Custom vs Commodity Processors

Bill Dally

October 26, 2005

95% of top 500 machines use commodity processors

Why?

Why 95% Commodity

- Are they faster?
- Are they more cost effective?
- Do they “ride the Moore’s law curve”?

- Or is it just the “easy path”

Definitions

Custom Processor: Processor built specifically for high-end scientific computing. Incorporates high-bandwidth memory system, latency hiding mechanisms, and ability to exploit data- and thread-level parallelism. Intended to scale to large numbers of processors.

Commodity Processor: Processor build primarily for mass market – workstation and enterprise database/web server. Incorporates cache-based memory system, and ability to exploit instruction-level parallelism

Objective

Capacity: Deliver maximum *sustained* performance per lifetime $ at modest scale ($1M-10M)

Capability: Deliver maximum *sustained* performance per lifetime $ at large scale (>$100M).

(You pay for scalability, but not necessarily at small scales)

Custom vs. Commodity – Pros and Cons

- Custom

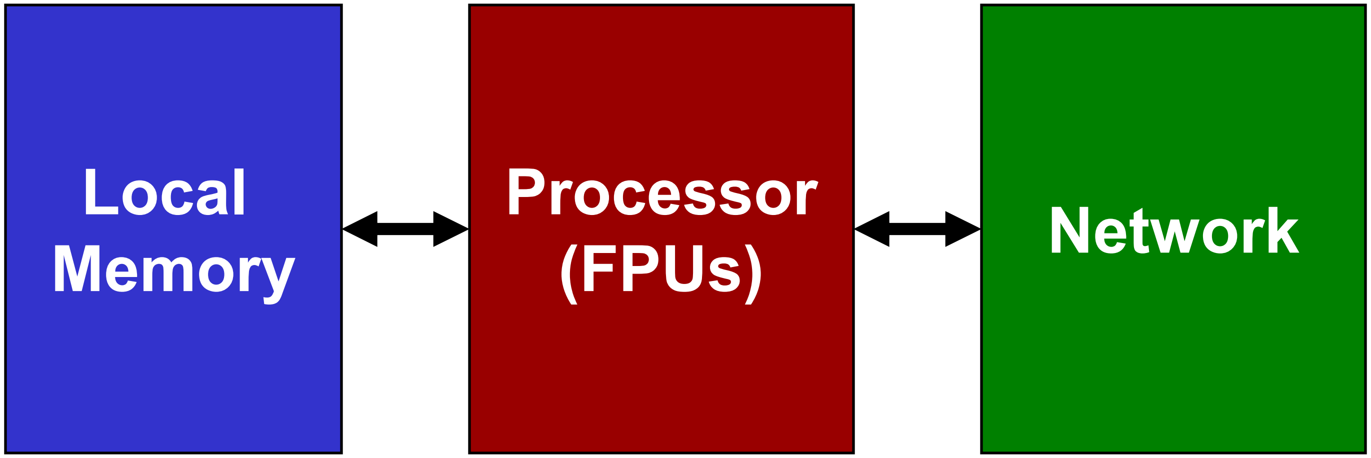
+ High bandwidth memory sys
  - 2-8x raw bandwidth
+ High bandwidth gather
  - 16x bandwidth on irregular acc.
+ Latency hiding
  - 100s of outstanding refs
  - 10x bandwidth*
+ Data/Thread parallelism
  - 100s of FPUs per chip (20x)
- Lower frequency (0.5x)
  - Circuits and process
- Non recurring costs
  - 10K to 100K units

