MASSACHUSETTS INSTITUTE OF TECHNOLOGY
Artificial Intelligence Laboratory
Concurrent VLSI Architecture Group

THE MIT MULTI-ALU PROCESSOR

Stephen W. Keckler

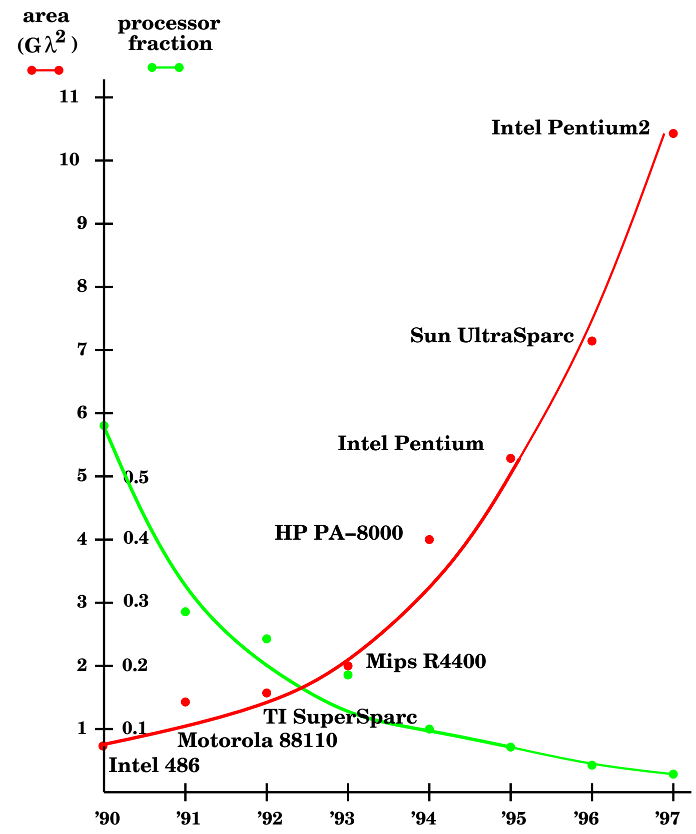
William J. Dally, Andrew Chang,
Nicholas P. Carter, Whay S. Lee

August 25, 1997

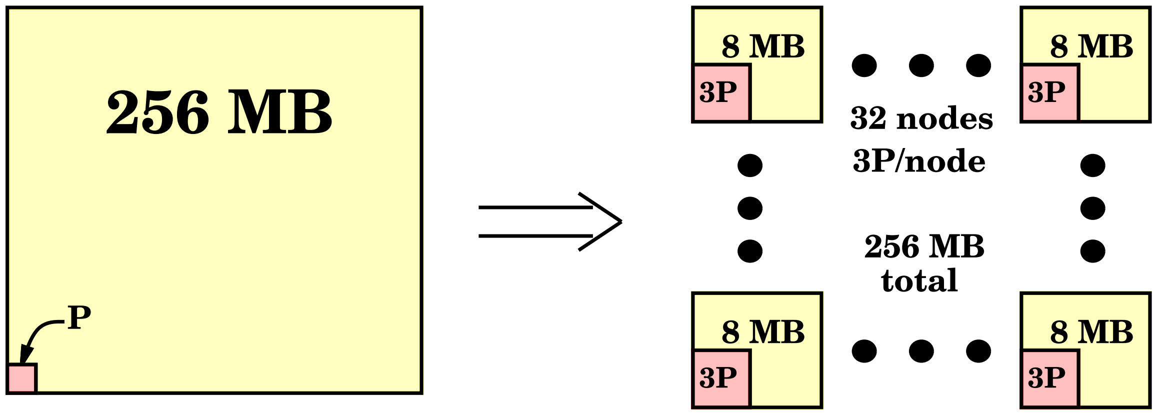
The Vanishing Processor

64 bit processor w/ pipelined FPU (R4600) = $400M\lambda^2$

- λ = 1/2 feature size, process independent
- 4% of today's die
- 0.13% of today's system (256MB)



What to Do?



