Autonomous, On-board Processing for Sensor Systems: High Performance Fault Tolerant Techniques

Matthew French, JP Walters, Mark Bucciero – USC / ISI

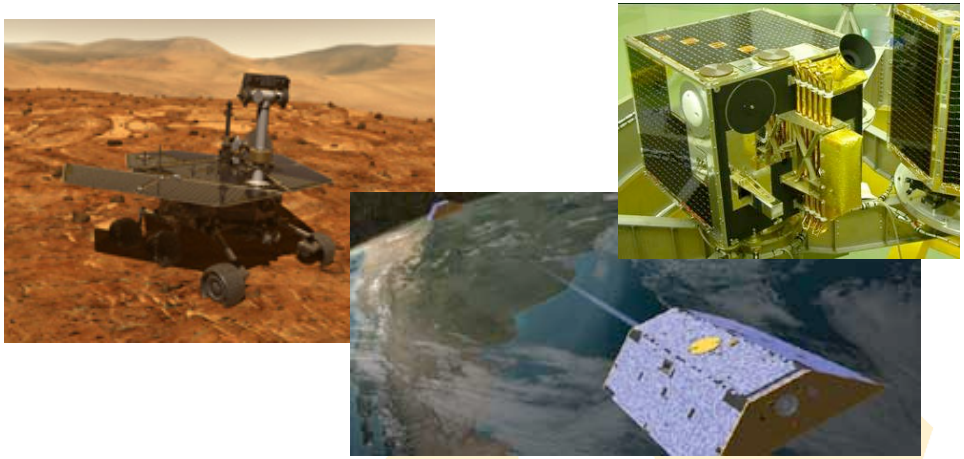
Tom Flatley – NASA GSFC

June 23rd, 2010

FPGAs in Space Background

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 Today

USC Viterbi School of Engineering

FPGAs have evolved, becoming heterogeneous

— PowerPC processors, Ethernet cores, Giga-bit transceivers

Legacy features (known mitigation techniques)

New features

		Virtex-5 FXT FPGA Platform Optimized for Embedded Processing with High-Speed Serial Connectivity (1.0 Volt)				
	Part Number	XC5VFX30T	XC5VFX70T	XC5VFX100T	XC5VFX130T	XC5VFX200T
	EasyPath™ Cost Reduction Solutions [1]	—	XCE5VFXT70T	XCE5VFXT100T	XCE5VFXT130T	XCE5VFXT200T
Logic Resources	Slices [2]	5,120	11,200	16,000	20,480	30,720
	Logic Cells [3]	32,768	71,680	102,400	131,072	196,608
	CLB Flip-Flops	20,480	44,800	64,000	81,920	122,880
Memory Resources	Maximum Distributed RAM (Kbits)	380	820	1,240	1,580	2,280
	Block RAM/FIFO w/ECC (36Kbits each)	68	148	228	298	456
	Total Block RAM (Kbits)	2,448	5,328	8,208	10,728	16,416
Clock Resources	Digital Clock Managers (DCM)	4	12	12	12	12
	Phase Locked Loop (PLL)/PMCD	2	6	6	6	6
I/O Resources [4]	Maximum Single-Ended Pins	360	640	680	840	960
	Maximum Differential I/O Pairs	180	320	340	420	480
	I/O Standards	HT, LVDS, LVDSEXT, RSDS, BLVDS, ULVDS, LVPECL, LVCMOS33, LVCMOS25, LVCMOS18, LVCMOS15, LVTTL, PCI33, PCI66, PCI-X, GTL, GTL+, HSTL I (1.2V,1.5V,1.8V), HSTL II (1.5V,1.8V), HSTL III (1.5V,1.8V), HSTL IV (1.5V,1.8V), SSTL2 I, SSTL2 II, SSTL18 I, SSTL18 II				
Embedded [5] Hard IP Resources	DSP48E Slices	64	128	256	320	384
	PowerPC® 440 Processor Blocks	1	1	2	2	2
	PCI Express Endpoint Blocks	1	3	3	3	4
	10/100/1000 Ethernet MAC Blocks	4	4	4	6	8
	RocketIO™ GTP Low-Power Transceivers	—	—	—	—	—
	RocketIO™ GTX High-Speed Transceivers	8	16	16	20	24

Xilinx V5FXT Datasheet

FPGA Embedded PowerPC outperforms radiation hardened RISC processors

Processor	Mongoose V	RAD6000	RAD750	Virtex4 PPC405	Virtex 5 PPC440
Dhrystone MIPS	8	35	260	900	2,200

Can RHBSW techniques be developed for new Hard IP Resources?
How can these features be leveraged to address autonomy?

Existing Embedded PPC Fault Tolerance Approaches

Problem: PowerPC state is not readable from the bitstream like all traditional FPGA circuitry

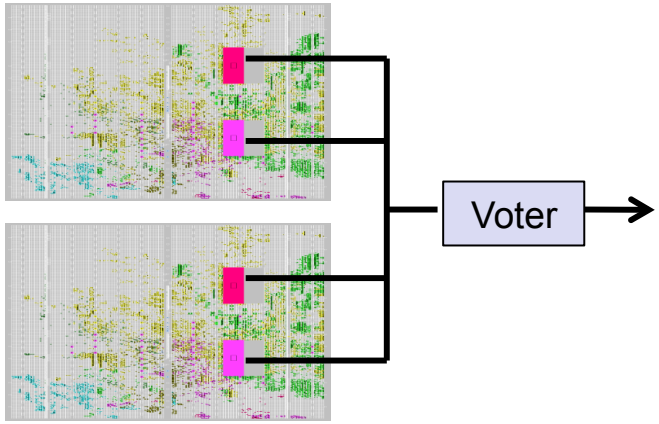
- Configuration scrubbing techniques have limited value
- Fault injection / emulation not feasible by this method

Quadruple Modular Redundancy

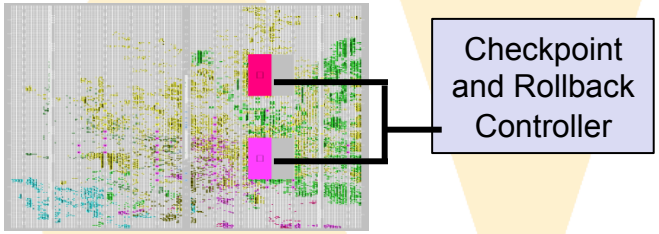
- 2 Devices = 4 PowerPCs
- Vote on result every clock cycle
- Fault detection and correction
- ~300% Overhead

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



QMR Approach



Dual Lock Step Approach

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

Autonomous, On-board Processing for Sensor Systems

PI: Matthew French, USC/ISI

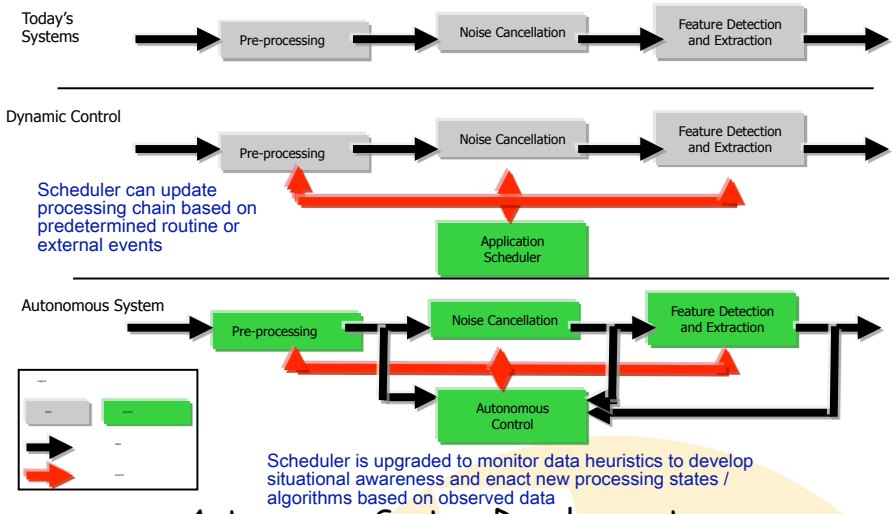
USC Viterbi School of Engineering

Objective

Fuse high performance reconfigurable processors with emerging fault-tolerance & autonomous processing techniques for a 10-100x decrease in processing time.

- This means more science experiments conducted per day & more thorough, timely analysis of captured data.
- Addresses the ability to quickly react & adapt processing or mission objectives in real-time, by combining autonomous agents with reconfigurable computing.
- Enables Autonomous On-board Processing for Sensor Systems (A-OPSS), via a tool-suite that generates a run-time system for sensor systems to autonomously detect changes in collected data & tune processing in a controlled manner to adapt to unforeseen events.

