



Merrimac Architecture



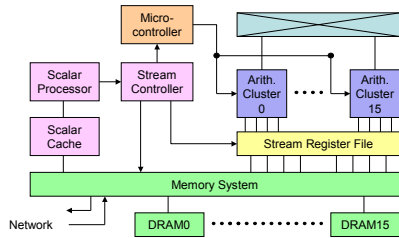
William J. Dally

Mattan Erez
Nuwan Jayasena

Jung Ho Ahn
Ujval Kapasi
Abhishek Das

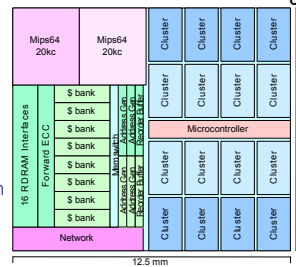
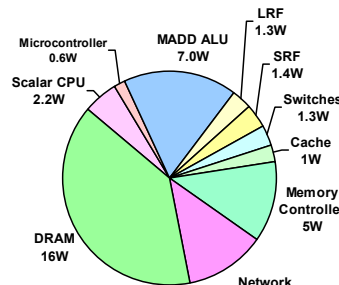
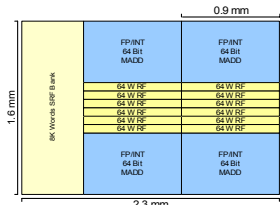
Francois Labonte
Timothy Knight

Merimac Node



- 16 data-parallel compute clusters
- Integrated scalar processor
- Indexable SRF (poster)
- Capable memory system
 - 16 Gbyte/s (random access)
 - Stream cache
 - ScatterOp (poster)

Floorplan



- 90nm tech (1 V)
- ASIC technology
- 1 GHz (37 FO4)
- 128 GFLOPs
- Inter-cluster switch between clusters
- 127.5 mm² (small ~12x10)
 - Stanford Imagine is 16mm x 16mm
 - MIT Raw is 18mm x 18mm
- 25 Watts per processor (P4 = 75 W)
- 41 Watts total per node (with DRAM)

Reliability, Availability, and Serviceability

- Defect data errors using parity
 - Parity protect all arrays
 - DRAM, SRF, Cache, μ code store, Registers, Stream buffers, Reorder buffers
 - Parity protect buses
- Defect control and execution errors by replication
 - Replicate scalar core
 - Replicate micro-controller
- Use only half the clusters and address generators
 - Duplicate the computation
 - Compare the results
 - High-performance mode
 - Use all clusters, but less reliable

- Checkpoint - rollback
- MTBF of 1e6 hours per board
 - Replace 1 board per month
 - MTBF of 1e8 per component
- Checkpoint duration ~5 minutes
- Recovery time ~10 minutes
- Checkpoint every 7 hours
- Slowdown of less than 1.5%

$$\text{slowdown} \leq \left(\frac{T_{cp}}{\Delta t_{cp}} + 1 \right) \left(1 - \frac{T_{repair} + T_{cp}^{-1} + \frac{\Delta t_{cp}}{2}}{E[\Delta t_j]} \right)^{-1}$$

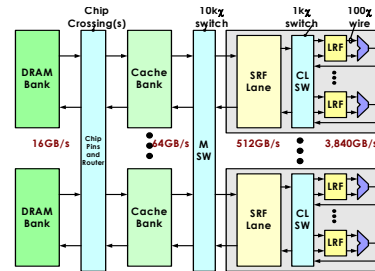
$$0 = \frac{\partial \text{slowdown}}{\partial \Delta t_{cp}}$$

$$= -\frac{T_{cp}}{\Delta t_{cp}^2} \left(1 - \frac{T_{repair} + T_{cp}^{-1} + \frac{\Delta t_{cp}}{2}}{E[\Delta t_j]} \right)^{-1} + \left(\frac{T_{cp}}{\Delta t_{cp}} + 1 \right) \left(1 - \frac{T_{repair} + T_{cp}^{-1} + \frac{\Delta t_{cp}}{2}}{E[\Delta t_j]} \right)^{-2} \frac{1}{2E[\Delta t_j]}$$

- Redundant power and cooling
 - diode voting on boards
- Hot extra board per cabinet
- Network gracefully degrades

