

ECE390
Computer Engineering II
Lecture 24



Dr. Zbigniew Kalbarczyk
University of Illinois at Urbana- Champaign

Outline



- Processor-Level Detection and Recovery

“Some” Design Objectives in Architecting Processors

- Performance, performance, and performance !!!!!!!!!!!
- Reduced power consumption
- **Reliability, reduce error sensitivity**

Fault Models

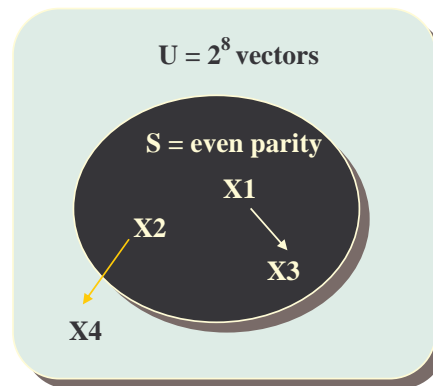
- Transients are a major factor in upsetting the correct operation of digital circuits.
 - Single and multiple upsets – device scaling and decrease in power make electronic devices highly sensitive to transients induced by ionizing particles and current and voltage spikes.
 - Wide range of software and hardware errors
- Significant fraction of processor logic is not visible and unprotected against transients

Error Detection through Encoding: Information Redundancy

- At logic level, codes provide means of masking or detecting errors
- Formally, code is a subset S of universe U of possible vectors
- A non-code word is a vector in set $U-S$

$X1$ is a codeword
<10010011>
due to multiple bit error,
becomes
 $X3 = <10011100>$
not detectable

$X2$ is a codeword,
becomes $X4$ noncode
detectable



Z. Kalbarczyk

ECE390

Hamming Distance Properties

- **Hamming distance** between two vectors x and y , $d(x,y)$ is number of bits in which they differ.
- **Distance of a code** is a minimum of Hamming distances between all pairs of code words.
- To **detect** all error patterns of **Hamming distance $\leq d$** , **code distance must be $\geq d+1$**
 - e.g., code with distance 2 can detect patterns with distance 1 (i.e., single bit errors)
- To **correct** all error patterns of **Hamming distance $\leq c$** , **code distance must be $\geq 2c + 1$**
- To **detect** all patterns of **Hamming distance d** , and **correct** all patterns of **Hamming distance c** , **code distance must be $\geq 2c + d + 1$**
 - e.g., code with distance 3 can detect and correct all single-bit errors

Example:

