Computer Architecture at Berkeley

Professor John Kubiatowicz

What is Computer Architecture?

... the attributes of a [computing] system as seen by the programmer, *i.e.* the conceptual structure and functional behavior, as distinct from the organization of the data flows and controls the logic design, and the physical implementation.

– Amdahl, Blaaw, and Brooks, 1964

-- **Organization of Programmable Storage**

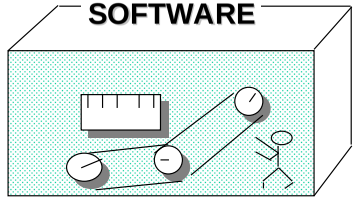
-- **Data Types & Data Structures: Encodings & Representations**

-- **Instruction Set**

-- **Instruction Formats**

-- **Modes of Addressing and Accessing Data Items and Instructions**

-- **Exceptional Conditions**



No!

- A Computer architect is like the architect of a building:
  - Must know building materials/properties:
    - transistors, circuits, wires
    - power consumption
  - Must know and understand construction styles (arches, reinforced concrete, etc.):
    - Hardware internals
    - Compilers
    - Networking
- Computer architecture is really SYSTEM architecture!
- Lessons of RISC: it is the complete, end-to-end system that must serve its purpose

Today: building materials prevalent

- Originally: worried about squeezing the last ounce of performance from limited resources
- Today: worried about an abundance (embarrassment) of riches?
  - Billions of transistors on a chip (17nm Yeah!)
  - Microprocessor Report articles wondering if all the lessons of RISC are now irrelevant
- Moore's laws: exponential growth of everything
  - Transistors, Performance, Disk Space, Memory Size
- So, what matters any more????

Examples of "Moore's Law's"

- Processor
  - logic capacity: about 30% per year
  - clock rate: about 20% per year
  - Performance: about 50-60% per year (2x in 18 months)
- Memory
  - DRAM capacity: about 60% per year (4x every 3 years)
  - Memory speed: about 10% per year
  - Cost per bit: improves about 25% per year
- Disk
  - capacity: about 60% per year

Simple answers: Performance is the wrong metric

- Complexity:
  - more than 50% of design teams now for verification
- Power
  - Processor designs hampered in performance to keep from melting
  - Why 3 or 4 orders of magnitude difference in power consumption between custom hardware and general Von Neuman architectures?
- Energy
  - Portable devices
- Scalability, Reliability, Maintainability
  - How to keep services up 24x7?
- Performance ("Cost conscious")
  - how to get good performance without a lot of power, complexity, etc.

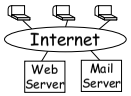
Shift in Focus

- Human time and attention, not processing or storage, are *the* limiting factors
- Givens:
  - Vast diversity of computing devices (PDAs, cameras, displays, sensors, actuators, mobile robots, vehicles); No such thing as an "average" device
  - Unlimited storage: everything that can be captured, digitized, and stored, will be
  - Every computing device is connected in proportion to its capacity
  - Devices are predominately compatible rather than incompatible (plug-and-play enabled by on-the-fly translation/adaptation)

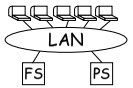
Case in point: Goals of the Endeavor Project

- Enhancing *understanding*
  - Dramatically more convenient for people to interact with information, devices, and other people
  - Supported by a "planetary-scale" Information Utility
    - Stress tested by challenging applications in decision making and learning
    - New methodologies for design, construction, and administration of systems of unprecedented scale and complexity
  - Figure of merit: how effectively we amplify and leverage human intellect
- A pervasive Information Utility, based on "fluid systems technology" to enable new approaches for problem solving & learning

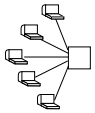
Computing Evolution



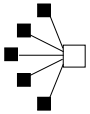
PC + Internet
Shared servers/
Dedicated computing
Remote access



