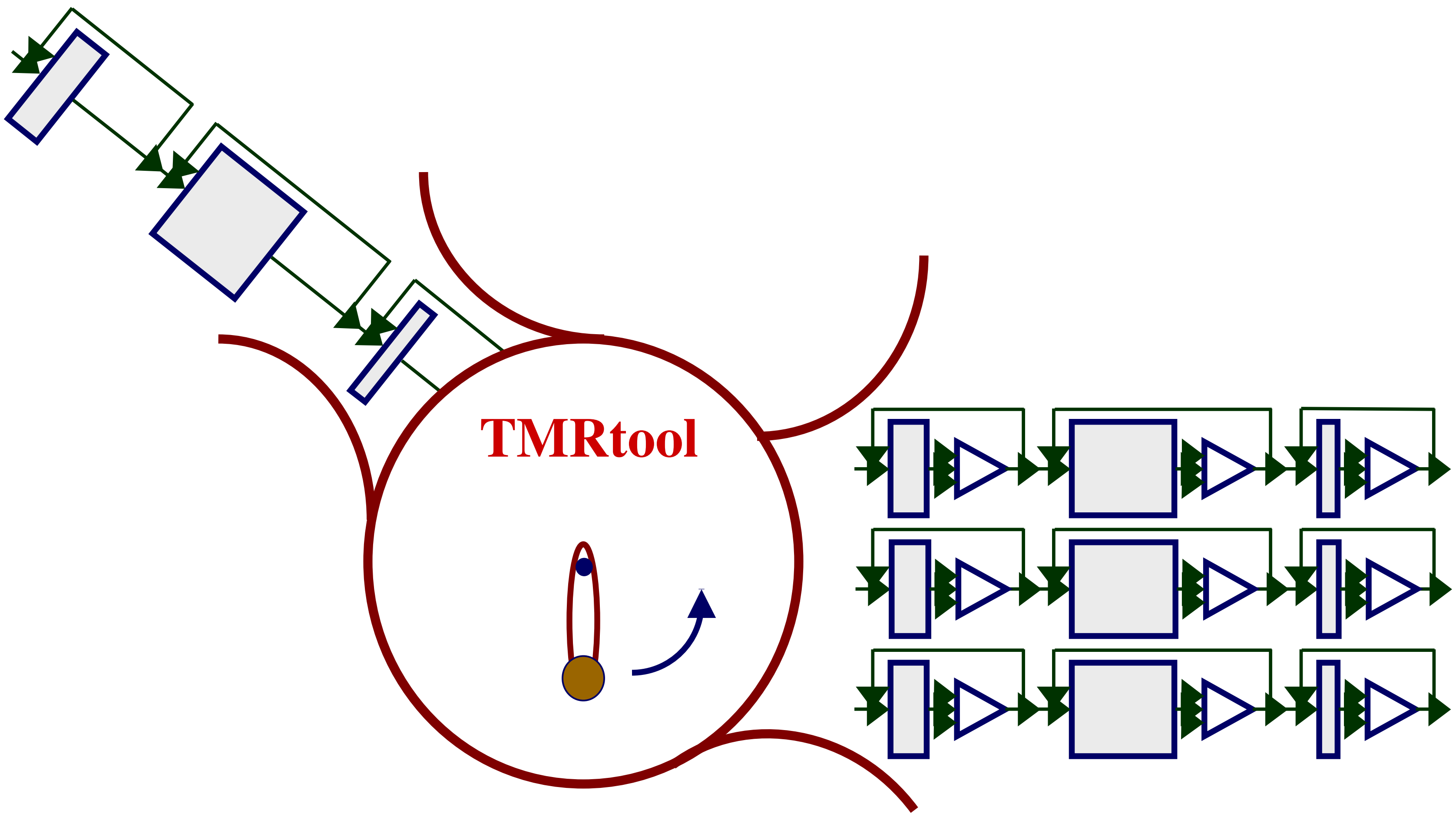


Multiple upsets but no error



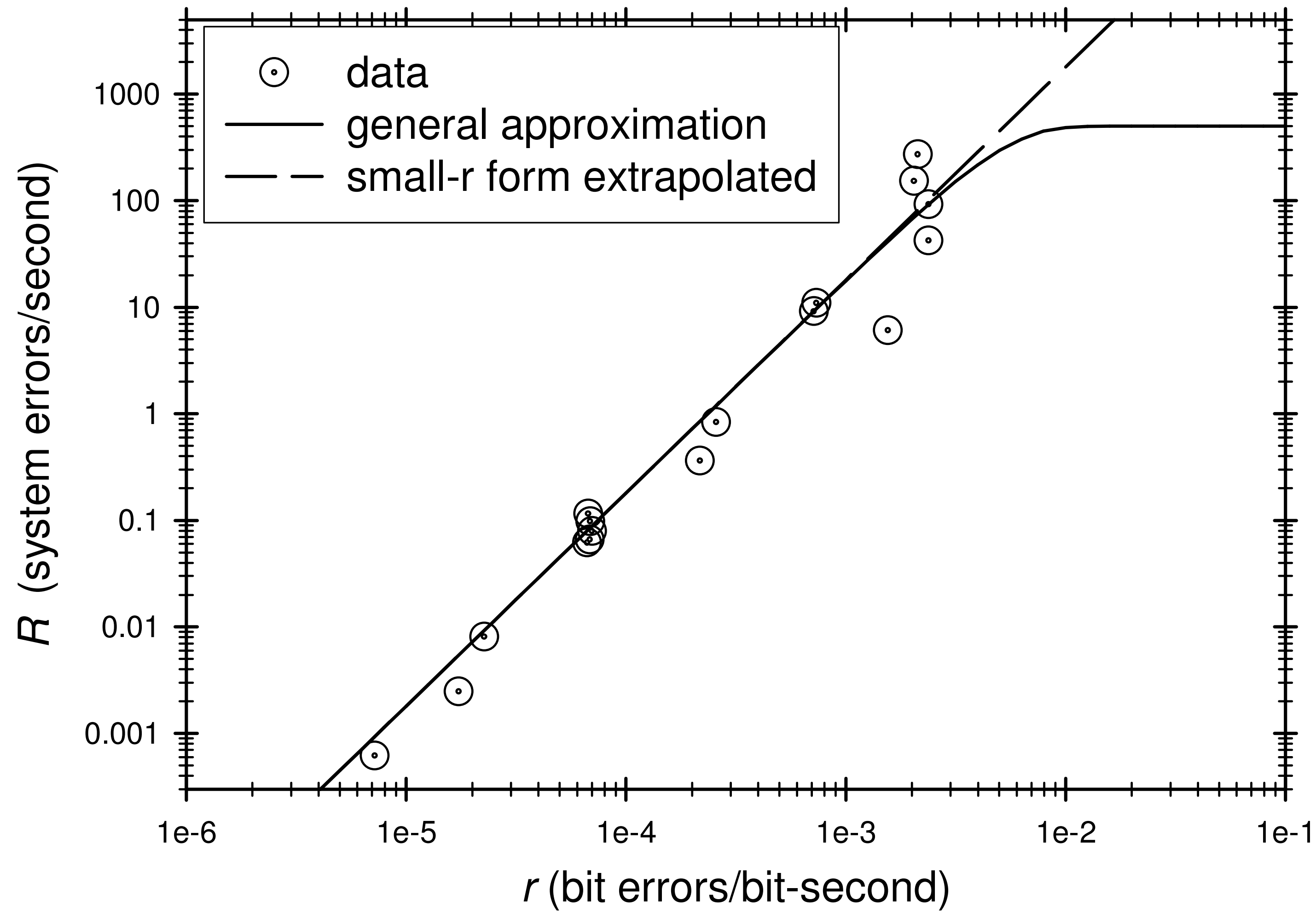
Error propagation requires upsets in two parallel modules (within a scrub cycle).

Designer’s TMR “Burden”



Run the working single-string design through the TMRtool to obtain the correct Xilinx-style triplicated and voted design.