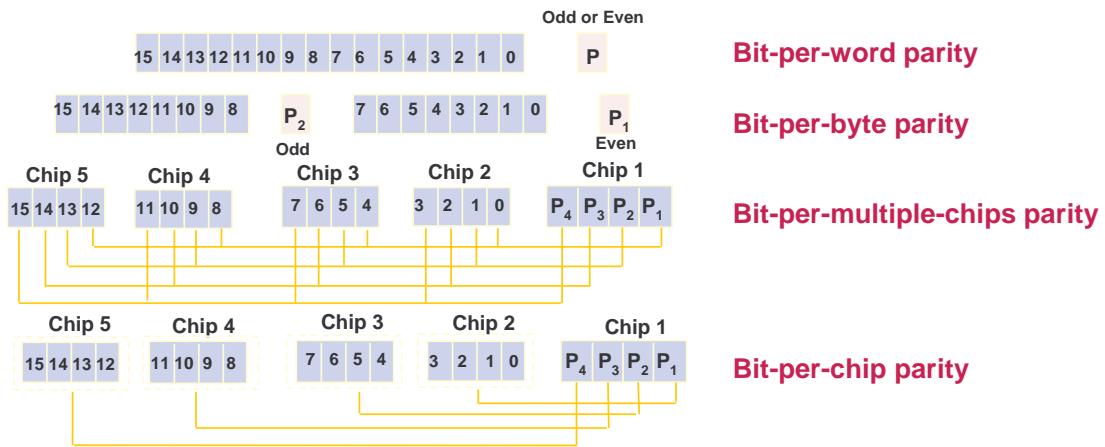
$x = (1011), y = (0110)$

$d(x, y) = 3$

Z. Kalbarczyk

ECE390

Parity Codes for RAMs



Z. Kalbarczyk

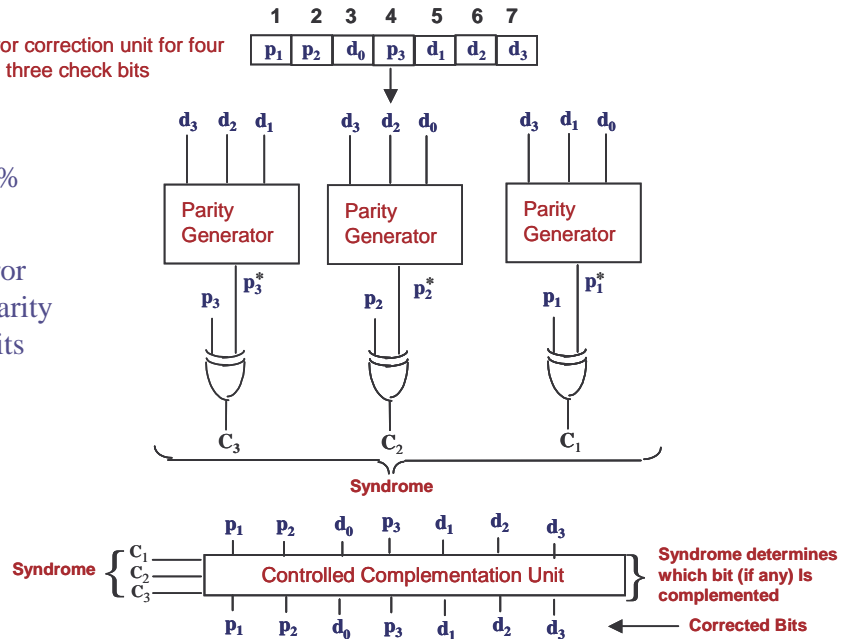
ECE390

Hamming Single-Error Correction Unit

- Require from 10% to 40% redundancy
- The Hamming single-error correcting code uses c parity check bits to protect k bits of information:

$$2^c \geq c + k + 1$$

Hamming single-error correction unit for four information bits and three check bits



Z. Kalbarczyk

ECE390

Chipkill Memory

- *Chipkill or Advanced ECC memory* is an IBM memory subsystem technology that significantly increases memory reliability
- This radically reduces the chances of system downtime caused by memory failures.
- This technology was developed specifically for the NASA pathfinder mission to Mars.

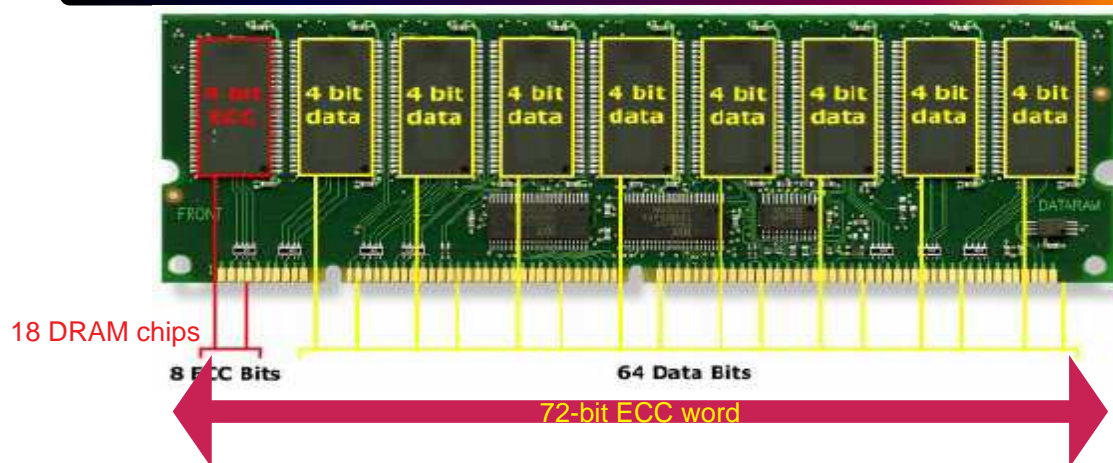
Why Is It Important?

- Standard ECC (Error Correcting Code) memory is able to detect and correct single bit memory errors, which make up the vast majority of memory errors.
- Higher risk of multi-bit memory errors due to increase in:
 - memory capacity, e.g., 32GB,
 - memory density on a single DIMM (Dual Inline Memory Modules), e.g., up to 1GB
 - memory subsystem speed
- Multi-bit errors cannot be corrected by standard ECC memory and result in the system hang.
- *Chipkill memory* has the ability to correct multi-bit memory errors and in doing so, increases system availability considerably.