Commodity

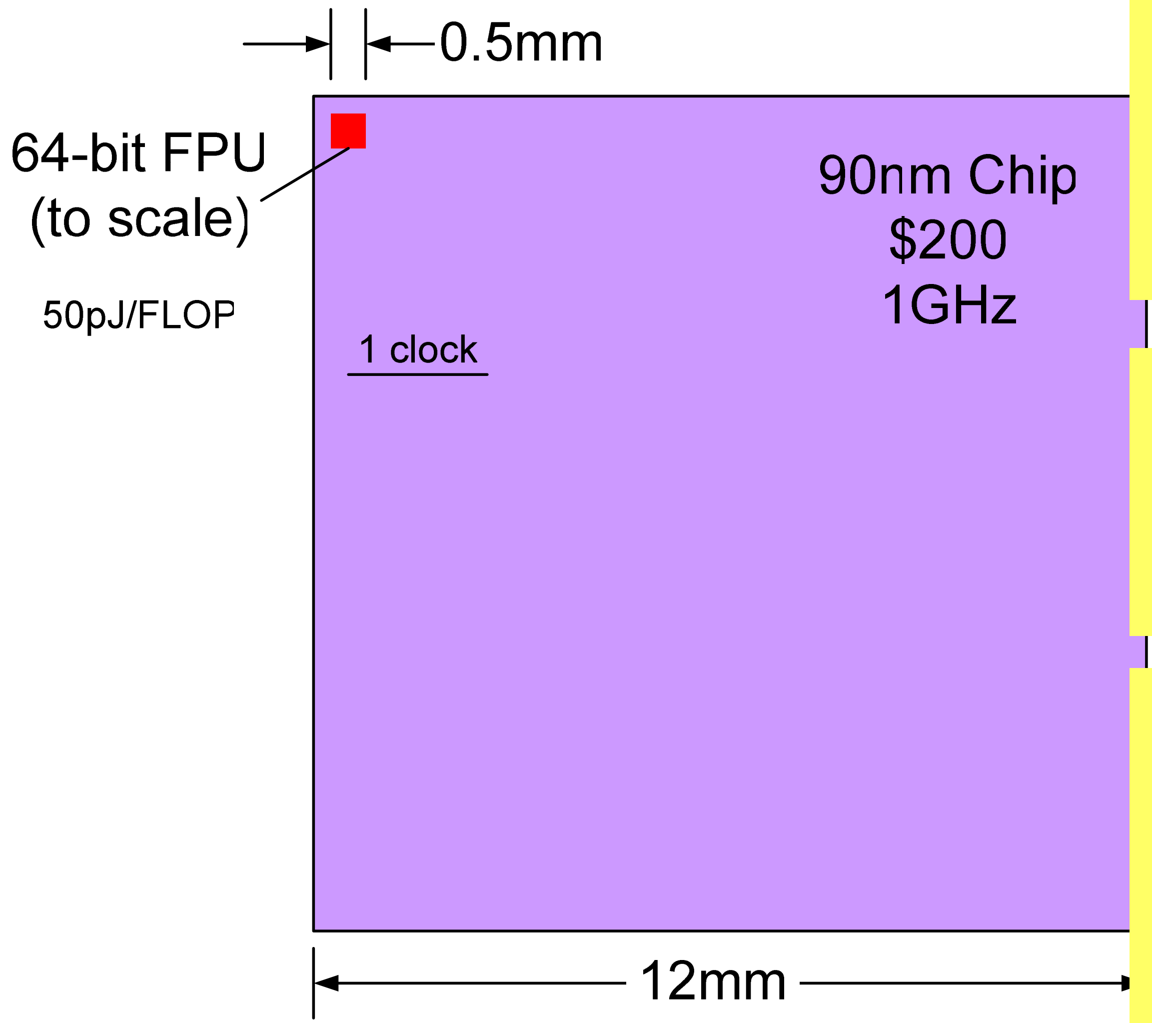
+ Better process
  - 1.4x freq
+ Better circuits
  - 1.4x freq
+ Amortized development costs
  - 100M units for desktops
  - 1M units for servers
- 128-byte memory access
  - 1/16 performance on gather
- 4-8 outstanding memory refs
  - Need 100s to hide latency
- Little DP or TLP
  - 2-4 FPUs

Balance in Machine Design

- Each *phase* of an application is limited by one of:
  - Arithmetic bandwidth
  - Local memory bandwidth
  - Global memory bandwidth (network bandwidth)



Technology makes arithmetic cheap and bandwidth expensive

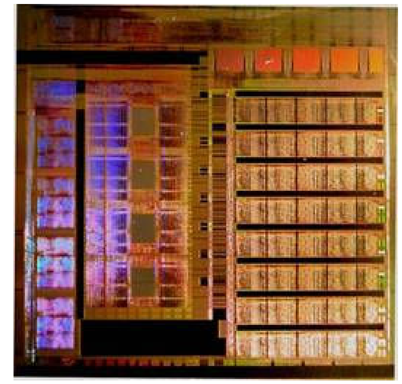


100s of FPUs per chip
$0.50/GFLOPS
50mW/GFLOPS

40GB/s off-chip BW
$5/GB/s
0.25W/GB/s

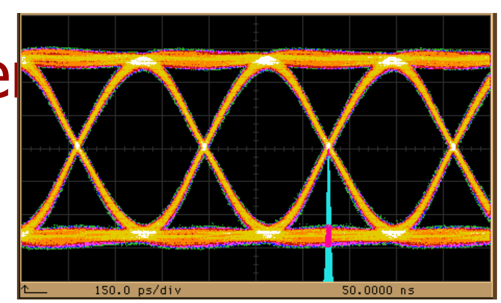
Cost of BW increases with distance
4x over backplane
6x over cable
25x VSR optics

Cost is dominated by bandwidth (and memory)



- Arithmetic is cheap $0.50/GFLOPS,
  - (200GFLOPS chips)
- Memory is $200/GByte, ~$10/GB/s
  - 1GByte of memory costs 400GFLOPS
  - 1GB/s of bandwidth costs 20GFLOPS

- Global bandwidth moderate cost
  - $1 (board), $4 (backplane), $25 (fiber) per GB/s
  - 2GFLOPS (board), 8GFLOPS (backplane), 50GFLOPS (global)



