ESE532:
System-on-a-Chip Architecture

Day 25: April 19, 2017

Fault Tolerance

Today

- Fault Sources
- Fault Tolerance
  - Memories
  - Interconnect
  - Compute

Message

- At small feature sizes, voltages, and high speed, not viable or economical to demand perfect operation across billions and trillions of operations
- Little redundancy can go a long way
- For large systems, reliability must be engineered
  - …all systems are becoming large

Reminder

Simple Implications: $P_{good}=P_g^N$

- As N gets large
  - must either increase reliability
  - …or start tolerating failures
- N
  - memory bits
  - disk sectors
  - wires
  - transmitted data bits
  - processors
  - transistors
  - molecules

As devices get smaller, failure rates increase
chemists think P=0.95 is good
As devices get faster, failure rate increases

Three Problems

1. **Defects:** Manufacturing imperfection
   - Occur before operation; persistent
     - Shorts, breaks, bad contact
2. **Transient Faults:**
   - Occur during operation; transient
     - node X value flips: crosstalk, ionizing particles, bad timing, tunneling, thermal noise
3. **Lifetime "wear" defects**
   - Parts become bad during operational lifetime
     - Fatigue, electromigration, burnout….
   - …slower
     - NBTI, Hot Carrier Injection

Faults

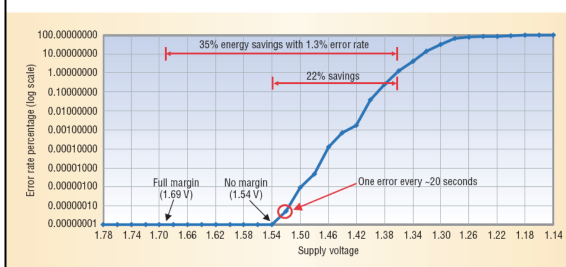
- Bits, processors, wires
  - May fail during operation
- Basic Idea same:
  - Detect failure using redundancy
  - Correct
- Now
  - Must identify and correct online with the computation

Fault Sources

Transient Sources

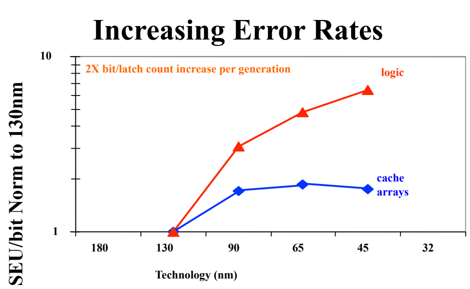
- Effects
  - Thermal noise
  - Supply voltage noise
  - Timing
  - Ionizing particles
    - α particle 10^5 to 10^6 electrons
      - Discharge DRAM cell (multiple)
    - Gates with 15—30 electrons
      - Even if CMOS restores, takes time

Voltage and Error Rate



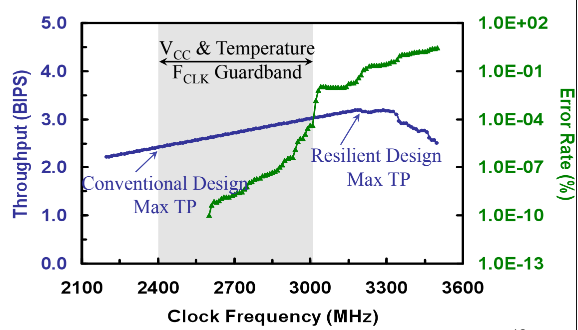
[Austin *et al.*--*IEEE Computer*, March 2004]

Scaling and Error Rates



 Source: Carter/Intel

Errors versus Frequency



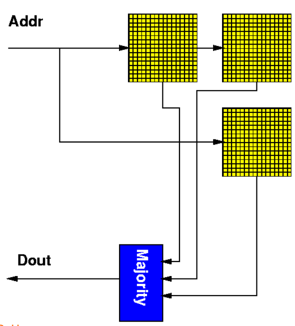
[Bowman, ISSCC 2008]

