Compiler-Assisted Fault Tolerance

Inexpensive Reliability with the CRAFT and SWIFT Approaches

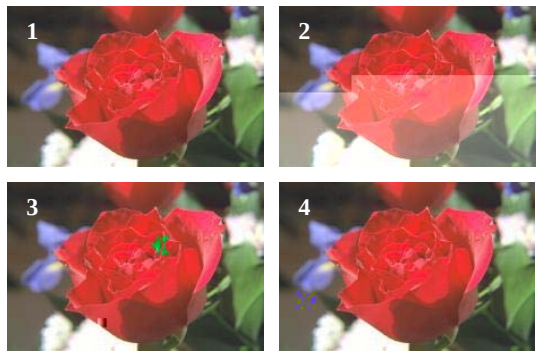
Can you trust results from your computer?

Even with bug-free software running on a validated processor, a single transient fault can make 1+1=3.

George A. Reis Jonathan Chang Neil Vachharajani Ram Rangan David I. August
{gareis,jcone,nvachhar,ram,august}@princeton.edu

Compiler-assisted fault tolerance (CRAFT) reduces the hardware necessary for reliability by introducing redundancy in software. Since processor designers have a fixed budget allotted for fault tolerance, this allows resources to be reallocated to previously unprotected regions. CRAFT minimizes its performance cost by efficiently managing redundancy and reclaiming unused instruction-level resources present during execution. SWIFT, the software-only variant of CRAFT, can add reliability to legacy systems or to those with strict hardware limitations because it requires no hardware support.

Why is this important? To improve performance and reduce power consumption, processor designers employ advances that shrink feature sizes, lower voltage levels, reduce noise margins, and increase clock rates. These advances, however, also make processors more susceptible to transient faults that can affect program correctness. Today, these faults are infrequent but still cause significant failures. In 2000, Sun Microsystems acknowledged that cosmic rays interfered with cache memories and caused crashes in server systems at major customer sites, including America Online, eBay, and dozens of others. Future processors will be more susceptible to lower energy radiation due to minimized voltages and tighter noise margins used in newer manufacturing processes. The density (flux) of cosmic rays are significantly higher at these lower energies than it is at higher energies, thus the transient fault rate will dramatically increase. Predictions state technologies may see a 10^8 increase in the transient fault rate within the next decade.



Images resulting from transient fault injections into jpeg encoder processing source image (1). Outputs range from blatant errors in the DC coefficient (2), to errors localized to a single block (3), to unnoticeable errors (4). With CRAFT, the compiler can provide different protection for different functions based on their effect on output.

Why not just use existing techniques? Fault tolerance can be achieved with redundant hardware, but that is simply too costly in terms of area and performance for most systems. Hardware-only approaches, such as lockstepped or RMT systems, require special hardware to create redundancy plus additional hardware for comparison and deadlock avoidance. Further, hardware-only schemes cannot utilize information at the application level to increase reliability. Such information can be used by CRAFT to protect program regions independently (see the figure above) as well as easily handle precise exceptions and concurrent shared memory systems. In addition, legacy systems can utilize the compiler-only approach (SWIFT) to enable fault tolerance in harsh environments (like high altitudes). SWIFT outperform previous software-only approaches by leveraging already protected ECC memory, reducing checkpoints, and using novel control flow protection mechanisms.

Our approach? Let the compiler protect what it can and let the hardware protect the rest. The CRAFT hybrid approach leverages compiler inserted redundancy to minimize the hardware required. This results in high reliability with minimal additional hardware. During code generation, the compiler duplicates program instructions and inserts comparison instructions at strategic points. During execution, values are computed twice and compared before any differences due to transient faults can adversely affect program output. Using SWIFT to execute the reliability enhanced program without any hardware support yields only a 39% average performance penalty due to the efficient utilization of otherwise unused resources. With minimal hardware, CRAFT achieves increased reliability with half the cost of SWIFT.

More Information:
http://www.liberty-research.org/Research/FT or contact the Liberty Fault Tolerance Team at the addresses above.