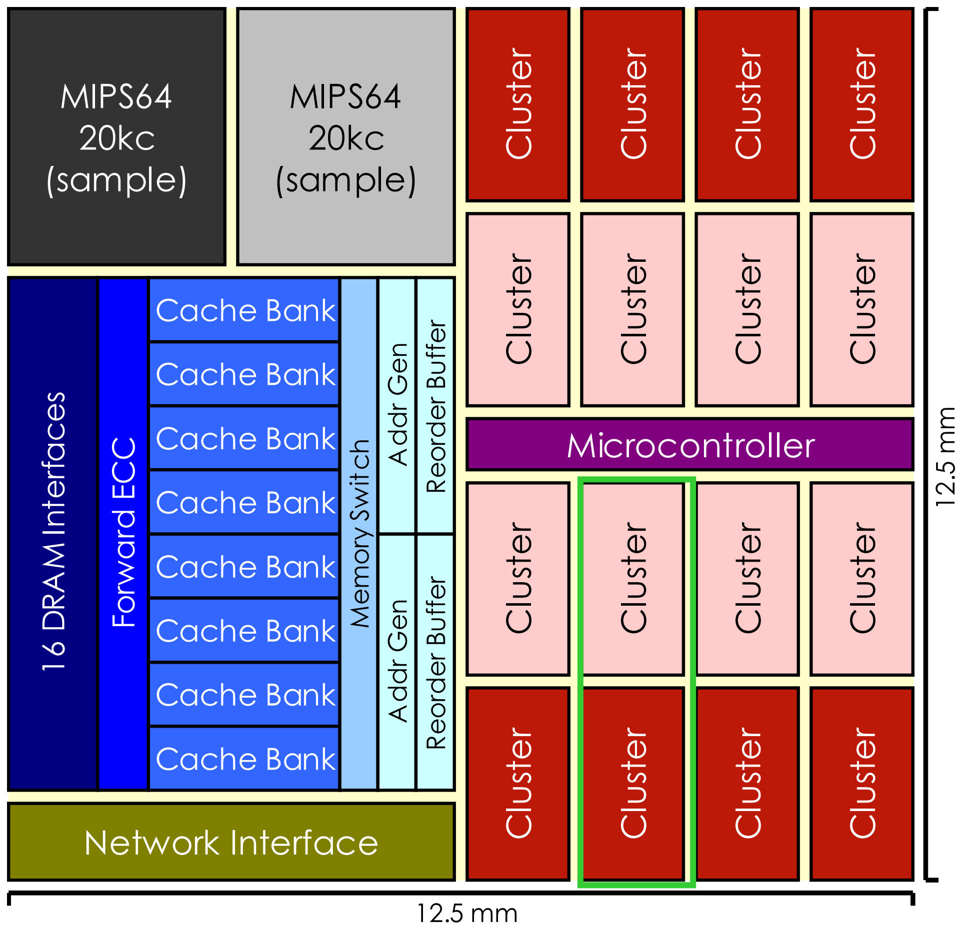
Recurring vs. Non-recurring costs

- Developing a custom processor costs $5-10M
  - Several examples
  - Quotes on Merrimac processor from 3 vendors ($6M)
  - Standard-cell design with semi-custom datapaths
  - Two mask sets in 90nm or 65nm
- Recurring costs $100-200 per unit
- Overall costs depend on volume
  - $1,200 per processor for 10K processors (20PFLOPS)
  - $300 per processor for 100K processors
- Costs $10M for the first one, then $200 per node
- NRE less than 10% the cost of a $100M Capability machine

Frequency can be misleading

- Commodity processors operate at 3+ GHz
  - The result of aggressive process and circuit design.
- However, what matters is
  - Total arithmetic performance
    - 200 FPUs at 1GHz (200GF) is better than 2 FPUs at 3GHz (6GF)
  - Latency around critical loops
    - Roughly the same
  - Memory bandwidth + latency hiding
    - Much better for custom
  - Performance per unit power
    - Better for custom
- Bottom line
  - A custom processor at 1GHz may greatly outperform a commodity processor at 3GHz (20x or more)