Memory

Simple Memory Example

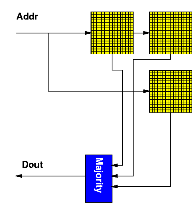
- **Problem:** bits may lose/change value
  - Alpha particle
  - Molecule spontaneously switches
- **Idea:**
  - Store multiple copies
  - Perform majority vote on result

Redundant Memory



Redundant Memory

- Like M-choose-N
- Only fail if >(N-1)/2 faults
- P=0.9
- Preclass: P(2 of 3)?



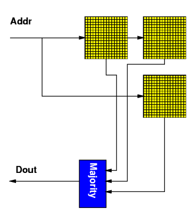
Redundant Memory

- Like M-choose-N
- Only fail if >(N-1)/2 faults
- P=0.9
- P(2 of 3)

  All good: $(0.9)^3 = 0.729$

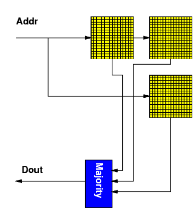
  + Any 2 good: $3(0.9)^2(0.1)=0.243$

  $= 0.971$



Redundant Memory

- Unsatisfying
  - Costs 3x (5x, 7x…) area
  - …and energy
- How can we do better?
  - Less overhead?



Better: Less Overhead

- Don't have to keep N copies
- Block data into groups
- Add a small number of bits to detect/ correct errors

Row/Column Parity

- Think of NxN bit block as array
- Compute row and column parities
  - (total of 2N bits)

						row parity
	0	1	0	1		0
	1	1	1	0		1
	1	1	0	0		0
	0	0	1	0		1
column parity	0	1	0	1		

Row/Column Parity

- Think of NxN bit block as array
- Compute row and column parities
  - (total of 2N bits)
- Any single bit error

						row parity
	0	1	0	1		0
	1	1	1	0		1
	1	1	1	0		0
	0	0	1	0		1
column parity	0	1	0	1		

Row/Column Parity

- Think of NxN bit block as array
- Compute row and column parities
  - (total of 2N bits)
- Any single bit error
- By recomputing parity
  - Know which one it is
  - Can correct it

						row parity	
	0	1	0	1		0	0
	1	1	1	0		1	1
	1	1	1	0		0	1
	0	0	1	0		1	1
column parity	0	1	0	1			
	0	1	1	1			

Preclass Exercise

- Which Block has an error?
- What correction do we need?

Row/Column Parity

- Simple case is 50% overhead
  - Add 8 bits to 16
  - Better than 200% with 3 copies
  - Overhead drop with block size
    - How scale?
  - More expensive than used in practice

						row parity	
	0	1	0	1		0	0
	1	1	1	0		1	1
	1	1	1	0		0	1
	0	0	1	0		1	1
column parity	0	1	0	1			
	0	1	1	1			

In Use Today

- Conventional DRAM Memory systems
  - Use 72b ECC (Error Correcting Code)
  - On 64b words [12.5% overhead]
  - Correct any single bit error
  - Detect multibit errors
- CD and flash blocks are ECC coded
  - Correct errors in storage/reading
- Learn More: ESE 676

Data Storage

CETS ENIAC Cluster

- Holds your home directory
- …and faculty and research data
- Stored on cluster of hard drives
  - Crude guess ~ 1000

Preclass 4

- Assume 300,000 hour MTBF/disk
- Expected number of disk failures/year for cluster of 1000?

RAID

- Redundant Array of Inexpensive Disks
- Disk drives have ECC on sectors
  - At least enough to detect failure
- RAID-5 has one parity disk
  - Tolerate any single disk failure
    - Parity enough to reconstruct when know which disk failed (technical term: Erasure code)
  - *E.g.* 8-of-9 survivability case
  - With *hot spare*, can rebuild data on spare

Interconnect

Gigabit Ethernet

- 1000BASE-T
- CAT-5 cables
- Specifies: Bit Error Rate (BER) $<10^{-10}$
- TCP/IP Packet ~ 1500Bytes
  - Call is 1250 to make math easy
- Probability of packet corruption?
- After takes 10 network hops?
- Errors in 2 hour movie?