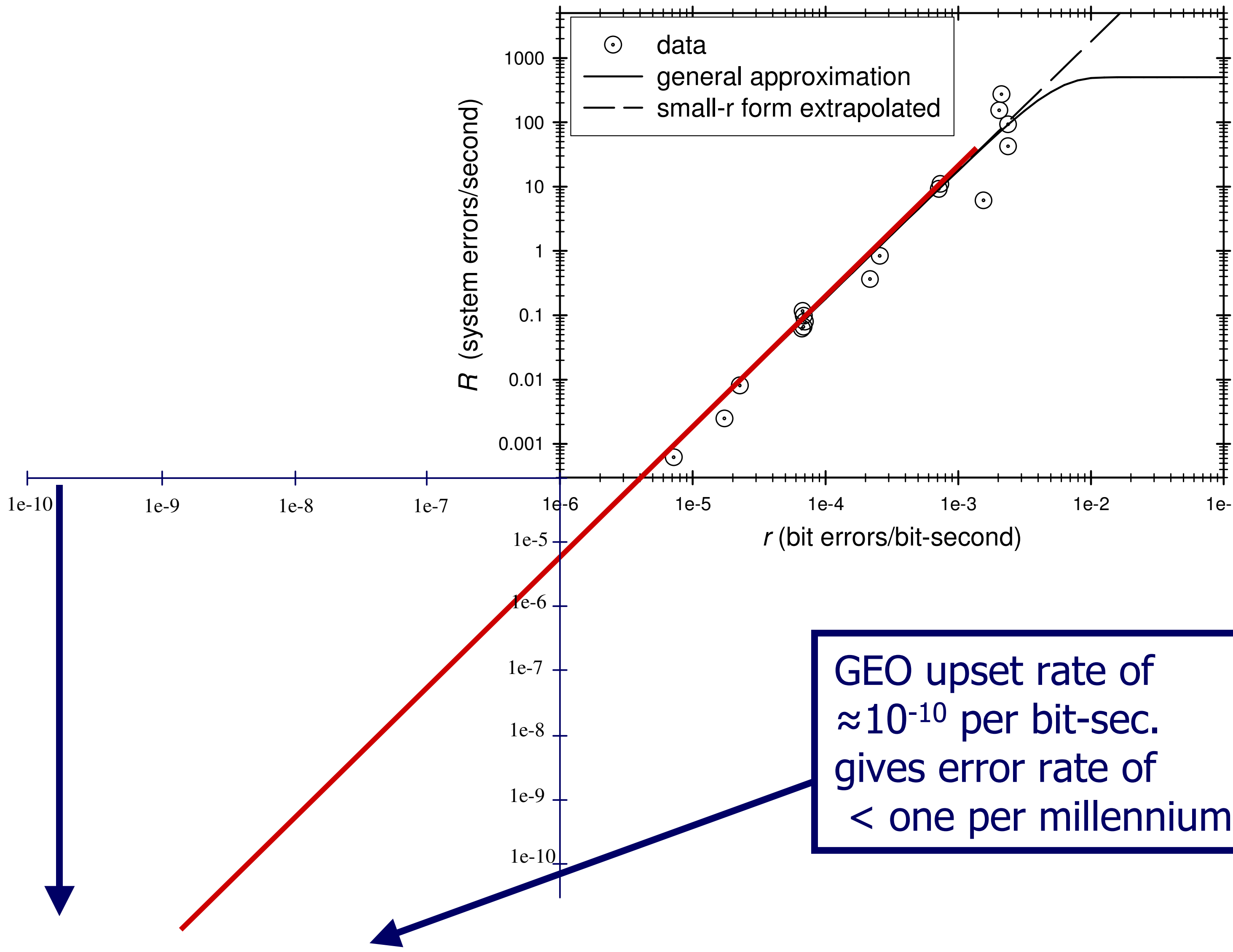
Example App - XQR2V6000 BRAM Scrubber



Given parameters: T=2 ms, M=48000

Fit parameter: M2=M3=M4= 250

Extrapolating to Space Rates



GEO upset rate of $\approx 10^{-10}$ per bit-sec. gives error rate of < one per millennium.

Three Options

1. Do nothing - live with intrinsic upset rate

    For rFPGA's not all config upsets yield errors

    However, nth error 'breaks' design ($n \approx 10$)

2. Upset mitigation - upsets ≠ errors

    Prevent a single upset from causing an error

    Prevent upset accumulation

3. Harden to upset - no upset = no error

Geometrical RHBD is *two-node* problem

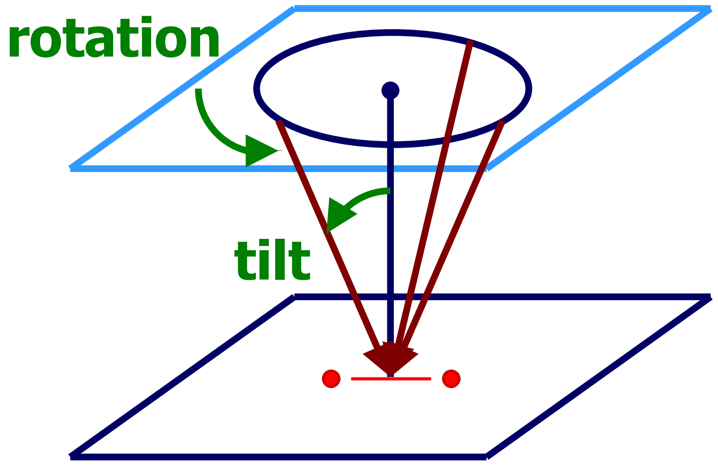
To upset a cell requires some charge collection at *both* of a pair of nodes, that is,