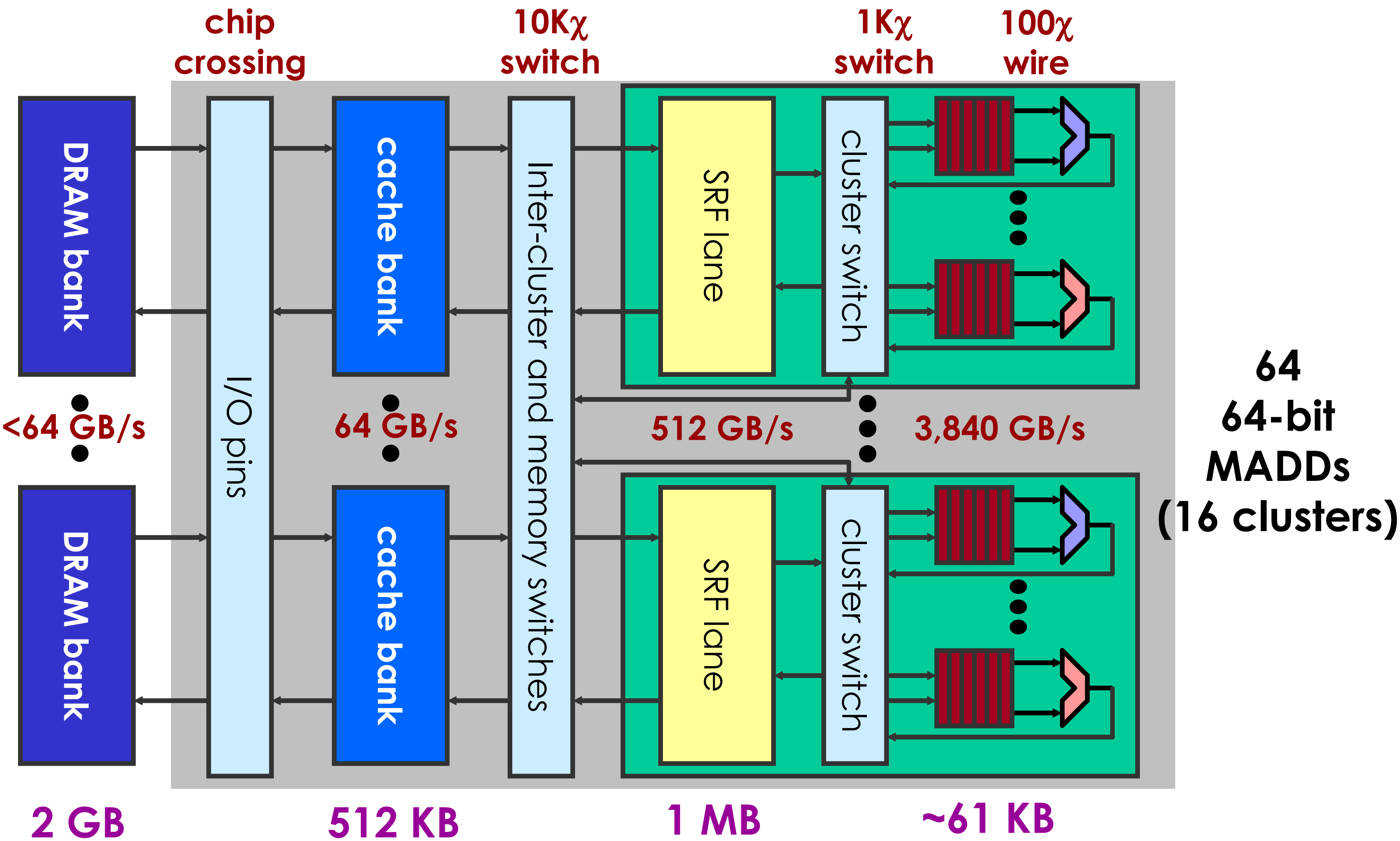
Example: The Merrimac Stream Processor



- 64 64-bit MULADD FPUs
  - Arranged in 16 clusters
- Capable memory system
- Designed for reliability
- 1 GHz in 90nm
  - 128 GFLOPS
- Area efficient
  - ~150mm^2 in 90nm
  - Pentium 4 is ~120mm^2 in 90nm but only 6.4 GFLOPS
- Efficient at ~25W
  - Pentium 4 is 100W
  - 28% of energy in ALUs

