Summary Part 2
A Systematic Method to Compute the Architectural Vulnerability Factors for a High-Performance Microprocessor

Mrinal Nath (ID: 3307043)
April 6, 2005

Did this paper address an important issue? Explain.

SEUs have become a major cause for reliability concerns in the latest and future microprocessors. This is because due to the decrease in transistor geometries, even a small amount of charge is now sufficient to corrupt the data stored in a latch or flip-flop. To ensure reliable operation, designers have to include fault tolerance mechanisms into the design or take some other measures to harden the transistors against SEUs (radiation hardened transistors). But an important observation is that not all faulty bits produce an observable error in the programs output. This paper defines the AVF of a structure in a microprocessor as the probability that a fault in that structure will lead to an error in the output. This metric is important, since it tells the designers which structures should be made robust and which structures can be left unprotected. Without this information, designers may over-protect their whole design which will increase the cost of the chip, or they may under-protect their design which will make the design less reliable. Thus AVF provides important information.

Are the proposed approaches valid? Describe its strength and weakness.

The approach is based on identifying the subset of all processor state bits required for `architecturally correct execution' (ACE). Only these bits have to be protected from SEUs. The other bits will not cause any observable error in the output and need not be protected. They start off with the assumption that all bits are ACE unless proved otherwise. Then they identify the various un-ACE bits at the architectural and microarchitectural levels.

Strengths

- They can identify the ACE bits with a fair amount of accuracy, which allows them to determine the AVF of a particular structure with reasonable accuracy
- Due to their assumption that all bits are ACE unless proven otherwise, they will usually provide a higher estimate of ACE than the actual value. This is helpful, since it will ensure the reliability of the design, while minimizing the cost
- This method of calculating the AVF is better than ‘traditional’ statistical fault injection methods, because in this method the ACE bits are tracked through the architectural registers and memory. Thus this a ‘direct’ measurement rather than a ‘statistical sampling’.

Weaknesses

I can’t think of any weaknesses as such, since the authors are just proposing a way to measure a metric. At the most, they have to determine the ACE bits using a large number of benchmarks with various characteristics, so that their results are not biased in any way.

Do the results support the conclusions? Explain.
The conclusion is that tracking the ACE bits gives and calculating the AVF as the average AVF of the bits in the structure gives a good estimate of the AVF. This seems to be a valid conclusion since it is known that not all bits affect the program output – only the ACE bits matter. Their technique of identifying the un-ACE bits is valid. Thus, their AVF calculations are also valid.

Describe the potential future works?

They could refine their un-ACE detection strategies so that they can come up with more accurate estimates of AVF. This will help in reducing the need for an RTL level analysis to get accurate fault-rate estimates. They plan to use ACE analysis at the RTL level, which may prove to provide very accurate fault-rate information. They could also consider evaluating the AVFs of other structures like reorder buffer or reservation stations, since these structures appear to have an AVF of nearly 100% at first glance.