Methods for testing fault tolerant systems

FAULT INJECTION

Overview

- Introduction

- Hardware Fault Injection
- Software Fault Injection
- Protocol Fault Injection

- Conclusions
- Questions

Introduction

- Test robust systems

- Run two copies
  - One is run normally
  - One is run with injected faults

  - Check that fault is detected
  - Compare behavior

HARDWARE FAULT INJECTION

Hardware Fault Injection

- Tests both hardware and software
- Physically cause faults
  - Heavy-Ion radiation
  - Pin level injection
  - EMI

- Focused on hardware testing

- Johan Karlsson, et al. "Application of Three Physical Fault Injection Techniques to the Experimental Assessment of the MARS Architecture," 1995

Hardware Fault Injection

- Could be used to test software
- Software based techniques work
  - Software doesn't know where fault came from
  - Can be used to test hardware
  - Tends not to trigger hardware fault detection

- Jean Arlat, et al. "Comparison of Physical and Software-Implemented Fault Injection Techniques," IEEE 2003

SOFTWARE FAULT INJECTION

Software Fault Injection

- Modify system state programmatically

- Only tests software
- Repeatable
- Possible to run on a larger scale

- Vary from full virtualization to none at all

Software Fault Injection

- Simulation based fault injection
  - Can trace the flow of an error
  - May need fewer test runs
  - Test runs are slower
  - Virtual environment not like real one
- Software implemented fault injection
- Many things fall in between

Software Fault Injection

- Full Simulation
- V-System VHDL-Simulator
- JVM exception injector
- RUGRAT
- Holodeck
- Xception
- FERRARI
- No simulation

