Reduced Cost Reliability via Statistical Model Detection

CHREC
NSF Center for High-Performance Reconfigurable Computing

Jon-Paul Anderson - PhD Student
Dr. Brent Nelson - Faculty
Dr. Mike Wirthlin - Faculty

Brigham Young University

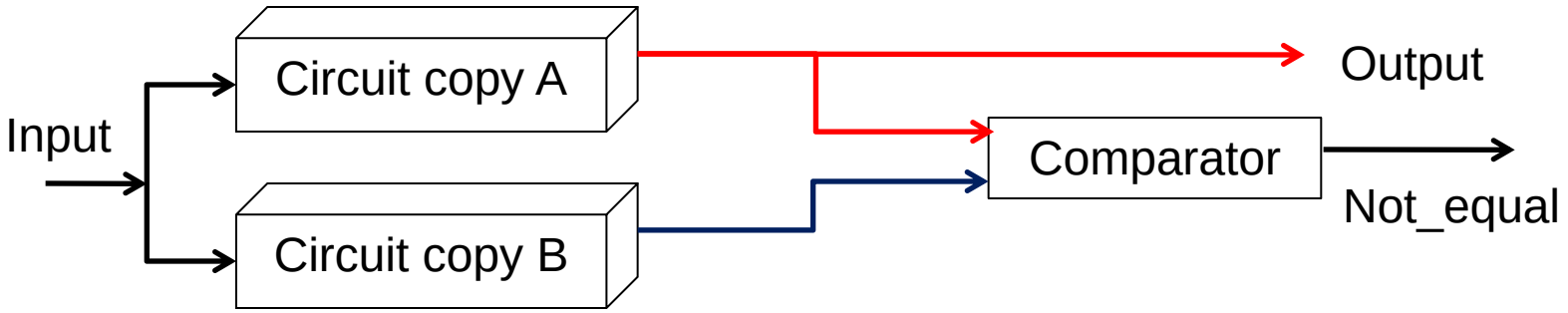
UF UNIVERSITY of FLORIDA
Virginia Tech VIRGINIA POLYTECHNIC INSTITUTE AND STATE UNIVERSITY
BYU BRIGHAM YOUNG UNIVERSITY
THE GEORGE WASHINGTON UNIVERSITY WASHINGTON DC

Alternative Mitigation Techniques

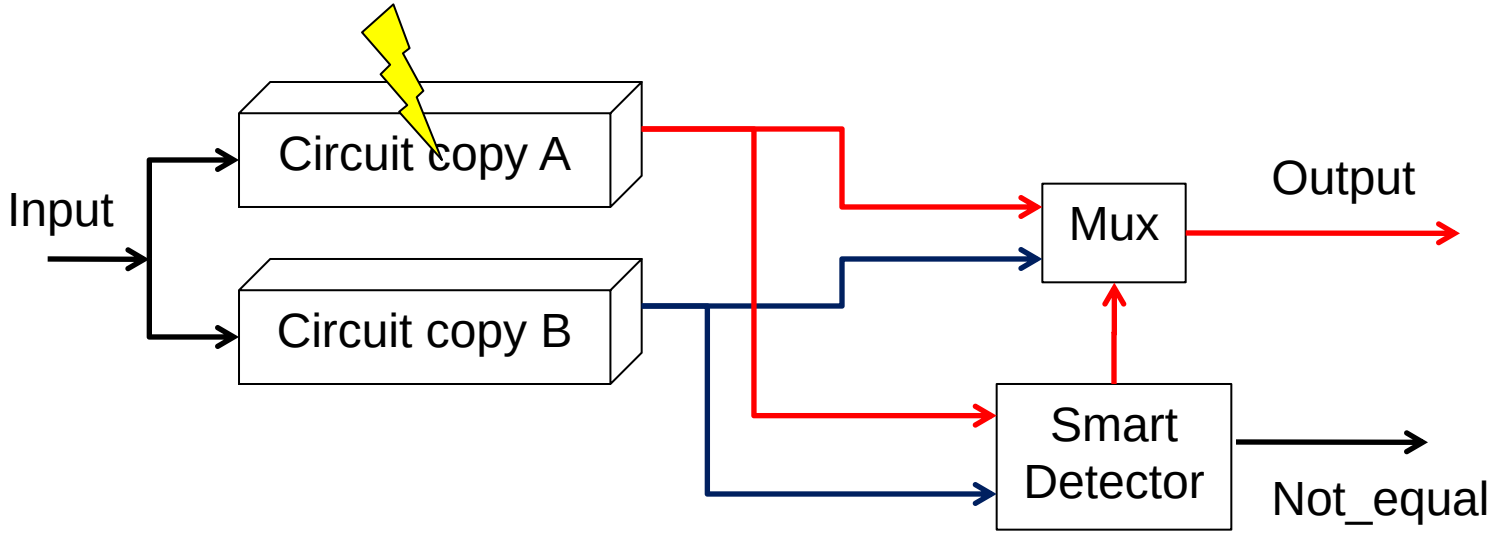
- Triple-Modular Redundancy (TMR) is expensive
  - Area 3-5x
  - Timing ~20%
  - Power 3-5x
- Need reduced-cost mitigation techniques
  - Trade off reliability for area/timing/power
- Motivating Example:
  - In-orbit experiment cannot be triplicated due to area cost
    - Some mitigation is better than none
    - Marking of which data is suspect would be useful

