Computer Architecture Futures 2007

Bob Colwell

FCRC June 2007

CompArch: Great Run So Far

1. World has paid top dollar for computers since 1952
2. Silicon has improved exponentially for several decades
   - Faster, denser, cooler, cheaper
3. CMOS let us ignore thermals/power for 30+ years
   - Gen purpose, anyway; embedded designers knew better long ago
4. Architects' job was to leverage silicon improvements into single-thread time-to-solution performance boosts
   - Nice, simple, unambiguous
5. We did this by
   - Improving CPUs first
     - Clock rates, CPI, ILP, caches, OOO, bag of tricks
   - Chipsets, memory, I/O, platforms second
     - PCI, graphics, CDROMs, DVDs,
   - Anticipating new SW that continually set bar higher

CompArch: Current Status

- Computer architecture field mutating
  - Underlying implementation tech changing
  - Historical progression halted by thermals
  - Big challenges
    - Possibility space is large, must tackle multiple issues
    - Multicore must fund industry into post-CMOS era
      - Software a huge concern
    - Usage models now subject to fashion trends
    - New implem contenders, but no clear favorite

Where We Started

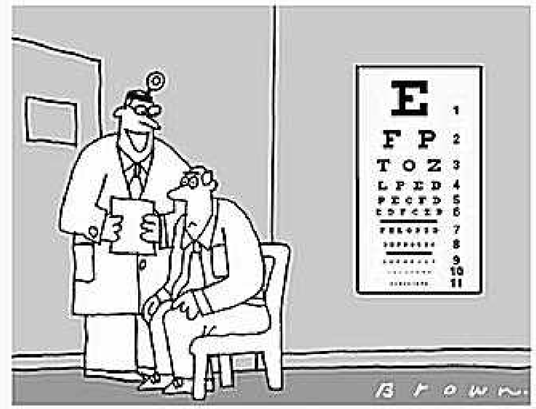
- Earliest machines built around hardware limitations
- Later machines more capable yet compatible with predecessors
  - Sys/360 and Sys/370 1960's
  - PDP-11 series 1970's
  - Intel Architecture 1990's to present
  - 1970's common wisdom: "SW will kill us, must bridge semantic gap"
  - 1980's-1990's: performance is everything
  - Meanwhile embedded processors took over the world
- Only experimental/research machines willing to trade performance for programmability
  - Intel iAPX-432, Scheme, Symbol, Lisp machines, SBN, Blaauw/Brooks, Myers' "semantic gap" books

Looking Back…

- **Software creation didn't kill us, but…**
  - **Viruses, worms, scam-ware take large toll on industry & internet**
  - **Security provided as afterthought at best**
  - **Large projects still fail because SW doesn't work**
  - **Costs too much to design and maintain**
  - **Computer systems fail gracelessly**
  - **Still don't know how to program MP**

- **HW: "every transistor must always work perfectly" attitude has scaled to today but will break badly at some point, maybe soon**

Power is now Public Enemy #1



"According to this test, your hindsight is 20-20."

Meanwhile, SW & Bottlenecks Changed

- Java interpretation
- Web browsers use large fraction of cycles
- I/O is limiter

- Worldwide, what is fraction of global cycles computers spend in idle loop?
  - **It's got to be pretty close to 1.0**
  - Does that strike you as weird?
  - Can we spend them to make active cycles more efficient?

    Search indexing…what else?

Usage Models Changing Radically

- **Embedded & general computing merging**
  - Cell phones & mobile computing have commenced an epic battle
  - HW/SW sales industry vs. subscription service industry

Cell phone/cable chief advantage monthly service fees, huge economic juggernaut; disadvantage is focus on self instead of focus on customers

Computer industry chief advantage innovation & aggressive cost reductions; disadvantage having to shift gears from historically lucrative products to less lucrative

Usage Models cont.

