Summary Part 2
DIVA: A Reliable Substrate for Deep Submicron Microarchitecture Design
Mrinal Nath (ID: 3307043)
April 6, 2005

Did this paper address an important issue? Explain.
Testability is becoming an issue with increasing design complexity, due to which there is the possibility of design bugs not being caught during testing. Also, as the device geometries decrease, the devices become more susceptible to transient faults e.g. SEUs. Due to such reasons, there is a need for integrating robust fault tolerance mechanisms in the design itself. This paper proposes one approach to providing such a fault tolerance mechanism in high performance microprocessors. So it addresses an important issue.

Are the proposed approaches valid? Describe its strength and weakness.
The proposed technique involves the verification of all the results generated by the processor core before committing the results. This is done by using a checker unit which sits between the processor core and the commit stage.

Strengths
- The checker is a simple in-order pipeline, so it is easy to verify and validate it thoroughly
- The checker design is different from the core design, so it can detect hard (design/manufacturing) errors as well as soft (transient) errors
- The inputs to the checker are ‘pre-computed’, so the checker (CHKcomp) pipeline never stalls. The CHKcomm pipeline may stall due to structural hazard on the memory ports, but this can be eliminated by providing a single exclusive memory port for the checker. The pipeline control mechanism is very simple
- The checker is independent of the (micro)architecture of the processor core
- The checker is reusable, so the cost of thoroughly testing it can be amortized over several processor designs
- It can be made of larger transistors, to that it is insensitive to transient faults due to noise, crosstalk and SEUs
- It is possible to use the error rate information collected from the checker to implement aggressive DVFS techniques and/or to provide graceful degradation in the face of processor core failure

Weaknesses
- The performance will be very poor if there is some hard fault which causes a very high error rate
- The power consumption due to the checker unit and due to the flushing of the core when an error is detected is not considered
• The DIVA architecture assumes that the memory and the register files are protected and always provide correct values.

• The information passed from the core to the checker has to be guaranteed to be correct, which will involve the overhead associated with error detection and correction codes.

• The throughput of the DIVA checker has to be at least as much as the maximum throughput of the processor core (in terms of number of instructions retired per clock). This can impose some limitations on the choice of architecture and microarchitecture for the DIVA checker, and introduce some complications in case of a deeply-pipelined checker.

Do the results support the conclusions? Explain.

The main conclusions are that DIVA is a simple, efficient and practicable approach to solve the testability issues and provide a robust fault tolerance mechanism in modern microprocessors. The design of the DIVA checker is such that it can detect and correct soft as well as hard errors. The results show that the overall performance overhead caused by the introduction of the DIVA checker is very small (about 3%) under error-free operating conditions. This can be brought down to negligible levels (0.7%) if an exclusive memory port is provided for the DIVA checker. Also, it is found that the overall performance is insensitive to the latency of the DIVA checker. The overall performance under relatively high error rates (1 error in every 1000 clocks) is quite small (3%). In fact, even under the worst case where the core fails completely, the checker can ‘execute’ the program completely on its own – though the performance in such a scenario would be very low. Thus, the results support all the conclusions.

Describe the potential future works?

• How well does the DVFS application of the DIVA architecture perform compared to other power-saving approaches like Razor, etc.?

• Is this ‘coarse-grained’ fault tolerance better than a more ‘fine-grained’ approach (for e.g. at the flip-flop or logic level) in terms of parameters like area, power, robustness, etc.?

• As the core clock frequency and (hopefully) throughput increases, can a simple DIVA checker keep pace with the additional throughput requirements?