if one node collects no charge, the cell will not upset no matter how much charge is collected at the other of the pair.

A cell may contain one or several such pairs, but the two nodes of a given pair must be as widely separated as possible.

Two-node case makes rotation important

The more an ion trajectory aligns with the line defined by the two nodes, the more likely it is to be able to cause an upset:



For a given tilt, different rotation angles give more or less alignment with line through the nodes.

Model to Guide Data Fits

Model necessary because ‘brute force’ :

requires too much data.

needs extrapolation to impossible tilts (90°).

Model assumes existence of a charge collection efficiency function with ellipsoidal volumes (like rounded RPPs).

Many (8) fitting parameters in current model:

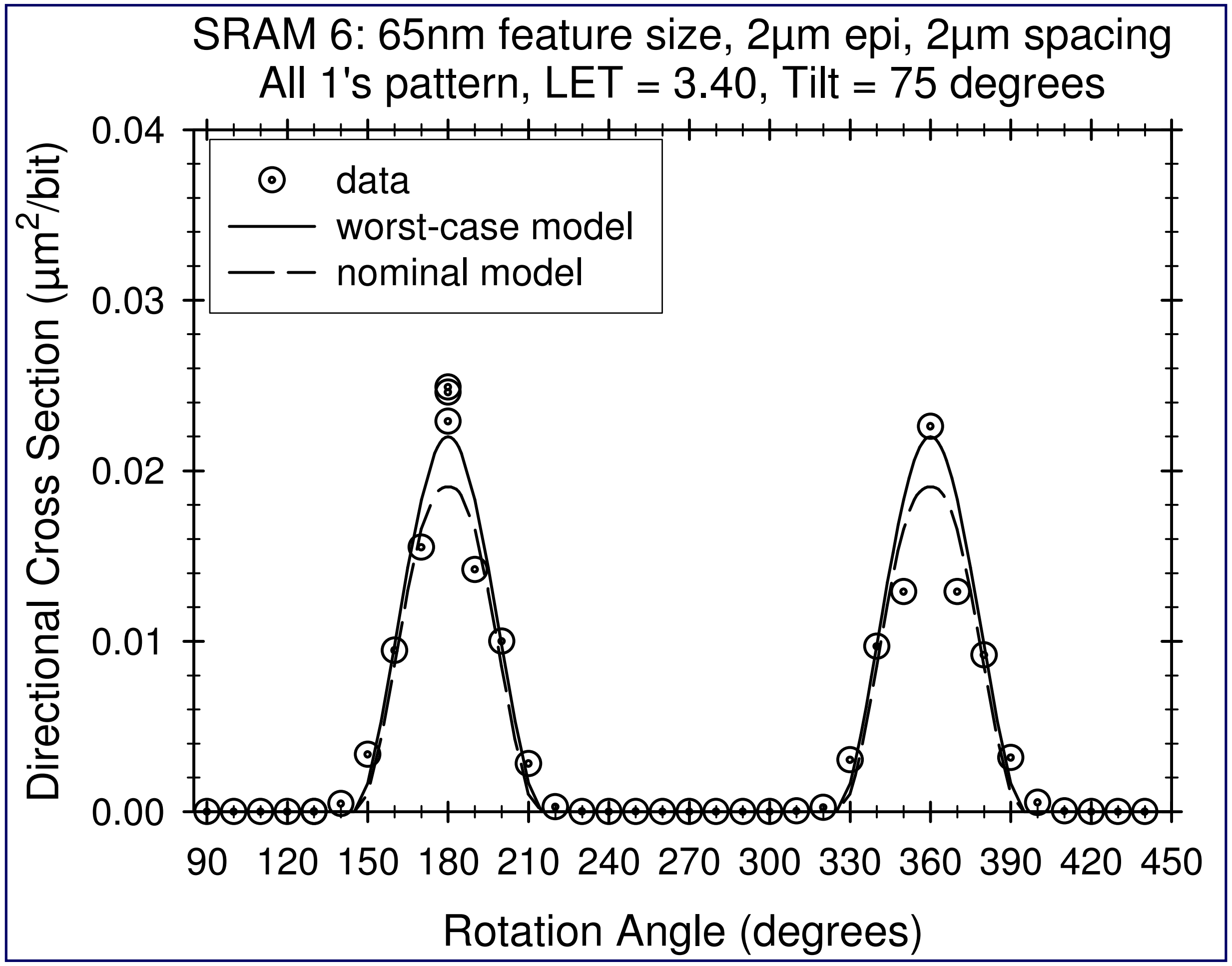
two (A, B) relate to ellipsoid shape

four – LET threshold and sat. cross section per node

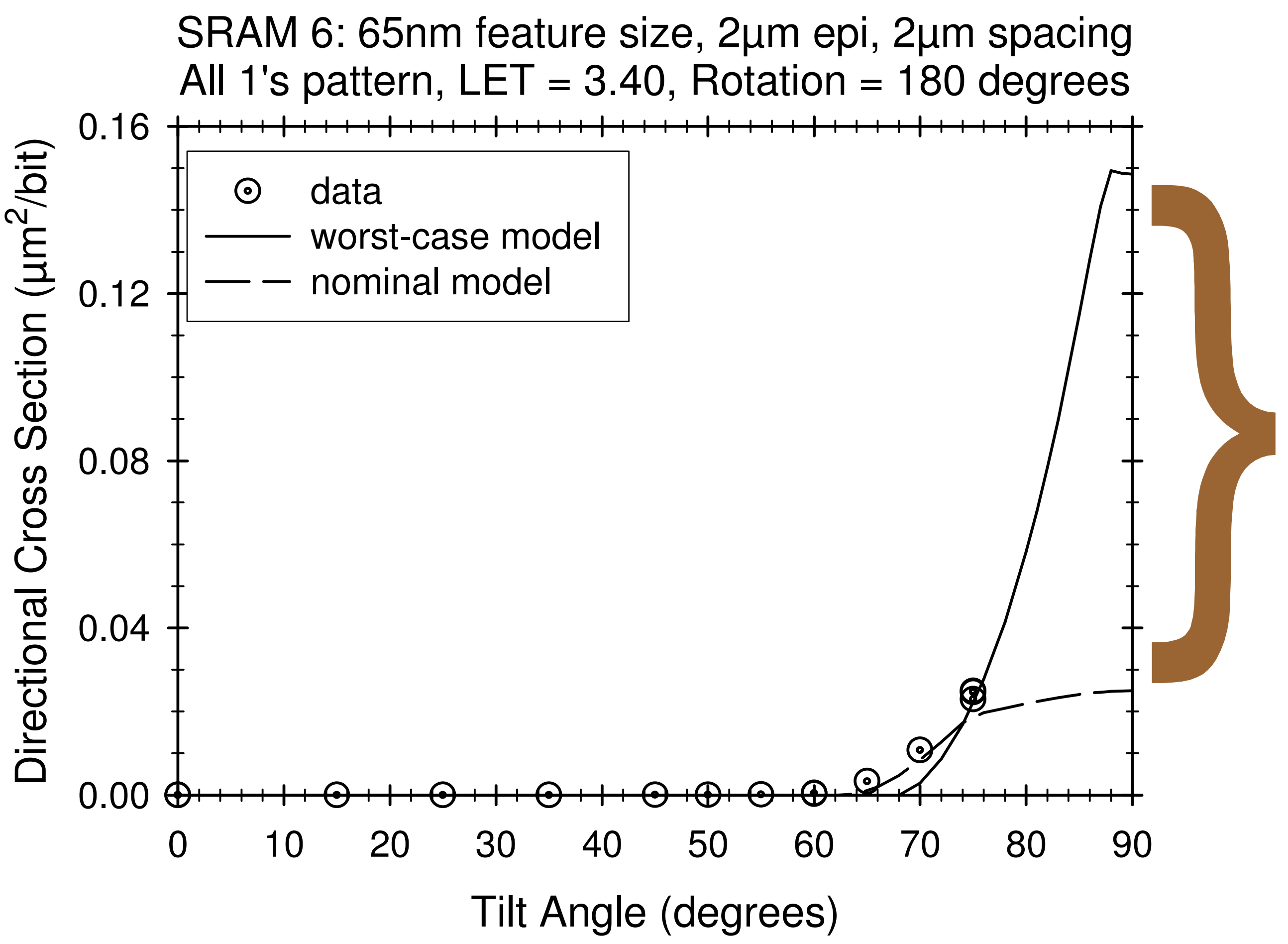
plus two others

Directional Upset Response

Clearly there is a strong dependence on rotation angle:



Necessary Extrapolation



Note factor of 4 or 5

Model Results

GEO rate for ones is <9E-10 upsets per bit-day.

