# CS184a: Computer Architecture (Structures and Organization)

Day1: September 25, 2000

Introduction and Overview



# Today

- Matter Computes
- Architecture Matters
- This Course (short)
- Who am I? Where did I come from? What do I want?
- Unique Nature of This Course
- Relation to other courses
- More on this course

# Review: Two Universality Facts

- Turing Machine is Universal
  - We can implement any *computable* function with a TM
  - We can build a single TM which can be programmed to implement any computable function
- NAND gate Universality
  - We can implement any computation by interconnecting a sufficiently large network of NAND gates



# Review: Matter Computes

- We can build NAND gates out of:
  - transistors (semicondutor devices)
    - physical laws of electron conduction
  - mechanical switches
    - basic physical mechanics
  - …many other things

# Starting Point

- Given sufficient raw materials:
  - can implement any computable function
- Our goal in computer architecture
  - is **not** to figure out how to compute new things
  - rather, it is an *engineering* problem



# Engineering Problem

- Implement a computation:
  - with least resources (in fixed resources)
    - with least cost
  - in least time (in fixed time)
  - with least energy
- Optimization problem
  - how do we do it best

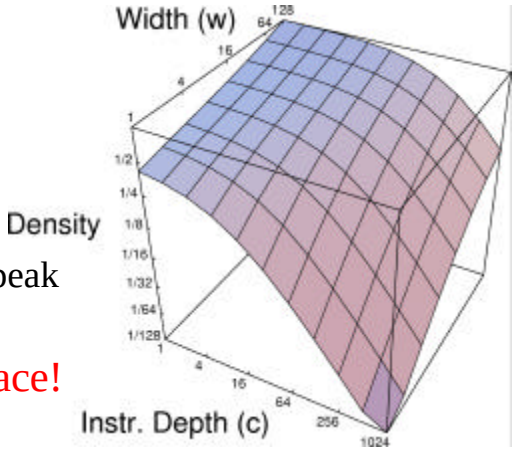
# Architecture Matters

- How much difference is there between architectures?
- How badly can I be wrong in implementing/picking the wrong architecture?
- How efficient is the IA-64?
  - Is there much room to do better?
- Is architecture done? A solved problem?



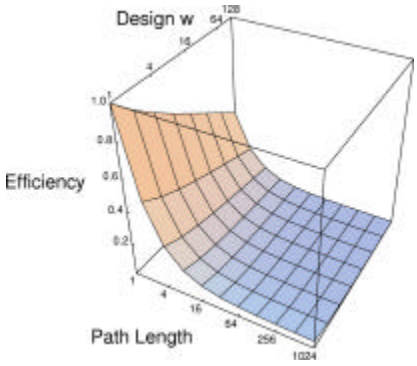
# Peak Computational Densities from Model

- Small slice of space
  - only 2 parameters
- 100× density across

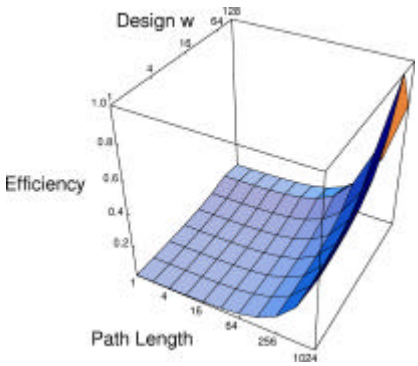



- Large difference in peak densities
  - large design space!

# Yielded Efficiency




FPGA ($c$=$w$=1)




"Processor" ($c$=1024, $w$=64)

- Large variation in **yielded** density
  - large design space!



# Architecture Not Done

- Many ways, not fully understood
  - design space
  - requirements of computation
  - limits on requirements, density...
- Costs are changing
  - optimal solutions change
  - creating new challenges and opportunities

## Architecture Not Done

- Not here to just teach you the forms which are already understood
  - (though, will do that and give you a strong understanding of their strengths and weaknesses)
- **Goal**: enable you to design and synthesize new and better architectures



## This Course (short)

- How to organize computations
- Requirements
- Design space
- Characteristics of computations
- Building blocks
  - compute, interconnect, retiming, instructions, control
- Comparisons, limits, tradeoffs

# This Course

- Sort out:
  - Custom, RISC, SIMD, Vector, VLIW, Multithreaded, Superscalar, EPIC, MIMD, FPGA
- Basis for design and analysis
- Techniques
- [more detail at end]