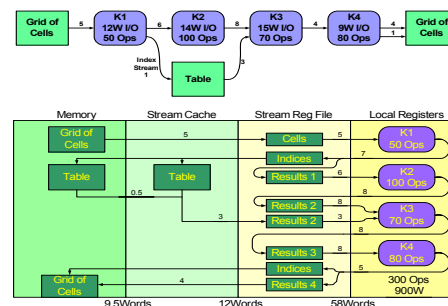
The Merrimac streaming supercomputer project aims to develop a scientific computer that offers an order of magnitude or more improvement in performance per unit cost compared to cluster-based scientific computers built from the same underlying semiconductor and packaging technology. We expect this efficiency to arise from two innovations: stream architecture and advanced interconnection networks. Organizing the computation into streams and exploiting the resulting locality using a register hierarchy enables a stream architecture to reduce the memory bandwidth required by representative computations by an order of magnitude or more.

Bandwidth/Register Hierarchy



- Exploits locality
 - producer-consumer within kernel in the LRF
 - producer-consumer across kernels in the SRF
- Reduces the distance data travels
- Support large number of ALUs

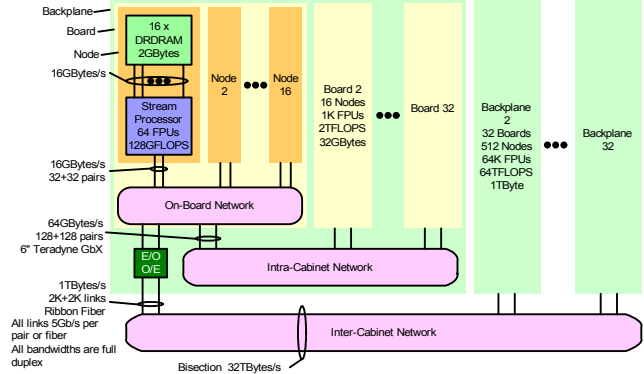
Stream Applications



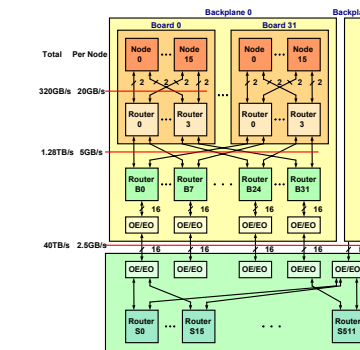
Application	Sustained GFLOPs	FP Ops / Mem Ref	LRF Refs	SRF Refs	Mem Refs
StreamFEM3D ¹ (Euler, quadratic)	31.6	17.1	153.0M (95.0%)	6.3M (3.9%)	1.8M (1.1%)
StreamFEM3D ¹ (MHD, constant)	39.2	13.8	186.5M (99.4%)	7.7M (0.4%)	2.8M (0.2%)
StreamMD ¹ (grid algorithm)	14.2 ²	12.1 ²	90.2M (97.5%)	1.6M (1.7%)	0.7M (0.8%)
GROMACS ¹	22.0 ²	7.1 ²	181M (95.4%)	5.3M (2.8%)	3.4M (1.8%)
StreamFLO	12.9 ²	7.4 ²	234.3M (95.7%)	7.2M (2.9%)	3.4M (1.4%)

1. Simulated on a machine with 64GFLOPs peak performance
2. The low numbers are a result of many divide and square-root operations

Merimac System

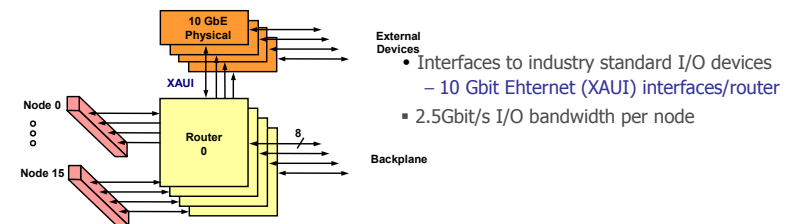


Board and Network



- Flat memory bandwidth within a 16-node board
- 4:1 Concentration within a 32-node backplane, 8:1 across a 32 backplane system
- Routers with bandwidth B=640Gb/s route messages with length L=128b
 - Requires high radix to exploit

Input/Output



Future Research

- Interface between stream and scalar processor
 - short-stream effects
 - amount of software control
- Mechanisms for variable-rate streams
- Efficient iterative operations
- Multi-node execution and simulation
- Aspect Ratio
- Work partitioning between static and run-time
 - stream scheduling
 - non-contiguous SRF