Interconnect

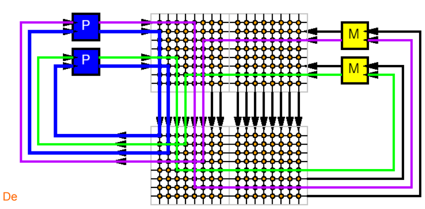
- Also uses checksums/ECC
  - Guard against data transmission errors
  - Environmental noise, crosstalk, trouble sampling data at high rates…
- Often just detect error
- Recover by requesting retransmission
  - *E.g.* TCP/IP (Internet Protocols)

Interconnect

- Also guards against whole path failure
- Sender expects acknowledgement
- If no acknowledgement will retransmit
- If have multiple paths
  - …and select well among them
  - Can route around any fault in interconnect

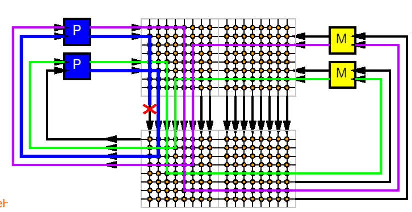
Interconnect Fault Example

- Send message
- Expect Acknowledgement



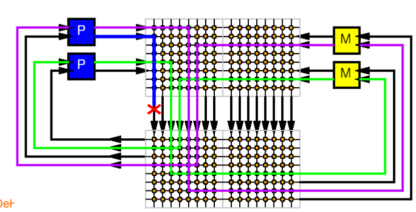
Interconnect Fault Example

- Send message
- Expect Acknowledgement
- If Fail



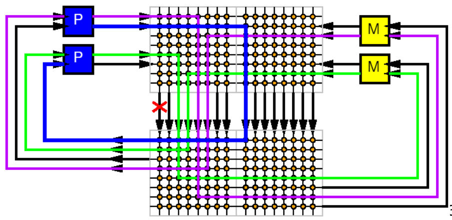
Interconnect Fault Example

- Send message
- Expect Acknowledgement
- If Fail
  - No ack



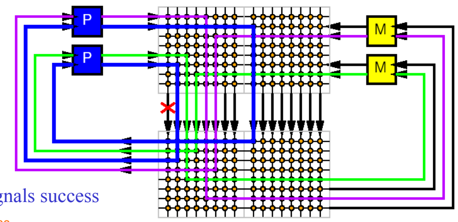
Interconnect Fault Example

- If Fail → no ack
  - Retry
  - Preferably with different resource



Interconnect Fault Example

- If Fail → no ack
  - Retry
  - Preferably with different resource

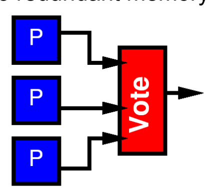


Ack signals success

Compute

Compute Elements

- Simplest thing we can do:
  - Compute redundantly
  - Vote on answer
  - Similar to redundant memory



Compute Elements

- Unlike Memory
  - State of computation important
  - Once a processor makes an error
    - All subsequent results may be wrong
- Response
  - “reset” processors which fail vote
  - Go to spare set to replace failing processor

In Use

- NASA Space Shuttle
  - Uses set of 4 voting processors
- Boeing 777
  - Uses voting processors
    - Uses different architectures for processors
    - Uses different software
    - Avoid Common-Mode failures
      - Design errors in hardware, software