Decadal Survey Missions: Primary - DESDynl, **HyspIRI**, GEO-CAPE; Secondary – SMAP, SWOT



Autonomous System Development

Approach

Phase I: Fault Tolerance

- Develop HPC fault techniques and tools for Virtex4FX
- Demonstrate on SAR application

Phase II: Single Node Autonomy

- Extend autonomous architecture to SpaceCube
- Demonstrate node level adaptation on dynamic scenarios

Phase III: Multi-layer Autonomy

- Extend architecture to system level (ground, other nodes)
- Demonstrate end-to-end adaptation

Co-I's/Partners

Tom Flatley/GSFC

Key Milestones

√ Initial documentation	5/1/09
√ Manual FT application demo	10/15/09
Automated FT application demo	3/30/10
Autonomous agent simulation demo	10/15/10
Autonomy hardware demo	3/30/11
End-to-end autonomy demo	10/15/12
End-to-end multi-node autonomy demo	3/30/12
Final documentation & report	

TRL_{in} = 3 $TRL_{current}$ = 3

Information Sciences Institute

Earth Science Technology Office

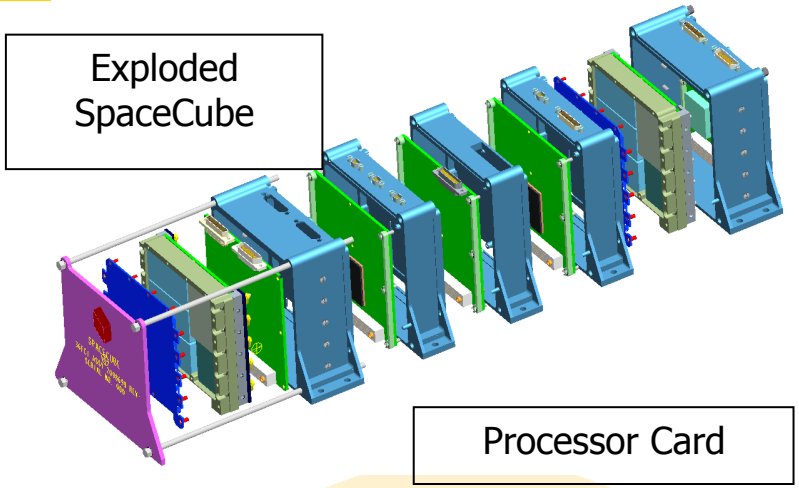
NASA HARDWARE and APPLICATIONS

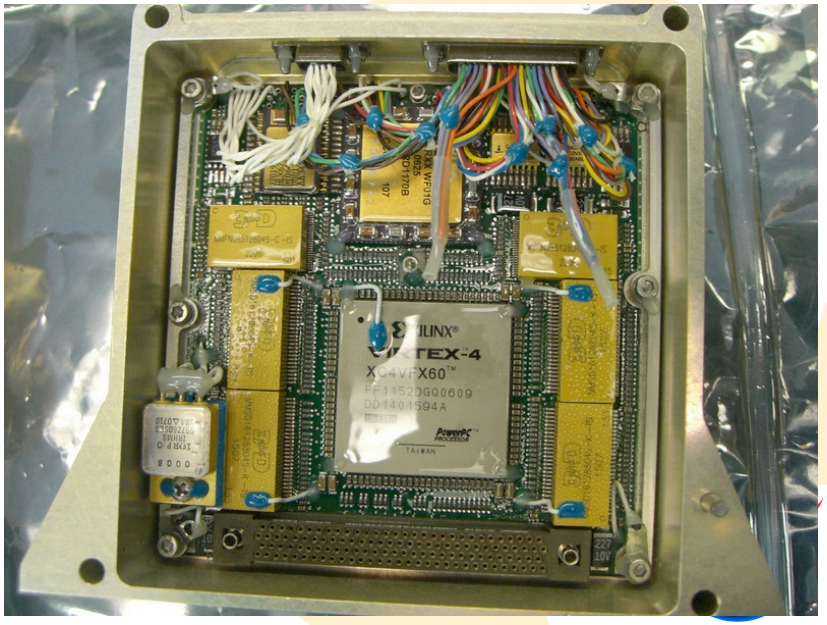
SpaceCube 1.0

USC Viterbi School of Engineering

SpaceCube Technology

- Multi-processing, reconfigurable platform
  - *2 Xilinx V4FX60 devices*
- Low cost, light weight, moderate power
- Custom stackable architecture
- >10x performance increase over existing flight processors
- Mechanical:
  - *7.5-lbs, 5"x5"x7"*
- Power:
  - *37W (HST RNS Application)*

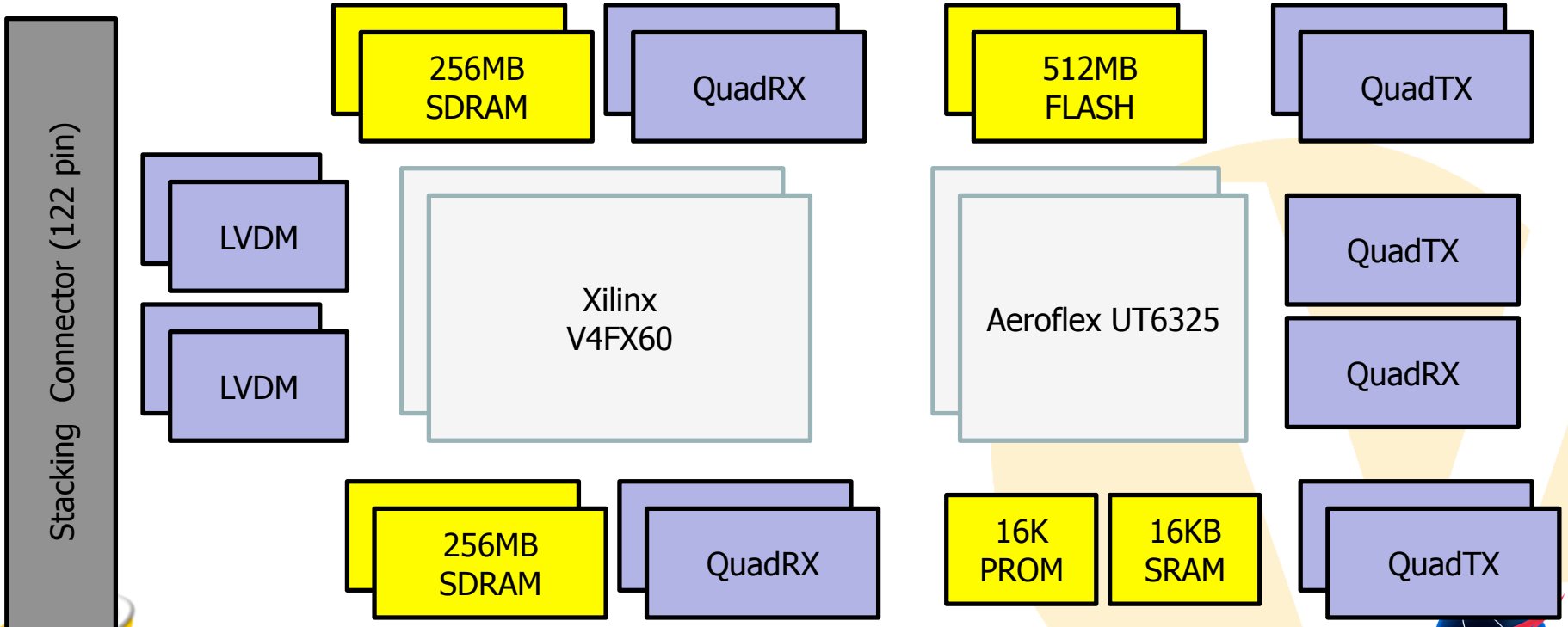




Information Sciences Institute

SpaceCube 1.0 Processor Card Details

General:	**4”x4” card, Back-to-Back FPGAs (x2), 7W typical power**
Memory:	**1GB SDRAM, 1GB Flash, 16KB SRAM, 16KB PROM**
Interfaces:	**20 bi-dir differential signals, JTAG**
Backplane:	**Power, 42 single-ended, 8 LVDM, 2 I2C, POR**



SpaceCube on MISSE7

USC Viterbi School of Engineering

Purpose

- On-orbit "Rad Hard By Software" test platform
- Collect radiation performance
- Collaborate
  - *Demonstrate partners' technology on-orbit*