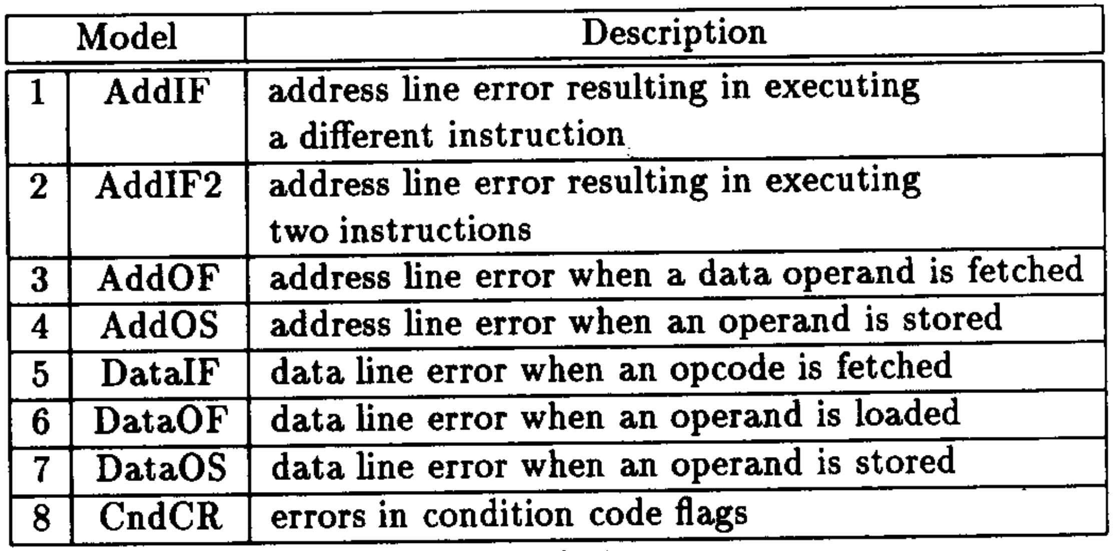
FERRARI

- Use known models of hardware faults
  - Simulate bit errors

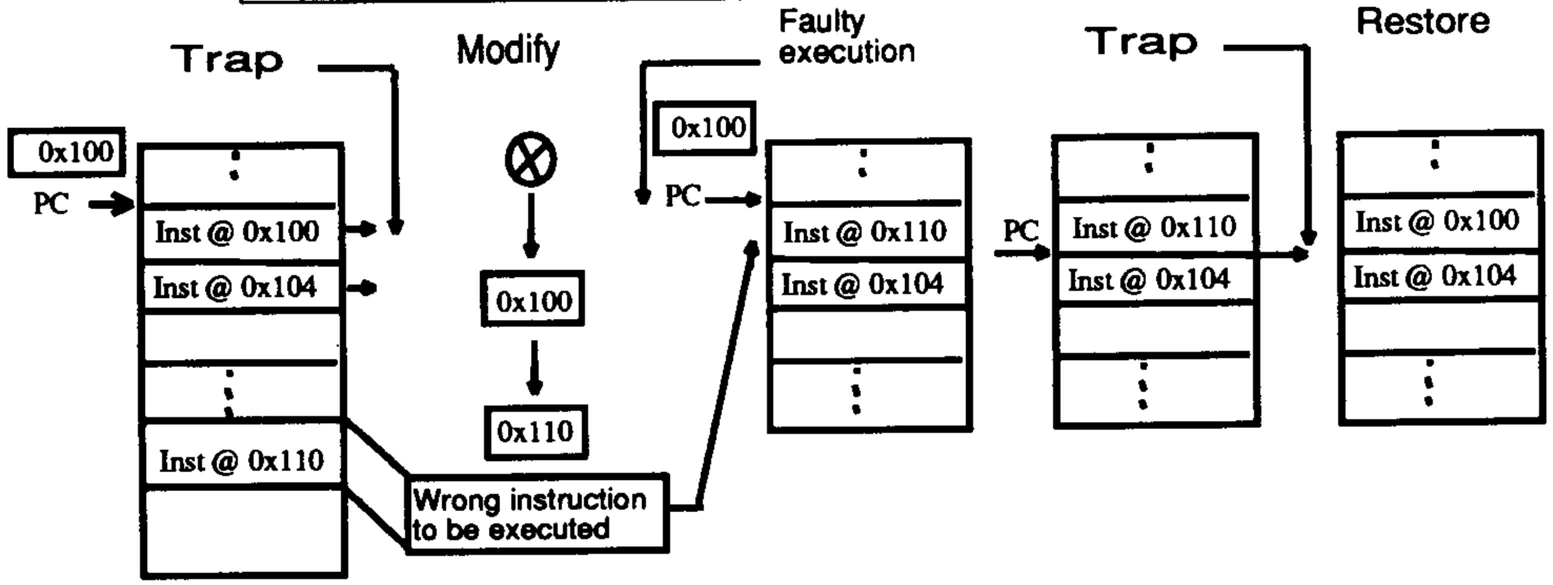
- Load target program into memory
- Inserts a trap to the injection process
- At runtime trap is executed
- Modifies program state

- G.A. Kanawati, N.A. Kanawati, and J.A. Abraham, "FERRARI: A Flexible Software-Based Fault and Error Injection System," IEEE Transactions on Computers, vol. 44, pp. 248, 1995

FERRARI



	Model	Description
1	AddIF	address line error resulting in executing a different instruction
2	AddIF2	address line error resulting in executing two instructions
3	AddOF	address line error when a data operand is fetched
4	AddOS	address line error when an operand is stored
5	DataIF	data line error when an opcode is fetched
6	DataOF	data line error when an operand is loaded
7	DataOS	data line error when an operand is stored
8	CndCR	errors in condition code flags



FERRARI

- Injected faults can be transient
- Can remove the trap after it is used

- Some faults are caught by the hardware
  - Illegal instructions, memory addresses, etc.

- Most faults are detected by software

FERRARI

- Dependent on system configuration
  - Has been ported to other platforms
- Validation given in terms of coverage
- Experiments run a large number of times
  - Only two different targets used

Xception

- Avoids modifying the target program at all
- Use processor features to inject faults

- Allows testing of run-time sanity checks
  - Run-time checksums of memory
  - Dynamic generation of function pointers

- João Carreira, Henrique Madeira, and João Gabriel Silva, "Xception: Software Fault Injection and Monitoring in Processor Functional Units," 1998

Xception

- Target program runs a full speed
- Use processor to jump to Xception code
- Generate hardware exception
  - Count a certain number of instructions
  - Set a memory address as inaccessible
  - Etc.
- Exception is handled by Xception code

Xception

- Has inject exception handlers into target
- Target may actually try to handle itself
- Needs to be tailored to the architecture
- Some exceptions are triggered frequently

PROTOCOL FAULT INJECTION

Protocol Fault Injection

- Distributed fault tolerant system
  - Needs to use some sort of protocol
  - Inject faults in the communications
  - Test that the system doesn’t fail
- Only need protocol description
  - Rest can be black box

ORCHESTRA

- Protocol injection platform
- Can be used to test any system
  - Needs code for intercepting messages
  - Needs code for relaying messages
  - Needs code for mutating messages
- Only really implemented for TCP

- Scott Dawson, Farnam Jahanian, Todd Mitton, and Teck-Lee Tung, "Testing of Fault-Tolerant and Real-Time Distributed Systems via Protocol Fault Injection," In Proceedings of the 26th International Symposium on Fault-Tolerant Computing (FTCS-26), 1996

ORCHESTRA

- Manipulates messages by
  - Dropping
  - Delaying
  - Reordering
  - Duplicating
  - Modifying contents
  - Generating extra messages

ORCHESTRA

- Has a nice GUI for writing scripts
  - State machine model
  - Control how messages are modified
  - Generation of messages
- Can simulate both faulty node or network

ORCHESTRA

- Validation of target depends on system
  - Quorum negotiates reaches consensus
  - Database is consistent and correct

- Can test only one kind of fault at a time

CONCLUSIONS

Conclusions

- Hardware fault injection is good for testing hardware, but not needed for testing software
- Software fault injection has many forms
    - Different advantages and disadvantages
- Protocol fault injection
    - Generally grouped with software fault injection
    - Relatively straight forward

Conclusions

- Case by case validation
  - Golden system
  - Assume handles are correct if they are triggered
  - Verify that system remains in a valid state

QUESTIONS?