- **Ubiquitous: always-on internet connectivity Goal #1**
- **Files being shared aren't text: video, MP3, pictures, movies…it's all about human interface**
- **Speech recognition still killer app of future (but *will* happen)**
- **3D graphics, shading, texturing mandatory**
- **How have our architectures changed to comprehend all this? Not much. We rely on massive GP computing overkill to smash problem instead of finessing it in power-efficient way**

Si Physics Gets Very Difficult

Variation-tolerant design

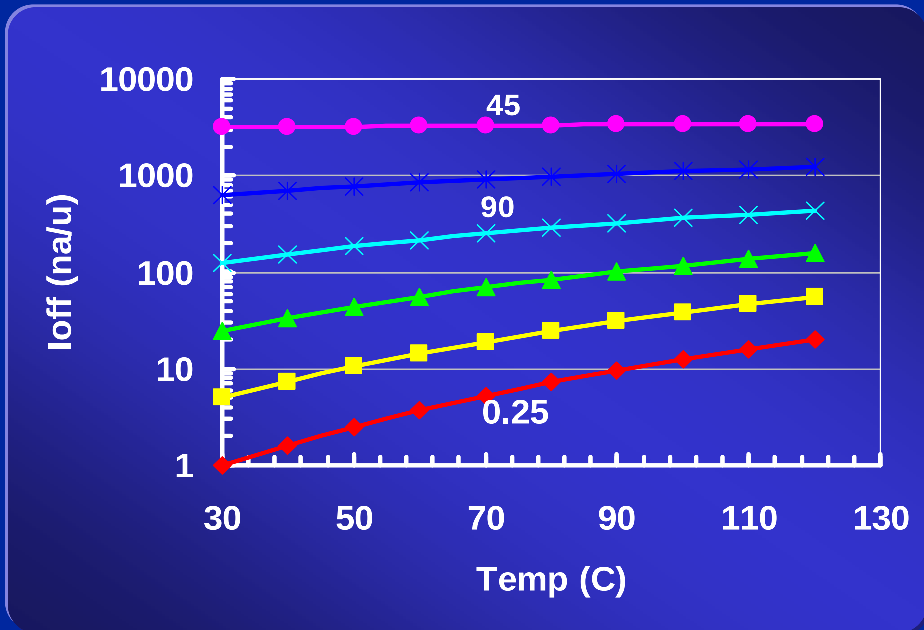
- Statistical doping a problem
- By 2014: 100B transistors: 20B unusable, 10B quit during product lifetime

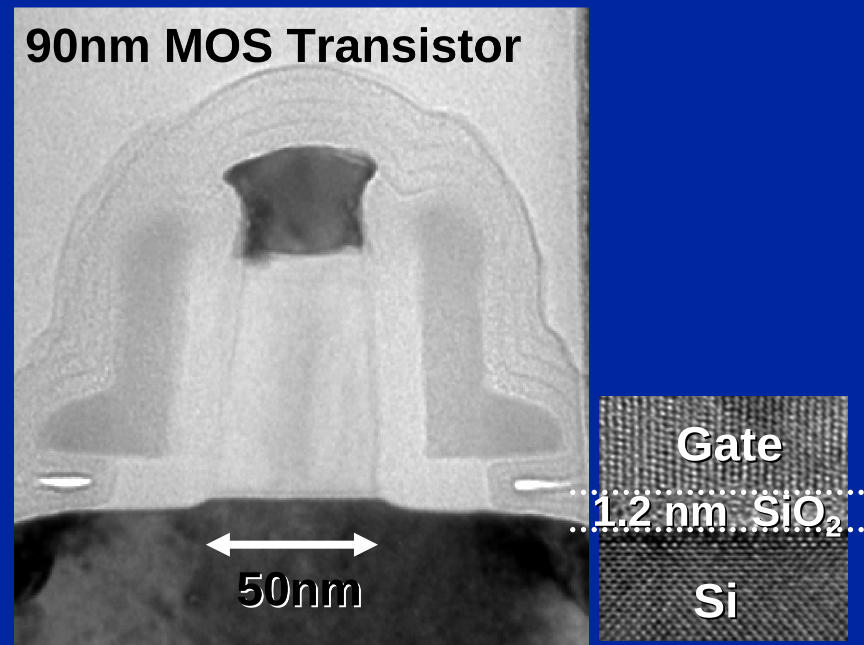
How do you design reliable products based on unreliable components?