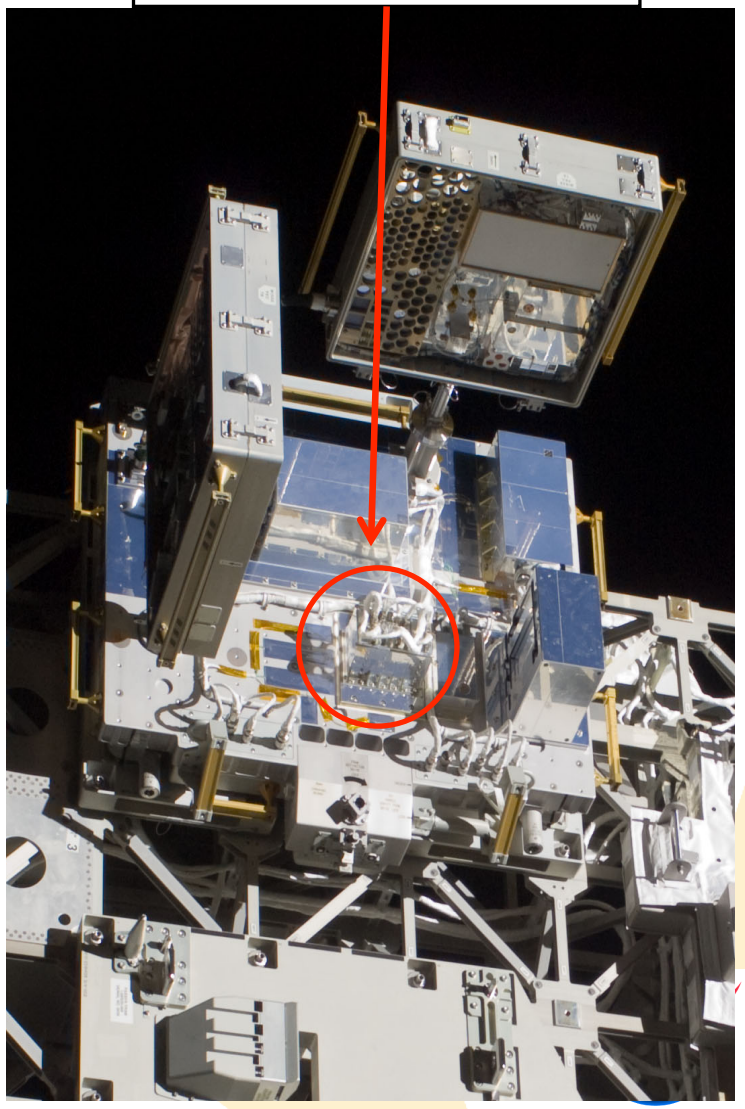
Capabilities

- Two SpaceCube processor cards
  - *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*
- Bandwidth (small but functional)
  - *With dedicated access to MISSE7 C&DH box*
    - *Uplink 106 bytes every 3 sec (~35 bytes/sec)*
    - *8hrs to uplink 1MB*
    - *Downlink 1024 bytes every 3sec (~341 bytes/sec)*

Flight test opportunities available for A-OPSS technology

Information Sciences Institute

SpaceCube on MISSE-7 experiment aboard the ISS

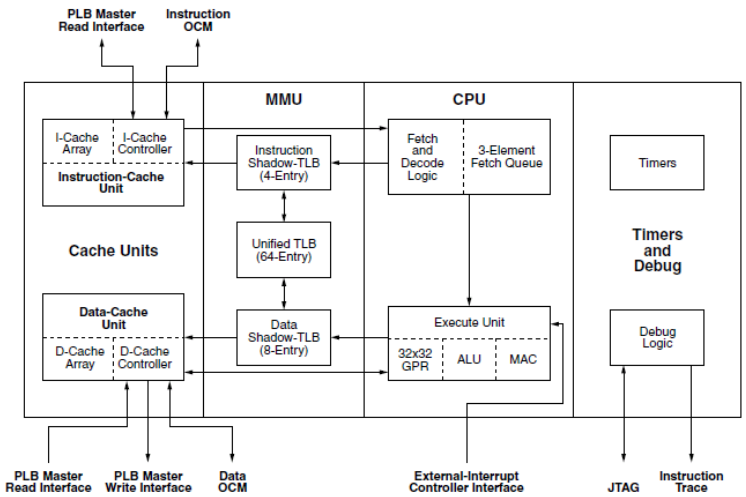


PowerPC Sensitive Cross Section Estimate

USC Viterbi School of Engineering

Do not have full visibility of PowerPC architecture, however good estimate can be made from data sheets

Feature	Size	Fault Injection Method	Comments
Instruction Cache	16 KB +64 control	Beam	
Data Cache	16 KB + 64 control	Beam	
General Purpose Register Set	32 x 32bit	SPFI, Beam	
Special Purpose Register Set	32 x 32bit	SPFI, Beam	OS dependant
Execution Pipeline	10 x 32bit	SPFI?, Beam	
ALU / MAC	~1,200 bits	Beam	
Timers	3x 64bit	SPFI?, Beam	
MMU	72 x 68bits	NA	OS dependant
Misc	1024	Beam	
Total	42,288 bits		36k w/ no OS



PowerPC 405 Functional Diagram

In comparison, Virtex4FX60 bitstream is 21,322,496 bits, or over 500x larger

Information Sciences Institute

Interferometer Synthetic Aperture Radar (SAR)

USC Viterbi School of Engineering

Simulates a synthetic "aperture" or antenna using the satellite's flight path

- Combine multiple radar images into a higher resolution result

InSAR used to detect

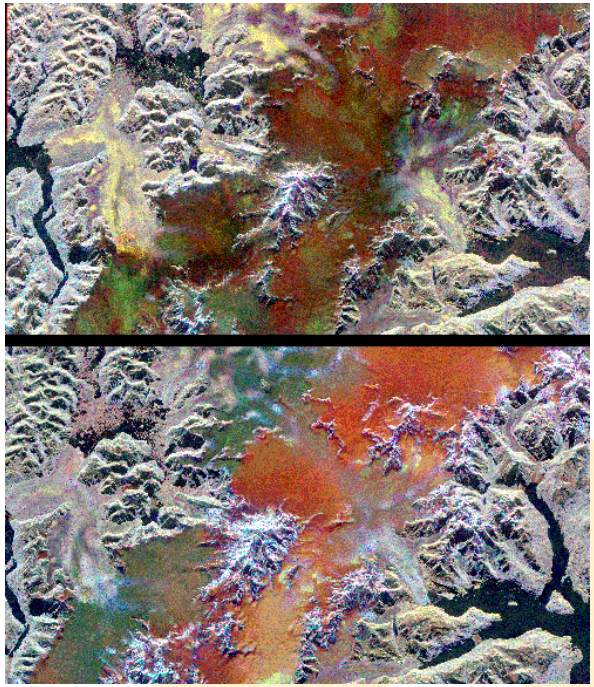
- Surface deformation
- Ice sheet dynamics
- Ecosystem structure

DESDynl Decadal Mission instrument

- L band
- 35m resolution
- 140 Mbps data rate

Science benefits

- Increase in public health and safety due to decreased exposure to tectonic hazards
- Response of ice sheets to climate change
- Effects of changing climate and land use on species habitats and CO2



Spaceborne Imaging Radar-C/X-band Synthetic Aperture radar image demonstrating ability to detect climate-related changes on the Patagonian ice fields in the Andes Mountains of Chile and Argentina. The images show nearly the same area of the south Patagonian ice field imaged during two space shuttle flights in 1994 conducted five-and-a-half months apart. Changes in color represent changes in glacier density.