How does it actually work?

- When writing data to the DIMM, a duplicate set of data in the form of a checksum is written to another part of the memory subsystem.
- If a memory failure occurs then the data is immediately recovered by re-calculating the data from the checksum information.
- This procedure allows the system to mask not only the single bit errors that standard ECC memory can correct but also 2,3 & 4 bit errors. In some cases, even a whole DRAM Chip failure.

512MB DIMM Using Standard ECC



A simple Hamming type ECC cannot recover the four bits of errors.

Standard ECC technology, which was highly reliable just a few years ago, is quickly becoming insufficient for today's dense DIMMs machines.

Configuration of DIMMs

- Usually DIMMs are install in groups of four
 - Four [64-bit data + 8 ECC bits]
- Data and ECC bits are scattered

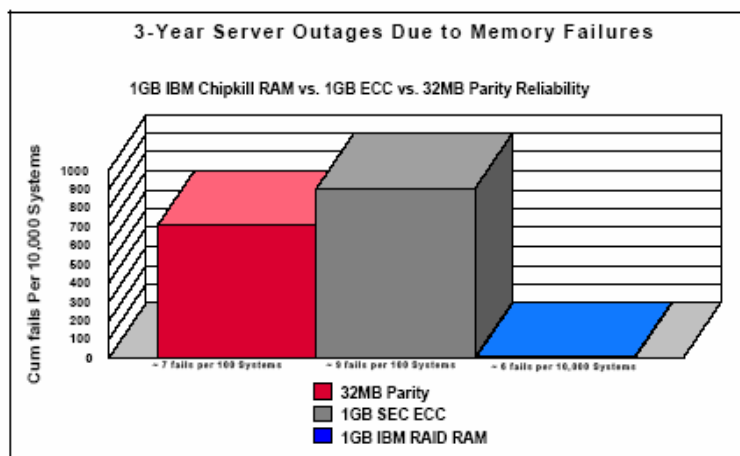


Z. Kalbarczyk

ECE390

How Reliable Is This?

- 32MB parity memory-equipped server received
 - 7 outages per 100 servers over 3 years.
- 1GB ECC memory-equipped server received
 - 9 outages per 100 servers over 3 years.
- 4GB Chipkill-equipped server received
 - 6 outages per 10,000 servers over 3 years!



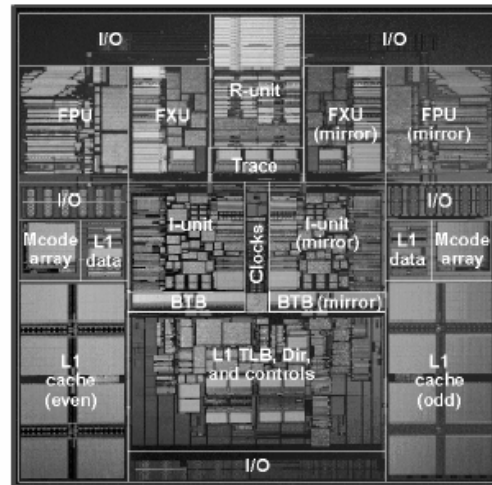
- The reliability of standard ECC memory 91% versus 99.94% of Chipkill.
- Simulation of continuous operation (720 power-on hours) over 36 months using 8x128MB DIMMs with 4 x 64Mb DRAMs

Z. Kalbarczyk

ECE390

IBM's S/390 G5 Microprocessor

- Not superscalar processor in IBM's CMOS technology
- Four logical units
 - The L1-cache, or buffer control element (BCE),
 - contains the cache data arrays, cache directory, translation-lookaside buffer (TLB), and address translation logic.
 - The I-unit
 - handles instruction fetching, decoding, and address generation and contains the queue of instructions awaiting execution.
 - The E-unit
 - contains the various execution units, along with the local working copy of the general access and floating point registers.
 - The R-unit
 - is the recovery unit that holds a checkpointed copy of the entire microarchitectured state of the processor



Z. Kalbarczyk ECE390

IBM G5 Microprocessor: Recovery Support

- R-unit
 - For every clock cycle in which the E-unit produces a result, that value is also written into the R-unit copy.
 - The R-unit checks whether the result is correct and then it generates ECC on that result.
 - The checkpointed result is written into the R-unit registers along with ECC.
 - The contents of R-unit registers represent the complete checkpointed state of the processor during any given cycle, should it be necessary to recover from a hardware error.
- Millicode
 - Millicode is used to implement instructions that are either more complex or relatively infrequently used
 - The millicode has complete read/write access to all R-unit registers.
 - Millicode also performs various service functions
 - logging data associated with any hardware errors that may have occurred, scrubbing memory for correctable errors, supporting operator console functions, and controlling low-level I/O operations.

