Fault-Tolerance Projects at Stanford CRC

Philip P. Shirvani
Nirmal Saxena
Edward J. McCluskey

Center for Reliable Computing
Computer Systems Laboratory
Departments of Electrical Engineering and Computer Science
Stanford University

1999 MAPLD Conference
September 28-30, 1999, Laurel, MD

Copyright © 1999 by the Center for Reliable Computing, Stanford University.
Center for Reliable Computing (CRC)
http://crc.stanford.edu/

- Fault-Tolerance and Adaptive Computing
  - ARGOS project
    - Real space environment experiment
    - Rad-hard v.s. COTS hardware
    - Only software techniques in COTS board
  - ROAR project
    - Reconfigurable hardware
    - Dependable adaptive computing
- Testing
- Design and Synthesis for Testability
 Dependability

- Crucial for Military, Avionics and Aerospace
- Costly Classical Solutions
  - Current systems mostly use old technology
- New Trends
  - Commercial off-the-shelf (COTS) components
    - ARGOS project (ONR)
  - Programmable logic devices (PLDs)
    - ROAR project (DARPA)
The ARGOS Project
http://crc.stanford.edu/projects/argos.html

- Reliable Computing in Space
  - Autonomous navigation and data processing
  - Radiation-hardened components
    - Expensive, old technology, unavailable
- ARGOS Satellite Computing Testbed
  - Rad-hard vs. COTS components
    - Evaluate effectiveness of software-implemented fault tolerance (SIFT)
  - Error data collection in a real space experiment (no simulation or fault injection)
The ARGOS Satellite

- Advanced Research and Global Observations Satellite
- Launch: Feb. 23, 1999
- Orbit: 800 km Altitude, Sun Synchronous, 98° Inclination
- 9 Experiments
  - Including USA (Unconventional Stellar Aspect) experiment of NRL
    - Contains the computing testbed
Computing Testbed

- Radiation-Hardened Board
  - Harris RH3000 radiation-hardened chip set
    - Self-checking pair
  - SOI SRAMs
  - ECC memory

- COTS Board
  - IDT R3081
  - No error detection hardware
    - No ECC
  - SRAM-based FPGA
System Features

- VxWorks Operating System
  - Real-time, multitasking, dynamic linking
- Telemetry
  - Upload programs and data @ 1.1 kbps
  - Download outputs @ 40/128 kbps
- Revolution Time: 101.6 min.
  - 8 min. window for telemetry
- Reprogrammable EEPROMs
  - System ROM and FPGA configuration
- Access to X-Ray Sensor Data
Requirements

- Error Detection
  - Programs to exercise functional units
    - e.g., FFT, sort, compression, ALU test
  - Software-implemented error detection added
  - Maximize error detection coverage

- Error Collection
  - Log type, time, position, etc.
  - Correct transmission of log

- Error Recovery
  - Automatically
Error Detection, Collection & Recovery Software

- ECC (COTS board)
- OS
- Diagnostic
- Profiler
- Collector
- Watchdog
- Computation
- Main Control
- Telemetry
- Ground Program
Design Framework

- Modular Design
  - Utilizes dynamic linking
  - Facilitates module update or repair
  - Efficient use of limited upload bandwidth

- Multitasking
  - Separate task for each module
  - Independent context eases error recovery

- Task Synchronization and Communication
  - Operating system library primitives
  - Machine/OS independent scheme
Main Control Program

- Running the Computations
- Command Interpreter
  - Adding/deleting modules
  - Changing parameters
- Controlling the Watchdog Timer
- Logging Error
  - With or without collector module
- Error Recovery and Restart
  - Mostly automatic
  - No need for interaction with ground
Software-Implemented Error Detection

- Time Redundancy
  - Stutter-Step Mode execution (SSM)
  - Software duplication/TMR
- Control Flow Checking
  - Signature Analysis by Instructions (SAI)
  - Watchdog task and timers
- Other
  - Algorithm-Based Fault Tolerance (ABFT)
  - Assertions
  - Programming practices
Stutter-Step Mode (SSM)