Image courtesy of NASA JPL

Information Sciences Institute

Hyperspectral Imaging

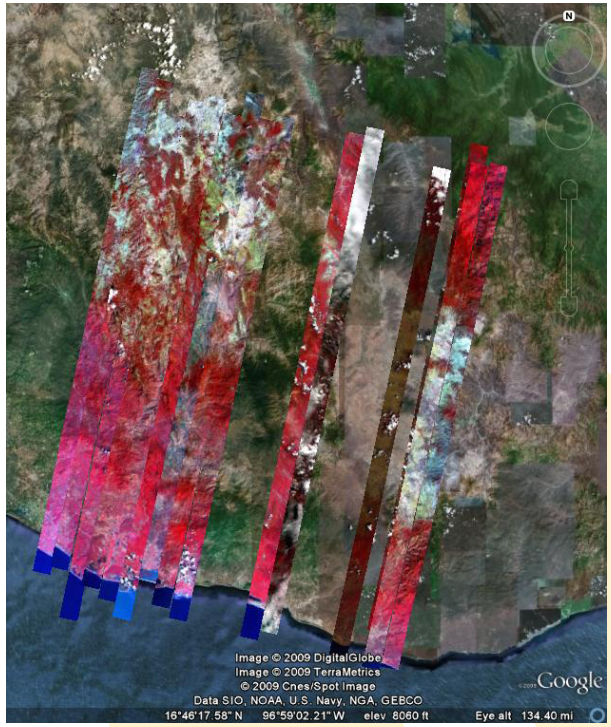
Images hundreds of frequency bands inside and outside of human visual system

HyspIRI Decadal Survey Mission

- Hyperspectral Visible ShortWave InfraRed (VSWIR) Imaging Spectrometer
  - *Range 380 to 2500 nm in 10 nm bands*
  - *60 m sampling*
  - *804 Mbps data generated*
- 15 MBPS downlink
- Onboard processing and autonomous prioritization of data product transmission likely needed

Science benefits

- Changes in vegetation type and deforestation
- Volcanic eruption and landslide forecasting
- Improved natural resource exploration



HyspIRI Test Image, courtesy NASA HyspIRI Science Workshop

USC

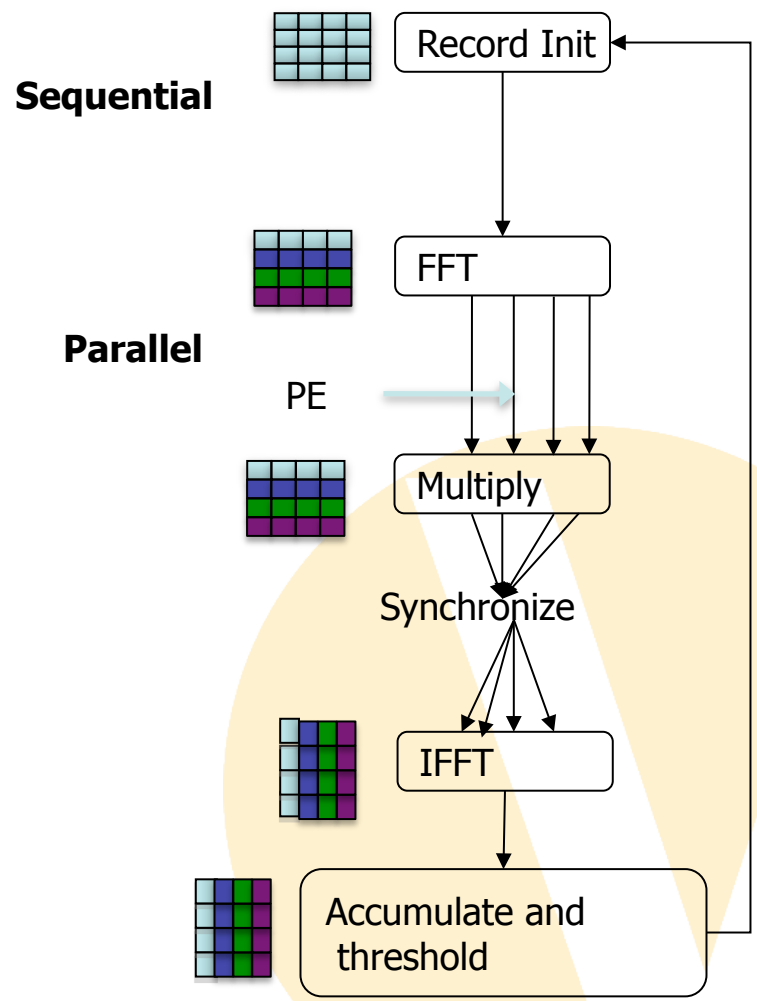
USC Viterbi
School of Engineering

SOFTWARE DEVELOPMENT

ISI
Information Sciences Institute

SAR Application Parallelization

- **SAR is highly parallel at the kernel level**
- **Perform the sequential computation at the outset**
- **Parallelize the loops**
  - Synchronize as necessary



Caching With Dual Processor

Primary challenge

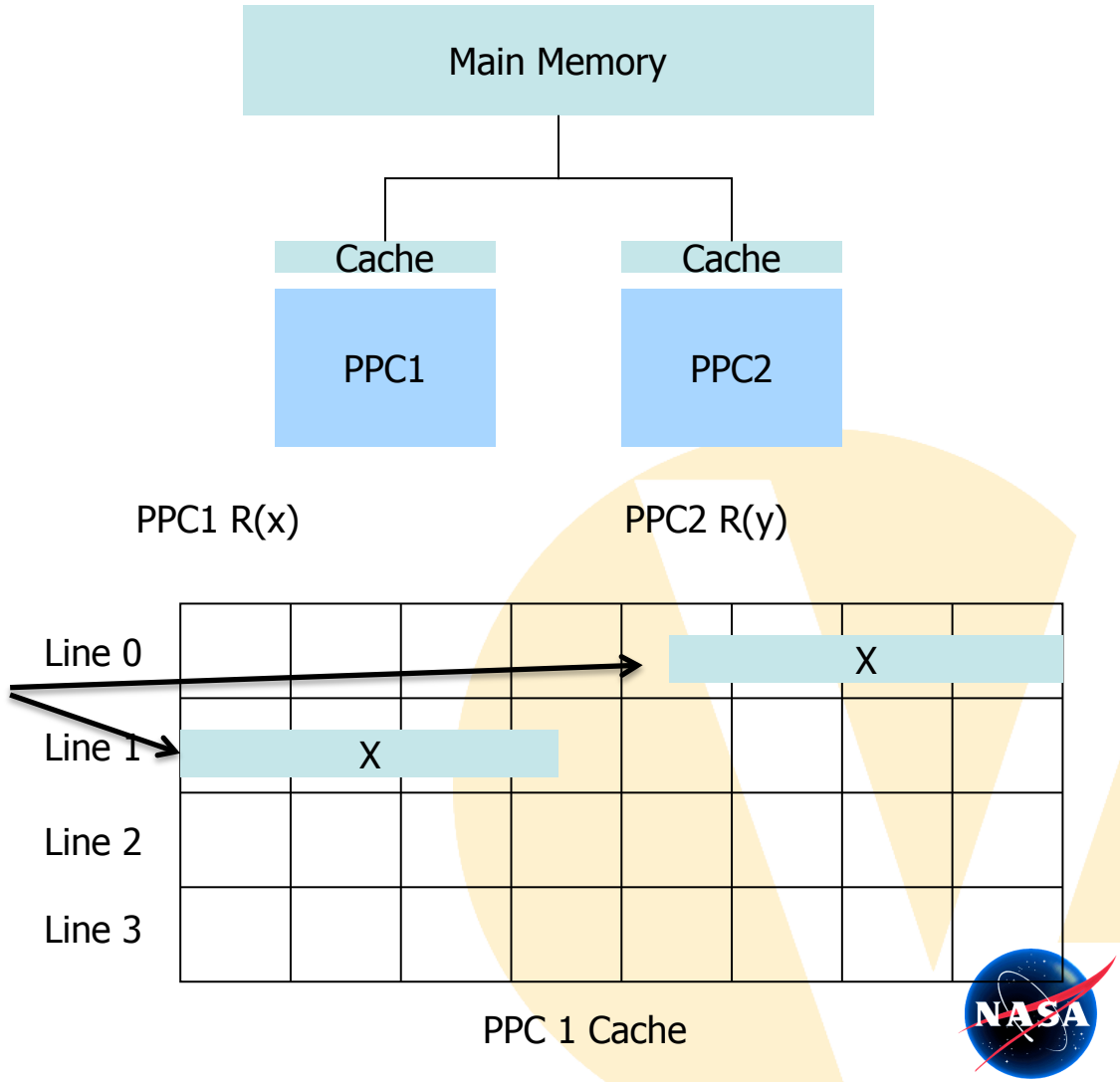
- Cache coherence
- PPCs are not "dual core."
- No hardware manages memory accesses and maintains synchronization

