Autonomous, On-board Processing for Sensor Systems: Initial Fault Tolerance and Autonomy Results

Matthew French, JP Walters, Mark Bucciero, Ken Zick
– USC / ISI
Tom Flatley – NASA GSFC
June 23rd, 2010
Field Programmable Gate Arrays (FPGAs) provide near Application Specific Integrated Circuit (ASIC) performance while being reprogrammable

- Resource Multiplexing
  - Multi-mission, multi-sensor
- Mission Obsolescence
  - Update Algorithms
- Design Flaws
  - Correct in Orbit

Static Random Access Memory (SRAM) based FPGAs are now common in space based systems

- Research such as that on the Reconfigurable Hardware in Orbit (RHinO) NASA AIST-03 project developed Radiation Hardening By Software (RHBSW) techniques to mitigate Single Event Upsets in commercial grade devices (COTS)
- 10-100x Processing Performance over Anti-fuse FPGAs
FPGAs have evolved, becoming heterogeneous
— PowerPC processors, Ethernet cores, Giga-bit transceivers

Legacy features (known mitigation techniques)

New features

FPGA Embedded PowerPC outperforms radiation hardened RISC processors

Can RHBSW techniques be developed for new Hard IP Resources? How can these features be leveraged to address autonomy?
A-OPSS Technology Roadmap

Applications

Technology Foundation

Core Fault Tolerance Technology Development

- ISS SpaceCube 1.0 Flight Test
- Radiation Beam Testing
- Software Fault Injection

Autonomous Hyperspectral Imaging

Increasing TRL
**Key Features:**
- **2 COTS Xilinx FPGAs**
  - 4 Total PowerPCs
- **Radiation Hardened Microcontroller**
Existing Embedded PPC Fault Tolerance Approaches

Problem: PowerPC state is not readable from the bitstream like all traditional FPGA circuitry
  • Configuration scrubbing techniques have limited value
  • Fault injection / emulation not feasible by this method

Quadruple Modular Redundancy
  • 2 Devices = 4 PowerPCs
  • Vote on result every clock cycle
  • Fault detection and correction
  • ~300% Overhead

Dual Processor Lock Step
  • Single device solution
  • Error detection only
  • Checkpointing and Rollback to return to last known safe state
  • 100% Overhead
  • Downtime while both processors rolling back

New fault tolerance techniques and error insertion methods must be researched.
Observations

- **Traditional Redundancy Techniques have increasing overhead**
  - PowerPC has ~500x smaller cross section than FPGA
  - 1 fault / 50 days
  - 1 fault / 2 x 10^15 clock cycles

- **Science Applications keep little ‘state’**
  - Streaming computations
  - Few sensitive constants to protect
  - Data errors ‘flush’

- **High Performance Computing community has similar problem**
  - 1000’s of nodes, running for days to weeks
  - A node will fail over runtime
  - HPC community does not use TMR
  - Too many resources for already large, expensive systems
  - Power = $,

- **HPC relies more on periodic checkpointing and rollback**

Cray HPC System
A-OPSS is developing a fault mitigation system of techniques

Sub-system Level Mitigation
- Relies on supporting radiation hardened devices
- High fault type coverage
- Slow response time (up to seconds)
- Low overhead

Application Level Mitigation
- Routines that can be inserted into application code
- Processor mitigates self

Register Level Mitigation
- Quick response time (clock cycles)
- High overhead

Approach: Focus on Sub-system level first, and tune for reliability performance
Heartbeats

- Sent from PowerPC to Radiation Hardened Controller to update status
- Sent at regular intervals
- Radiation Hardened Controller can rollback or restart PowerPC if fault occurs

Heartbeat Contents

```c
// On a Timer Interrupt
msg[0] = (PPC_ID<<4) | RAD_HARD_ID;
msg[1] = heartbeat_number++;
msg[2] = HEARTBEAT_TYPE;
msg[3] = DATA_LENGTH_ZERO;
Send_Message(msg);
```
Checkpoint and Rollback