- Increase processing per unit memory
  - 16% of 256MB system
  - 96 times peak performance
  - 1.5 times silicon area cost

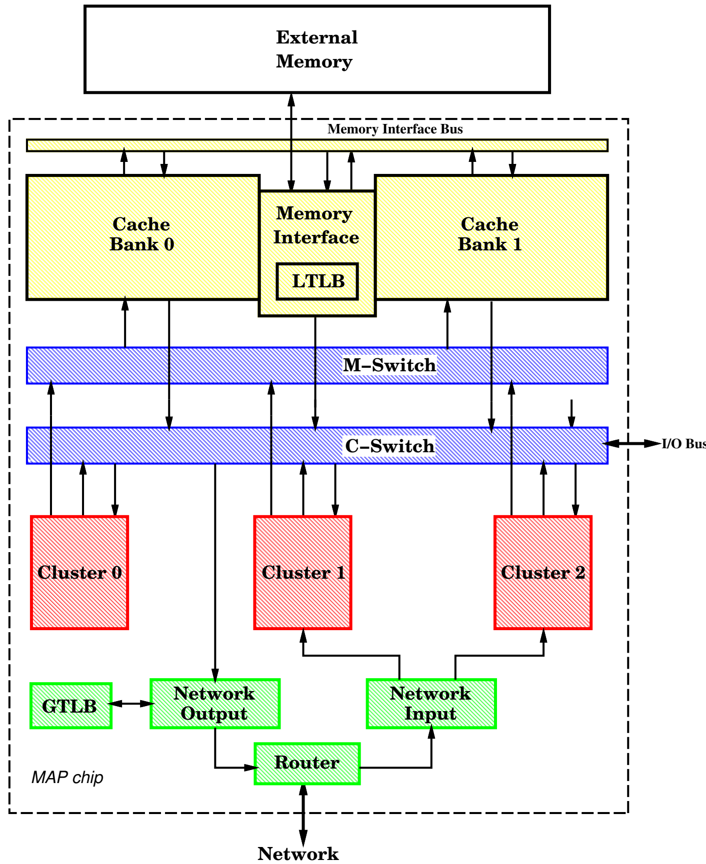
Outline

1. Motivation
2. M-Machine Architecture
   - Instruction Level Parallelism
   - Communication
3. MAP Chip Implementation
4. Summary

The Multi-ALU Node

Highly integrated node (6 chips) containing:

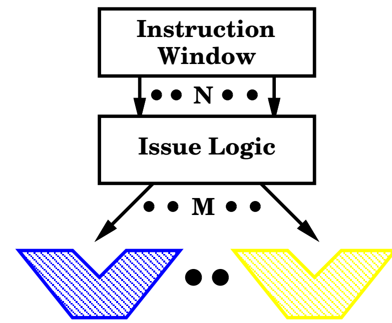
- 8 MBytes of external memory
- MAP Processor
  - 44KB cache (D+I)
  - 4 register files
  - 6 Integer units, 1 FPU
  - NI + Router



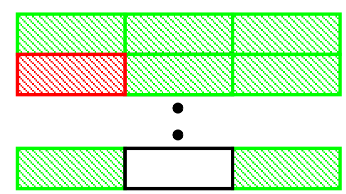
Multiple ALUs

Problem: Current multi-ALU control is inadequate

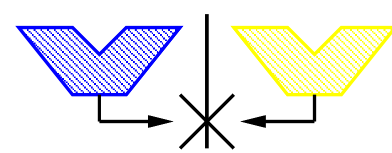
- Superscalar
  - Issue logic and register file at scaling limits
  - Empty issue slots



- VLIW
  - Variable latency
  - Empty issue slots

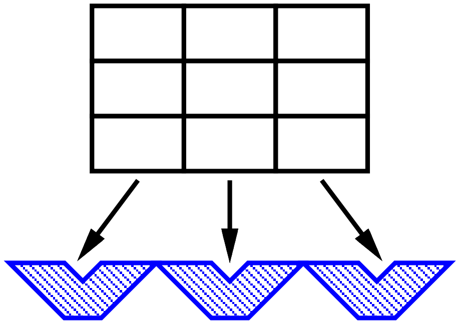


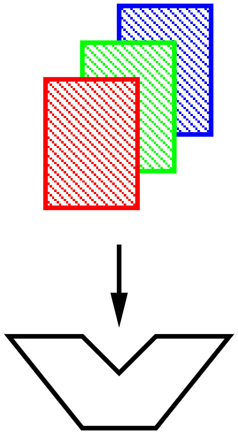
- Multiprocessor
  - Long interaction latency