Workstation
Shared servers/
Dedicated computing
Remote access



Time Sharing
Shared resources
Remote access

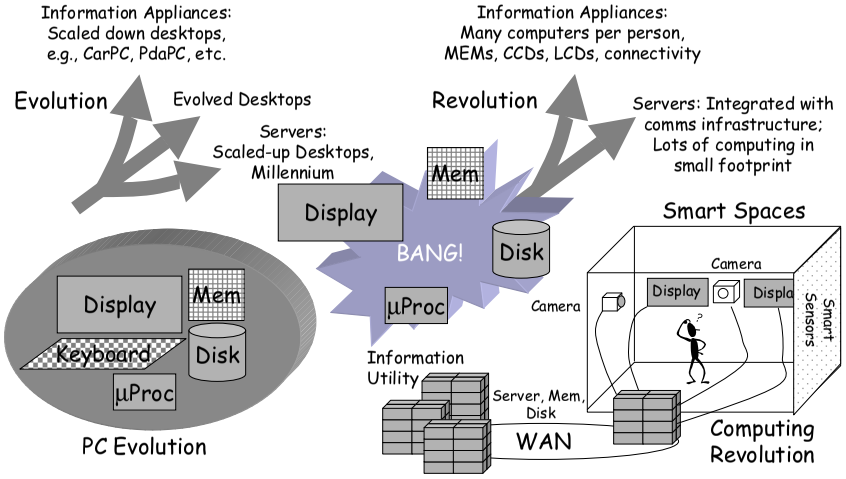


Remote Job Entry
One at a time use
Remote access to machine

Batch processing
One at a time use
User comes to machine

Increasing Freedom from Colocation
Increasing Sharing & Distribution
Increasing Personalization
Increasing Ratio of Computers:Users

Computing Revolution: eXtreme Devices



What does the future of Architecture hold?

- PostPC Era will be driven by 3 technologies:
  - Networking: Everything connected to everything else
  - Ubiquitous computing
    - e.g., successors to PDA, cell phone, wearable computers
    - Processing everywhere
    - Sensors everywhere
  - Infrastructure to Support such Devices
    - e.g., successor to Big Fat Web Servers, Database Servers

Major Emphases:

- Addressing the Processor/Memory Gap
- Power
- Reconfigurablity
- SAM (Scalability, Availability, Maintainability)
- Introspection and dynamic adaptability
- Quantum Computing (hobby of mine)

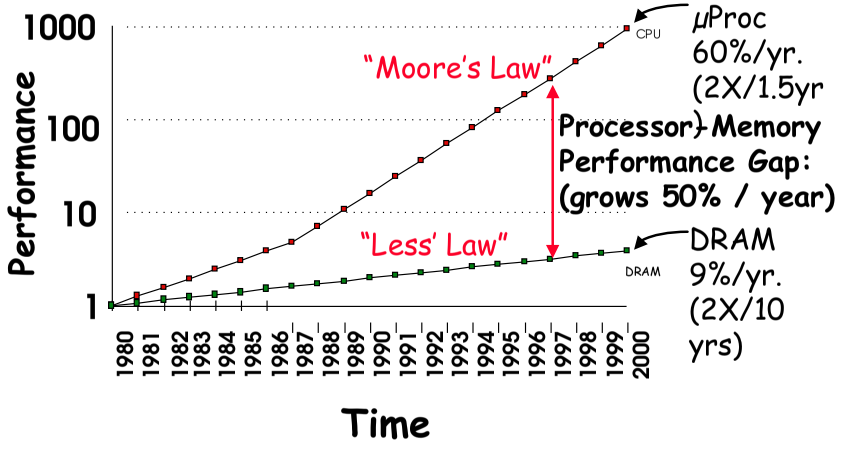
Some Projects:

- IRAM: "Intelligent RAM project"
  - Patterson, Yelick, Kubiatowicz
- BRASS: Reconfigurable Computing
  - Wawyrznek
- ISTORE: "The Intelligent Storage Project"
  - (Actually, Introspective Storage project)
  - Patterson, Yelick, Kubiatowicz
- DynaComp: "Introspective Computing Project"
  - Kubiatowicz
- OceanStore: Utility Storage
  - Kubiatowicz