- Duplicate Instructions
  - Master and shadow instructions
- Compare Master and Shadow Results
  - Detect transient errors in computations

```
ADD R3, R1, R2 ; R3 <- R1 + R2
MUL R4, R3, R5 ; R4 <- R3 * R5
ST 0(SP), R4 ; store R4 in location pointed by SP

ADD R3, R1, R2 ; R3 <- R1 + R2 master
ADD R23, R21, R22 ; R23 <- R21 + R22 slave
MUL R4, R3, R5 ; R4 <- R3 * R5 master
MUL R24, R23, R25 ; R24 <- R23 * R25 slave
BNE R4, R24, ErrorHandler ; compare master and slave results
ST 0(SP), R4 ; store master result
ST offset(SP), R24 ; store slave result
```
Signature Analysis by Instructions

- Assigned Signature Analysis Method
  - Unique signature for each basic block
- Interblock Control Flow Checking
  - Correct sequence of blocks followed
- Signature Comparison
  - Pure software
  - No extra hardware
Flow for Adding SSM and SAI

C source \( \rightarrow \) CC (gcc) \( \rightarrow \) Assembly code

Post Processor

Assembly code with EDI \( \rightarrow \) Assembler \( \rightarrow \) Object code
Errors

- Main Type of Errors
  - Radiation-induced transient errors
    - e.g., Alpha particles and cosmic rays
  - Single-Event Upsets (SEUs)
  - Multiple-bit error caused by a single SEU
- Memory Scrubbing
  - Hardware in rad-hard board
  - Software in COTS board
Fault-Tolerance in FPGAs

- SRAM-based FPGAs on COTS Board
  - Configuration ROM is reprogrammable
- Spares Replace Faulty Blocks
  - Configure for built-in self-test (BIST)
  - Run test to find faulty block
  - Reprogram isolating faulty block
- Adaptive Computing
- Experiments on Upset Rate in FPGAs
Status and Future Work

- Initial Test of Programs Completed
- Long Term Testing in Progress
  - Collect statistically significant number of errors
  - Fine tune the software according to results
- Future Research
  - More fault tolerance techniques
  - FPGA experiments
Summary

- Fault-Tolerant Computing Important in Space
- Techniques
  - Radiation hardening
  - ECC, duplication, etc.
  - Software techniques
- Great Opportunity to Collect Valuable Data
- Compare Hardware and Software FT Techniques
Acknowledgments

- This work was supported in part by the Ballistic Missile Defense Organization, Innovative Science and Technology (BMDO/IST) Directorate and administered through the Department of the Navy, Office of Naval Research under Grant Nos. N00014-92-J-1782 and N00014-95-1-1047.

- This is a collaborative project with the NRL USA experiment group
  - Kent Wood, principal investigator
The ROAR Project

Reliability Obtained by Adaptive Reconfiguration

Dependable, Adaptive Computing Systems (ACS)
In Collaboration with University of Texas, Austin
Sponsored by DARPA ACS Program

- Economical Deployment of FT Technology
  - In commercial applications
- Common ACS Architecture
  - High-performance, and
  - High-reliability
New Design Diversity Metric

- Common-Mode Failures
  - Diverse designs better

- Old Definition
  - Qualitative and lacks design insight
    - No framework for comparison

- Our New Definition
  - Quantitative
  - Framework for comparison
  - Guidance for synthesis process
Cost of Diversity

- Traditional
  - Manufacturing cost
  - Development cost

- ACS
  - Enabling technology for diversity
  - Field programmability of FPGAs
    - No manufacturing cost
  - Development cost
    - Mitigated by automation tools
Fault-Tolerance with Multithreaded Processors

Observations
- Common architecture for FT & high performance
- Economical deployment of FT in general purpose processors