Smart Detection

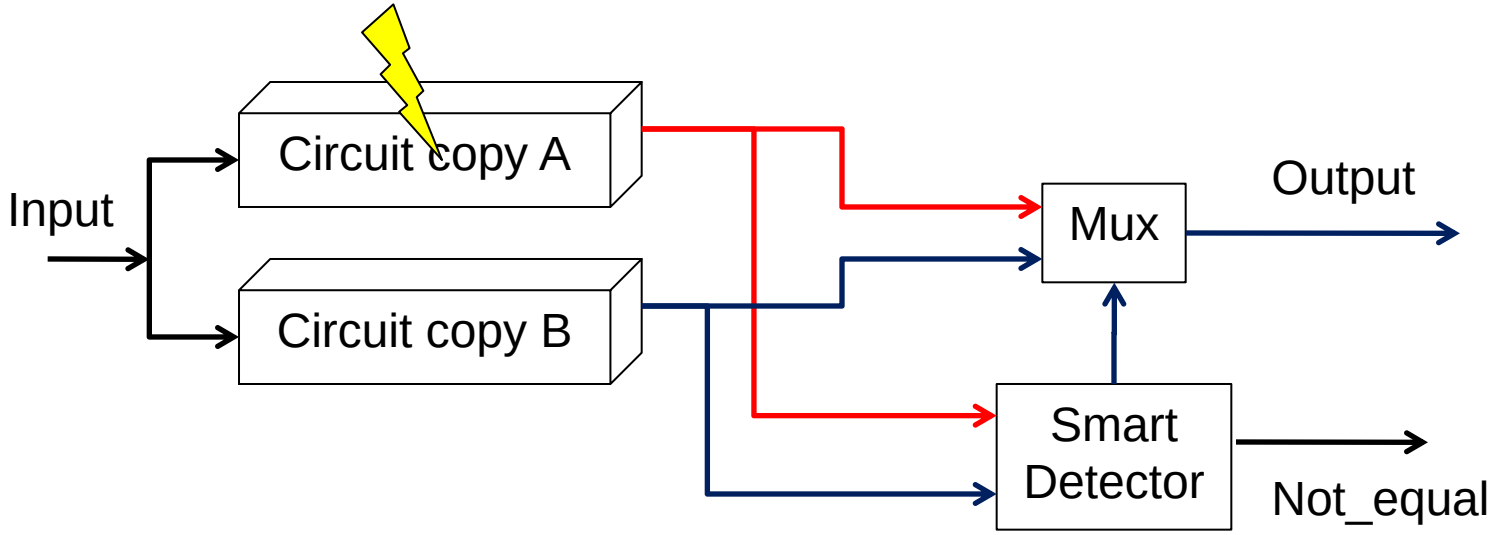
- Duplication + Detection as lower cost alternative?
  - Duplication is 2/3 the size of TMR
  - Duplicate With Compare only detects errors - doesn't mask them
    - Can DWC be modified to mask?
      - Use of 'smart detector' to attempt to mask errors.