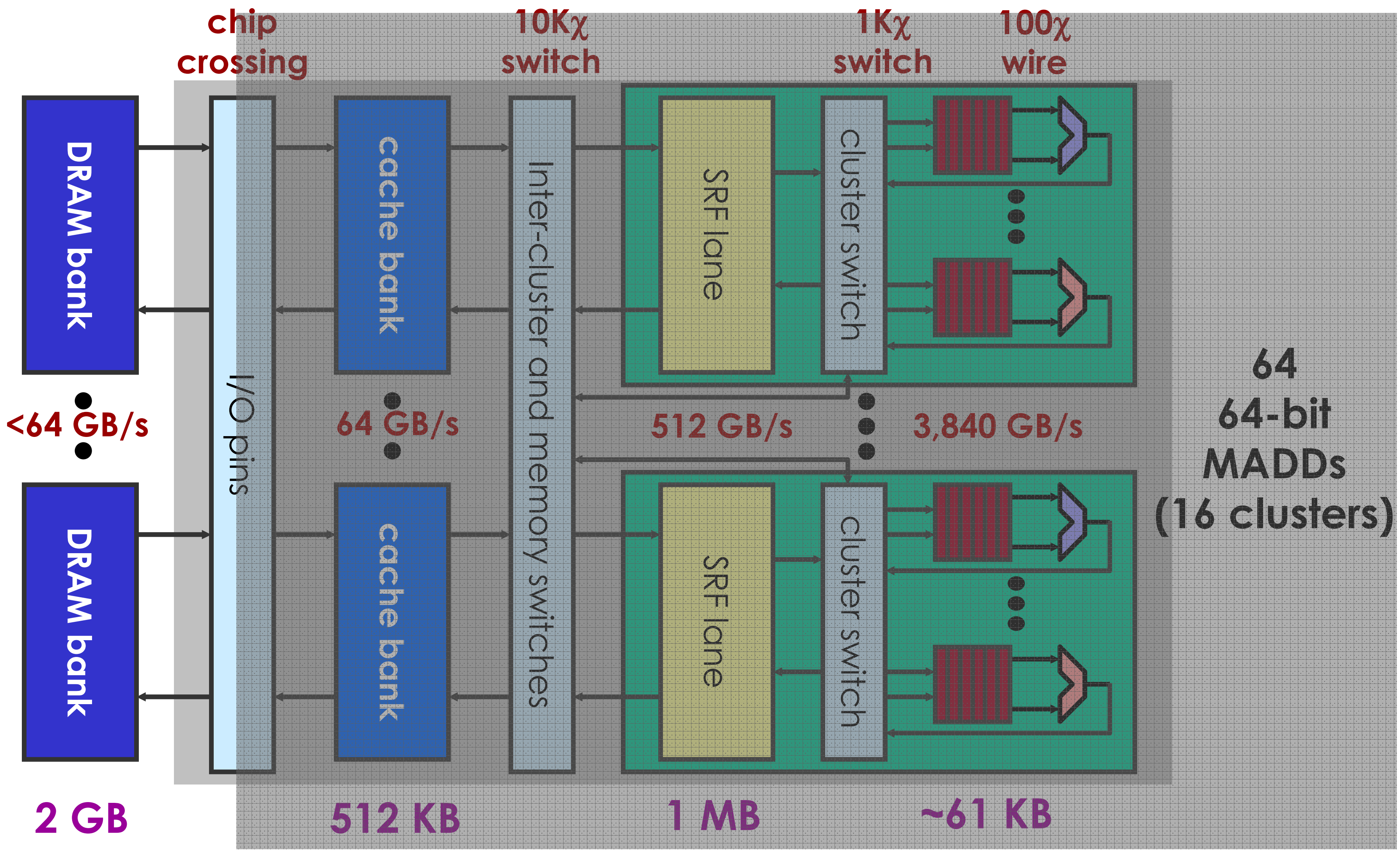
Merrimac processor is tuned for scientific computing

Merrimac Bandwidth Hierarchy