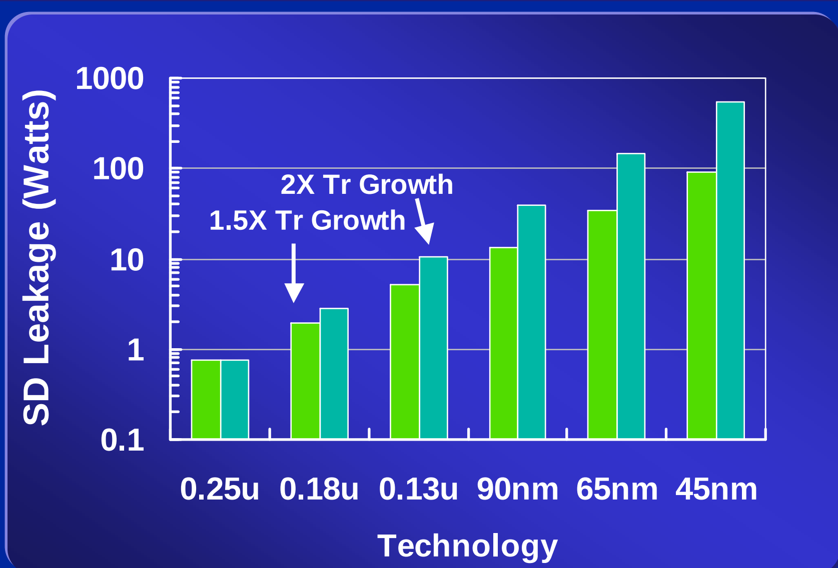
NASA has, but N-mod redundancy not useful here
Need extensive ability to detect, fix, work around errors
Synergistic with multicore? Maybe