IRAM: Intelligent RAM

David Patterson, Kathy Yellick,
John Kubiatowicz

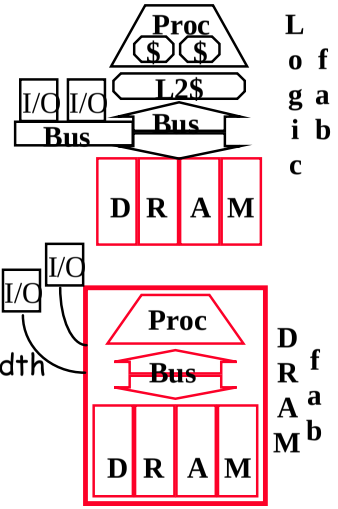
Moore's Law vs Processor Memory Gap



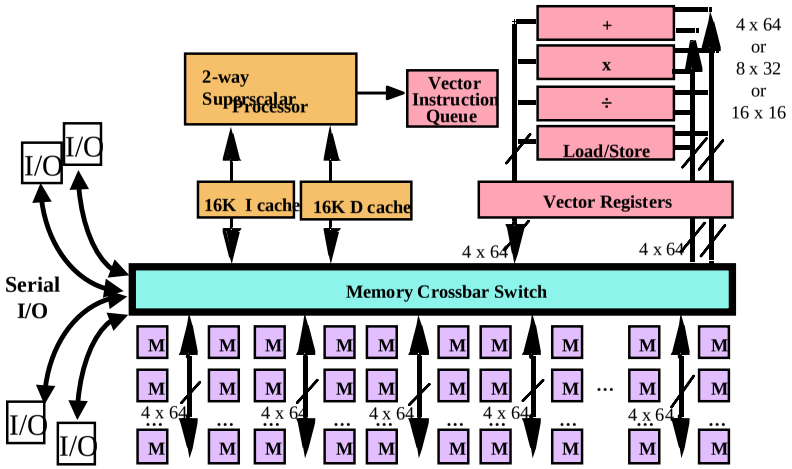
IRAM Vision Statement

Microprocessor & DRAM on a single chip:

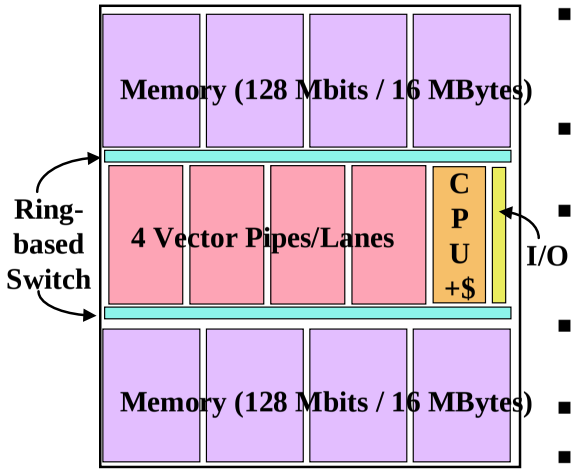
- on-chip memory latency 5-10X, bandwidth 50-100X
- improve energy efficiency 2X-4X (no off-chip bus)
- serial I/O 5-10X v. buses
- smaller board area/volume
- adjustable memory size/width



V-IRAM1: 0.18 µm, Fast Logic, 200 MHz
1.6 GFLOPS(64b)/6.4 GOPS(16b)/16MB



Tentative VIRAM-1 Floorplan



- 0.18 µm DRAM 16-32 MB in 16 banks x 256b
- 0.18 µm, 5 Metal Logic
- ¡ 200 MHz MIPS IV, 16K I$, 16K D$
- ¡ 4 200 MHz FP/int. vector units
- die: ¡ 20x20 mm
- xtors: ¡ 130-250M
- power: ¡ 2 Watts

BRASS
(Berkeley Reconfigurable Architectures, Software and Systems)

John Wawyrznek

BRASS

Early successes of FPGA based computing machines.

- DEC PRL PAM achieves fastest RSA implementation beating out supercomputers and custom ICs
- SRC Splash performs DNA sequence matching at 300X Cray2 speed, and 200X 16K CM2.

Density advantages (and dynamic reconfiguration) has motivated a new interest in FPGAs as computing devices.