Z. Kalbarczyk

ECE390

IBM G5 Microprocessor: Recovery Support

- **Full duplication of the I-unit and E-unit.**
 - On every clock cycle, signals coming from these units, including instruction results, are cross-compared in the R-unit and the L1-cache.
 - If the signals do not match, hardware error recovery is invoked.
 - All arrays in the L1-cache unit are protected with parity except for the store buffers, which are protected with ECC.
 - If the R-unit or L1-cache detects an error, the processor automatically enters an error recovery mode of operation.

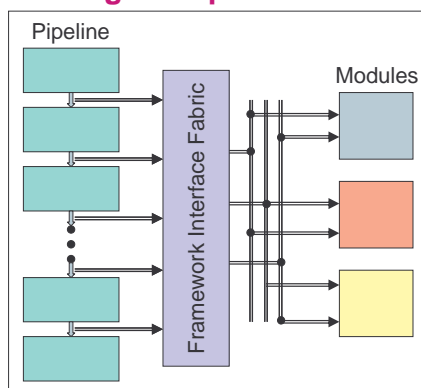
IBM G5 Microprocessor: Recovery Procedure

1. The R-unit freezes its checkpoint state and does not allow any pending instructions to update it.
2. The L1-cache forwards any store data to the L2 for instructions that have already been checkpointed.
3. All arrays in the L1 cache unit are reset.
4. Each R-unit register is read out in sequence, with ECC logic correcting any errors it may find, and the corrected values are written back into the register file and to all shadow copies of these registers in the I-unit, E-unit, and L1-cache.
5. All R-unit registers are read a second time to ensure there are no solid correctable errors. If there are, the processor is check-stopped, i.e., that chip is no longer available for system
6. The E-unit forces a serialization interrupt, which restarts instruction fetching and execution.
7. An asynchronous interrupt tells millicode to log trace array and other data for later analysis by IBM product engineering.

Processor-Level Detection and Recovery Framework

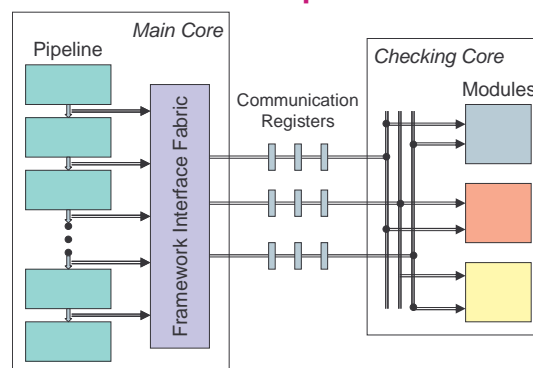
Reconfigurable Processor-Level Framework

On-Core Technique Single-Chip Architecture



- Processor, framework, and modules are part of the same core on a single die
- Framework and modules are reconfigurable
- Inputs to the framework are the outputs of the pipeline stages.

Off-Core Technique Multi-Chip Architecture



- Processor and the framework interface implemented as one core (the main core),
- Modules embedded in another (reconfigurable) core (the checking core).
- The two cores are located on the same die
- Communication path pipelined