# Who Am I?

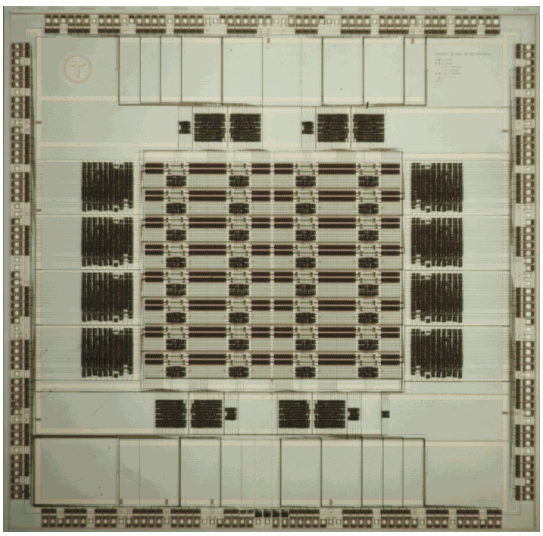
- Academic History:
  - LSMSA [state gifted high school, LA]
    - *Real Genius* summer before senior year
  - $(MIT)^3$
  - UCB postdoc
    - co-ran BRASS group
  - Caltech
    - start Sept. 1999

# What have I done?

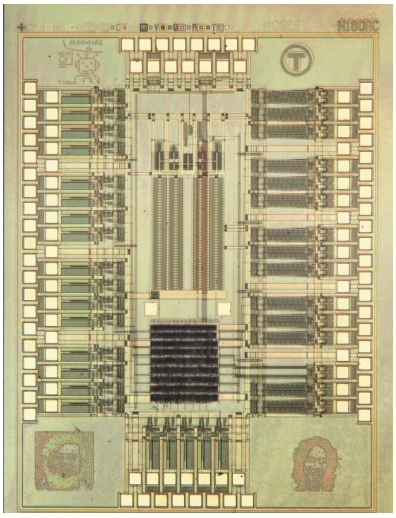
- Started research as a UROP
  - (Undergrad. Researcher…like SURF)
- Transit Project
  - RN1, TC1, Metro, Mlink, MBTA
  - parallel theory and architecture
  - SB on fat-tree networks
  - SM on fault-tolerant, low-latency, large-scale routing networks



# RN1



# TC1





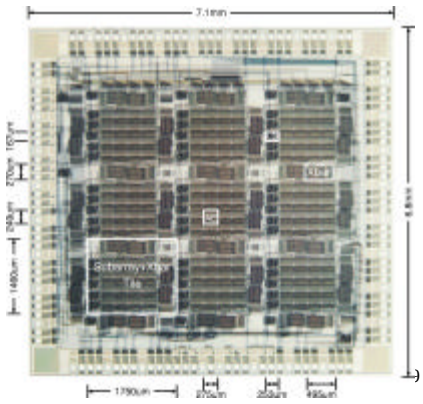
# Reinventing Computing

- FPGA-coupled processor
- DPGA (first multicontext FPGA)
- TSFPGA
- MATRIX
- How compare FPGAs and Processors?
- PhD - Reconfigurable Architectures for General-Purpose Computation

# MIT DPGA Prototype

- $w$=1, $d$=1, $c$=4
  $p$ small
- 9 ns cycle, 1.0µm
  - LUT
  - Interconnect
  - Context read
- Team:
  - Jeremy Brown,Derrick Chen
  - Ian Eslick, Ethan Mirsky
  - Edward Tau
  - André DeHon

- Automatic CAD
  - multicontext evaluation
  - FSM partitioning/mapping

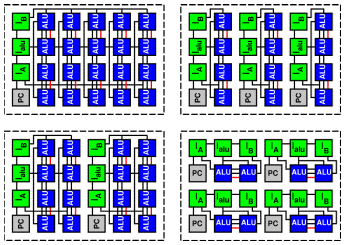


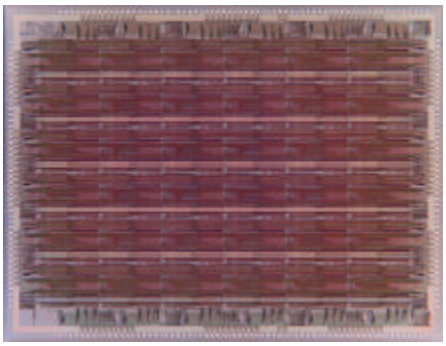
FPD'95



# MIT MATRIX Testchip

- Efficient/flexible word size and depth
- Base unit:
  - $c$~4 or 256, $d$~1 or 128
  - $w$~8 expandable
- 50MHz, 0.6µm
- Team:
  - Ethan Mirsky
  - Dan Hartman
  - André DeHon