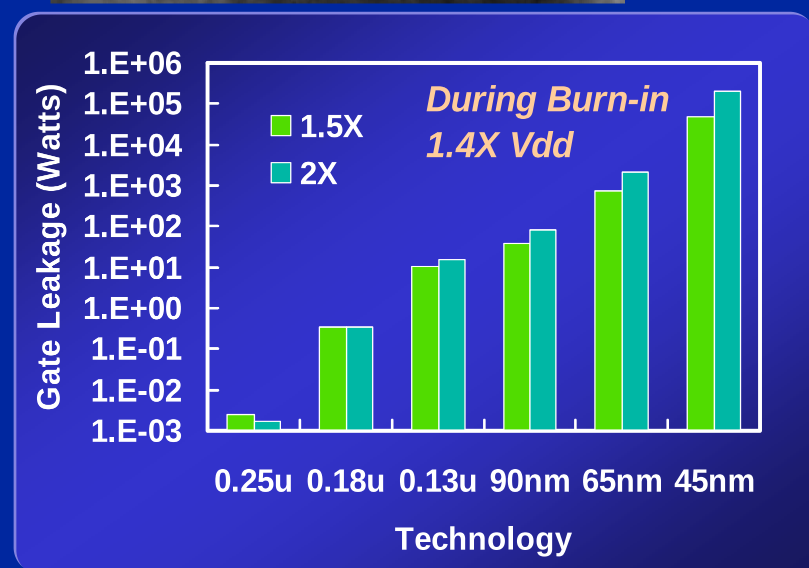
The Leakage(s)…

Shekhar Borkar, Intel, Micro-37



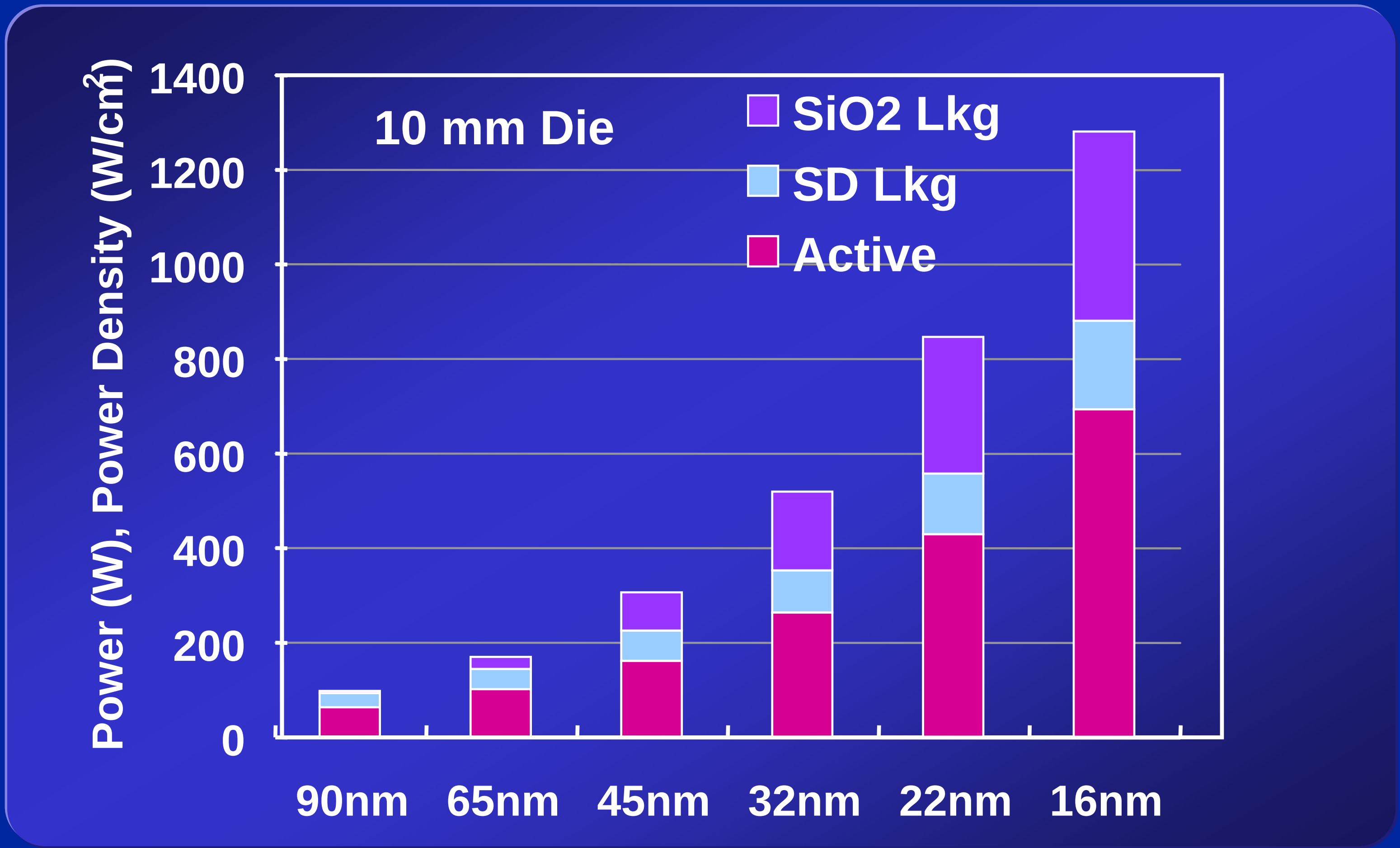






Shekhar Borkar, Intel, Micro-37

Projected Power (unconstrained)



Active and Leakage power will become prohibitive

CMOS Won't Just Die Quietly, Of Course

Other technologies

Some "showstoppers" will turn out to have solutions

- Can probably count on pushing CMOS a generation or two beyond what's visible right now
- But end-game in sight

Future CompArch Mileposts

- Comp arch is <u>system</u>, not just CPU
- Dependability: when emphasis is on system, not CPU, can no longer just "blame the software"; what can we architects do?
  - Systems must become inherently self-checking
    - Like phone system, or internet
  - Systems that survive own design errata
  - Systems that successfully communicate status to naïve users

What will float 1B units/yr industry in 10 years?

Lessons from cell phone industry

CompArch Mileposts cont.

- More single-thread efforts, focused on low power & multicore use
- Multicore systems, caches, memories, communications, whatever compilers want beyond traditional fast ISAs, all with power as 1st order constraint
- Reconfigurable fabrics
  - Standalone & as adjuncts to existing ISAs
- Streaming
  - Any technology that fits silicon so well must be explored!

Encouraging Signs @ ISCA07

Comp arch confs need to encourage & accept papers that assume imperfect implementation tech and having goals other than single thread time-to-solution perf

Good News: ISCA 2007 on its way to grappling with this new diversity (instead of arguing over SPEC benchmarks ☺)

2007 "new" topics:

- Transactional memory
- Power management
- Virtualization
- Quantum/physics engines
- Security

And some trusty old warhorses:

- Branch prediction
- Caches & memory
- Networking
- Fine grain parallelism

Intel seems awfully well-represented here…where is everybody?

More Work Needed

In order of decreasing urgency

Research Needed

1. CMOS end-game electricals problems
2. Multicore SW
3. Power/thermals management
4. Thread and manycore sync: SW needs help
5. Expand synergies between embedded & GP
6. Design-in-the-Large
7. Grand Challenges
8. New technologies like reconfig fabrics, streaming machines, quantum, bio, nano

Conclusions

1. **Keep developing CMOS engines**
   - But be ready for its senility phase
   - Multicore challenges abound (more SW than HW, but SW needs help)
2. **Ready all plausible alternative techs**
3. **Absorb lessons from cell phones & embedded arenas**
4. **Remember that Moore's Law is pushing us into Large System Design**
   - Design accordingly
5. **When compatibility mandate has run its course, new opportunities arise: be ready**

Q&A