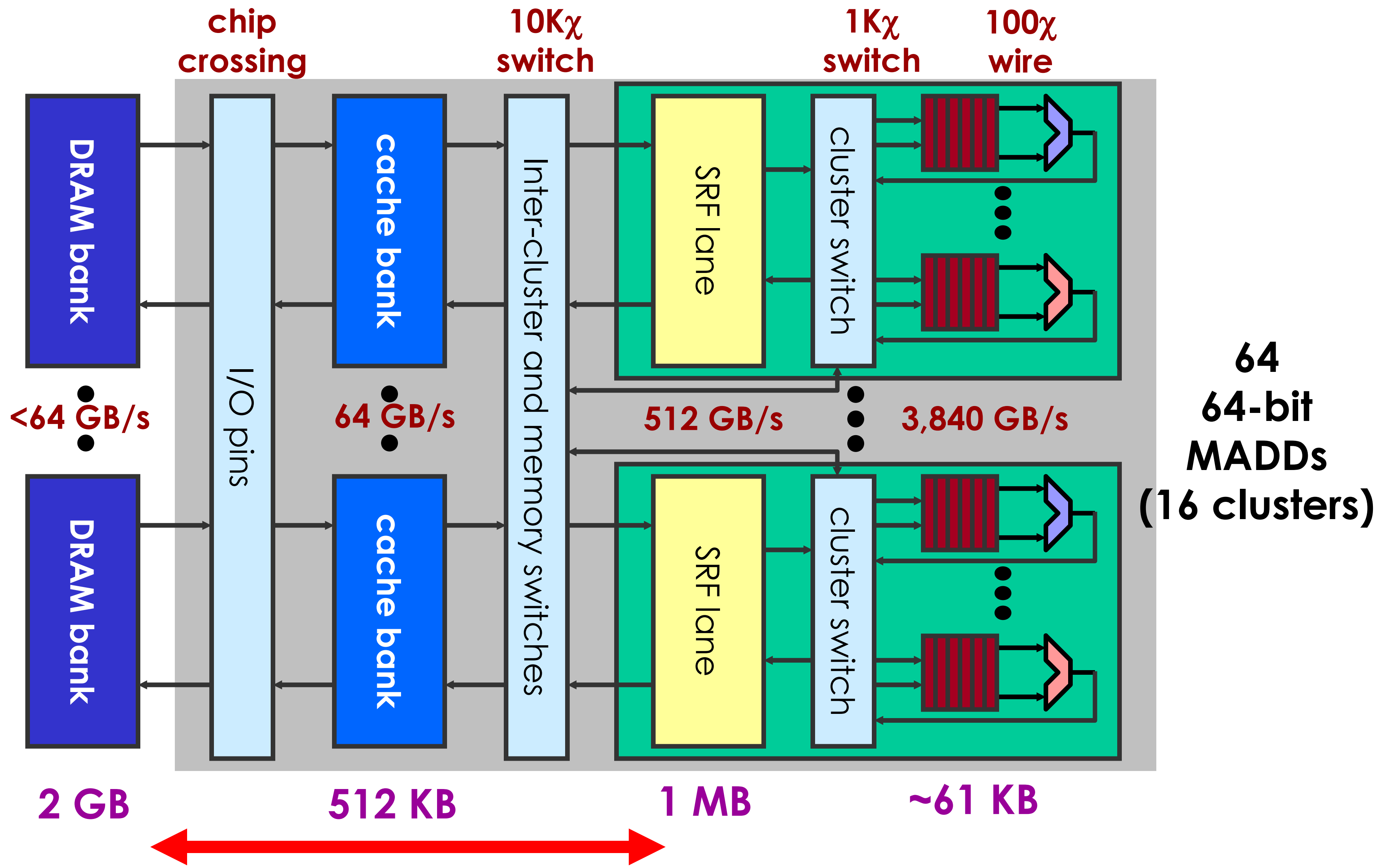
Register hierarchy and data-parallel execution
enable high performance and efficiency – 128 GFLOPS ~25 W