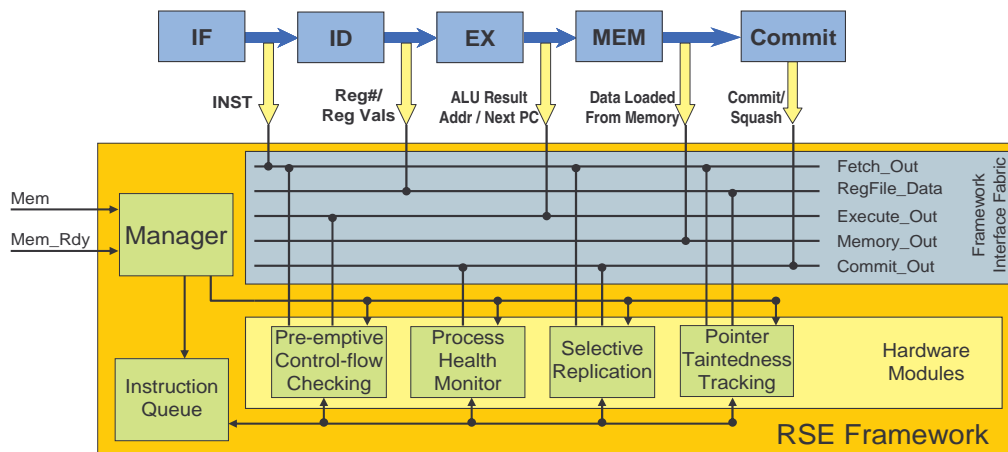
Reliability and Security Engine

- A common framework to provide a variety of application-aware techniques for error-detection, masking of security vulnerabilities and recovery under one umbrella, in a uniform, low overhead manner.
- Implemented as an integral part of a superscalar microprocessor
- Hardware-implemented error-detection and security mechanisms embedded as modules in the framework
- The framework serves two purposes
 - Hosts hardware modules that provide reliability and security services, and
 - Implements interface of the modules with the main pipeline and the executing software (OS and application)

Z. Kalbarczyk

ECE390

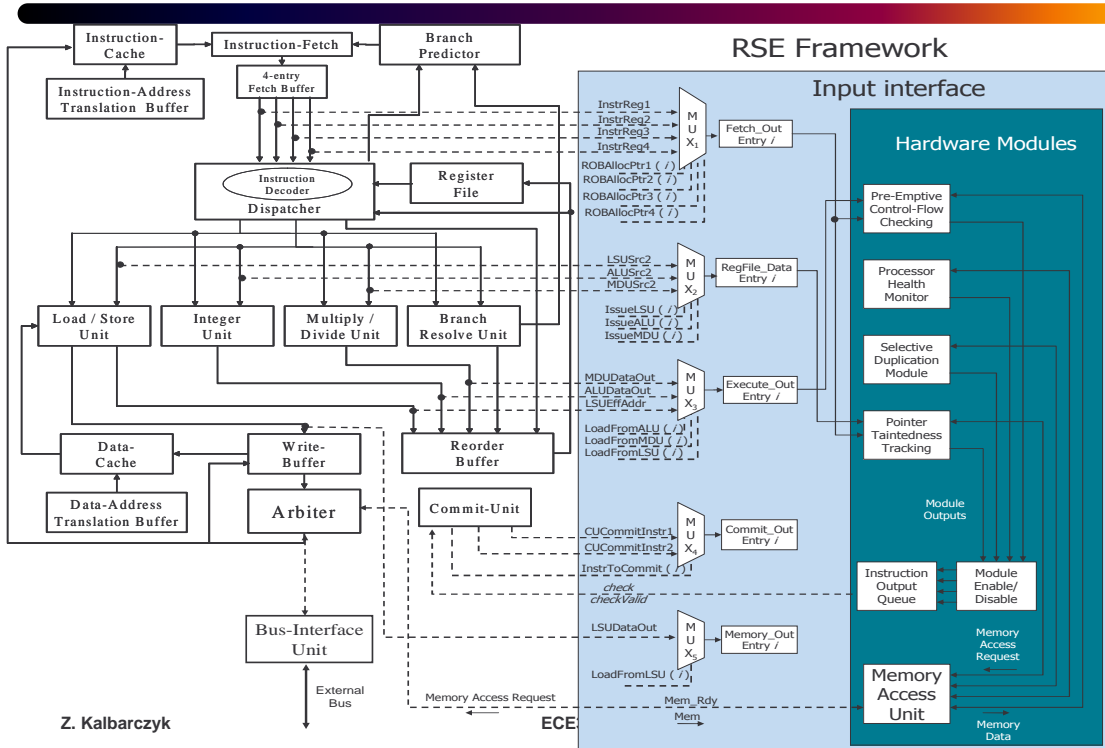
Reliability and Security Engine (RSE): Architecture



Z. Kalbarczyk

ECE390

Reliability and Security Engine (RSE): Example Implementation



Assessment of Microprocessors Failure Behavior Using Fault Injection at Gate-level