BRASS Project Motivation

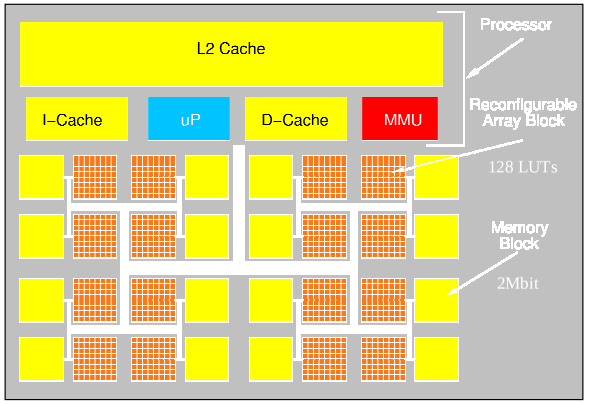
What would make a reconfigurable device a general purpose computing platform?

- Device must solve the **entire problem**, not just the computational kernel.
- Must gracefully handle **heavy memory** bandwidth and capacity demands.
- Must support a **convenient programming** environment.
- **Software must survive** hardware evolution.

Answer:

- Hybrid Processor
  - Reconfigurable array + MPU core + memory system
    - gives best of temporal (MPU) versus spatial (RC array) organizations
    - conventional runtime environment (OS, etc.)
    - convenient development path
- Compute Model ("architecture")
  - critical for:
    - application longevity
    - rapid insertion of new hardware
    - hardware resource virtualization

Architecture Target

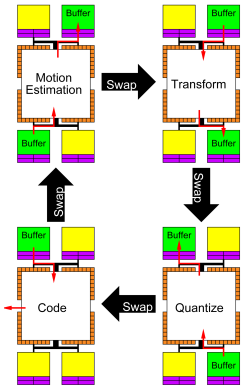


- Integrated RISC core + memory system + reconfigurable array.
- Combined RAM/Logic structure.
- Rapid reconfiguration with many contexts.
- Large local data memories and buffers.
- These capabilities enable:
  - **hardware virtualization**
  - **on-the-fly specialization**

SCORE: Stream-oriented computation model

Goal: Provide view of reconfigurable hardware which exposes strengths while abstracting physical resources.

- Computations are expressed as data-flow graphs.
- Graphs are broken up into **compute pages**.
- Compute pages are linked together in a data-flow manner with **streams**.
- A **run-time manager** allocates and schedules pages for computations and memory.



ISTORE: Intelligent Storage

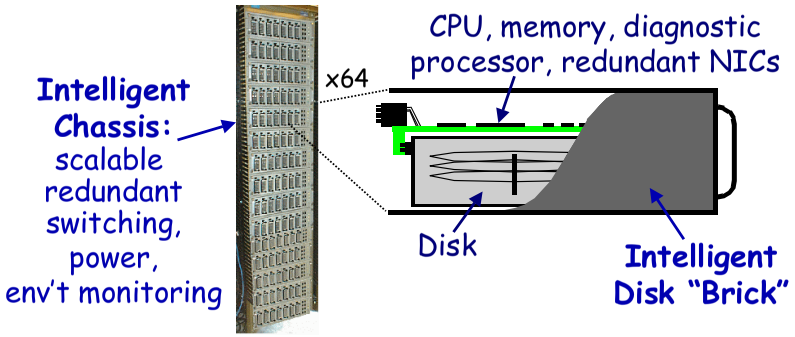
David Patterson
Kathy Yelick, John Kubiatowicz

ISTORE Hardware Vision

- System-on-a-chip enables computer, memory, redundant network interfaces without significantly increasing size of disk
- Target for + 5-7 years:
  - building block: 2006 MicroDrive integrated with IRAM
    - 9GB disk, 50 MB/sec from disk
    - connected via crossbar switch
  - 10,000+ nodes fit into one rack!