High memory bandwidth provided



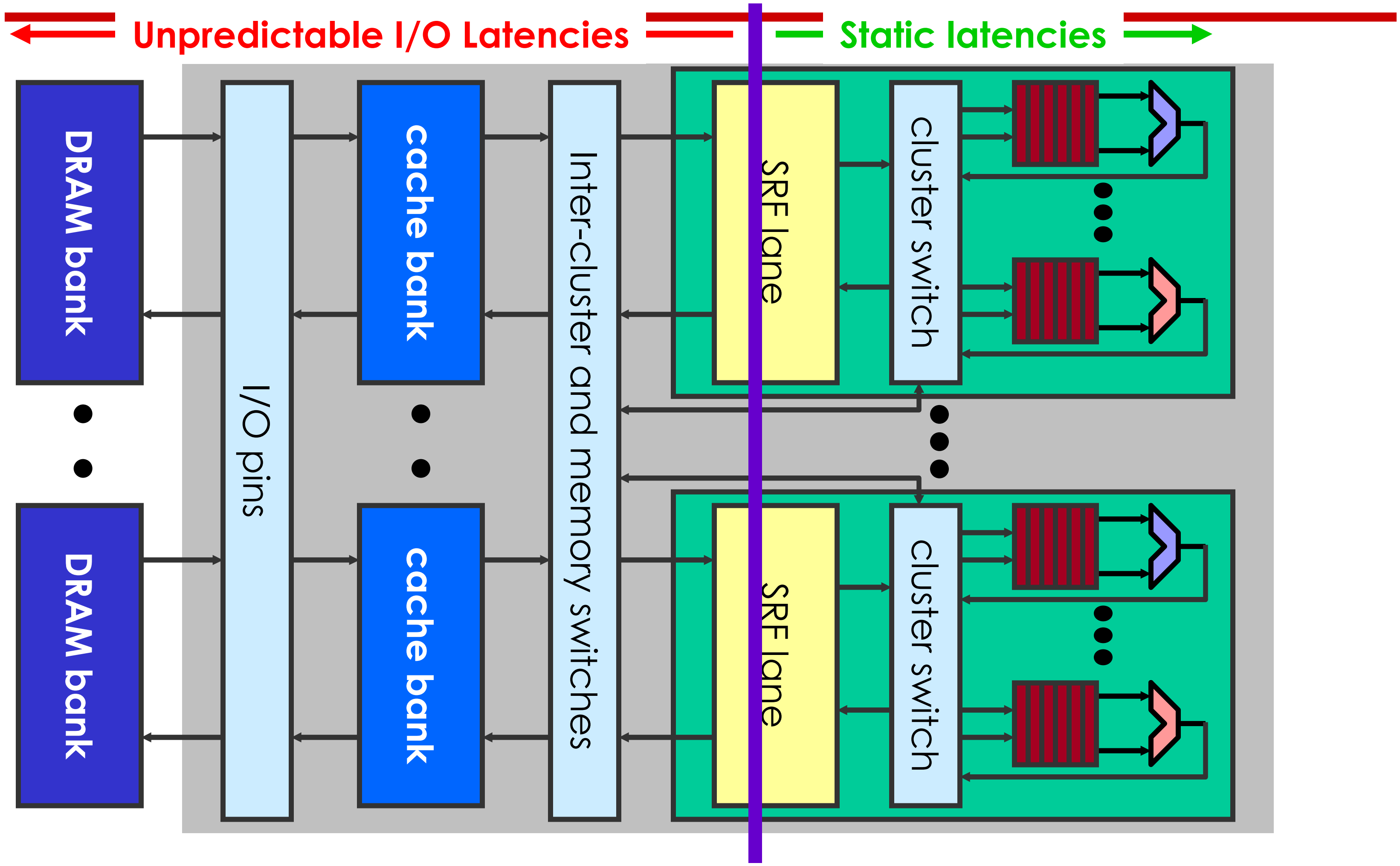
64 GB/s sequential access bandwidth

Latency hiding via stream transfers



1,000s of words tranferred in one instruction

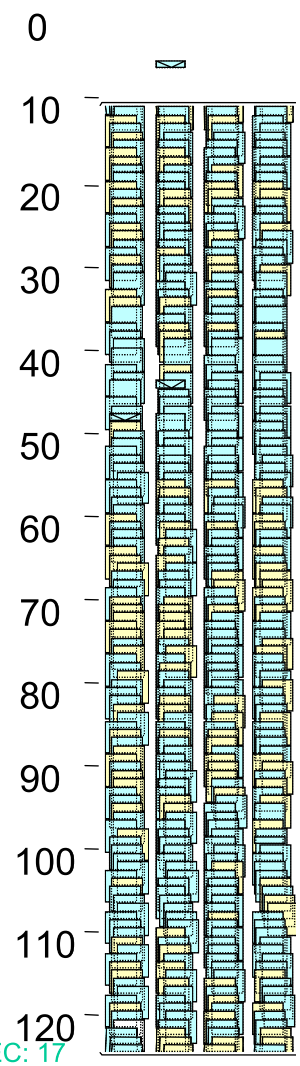
SRF Decouples Execution from Memory



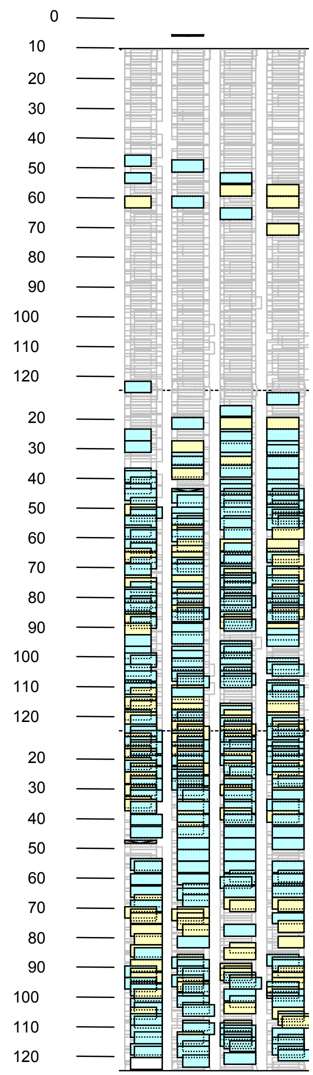
Decoupling allows efficient SRF allocation
Decoupling allows efficient scheduling of instructions on FPUs