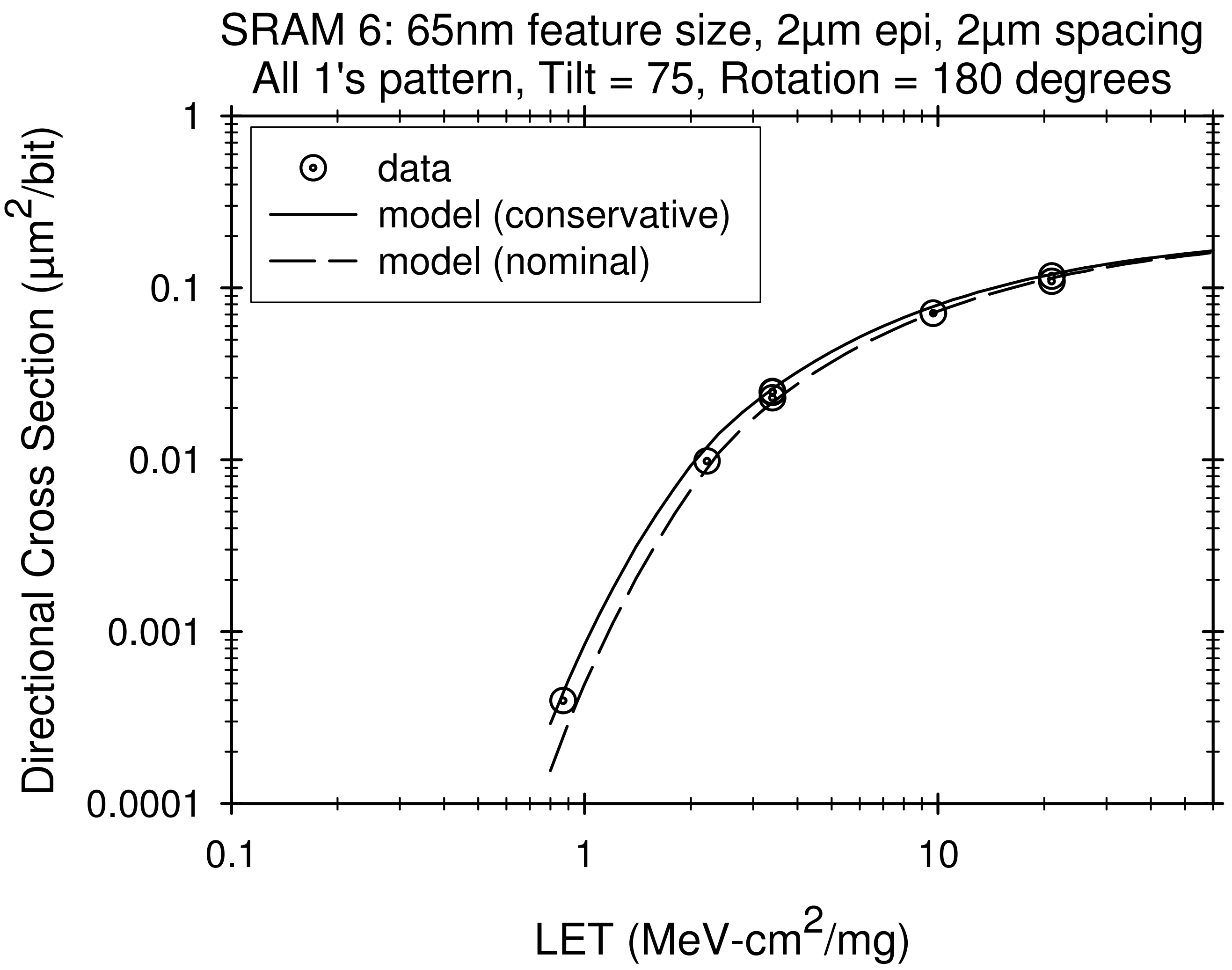
GEO rate for zeros is <9E-11 upsets per bit-day.