ISTORE-1 Hardware Prototype

- Hardware architecture: **plug-and-play intelligent devices with integrated self-monitoring, diagnostics, and fault injection hardware**
  - intelligence used to collect and filter monitoring data
  - diagnostics and fault injection enhance robustness
  - networked to create a scalable shared-nothing cluster

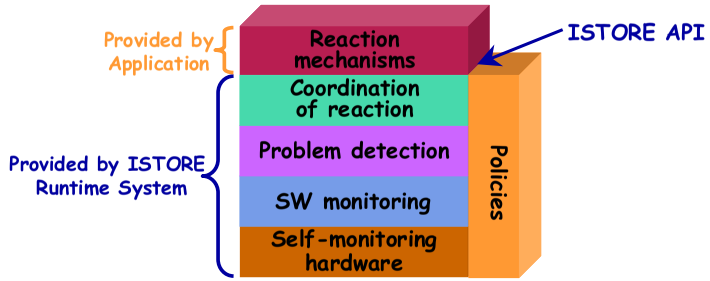


ISTORE Software Approach

- Two-pronged approach to providing reliability:
  1) **reactive self-maintenance:** dynamic reaction to exceptional system events
     - self-diagnosing, self-monitoring *hardware*
     - software *monitoring* and problem *detection*
     - automatic *reaction* to detected problems
  2) **proactive self-maintenance:** continuous online self-testing and self-analysis
     - automatic *characterization* of system components
     - *in situ fault injection*, *self-testing*, and *scrubbing* to detect flaky hardware components and to exercise rarely-taken application code paths before they're used

Reactive Self-Maintenance

- ISTORE defines a layered system model for monitoring and reaction:



- ISTORE API defines interface between runtime system and app. reaction mechanisms
- Policies define system's monitoring, detection, and reaction behavior

Proactive Self-Maintenance

- Continuous online self-testing of HW and SW
  - detects flaky, failing, or buggy components via:
    - **fault injection:** triggering hardware and software error handling paths to verify their integrity/existence
    - **stress testing:** pushing HW/SW components past normal operating parameters
    - **scrubbing:** periodic restoration of potentially "decaying" hardware or software state
  - automates preventive maintenance
- Dynamic HW/SW component characterization
  - used to adapt to heterogeneous hardware and behavior of application software components

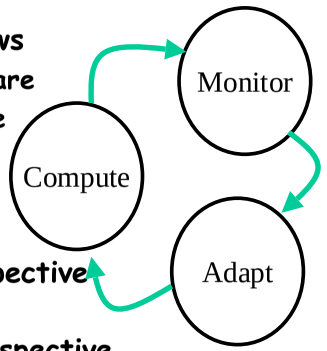
DynaComp:

The Berkeley Introspective Computing Project

John Kubiatowicz

Introspective Computing

- **Biological Analogs for computer systems:**
  - **Continuous adaptation**
  - **Insensitivity to design flaws**
    - **Both hardware and software**
    - **Necessary if can never be sure that all components are working properly…**
- **Examples:**
  - **ISTORE -- applies introspective computing to disk storage**
  - **DynaComp -- applies introspective computing at chip level**
    - **Compiler always running and part of execution!**



Introspective Computing

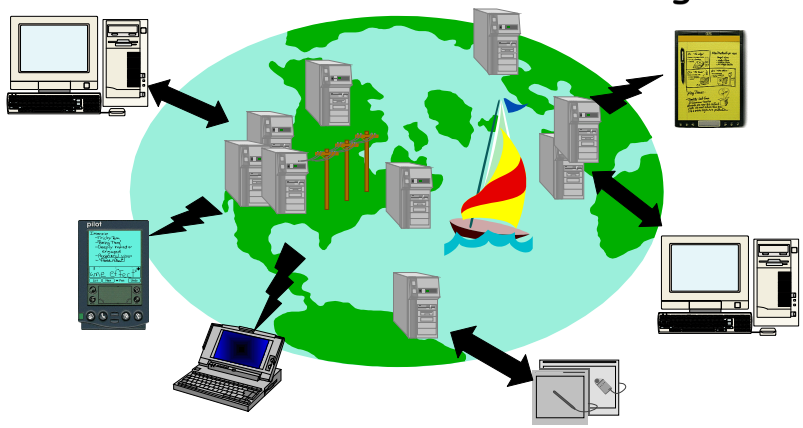
- Two high-level goals:
  - Performance:
    - squeeze last ounce of performance through on-line compiler analyses
    - Better adaptation to extremes of performance
    - Better use of parallel resources (dynamic parallelism)
  - Reliability, Maintainability:
    - Automatic recognition of hardware flaws through use of proof checking (such as PCC) and redundancy
    - Adaptation to "compile around" problems