FCCM'96/HotChips'97

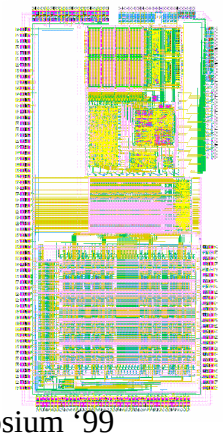
# BRASS

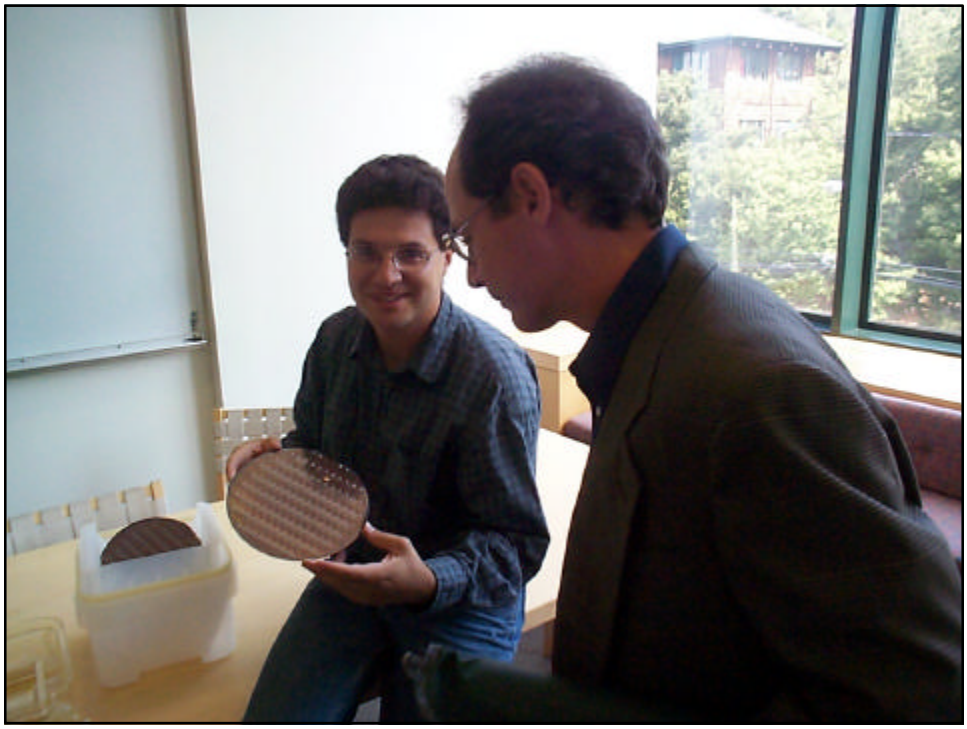
- Processor + FPGA Architecture
- HSRA
  - fast array, balance interconnect, retiming
  - mapping focus
- DRAM integration (heterogeneous arch.)
- SCORE
  - Models/architectural abstractions for RC and beyond



# UCB HSRA Testchip

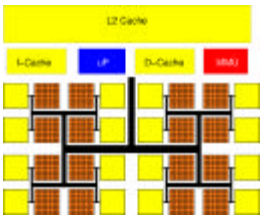
- Spatial, bit-level
  - $c$=1, $w$=1, $d$=8, $p$=2/3
- 250MHz, 0.4µm DRAM
- 2Mbit DRAM macro
  - $c$~50, $d$~16K, $w$~64
- Team:
  - William Tsu, Stelios Perissakis, Randy Huang, Atul Joshi, Michael Chu, Kip Macy, Varghese George, Tony Tung, Omid Rowhani, Norman Walker, John Wawrzynek, André DeHon
- Automatic retiming
  - accommodate interconnect pipelining





# BRASS RISC+HSRA (heterogeneous mix)

- Integrate:
  - temporal (processor)
  - spatial (HSRA)
  - DRAM
    - instruction
    - data retiming