Write-back cache

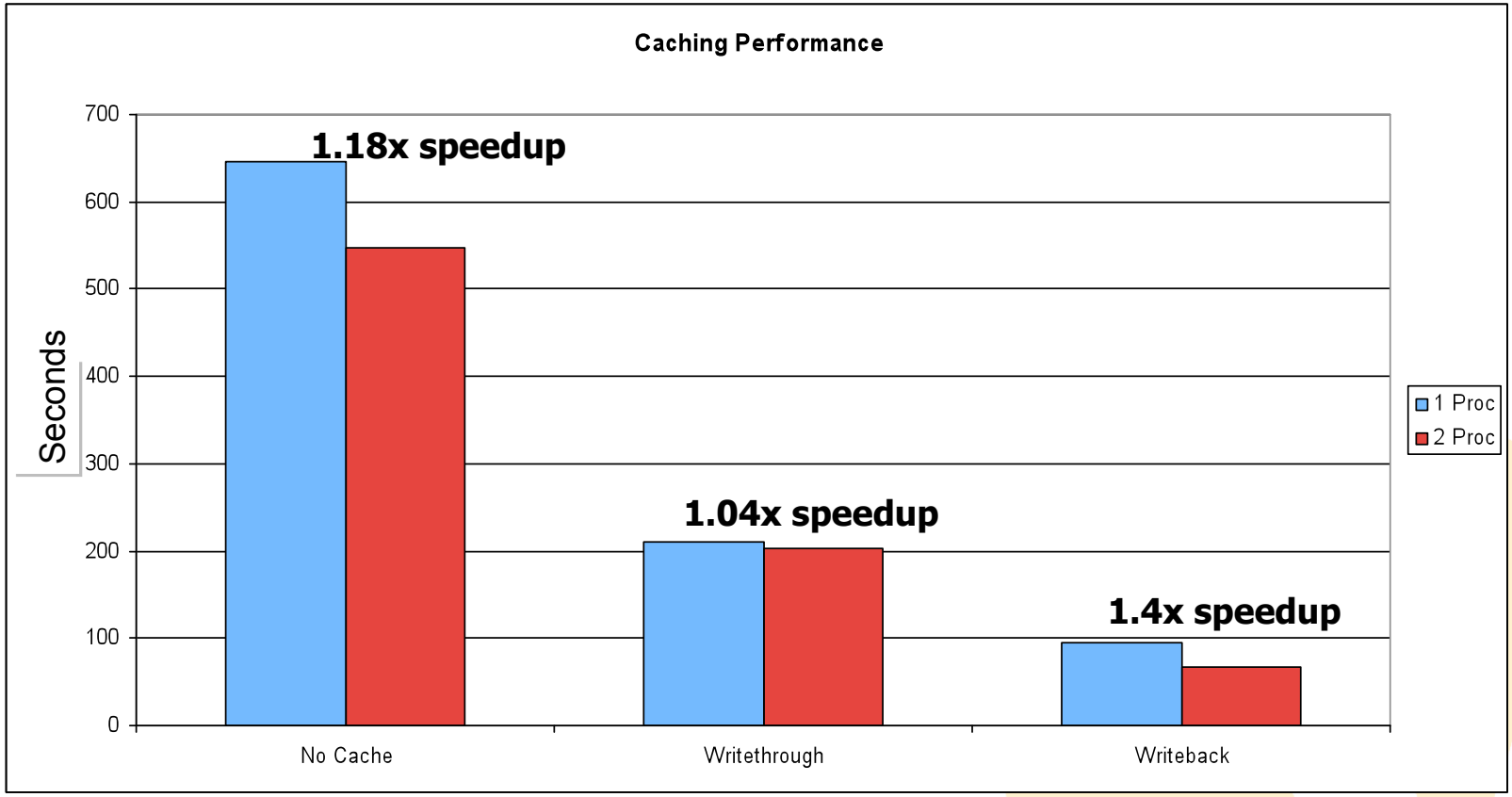
- Best performance
- Most complex implementation

Data endpoints corrupt due to cache misalignment

Solution: Programmers align ALL shared data to cache boundaries



Performance



FAULT TOLERANCE TECHNIQUES

Mission Analysis

Upsets constitute an extremely small fraction of overall cycles

- PowerPC 405 – 3.888 x 10^13 clock cycles per day vs ~1 error per 50 days

Communication Downlink is largest bottleneck

- Data typically buffered – enables out of order execution

Science Applications

- Tend to be streaming computations with little feedback or state needed to be kept
- Ground processing can clean up single, non-persistent errors

High Performance Computing Community has similar problem

- Is checkpointing and rollback viable for embedded real time systems?