Introspective Prototype

- Multiprocessor on a chip + some support for monitoring
- Hierarchical Compiler technologies:
  - Compiling can occur at different times and at different levels of completeness

OceanStore: The Oceanic Data Utility:

Global-Scale Persistent Storage

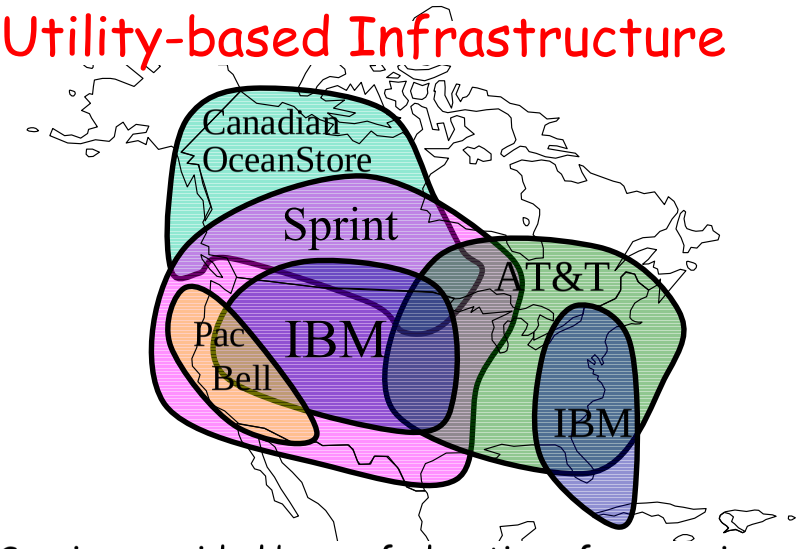


John Kubiatowicz

Ubiquitous Devices ⇒ Ubiquitous Storage

- Consumers of data move, change from one device to another, work in cafes, cars, airplanes, the office, etc.
- Properties *REQUIRED* for Endeavour storage substrate:
  - Strong Security: data must be encrypted whenever in the infrastructure; resistance to monitoring
  - Coherence: too much data for naïve users to keep coherent "by hand"
  - Automatic replica management and optimization: huge quantities of data cannot be managed manually
  - Simple and automatic recovery from disasters: probability of failure increases with size of system
  - Utility model: world-scale system requires cooperation across administrative boundaries

Utility-based Infrastructure



- Service provided by confederation of companies
  - Monthly fee paid to one service provider
  - Companies buy and sell capacity from each other

OceanStore Assumptions

- Untrusted Infrastructure:
  - Infrastructure is comprised of untrusted components
  - Only cyphertext within the infrastructure
  - Must be careful to avoid leaking information
- Mostly Well-Connected:
  - Data producers and consumers are connected to a high-bandwidth network most of the time
  - Exploit mechanism such as multicast for quicker consistency between replicas
- Promiscuous Caching:
  - Data may be cached anywhere, anytime
  - Global optimization through tacit information collection
- Operations Interface with Conflict Resolution:
  - Applications employ an operations-oriented interface, rather than a file-systems interface
  - Coherence is centered around conflict resolution

Interesting Issue: Rapid Update in an Untrusted Infrastructure

- Requirements:
  - Scalable coherence mechanism which provides performance even though replicas widely separated
  - Operate directly on encrypted data
  - Updates should not reveal info to untrusted servers
- OceanStore Technologies:
  - Operations-based interface using conflict resolution
  - Use of incremental cryptographic techniques: No time to decrypt/update/re-encrypt
  - Use of oblivious function techniques to perform this update (fallback to secure hardware in general case)
  - Use of automatic techniques to verify security protocols

Conclusion:

- Computer Architecture Research is targeting problems of the 21st century
  - The Network is Central
- Users matter, not hardware
- Hardware Issues:
  - Complexity, Power, Availability, Fault Tolerance
- Software Issues:
  - Complexity, Adaptability, Availability, Maintainability, Scalability