- PowerPC periodically saves key application variables and state to Radiation Hardened Controller.
- If PowerPC failure occurs, Rollback allows PowerPC to rewind to last known good operational state avoiding vast recomputation.
- If severe PowerPC error occurs, computation can be restarted on another PowerPC node.
### Control Flow Assertions
- PowerPC Code tagged with signatures
- During execution, signatures checked against expected values
- If mismatch, PowerPC sends message to Radiation Hardened Controller for Rollback

**Tagged Code**

```
ES_1 = ES_1 ^ 01;
x = 50;
if (condition == 1) {
    ES_1 = ES_1 ^ 010;
    new_x = x - 5;
} else{
    ES_1 = ES_1 ^ 010;
    New_x = x - 3;
}
ES_1 = ES_1 ^ 0100;
if (ES_1 != 0111) error();
z = new_x - x;
```
A-OPSS vs Traditional Mitigation Preliminary Results

- A-OPSS approach leverages additional hardware for useful computation
- Heartbeats and assertions cause minimal overhead
- Checkpoints are taken according to the expected upset rate

Comparison of Fault Tolerance (FT) Strategies

- Duplication
- TMR: Triple Modular Redundancy
- QMR: Quadruple Modular Redundancy

Computational resources saved can be used for autonomous operations

Dup: Duplication, TMR: Triple Modular Redundancy, QMR: Quadruple Modular Redundancy
Memory Sentinel and Injection System

Fault Injection emulator for PowerPC
Injects faults directly onto executing hardware
Estimated 99% sensitive bit coverage
Enables long tests runs >10,000’s injections
Available for government use
Published in 2011 IEEE Field Customizeable Computing Machines conference
Software Injection Results

Value Added

Quickly recover from locked processor (reset)
Lost computation can be tuned to mission requirements.
Currently investigating data errors: can we learn anything from failure characteristics?
Checkpointing and rollback also allows speculative execution. Will be used for autonomy.

<table>
<thead>
<tr>
<th>Error classification</th>
<th>Before (no FT)</th>
<th>After (with FT)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unrecoverable crash/hang</td>
<td>9%</td>
<td>0%</td>
</tr>
<tr>
<td>Error recovery via processor reset</td>
<td>n/a</td>
<td>2%</td>
</tr>
<tr>
<td>Silent data corruption error</td>
<td>5%</td>
<td>4%</td>
</tr>
<tr>
<td>Error recovery via rollback &amp; restart</td>
<td>n/a</td>
<td>9%</td>
</tr>
<tr>
<td>No error</td>
<td>86%</td>
<td>85%</td>
</tr>
</tbody>
</table>

Total 96% data error free results after fault injection using radiation hardening by software.
Radiation Testing Plans

**Application level mitigation driving radiation experimental setup**

- Traditional approaches would saturate device, causing unrealistic rate of errors per application execution control loop

**Application level fault mitigation test plan**

- Testing at Naval Research Laboratory laser facility
  - Can control error injection rate
- NASA GSFC Radiation Effects Group supporting efforts
- Testing scheduled for July
MISSE7/8 In-orbit Testing

**Purpose**
- On-orbit “Rad Hard By Software” test platform
- Operated by NRL / NASA Langley
- Collect radiation performance
- Collaborate
  - Demonstrate partners’ technology on-orbit

**Capabilities**
- Two SpaceCube processor cards operated by NASA GSFC
  - Independent experiment units
- On-orbit reconfiguration
  - Uplink compressed data files from the ground
    - new bit files, new PPC code, new microcontroller code, new data files

Integrated with NASA to create an on-orbit test of software fault tolerance methods

Upload in progress – ETA August
Developing demonstration of on-board processing for representative HyspIRI applications

Increased computational yield from RHBSW enables capability to perform look-ahead computations

- Can rapidly send time sensitive data to decision makers
A-OPSS enables spiral development, allowing path to produce rapid prototypes and gradually increase performance as funding and schedule allow.

System on chip architectures provide best of both worlds:
- Branching algorithms can operate on PowerPC
- Mass parallelism can be achieved on streaming functions
Software fault emulation results promising

- No hard failures
- 96% data correct with no data mitigation techniques added
- Currently reviewing data error types

Radiation and In-space data eminent

Autonomy

- Architecture lends itself favorably for high performance autonomous processing