Shirvani
Multithreading Emulation Experiment

- Experimental Code
  - LZW compress algorithm
- Explicit Threading in Source
  - Two threads emulated
  - Replicated data structures
  - Replicated control
- Performance Measurement
  - Fault-tolerance SlowDown
  - Instruction throughput SpeedUp

\[ \text{SlowDown} = \frac{ET_2}{ET_1} \]

\[ \text{SpeedUp} = 2 \frac{ET_1}{ET_2} \]

<table>
<thead>
<tr>
<th>Processor</th>
<th>OS</th>
<th>One Thread</th>
<th>Two Threads</th>
<th>Slow Down</th>
<th>Speed Up</th>
</tr>
</thead>
<tbody>
<tr>
<td>PowerPC 604e</td>
<td>MkLinux</td>
<td>9.37</td>
<td>11.33</td>
<td>1.21</td>
<td>1.65</td>
</tr>
<tr>
<td>233 MHz</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AMD K6</td>
<td>Linux</td>
<td>7.10</td>
<td>8.69</td>
<td>1.22</td>
<td>1.63</td>
</tr>
<tr>
<td>233 MHz</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UltraSPARC -II</td>
<td>Solaris</td>
<td>8.00</td>
<td>9.00</td>
<td>1.13</td>
<td>1.77</td>
</tr>
<tr>
<td>300 MHz</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For Three Threads (Multithreaded TMR) SlowDown \(~ 1.6\)
Different TMR Schemes

Non-Multithreaded Software TMR (NMT-STMR)
- N cycles
- Error Rate = 1-p
- Context Switch
- Voting

Multithreaded Software TMR (MT-STMR)
- 3N/Su

Multiprocessor TMR (MP-TMR)
- N cycles
- Processor 1
- Processor 2
- Processor 3
Reliability Analysis

Error Rate = $10^{-6}$/cycle
Speed Up = 2.2

Cut-off with respect to Simplex
- NMT-STMR: 50K Cycles
- MT-STMR: 300K Cycles
- MP-TMR: 700K Cycles

Reliability

N x $10^3$ Clock Cycles $\rightarrow$
ACS Robotics

- Control Computer (Old Way)
  - Implements PID control algorithm
  - C program running on a CPU
- FPGA Coprocessor (New Way)
  - Controller in FPGA hardware
    - With FT features
    - Faster feedback response
- Experimental Results
  - Pentium CPU (100 MHz)
    - 2670 ns (non-optimized)
    - 1170 ns (optimized)
  - FPGA implementation
    - 600 ns (Quickturn emulator 1.6MHz)
ACS Robotics

- New FT Techniques
  - Multithreaded control
- Median Voting
- Precision vs. FT

### Original

<table>
<thead>
<tr>
<th>Threads</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td># of Bits</td>
<td>32</td>
<td>32</td>
<td>32</td>
</tr>
<tr>
<td># of Regfiles</td>
<td>0</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Logic Area</td>
<td>54255</td>
<td>36583</td>
<td>40852</td>
</tr>
<tr>
<td>Pipeline Stages</td>
<td>1</td>
<td>9</td>
<td>11</td>
</tr>
<tr>
<td>$f$ (MHz)</td>
<td>1.6</td>
<td>2.5</td>
<td>2.5</td>
</tr>
</tbody>
</table>

### Multithreading

Thread 2

- Register File
- Instruction Memory (VLIW)
- ADD
- ADD
- MUL

- # of Bits: 32
- # of Regfiles: 2
- Logic Area: 36583
- Pipeline Stages: 9
- $f$ (MHz): 2.5
ACS Robotics

- **Median Voting vs. Majority Voting**
- **Fault Injection Experiments**
- **Different Precision**