Processor Coupling

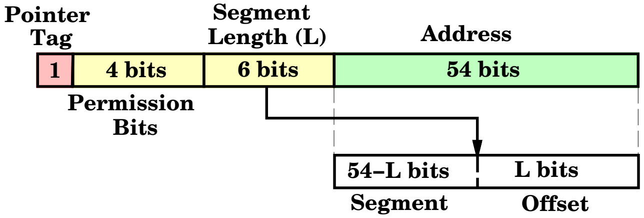
- Compile-time scheduling (across clusters)
  - Instruction Level Parallelism
  - Independent cluster execution
  - Register-register communication
  - Tolerates slip between clusters
- Runtime multithreading (each cluster)
  - Hides variable latencies
  - Exploits slip between clusters



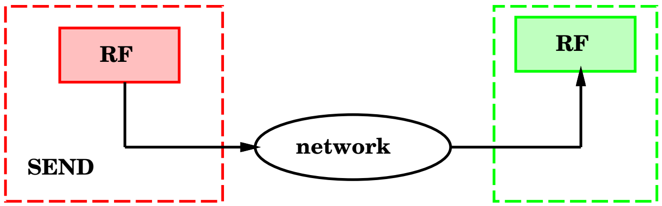


Addressing and Communication

- Guarded Pointers
  - Capability based addressing
  - No table lookup
  - Independent addressing and protection



- Send Instruction
  - Register-register transmit
- Fast message handling
  - Dedicated thread



Node Compilation

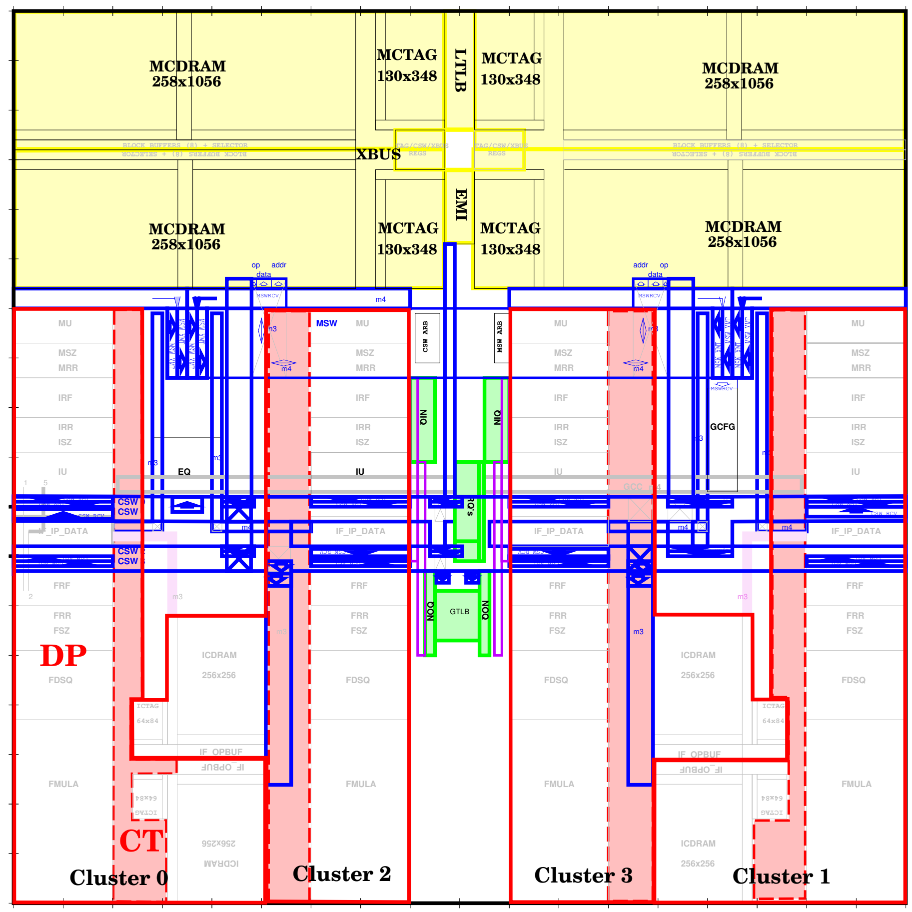
- Multiflow Compiler port
  - C Compiler
  - Optimized single cluster code
  - Statically scheduled code across clusters
- Runtime System
  - C Library
  - Lightweight threads (local and remote)
- SCP Group – Caltech/Syracuse
  - Steve Taylor, Daniel Maskit, Bryan Chow

MAP Chip Implementation

- The Goals
  - Validate mechanisms
  - 4 clusters/1600MFlops
  - 13 million transistors, 100MHz
- The Resources
  - 0.7μm (0.5μm effective), 5-metal process; 18mm $\times$ 18mm die
  - MIT personnel: average 8 students and staff
    - Architecture, logic/circuit design
  - Cadence Spectrum Design
    - Chip assembly, layout
    - Design flow
    - Clock distribution design and analysis