Smart Detector

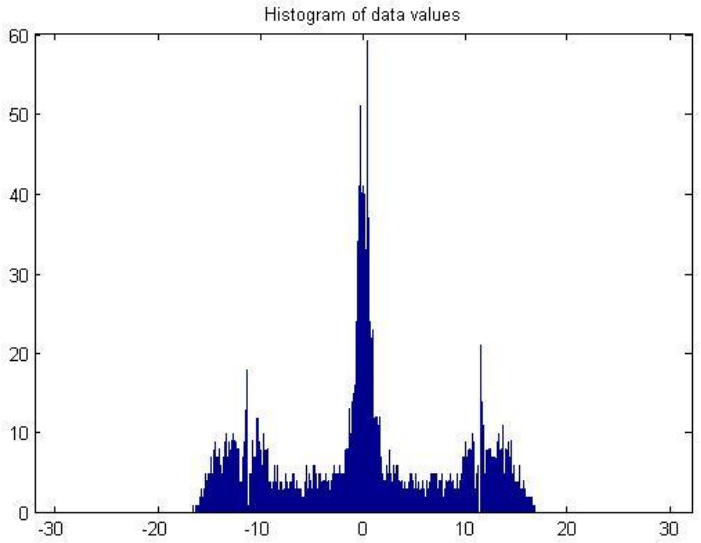


Smart Detector



Statistical Smart Detector

- Statistical detection
  - Use a histogram of data values to try and determine which branch is without error
  - 3 possible outcomes
    - Correct detection
    - Incorrect detection
    - Ambiguous outcome



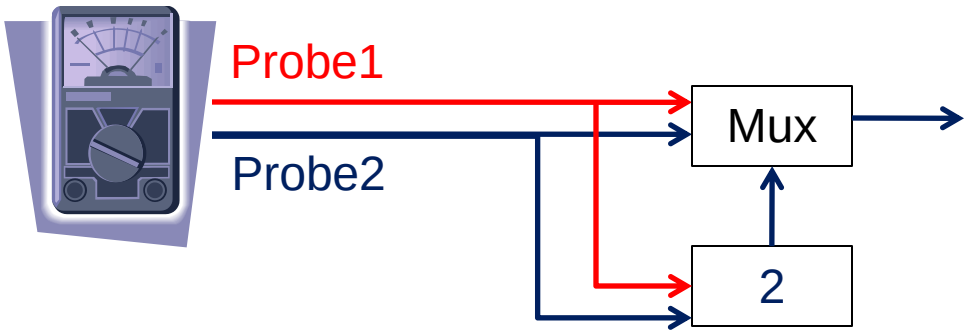
Simple Statistical Example

- Volt meter has redundant probes
  - TMR would be too expensive so we use two with a statistical model
- Three possible outcomes
  - One probe reads 1.2V and other reads 5V
    - Result – Ambiguous detection
  - Correct circuit reads 3.3V and circuit in error reads 15V
    - Result – Statistical model chooses correct voltage
  - Correct circuit reads 15V and circuit in error reads 3.3V
    - Result – Wrong voltage chosen.

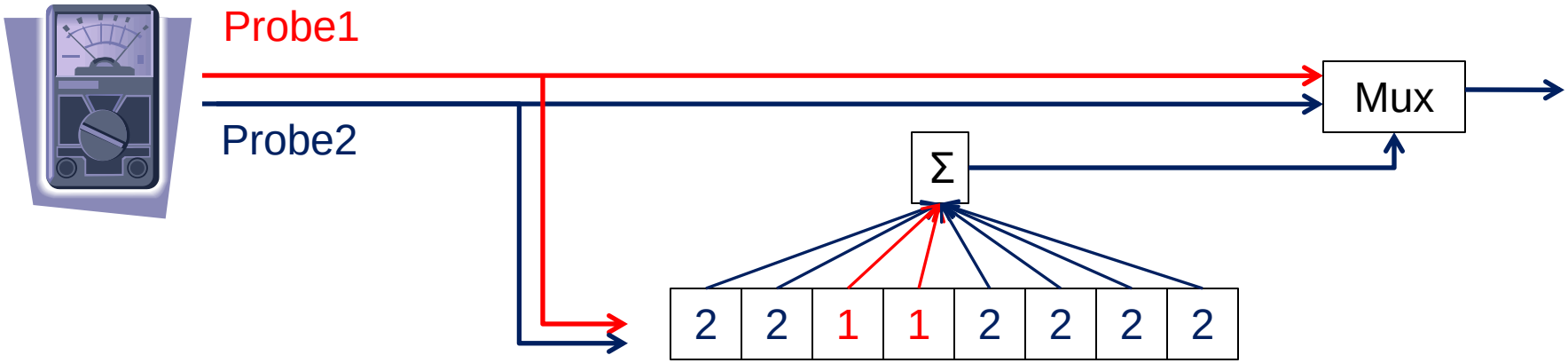
Voltage	Probability
3.3V	50%
1.2V	20%
5V	20%
15V	10%