### Error Free Result

<table>
<thead>
<tr>
<th>Bits</th>
<th>32</th>
<th>24</th>
<th>16</th>
<th>12</th>
<th>8</th>
<th>Does not work</th>
</tr>
</thead>
<tbody>
<tr>
<td>Robot</td>
<td>16</td>
<td>12</td>
<td>8</td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Control</td>
<td>16</td>
<td>12</td>
<td>12</td>
<td>9</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Overshoot</th>
<th>43%</th>
<th>42%</th>
<th>41%</th>
<th>47%</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Steady state error</th>
<th>1.0%</th>
<th>1.0%</th>
<th>2.9%</th>
<th>3.6%</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Stable time (cycles)</th>
<th>&lt;5%</th>
<th>15</th>
<th>15</th>
<th>31</th>
<th>37</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;2%</td>
<td>27</td>
<td>27</td>
<td>---</td>
<td>---</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Response time (cycles) (90%)</th>
<th>1</th>
<th>1</th>
<th>2</th>
<th>5</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Prob of bit flip</th>
<th>$10^2$</th>
<th>$10^{-3}$</th>
<th>$10^4$</th>
<th>$5 \times 10^{-6}$</th>
<th>$10^{-5}$</th>
<th>$10^{-6}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 thread</td>
<td>98.9%</td>
<td>98.4%</td>
<td>95.2%</td>
<td>95.2%</td>
<td>39.6%</td>
<td>0%</td>
</tr>
<tr>
<td>3 threads traditional voting</td>
<td>98.9%</td>
<td>97.9%</td>
<td>77.1%</td>
<td>38.5%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>3 threads median voting</td>
<td>98.9%</td>
<td>97.3%</td>
<td>79.7%</td>
<td>45.5%</td>
<td>0%</td>
<td>0%</td>
</tr>
</tbody>
</table>
ACS LZ-77 Compression

- Low Overhead Error Detection
  - Idle PEs utilized for error detection
- More Efficient than Duplex
- Quickturn Emulation
  - 28 Xilinx 4013 chips
  - 4 Xilinx 4036 chips
- Performance
  - Negligible delay penalty
  - Throughput
    - One source char/cycle
    - Speed up 32-100x over LZ on a 300MHz UltraSPARC-II
ACS Fault-Tolerant FFT

- Concurrent Error Detection (CED)
  - Weighted input checksum
  - Weighted output checksum
  - Checksum comparison

- Features
  - 100% error coverage
  - Area overhead
    - 50% less than prior art
  - Negligible delay penalty

- 8-point FFT
  - 6 Xilinx 4013 FPGAs
  - 7 FPGAs with CED
Diagnosing Interconnect Faults

- Configuration-Dependent Interconnect Test and Diagnosis
  - Rapidly test interconnects used in a particular config.
  - Can be done each time system is reconfigured.
  - Detect and locate stuck-at and bridging faults.
  - Can reconfigure to avoid faulty hardware.

- Form Test Configuration
  - Only change function of CLBs to form pseudo scan paths.
  - Interconnects kept intact.
    - No place or routing required.
    - Test physical interconnects used in original config.
  - Perform walking-1 test.
    - Make each net a 1 while all others are 0.
# Number of Reconfigurations for Fault Diagnosis

<table>
<thead>
<tr>
<th>Circuit</th>
<th>FPGA</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
<td><strong>Pls</strong></td>
</tr>
<tr>
<td>C2670</td>
<td>233</td>
</tr>
<tr>
<td>C3540</td>
<td>50</td>
</tr>
<tr>
<td>C5315</td>
<td>178</td>
</tr>
<tr>
<td>C6288</td>
<td>32</td>
</tr>
<tr>
<td>C7552</td>
<td>207</td>
</tr>
<tr>
<td>s1423</td>
<td>17</td>
</tr>
<tr>
<td>s5378</td>
<td>35</td>
</tr>
<tr>
<td>s9234</td>
<td>36</td>
</tr>
<tr>
<td>s13207</td>
<td>62</td>
</tr>
<tr>
<td>s15850</td>
<td>77</td>
</tr>
</tbody>
</table>
Summary

- New Dependable ACS Architecture
  - High performance and high reliability
  - Low cost
- Diversified Designs
  - New ACS opportunity
- New CED and Test Techniques
- Fault-Tolerance Using Multithreading
  - Processor and configurable logic