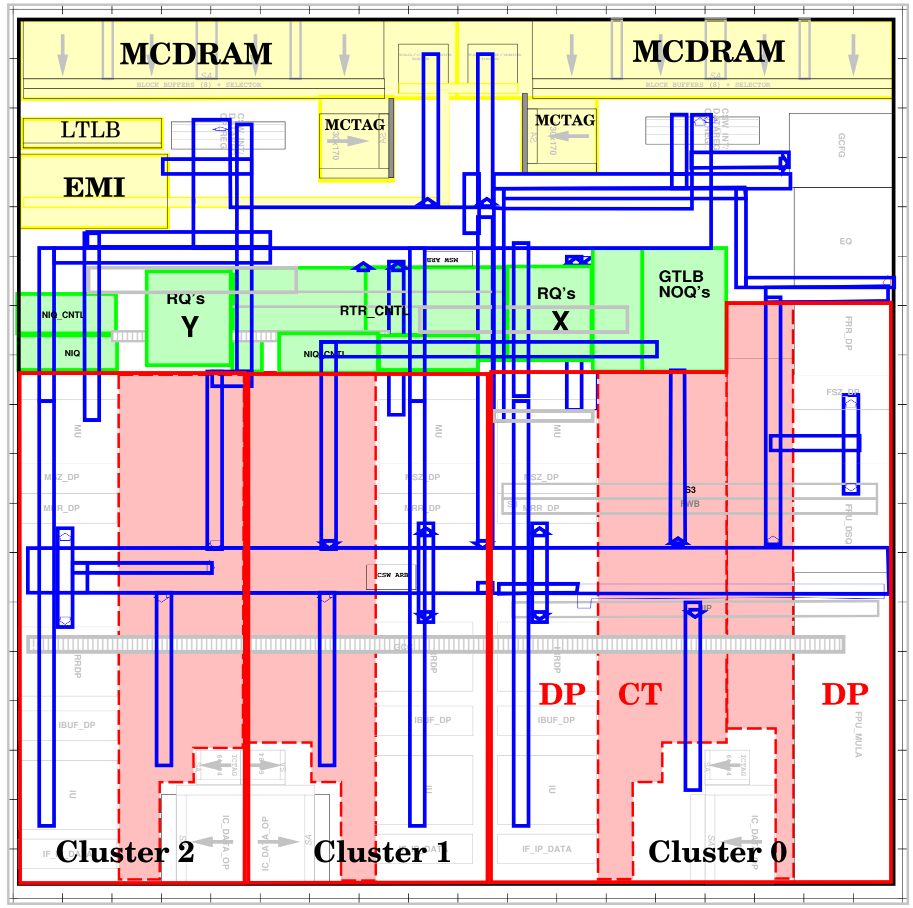
Initial Floorplan: 4/8/96

- 8 IUs, 4 FPUs
- 3D Router
- 160KB SRAM



Final Floorplan: 4/15/97

- 6 IUs, 1 FPU
- 2D Router
- 44KB SRAM



MAP Team Accomplishments

- 5 Million transistor, 64-bit custom microprocessor
- Fully characterized standard cell library
- Composable datapath cell library
- 5 SRAM arrays
- Radix 8 multiplier array w/ domino logic
- IEEE format FPU
  - 4 cycle pipelined MULA
  - 20 cycle DIV/SQRT
- 7 ported register file
- 64 bit custom adder
- Low voltage simultaneous bidirectional pads

Lessons from the Implementation

1. Custom Cell Placement vs. Full Custom datapaths
   - Cost: 40% increased area
   - Benefit: Creation and modification flexibility
2. Architecture greatly affects estimation accuracy
   - 55% utilization in arithmetic control
   - 40% utilization in pipeline control
3. Cadence Spectrum Design was critical
   - Reality check for density
   - Tool flow and physical design
   - Expertise with the fabrication process

Summary

- M-Machine makes better use of silicon area
  - Instruction Level Parallelism: Processor Coupling
  - Coarser grained parallelism
    - Protection: Guarded Pointers
    - Communication: Send instruction
- Clean Sheet Design: Be careful what you wish for!
  - Difficult to predict area, critical path, effort...
- Why build it?
  - Because it is *not* there
- *http://www.ai.mit.edu/projects/cva*