Fault Tolerance System Hierarchy

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

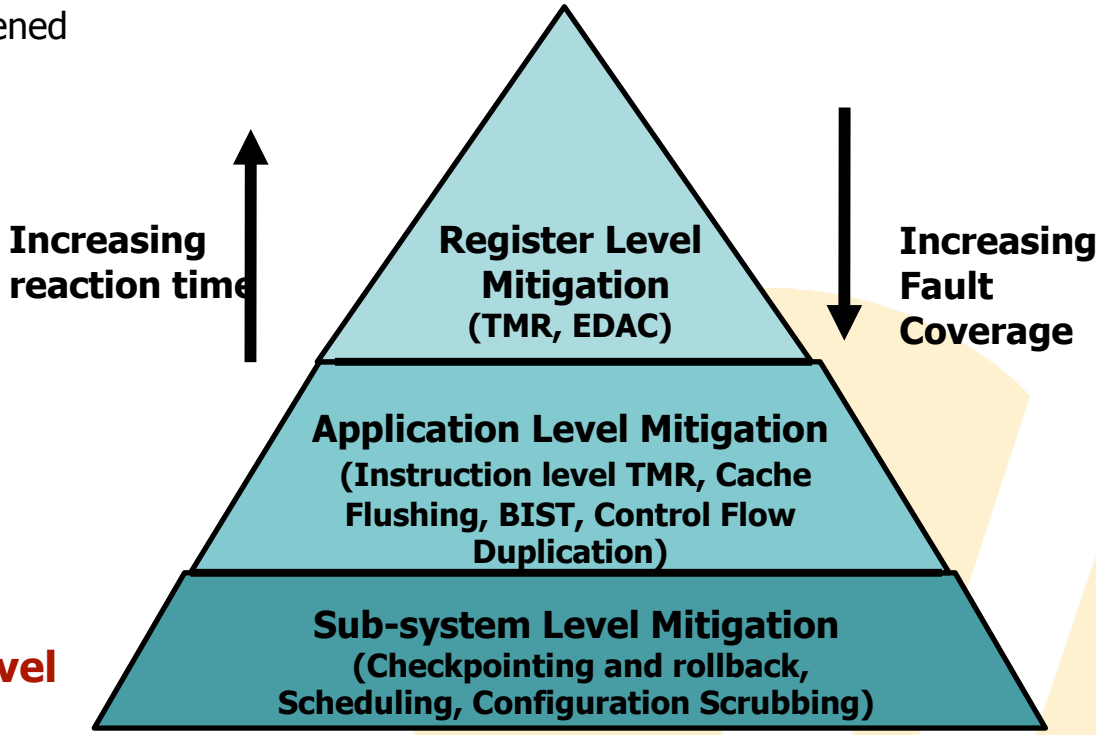
Application Level Mitigation

- 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



Sub-system Level Mitigation

USC Viterbi School of Engineering

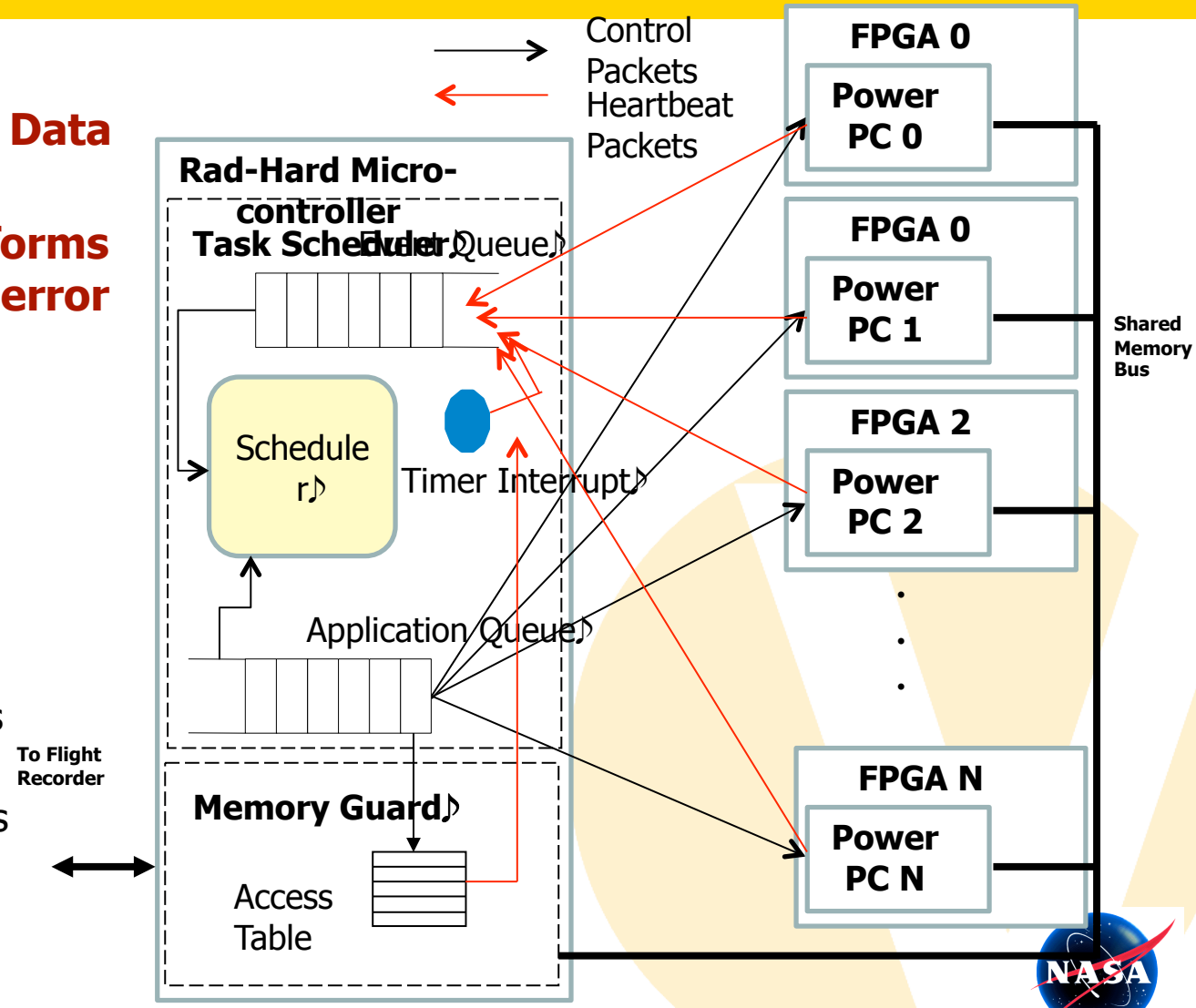
Implement Single Instruction, Multiple Data (SIMD) model

RadHard controller performs data scheduling and error handling

- Control packets from RadHard controller to PowerPCs
- Performs traditional bitstream scrubbing

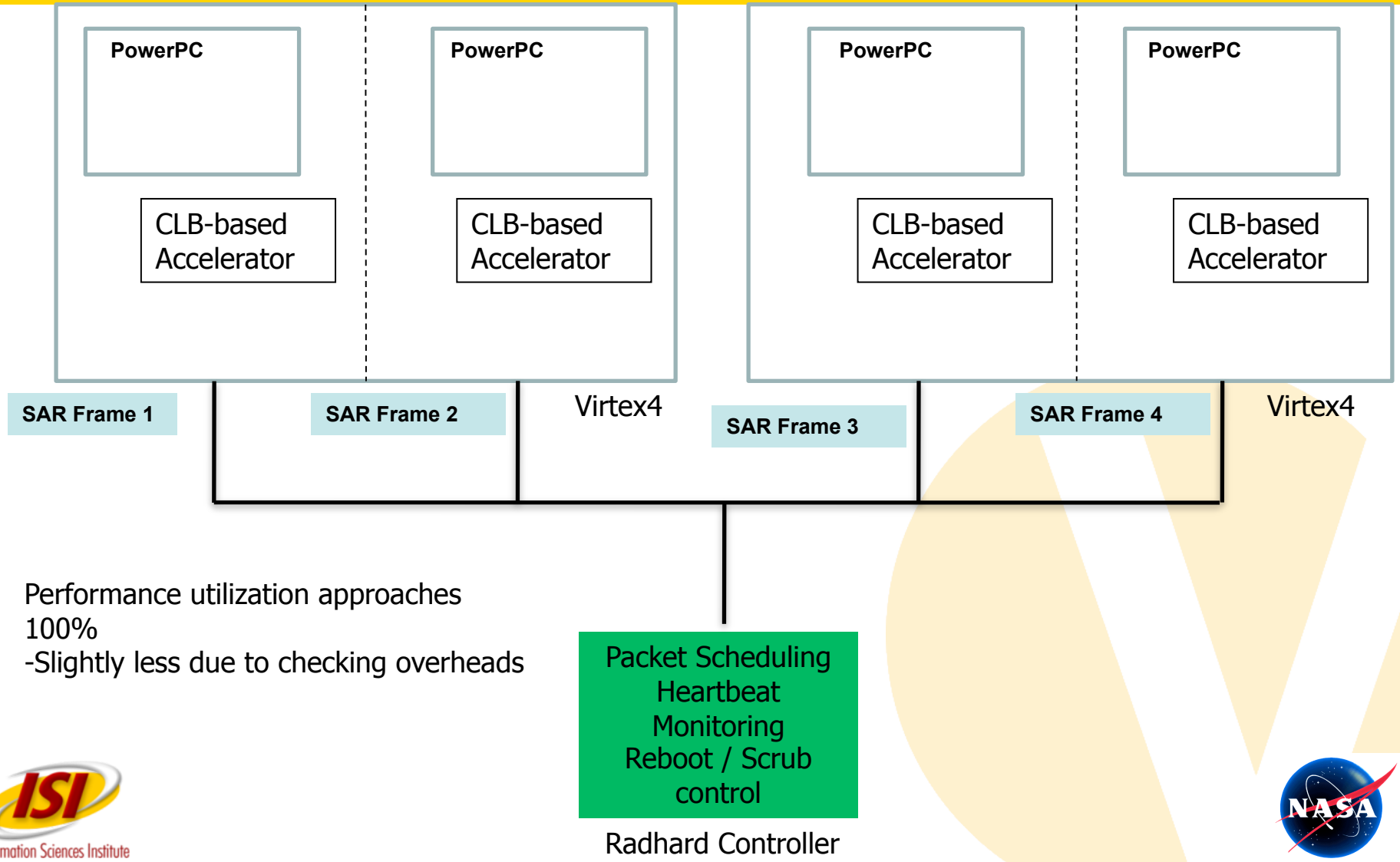
PowerPC node

- Performs health status monitoring (BIST)
- Sends health diagnosis packet 'heartbeats' to RadHard controller