Statistical Example

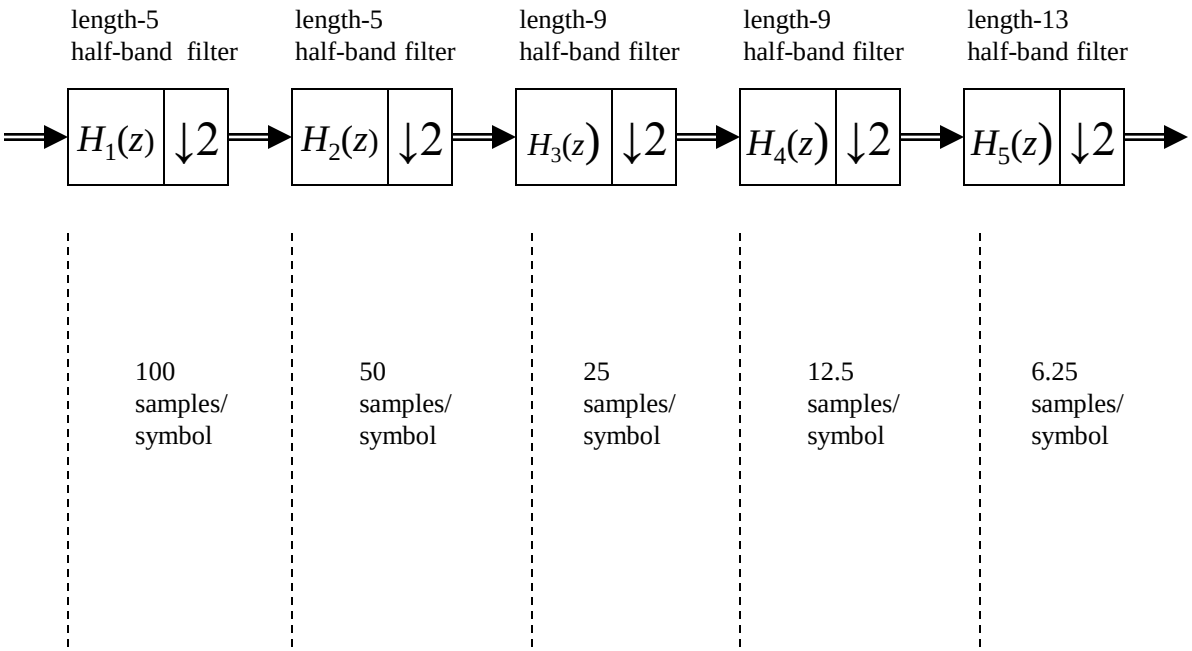
Statistical model with no history



Statistical model with 8 deep sample history

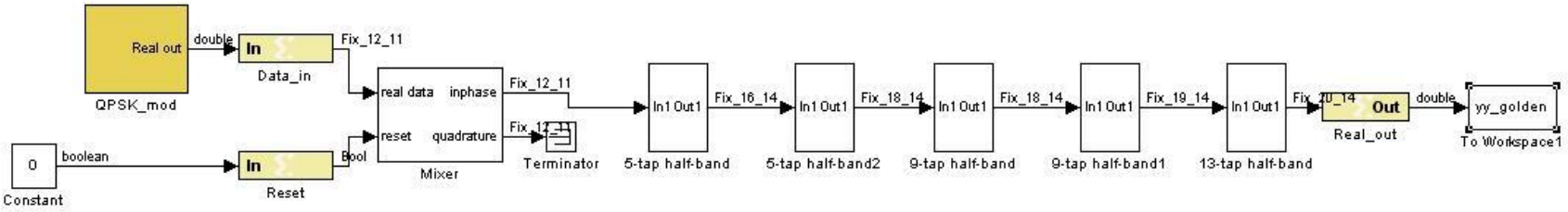


System under test

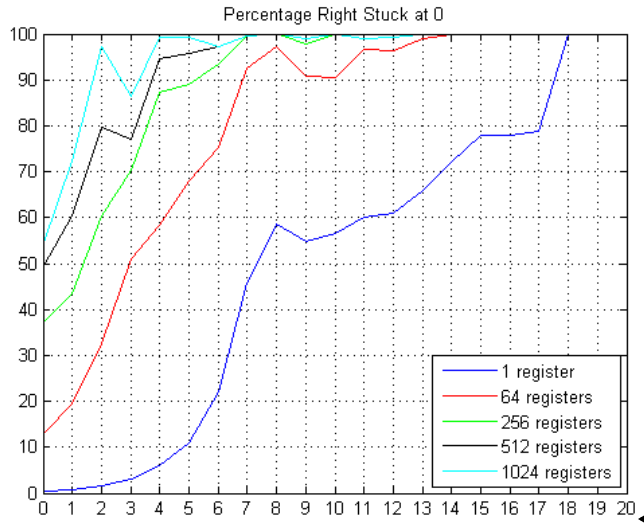


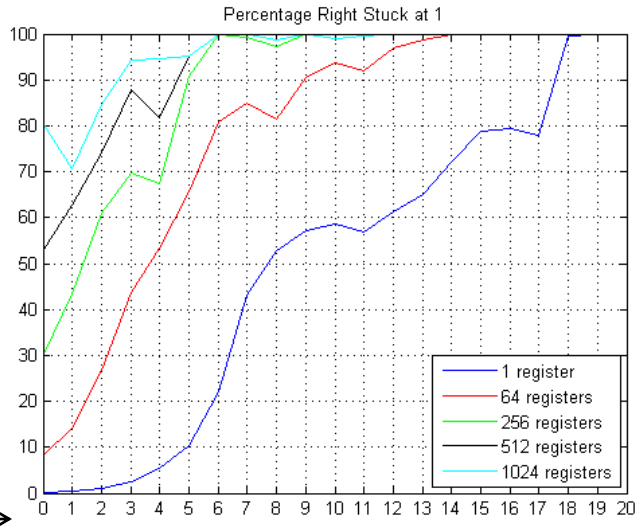
Initial tests – Stuck at faults

- Downsampler was created in System Generator
  - Matlab was then used to create artificial stuck at faults and tabulate the results
    - Tests run for stuck at 1 and stuck at 0 faults for all bits in the 20 bit result