- Ideas:
  - best of both worlds temporal/spatial
  - exploit 10× DRAM density
  - SCORE
    - manage spatial pages as virtual resources (like virtual memory)
  - Compute model→ Language → Mapping → Scheduling run-time

# Silicon Spice

- Founded 1997
  - by two of my MIT/RC M.Eng. Students
  - commercialize reconfigurable computing ideas
- Focus on telecommunication solutions
- consult for
- Acquired by Broadcom for $1.2B last month
- CALISTO 240 channel, single-chip VoIP



# What do I want?

- Develop systematic design
- Parameterize design space
  - adapt to costs
- Understand/capture req. of computing
- Efficiency metrics
  - (similar to information theory?)

# What do I want?

- Research vectors:
  - architecture space
  - interconnect (beyond one/few PE per die)
  - SCORE (beyond ISA model)
  - heterogeneous architectures (beyond monolithic, homogeneous components)
  - molecular electronics (beyond silicon)



# Uniqueness of Class

# Not a Traditional Arch. Class

- Traditional class
  - focus RISC Processor
  - history
  - undergraduate class on uP internals
  - then graduate class on details
- This class
  - much broader in scope
  - develop design space
  - see RISC processors in context of alternatives



# Authority/History

- ``Science is the belief in the ignorance of experts." -- Richard Feynman
- Traditional Architecture has been too much about history and authority
- Should be more about engineering evaluation
  - physical world is “final authority”
- **Goal**: Teach you to think critically and independently about computer design.

# Tension

- Trying to develop one class to satisfy everyone
  - what cover is sufficiently different should be unique from undergrad. Architecture may have had elsewhere
  - trying to develop the “right” introduction for those seeing for first time
  - not completely sure what background I can assume for Caltech undergrads



# On Prerequisites

- Suggested:
  - CS20 (compute models, universality)
  - EE4 (boolean logic, basic logic circuits)

# Next Few Lectures

- Quick run through logic/arithmetic basics
  - make sure everyone remembers
  - (some see for first time?)
  - get us ready to start with observations about the key components of computing devices
- Trivial/old hat for many
- May be fast if seeing for first time
- (Diagnostic quiz intended to help me tune)



# Experimental: feedback

- Will want feedback on how this works:
  - Need another class as staging to get here?
  - Such class already exist at caltech?
  - Where this class overlap with others at caltech?
  - Too much elementary stuff in class?

## Relation to Other Courses

- CS181 (VLSI)
- EE4 (Fundamentals of Digital Systems)
- CS184 (Architecture)
- CS137 (Electronic Design Automation)
- CS134 (Compilers and Systems)
  - also CS237 (Compiler Design)
- CS20 (Computational Theory)



## Content Overview

- This quarter:
  - building blocks and organization
  - raw components and their consequences
- Next two quarter:
  - abstractions, models, techniques, systems
- Second quarter
  - will include stuff from typical architecture class, but placed in broader context

## Themes (this quarter)

- Design Space
- Parameterization
- Costs
- Change
- Structure in Computations



## This Quarter

- Focus on raw computing organization
- **Not** worry about
  - nice abstractions, models
- Will come back to those next quarter

# Change

- A key feature of the computer industry has been rapid and continual change.
- We must be prepared to adapt.
- For our substrate:
  - capacity (orders of magnitude more)
    - what can put on die, parallelism, need for interconnect and virtualization, homogeneity
  - speed
  - relative delay of interconnect and gates



# Fountainhead Parthenon Quote

"Look," said Roark. "The famous flutings on the famous columns---what are they there for? To hide the joints in wood---when columns were made of wood, only these aren't, they're marble. The triglyphs, what are they? Wood. Wooden beams, the way they had to be laid when people began to build wooden shacks. Your Greeks took marble and they made copies of their wooden structures out of it, because others had done it that way. Then your masters of the Renaissance came along and made copies in plaster of copies in marble of copies in wood. Now here we are making copies in steel and concrete of copies in plaster of copies in marble of copies in wood. Why?"

## Computer Architecture Parallel

- Are we making:
  - copies in submicron CMOS
  - of copies in early NMOS
  - of copies in discrete TTL
  - of vacuum tube computers?



## Big Ideas

- Matter Computes
- Efficiency of architectures varies widely
- Computation design is an engineering discipline
- Costs change ⇒ Best solutions (architectures) change
- Learn to cut through hype
  - analyze, think, critique, synthesize