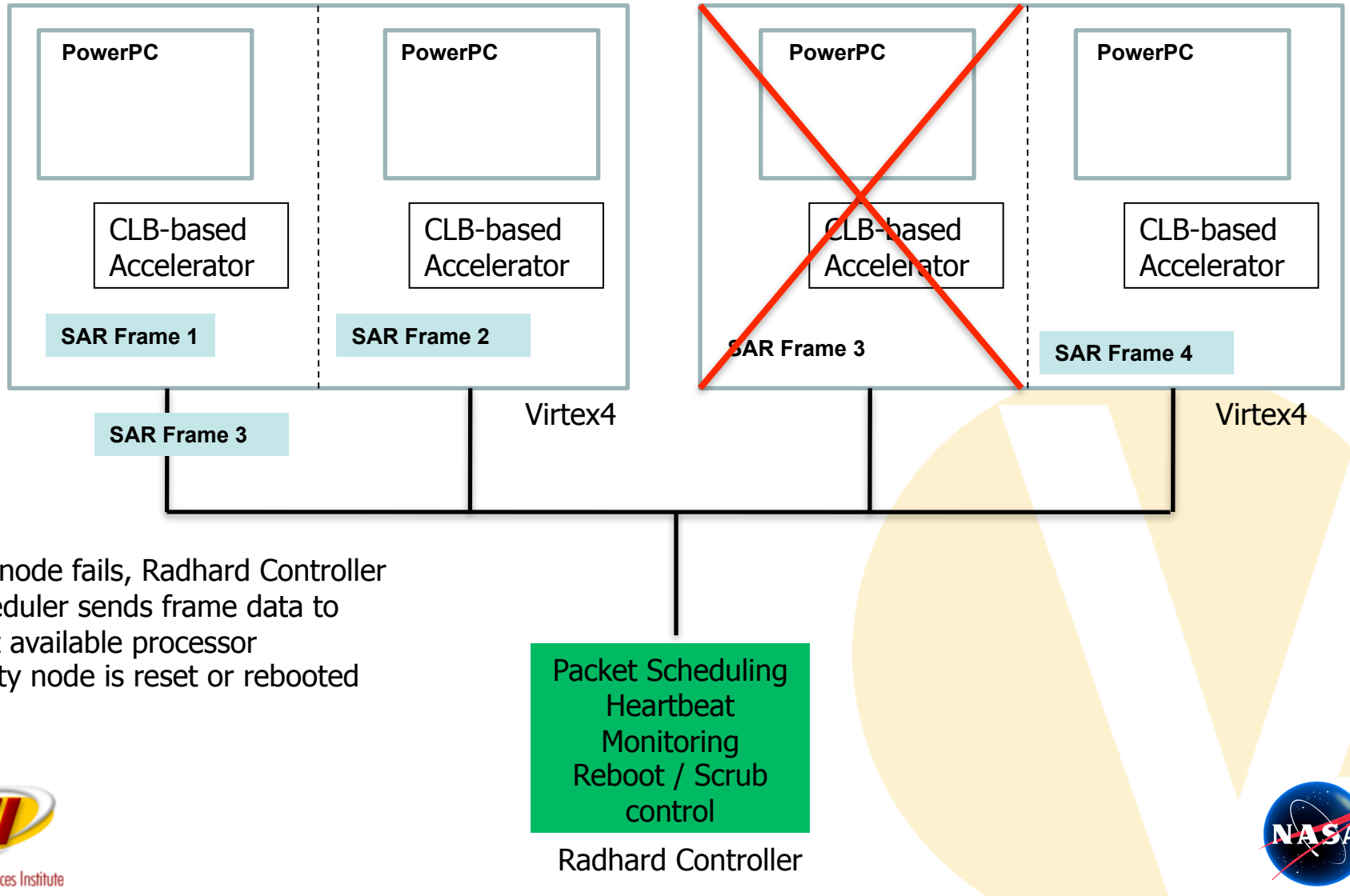
Information Sciences Institute

Sub-system Architecture: No Errors



Performance utilization approaches 100%
-Slightly less due to checking overheads

Sub-system Architecture: Failure Mode



If a node fails, Radhard Controller scheduler sends frame data to next available processor
Faulty node is reset or rebooted

Checkpoint and Rollback

USC Viterbi School of Engineering

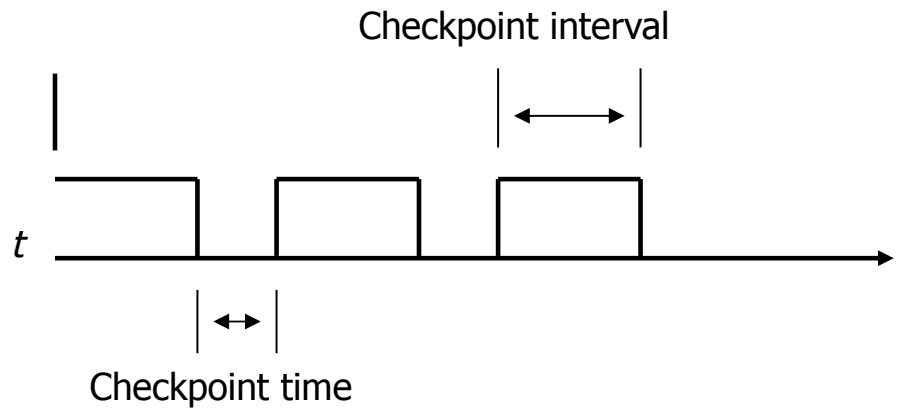
User-level checkpoint/rollback

General purpose

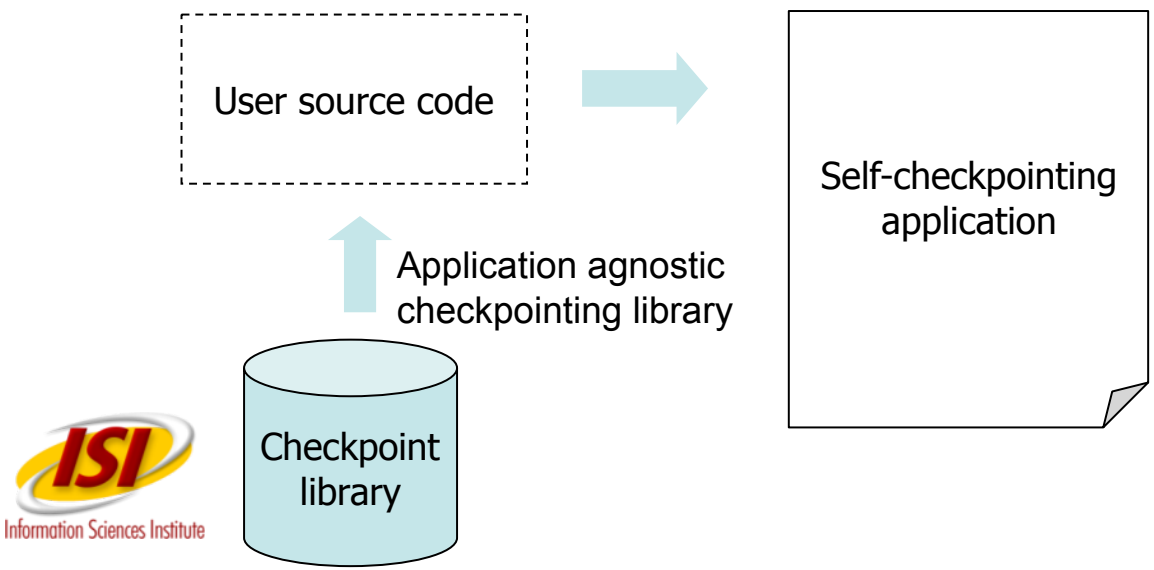
Provides user-defined callbacks

- Helpful for graceful cleanup of files, networks, FPGA fabric

Enables rapid context switching



Balance checkpoint interval to upset rate



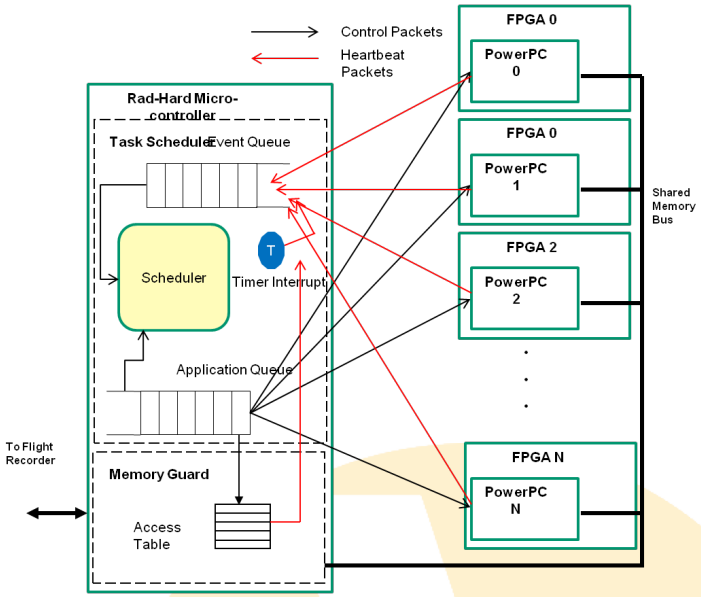
- User links in checkpoint library
- Library provides checkpoint() and restart() functions
- User inserts calls to checkpoint() at desired location(s)

ISI Information Sciences Institute

NASA

Heartbeats

- Heartbeats are generated by an FPGA based timer interrupt
- Each Heartbeat includes at least the following:
  - Destination ID / Source ID (1 byte)
  - Message Number (1 byte)
  - Message Type (1 byte)
  - Data Length (N bytes)
  - N data bytes
- Heartbeats output when:
  - Program Starts
  - Program Ends
  - Autonomous Events




```
// 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);
```

Control Flow Assertions

- **Tag blocks of code with signatures**
- **As code progresses check signatures against expected value**
- **Programmer indicates where to put assertions**

Original Code

```
x = 50;
if (condition == 1)
    new_x = x-5;
else
    new_x = x - 3;
z = new_x - x;
```