Goals

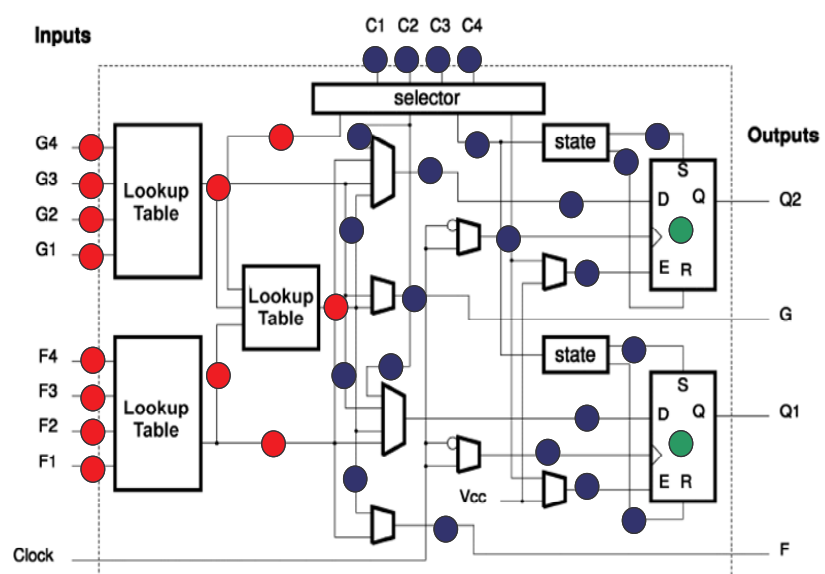
- Which parts of the processor are more sensitive to faults, and how do they fail?
- How can we protect them?
- What is the effect of faults in the control logic?
- What is the error sensitivity of the speculation logic?
- What are application level effects of faults

Z. Kalbarczyk

ECE390

FPGA Fault Model

- Faults in Xilinx FPGA architecture
- Bit-flip faults in flip-flops
- Transient disturbances (varying duration) on input/output of LUTs / gate of glue logic
- Time of fault occurrence is equi-distributed in the simulation time



Z. Kalbarczyk

ECE390

Estimation of fault sensitivity of processor components

Functional Units	1/8 clk	1/4 clk	1/2 clk	1 clk
	Sensitivity	Sensitivity	Sensitivity	Sensitivity
Execution	0.5	0.6	1.3	2.4
Control	0.8	1.3	2.3	4.4
Speculation	1.0	4.0	6.5	12.8
Memory Interface	2.6	4.2	6.4	11.1

- ◆ **Control** and **Speculation** units are 2 and 5 times more sensitive than **Execution** unit
 - ◆ Speculation is *not* fault-tolerant
 - ◆ Speculation sensitivity mainly due to Branch Target prediction (in IF) and Commit Unit
- ◆ We expect this problem being worse for more complex microprocessors (such as Pentium 4 and AMD Athlon 64)

Z. Kalbarczyk

ECE390

Impact on the application

- The processor is connected to an external memory (not injected), containing the process image (code and data) written / read by the processor

Outcome	Definition
Crash	A memory location loaded or stored is out of the boundaries of the application text segment or out of the boundaries of the application data segment.
Fail-silent data violation (SDC)	The application terminates without crashing but the result of the computation, i.e., the content of the memory, is incorrect as compared with the golden run.
Incomplete execution	The program does not complete in the expected time (normal execution time + extra time margin).
No effect	There is a mismatch at the pin-level in the cycle-accurate behavior of the processor, and the application program terminates correctly, i.e., the content of the memory matches the golden run.

Outcome	% of errors
Crash	23%
Fail-silent data violation	13%
Incomplete execution	12%
No effect	53%

Z. Kalbarczyk

ECE390

Classification of the processor failure behavior

Functional Block	Crash	Fail silent data violation	Incomplete execution
Execution	45%	40%	17%
Control	17%	24%	35%
Speculation	17%	10%	34%
Memory Interface	21%	26%	14%

- ◆ Faults in **Execution** contribute to 45% Crashes and 40% FSDV
- ◆ **Control** and **Speculation** together contribute to 70% of Incomplete Executions
 - ◆ E.g., stalled commit unit, corrupted reorder buffer
 - ◆ Mechanisms to contain these errors