Enables high utilization of large numbers of FPUs

One iteration



SW Pipeline



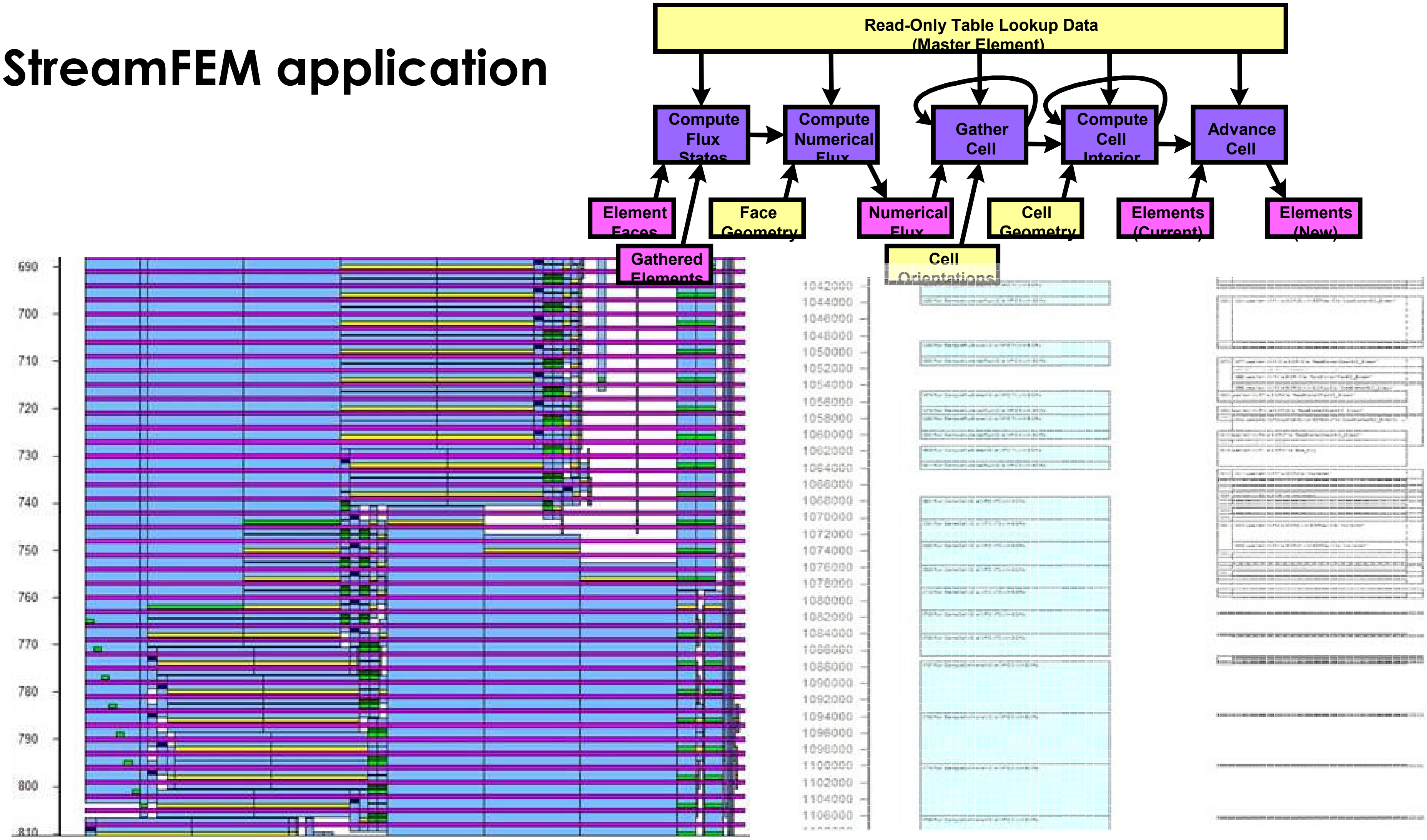
ComputeCellInt kernel from StreamFem3D

Over 95% of peak with simple hardware

Depends on explicit communication to make delays predictable

And efficient use of on-chip storage

StreamFEM application



Prefetching, reuse, use/def, limited spilling

Merrimac Application Results

Application	Sustained GFLOPS	FP Ops / Mem Ref	LRF Refs	SRF Refs	Mem Refs
StreamFEM3D (Euler, quadratic)	31.6	17.1	153.0M (95.0%)	6.3M (3.9%)	1.8M (1.1%)
StreamFEM3D (MHD, constant)	39.2	13.8	186.5M (99.4%)	7.7M (0.4%)	2.8M (0.2%)
StreamMD (grid algorithm)	14.2*	12.1*	90.2M (97.5%)	1.6M (1.7%)	0.7M (0.8%)
GROMACS	38.8*	9.7*	108M (95.0%)	4.2M (2.9%)	1.5M (1.3%)
StreamFLO	12.9*	7.4*	234.3M (95.7%)	7.2M (2.9%)	3.4M (1.4%)

Simulated on a machine with 64GFLOPS peak performance and no fused MADD
* The low numbers are a result of many divide and square-root operations

Applications achieve high performance and make good use of the bandwidth hierarchy

What about software?

- Software costs are typically much greater than hardware costs
  - Particularly applications software
- Custom processors can make software easier
  - High local and global bandwidth
  - Less sensitivity to "cache issues"
  - Fewer "performance surprises"
- Compilers for custom processors are not difficult
  - Leverage existing compiler infrastructure
- However, little application software is written to take advantage of such processors
  - MPI encourages LCD applications

Summary

- Custom processors can provide more sustained performance per $ than commodity processors.
  - Better at Capability and Capacity
- Bandwidth is expensive and scarce
- Tailor memory system to characteristics of scientific applications
  - Lots of bandwidth (64 GB/s per node)
  - Latency hiding (1000s of cycles)
  - Good gather/scatter performance (about 20GB/s per node)
- Provide explicit on-chip storage hierarchy to reduce BW demand and enable FPU scheduling
- Overprovision inexpensive FPUs
- Design in RAS for large configurations
- Can deliver 128GFLOPS node for $1K (parts cost)
  - 1 PFLOPS at 8K nodes and $8M