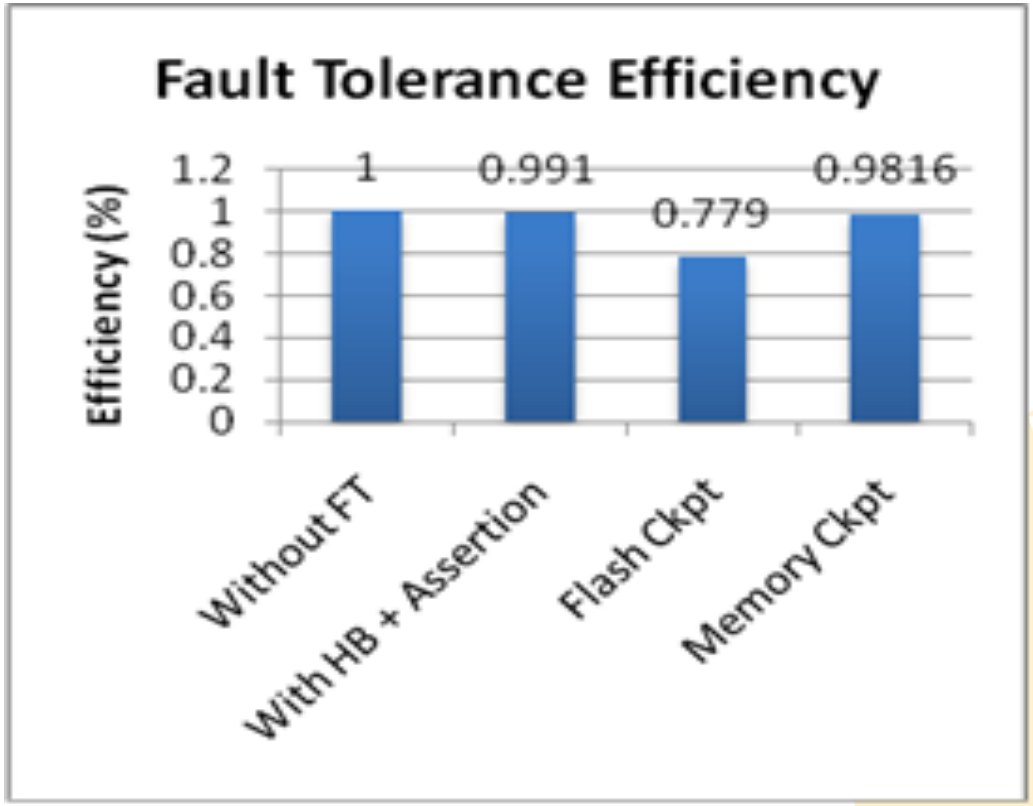
Transformed Code

```
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;
```

- When an error is detected, alert heartbeat and initiate a rollback
- Coordinate rollback/restart with 2nd PPC

PRELIMINARY RESULTS

Performance Overhead



- Checkpointing largely dependant on off-chip memory speed
- SpaceCube will check point in memory, not Flash

Fault tolerance only costing < 2% overhead

SAR Fault Injection Results (Unmitigated)

Using SPFI fault injector for baseline testing

- Automatically injecting faults into register set and memory

Observations

- Only 10% of the injected errors resulted in failure of any kind.
- 89% of injections had no effect
- 1% failed to inject

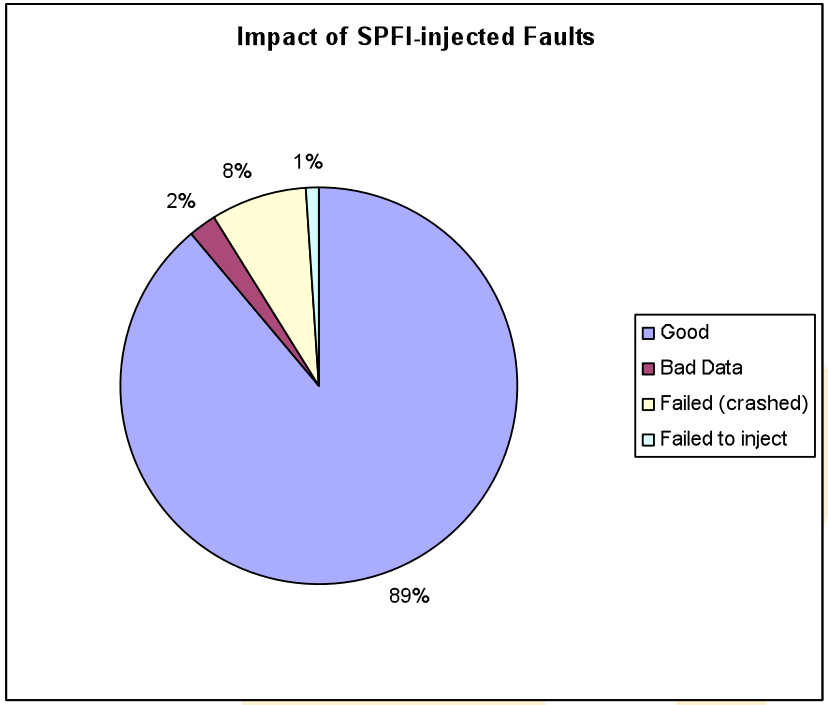
Of those injections that resulted in failures

- Only 2 resulted in bad data
- 8 crashed the application

Of those failures that crashed the application

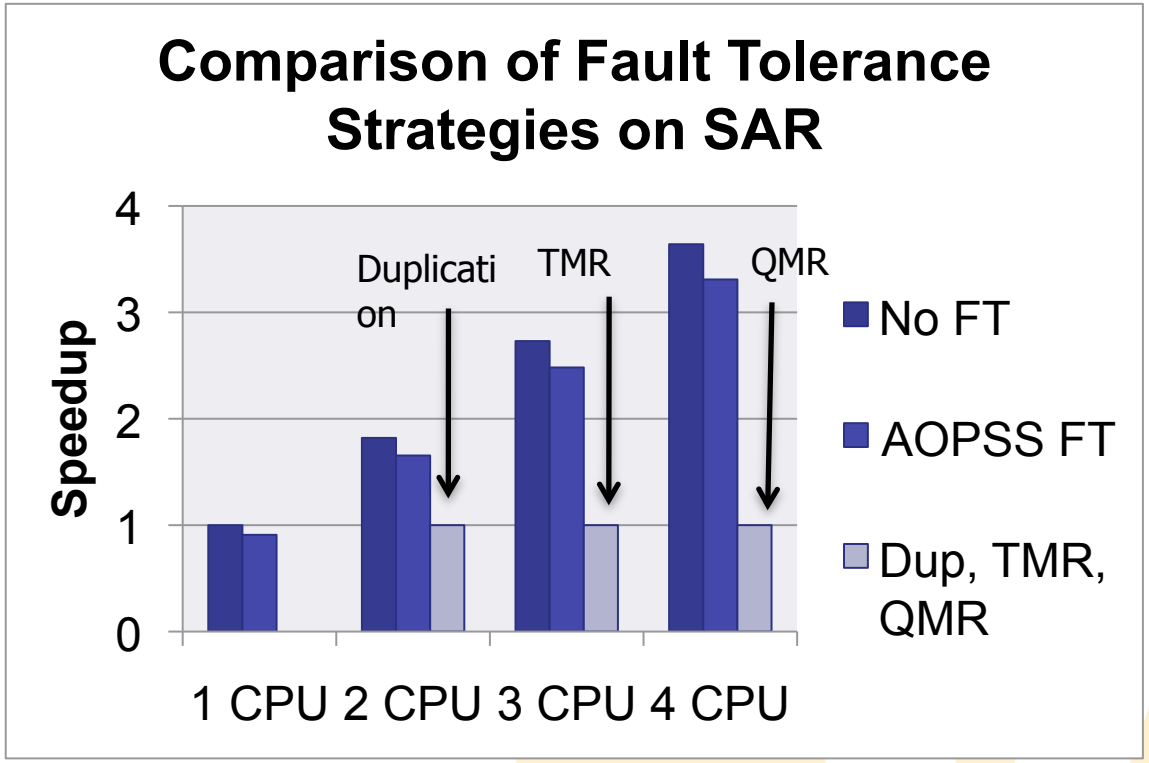
- Only 1 was a GPR
- Others were LR, SP, PC
  - *Mostly control flow*

A-OPSS fault mitigation can detect and recover from many control flow failures



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



SUMMARY

Developing a library of fault tolerance routines available to NASA community

- Targeted for science data processing

Initial tests promising

- Observed faults in unmitigated processor in LEO extremely low
- <2% overhead for fault tolerant routines
- ~2% of faults result in data errors

Upgrading Fault Injection

- Developing new techniques to inject faults from FPGA fabric which emulate faults in caches, local buses etc

Test Plans

- Beam testing 2nd half 2010
- ISS testing on MISSE-7