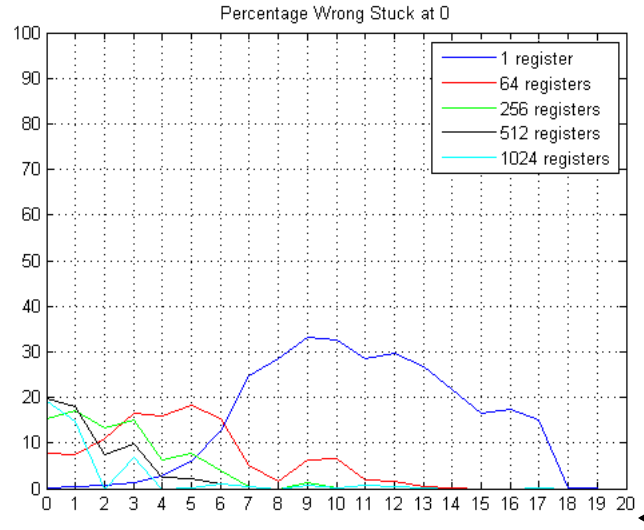


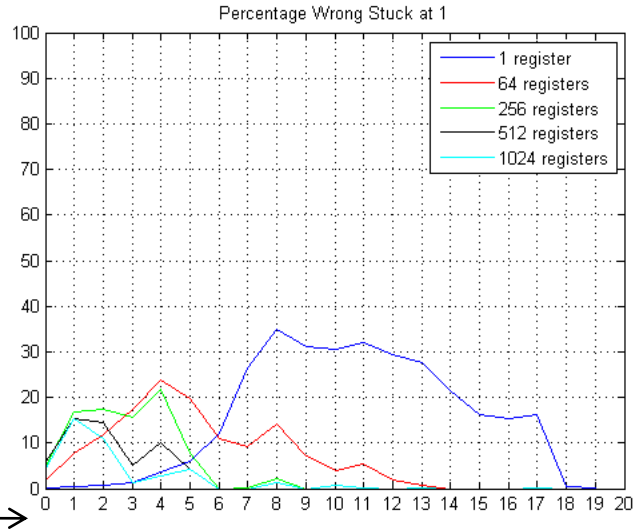
Stuck at results





← Bit position →

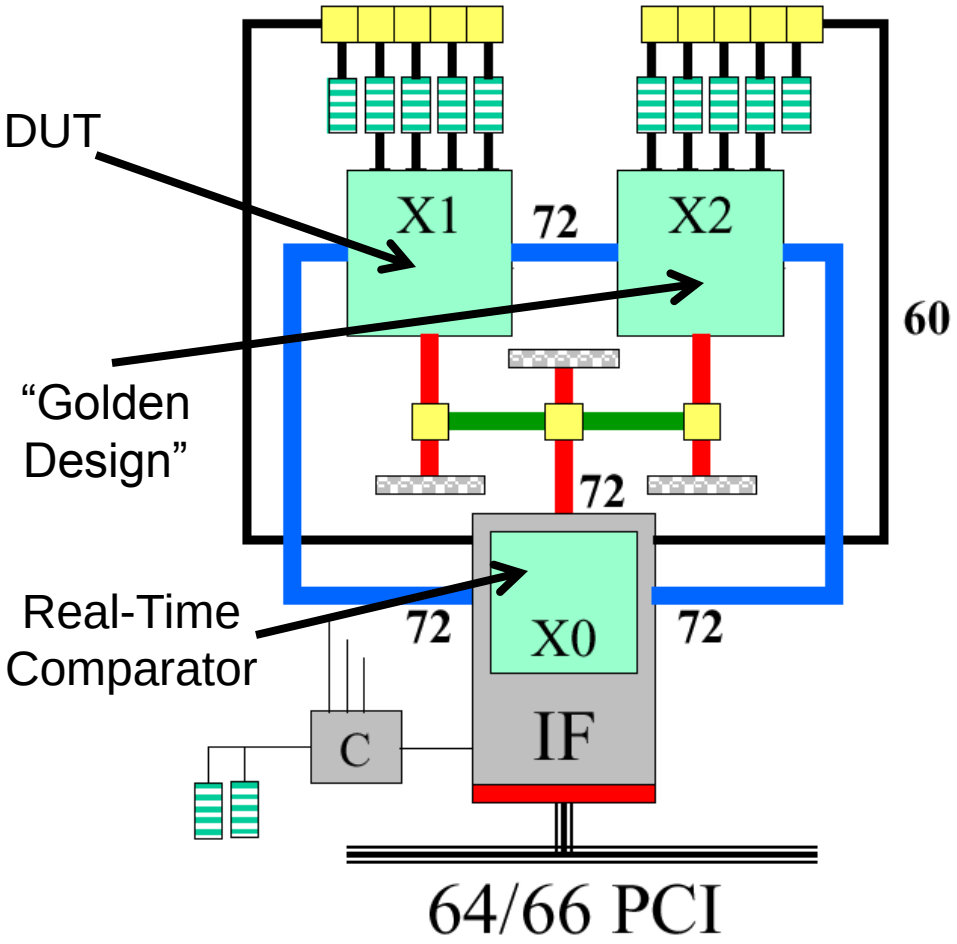




← Bit position →

Fault simulator

- BYU/LANL fault injection tool
  - Based on SLAAC-1V board
    - PCI card with 3 Virtex 1000 FPGAs
  - Previously validated with radiation testing
  - Sensitive configuration bits are tabulated and then tested one by one



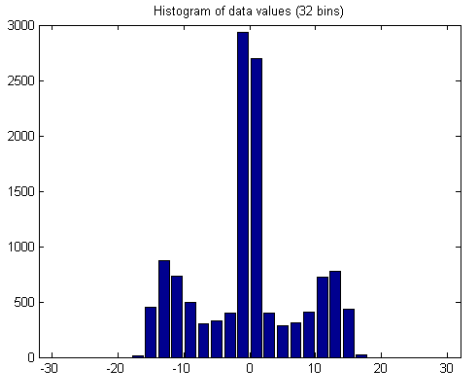
Virtex SEU Emulator (LANL/BYU)