Typical design has more than 90% zeros and takes about ten (or more) upsets to cause an error:

GEO rate for typical design is
<2E-11 errors per bit-day
or approx. one every 2 years.

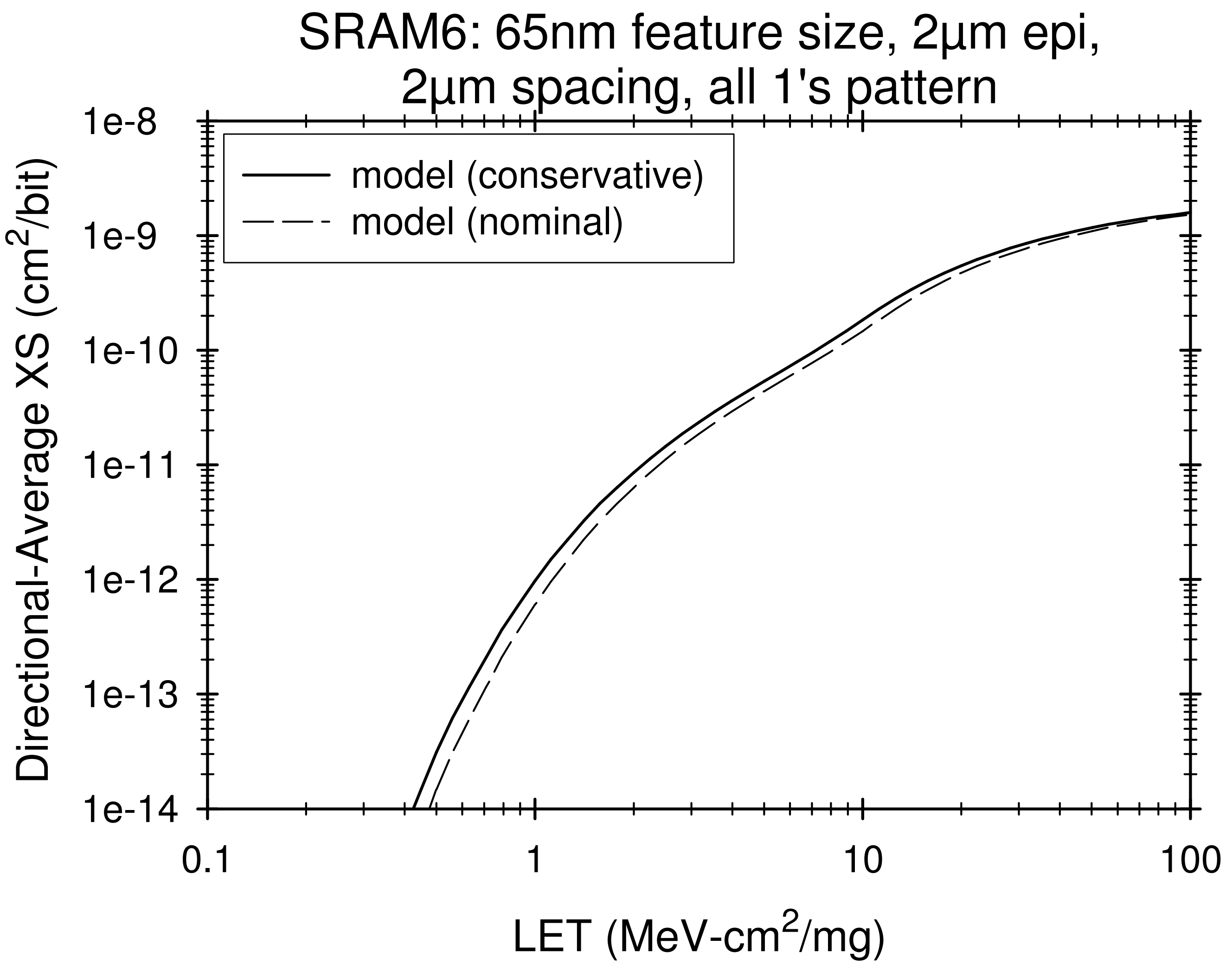
Data & Model for an LET Sweep

Good agreement at worst rotation:



Average Cross Sections

useful for 'estimating' rates via standard calculation



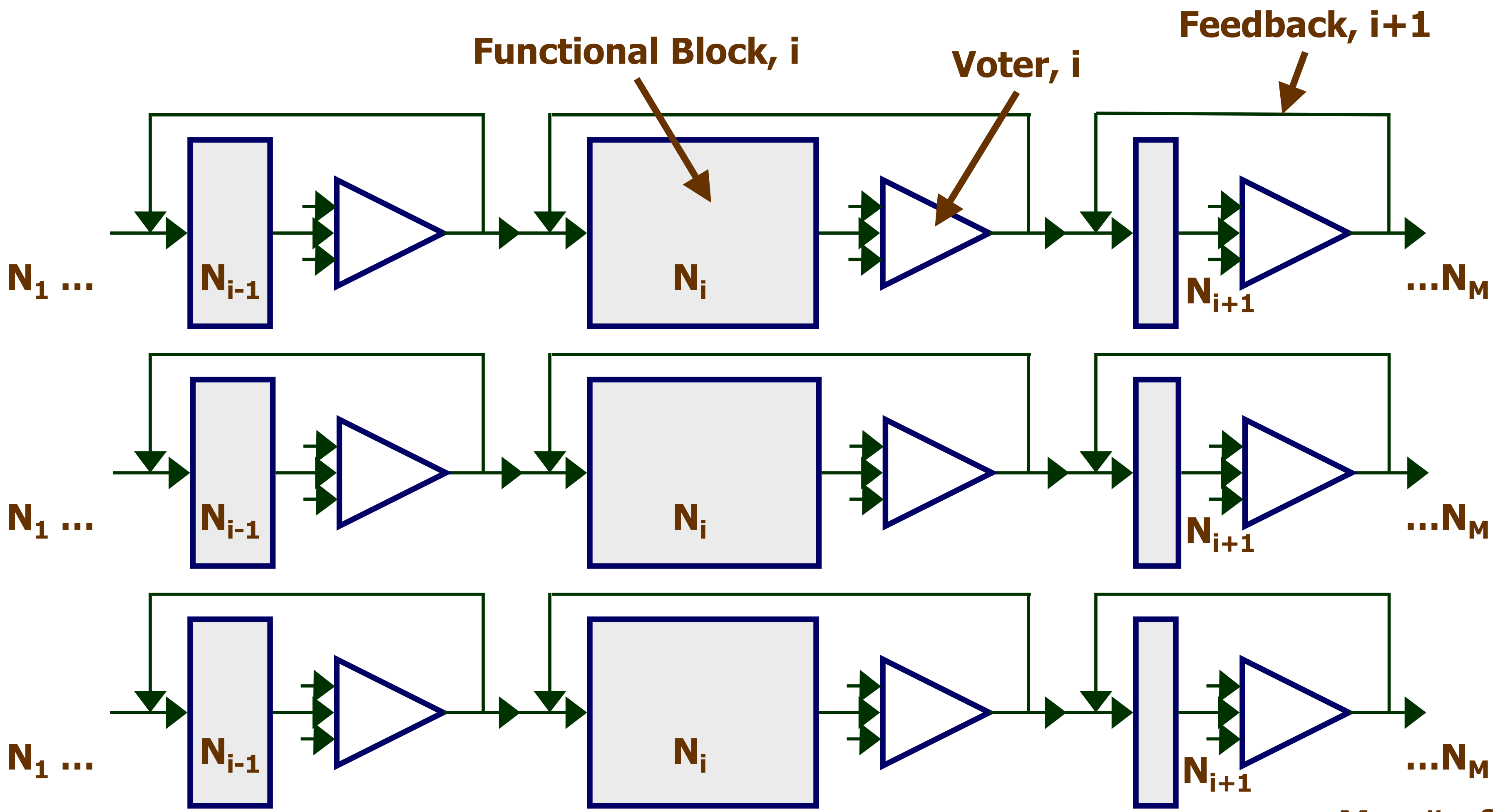
Conclusions - RHBD vs. TMR

Both can yield good system robustness.

- TMR
  - × Requires designer involvement
  - × Costs times 3+ in gates and power
  - × Extrapolation required for space error rates

- RHBD
  - × Transparent to the designer
  - × Requires extra silicon area
  - × Extrapolation required for space error rates
  - × Potentially more robust in “extreme” environments

Appendix: TMR System Model



N_i = # of "care' bits in each block and voter

M = # of modules

T_C = scrub time

Designs are likely to be "lumpy"