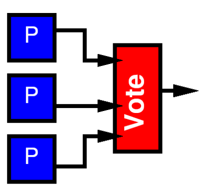
Forward Recovery

- Can take this voting idea to gate level
  - VonNeuman 1956
- Basic gate is a majority gate
  - Example 3-input voter
- Alternate stages
  - Compute
  - Voting (restoration)
- Number of technical details…
- High level bit:
  - Requires P_{gate}>0.996
  - Can make whole system as reliable as individual gate

Detect / Correct

Correction

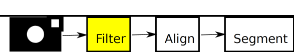
- Forward error correction with voting unsatisfying
- Paying 3x (5x, 7x, …)
  - Area
  - Energy



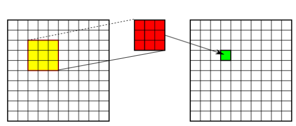
Detect vs. Correct

Detection is cheaper than correction

1. Redundancy: To handle k-faults
   - Voting correction requires 2k+1
     - K=1 → 3
   - Detection requires k+1
     - K=1 → 2
2. Computation
   - E.g.: Sorting (N*log(N)) vs. check in sorted order (N)



2D Window Filtering



- Compute output image as weighted sum of pixels
  - Demosaic
  - Gaussian filter
- Weight of subregion proportional to weight of original
  - Except edging effects

$$\frac{Check}{Compute} = \frac{2C_{add}}{W_{coeff}(C_{add}+C_{mpy})}$$

E.g. 3x3 window with no zeros

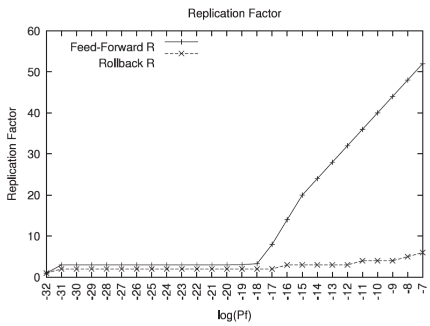
$C_{mpy}/C_{add}=10$

Check/Compute = 2/99 ~= 2%

Rollback Recovery

- Commit state of computation at key points
  - to memory (ECC, RAID protected...)
  - …reduce to previously solved problem of protecting memory
- On faults (lifetime defects)
  - recover state from last checkpoint
  - like going to last backup….
  - …(snapshot)

Rollback vs. Forward



[Naeimi, Nanotechnology 2008]

Networking, Data Center

Cloud Providers

- How does Google deal with millions of simultaneous user search requests?
- What effect of one computer crashing in their cloud?

Google (2008)

- Building block 1800 server cluster
- *In each cluster's first year, it's typical that*
  - *1,000 individual machine failures will occur;*
  - *thousands of hard drive failures will occur;*
  - *… and there's about a 50 percent chance that the cluster will overheat, taking down most of the servers in less than 5 minutes and taking 1 to 2 days to recover.*

http://www.datacenterknowledge.com/archives/2008/05/30/failure-rates-in-google-data-centers/

Defect vs. Fault Tolerance

- Defect
  - Can tolerate large defect rates (10%)
    - Use virtually all good components
    - Small overhead beyond faulty components
- Fault
  - Require lower fault rate (*e.g.* VN <0.4%)
    - Overhead to do so can be quite large

Summary

- Possible to engineer practical, reliable systems from
  - Imperfect fabrication processes (defects)
  - Unreliable elements (faults)
- We do it today for large scale systems
  - Memories (DRAMs, Hard Disks, CDs)
  - Internet
  - Multiprocessor chips
  - Data Centers
- …and critical systems
  - Space ships, Airplanes
- Engineering Questions
  - Where invest area/effort?
    - Higher yielding components? Tolerating faulty components?

Big Ideas

- Left to itself:
  - reliability of system << reliability of parts
- Can design
  - system reliability >> reliability of parts [defects]
  - system reliability ~= reliability of parts [faults]
- For large systems
  - must engineer reliability of system
  - …all systems becoming "large"

Admin

- Final Project Report
  - Due Friday
- Collect Zed Boards
  - Class Monday 4/24