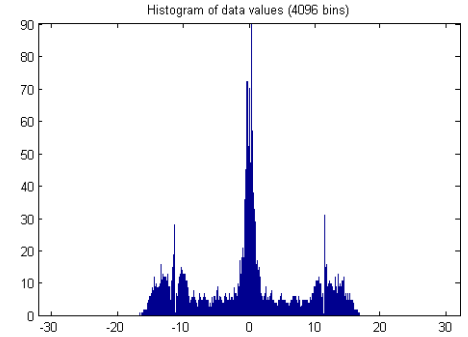
Test Methodology

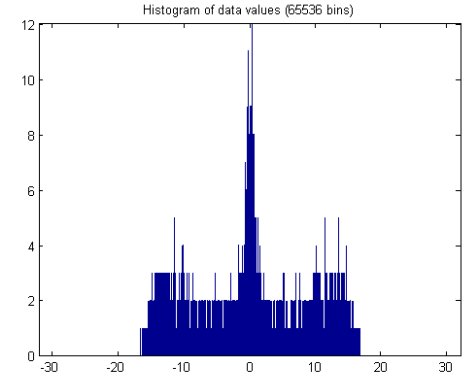
- Design is loaded onto the SLAAC board and the sensitive configuration bits are tabulated
  - Every bit in the configuration bitstream on the DUT is flipped individually and if there is a difference on the output with the golden copy then the bit is recorded as ‘sensitive’.
- Random numbers are fed through a QPSK modulator in Matlab to generate the input vector.
- The vector is then run through the original design without injecting faults to gather a golden output.

Test Methodology

- The histogram is generated with the golden data in Matlab by specifying the bin size.
  - If the bin size is too large, too many faults will map to the same bin resulting in ambiguity.
  - Small bin sizes cause multiple bins to have the same counts, once again resulting in ambiguity.
  - To simplify the hardware, bin sizes are constrained to powers of 2.







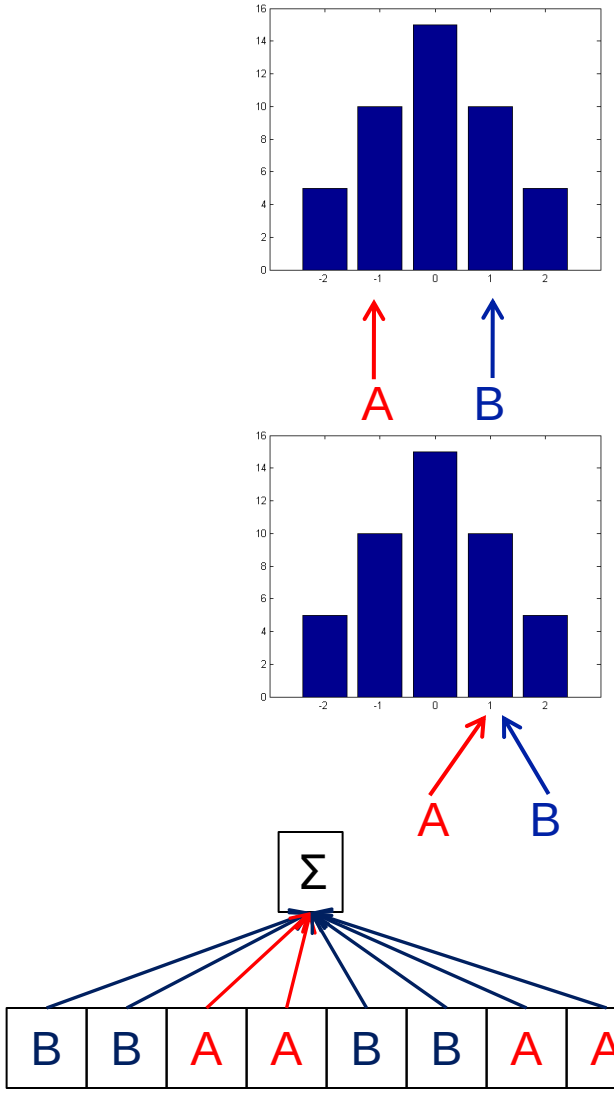
Test Methodology

- The input vector is then run through the design for each sensitive bit and the output captured.
  - This design has 73146 sensitive bits
  - The fault is inserted into the design roughly halfway through the execution to give a certain amount of fault free operation
- Matlab is then used to implement the smart detector and analyze the results.

Ambiguous contribution

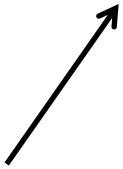
- Ambiguous detection occurs in three ways

- Four possible ways to count ambiguous results
  - Don't count them at all
  - Record all as a wrong choice
  - Record all as a right choice
  - Record half as correct
    - Assuming it is fair, 50% of the time you should get it right

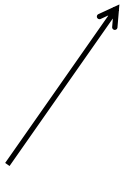


Results – Correct decisions for 1024 bins

Number of samples considered for decision	Percent decisions with ambiguous outcome	No ambiguous decisions counted	All ambiguous counted as wrong	Half ambiguous counted as right	All ambiguous counted as right
1	34.14%	86.13%	56.73%	73.80%	90.87%
64	15.80%	88.60%	74.60%	82.50%	90.40%
256	9.83%	90.15%	81.29%	86.20%	91.12%
512	7.55%	90.96%	84.09%	87.87%	91.64%
1024	5.66%	92.05%	86.84%	89.67%	92.50%



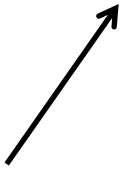
Lower Bound



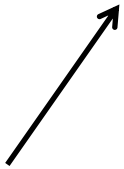
Upper Bound

Results – Correct decisions for 4096 bins

Number of samples considered for decision	Percent decisions with ambiguous outcome	No ambiguous decisions counted	All ambiguous counted as wrong	Half ambiguous counted as right	All ambiguous counted as right
1	28.29%	85.49%	61.30%	75.45%	89.59%
64	9.59%	93.05%	84.12%	88.92%	93.71%
256	4.93%	95.29%	90.60%	93.06%	95.53%
512	3.48%	95.87%	92.53%	94.27%	96.02%
1024	2.54%	96.57%	94.12%	95.39%	96.65%



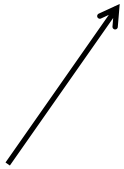
Lower Bound



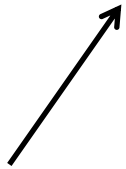
Upper Bound

Results – Correct decisions for 16384 bins

Number of samples considered for decision	Percent decisions with ambiguous outcome	No ambiguous decisions counted	All ambiguous counted as wrong	Half ambiguous counted as right	All ambiguous counted as right
1	23.70%	88.49%	67.51%	79.36%	91.21%
64	4.39%	97.04%	92.78%	94.97%	97.17%
256	1.83%	97.69%	95.90%	96.81%	97.73%
512	1.18%	97.93%	96.77%	97.36%	97.96%
1024	.81%	98.25%	97.45%	97.86%	98.27%

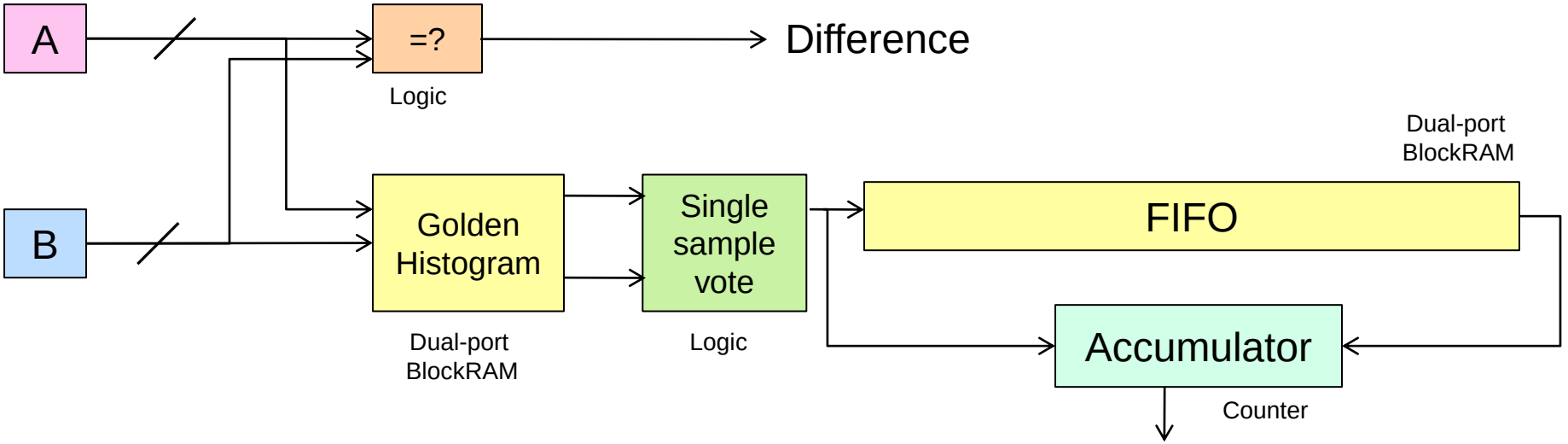


Lower Bound



Upper Bound

Possible Hardware Implementation



- Smart detector consists of
  - Small number of BlockRAMs
  - Counter
  - Small amount of logic

Conclusion

- Smart detection using a simple histogram was discussed
  - High accuracy with very low resource costs
- Future work
  - Expand statistical approach to more than just FIR filters.
  - Investigate using machine learning techniques for an even more accurate smart model