
Stream Processors

William J. Dally
Computer Systems Laboratory
Stanford University
billd@csl.stanford.edu

1

Stream Processors

- What is *Stream Processor Architecture*?
 - What problem is being solved
 - Performance scaling, power efficiency, bandwidth bottlenecks
 - What is a stream program?
 - What is a stream processor

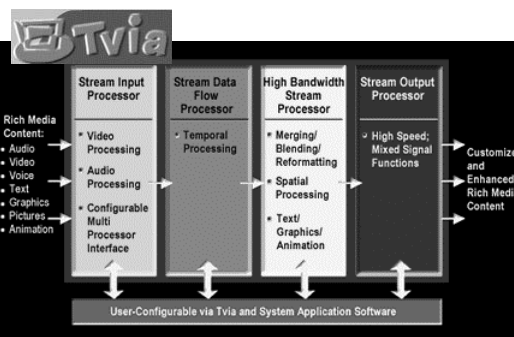
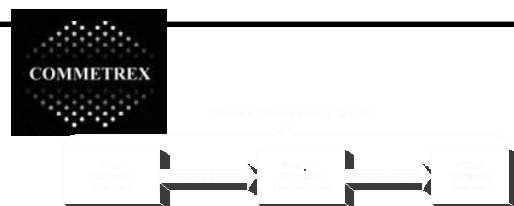
2

What is a Stream Processor?

- ï A stream program is a computation organized as streams of records operated on by computation kernels.
- ï A stream processor is optimized to exploit the locality and concurrency of stream programs

3

Stream Processing is becoming pervasive



Forbes
.com

Insights

Peter Huber, 01.07.02

A new type of computing--"stream computing"--has emerged to handle the back end of sonar, radar, X-ray sources and certain broadband applications such as voice-over Internet and digital TV.

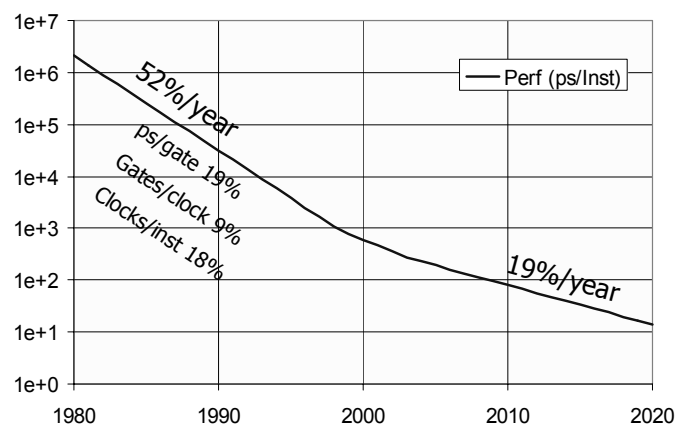
4

Motivation ñ Why do we need a stream processor?

- ï Application demand
 - ñ Power
 - ñ Bandwidth
 - ñ Performance Scaling
- ï Emerging media applications demand 10s of GOPS to TOPs with low power.
 - ñ Video compression
 - ñ Image understanding
 - ñ Signal processing
 - ñ Graphics
- ï Scientific applications also need TOPs of performance with reasonable cost

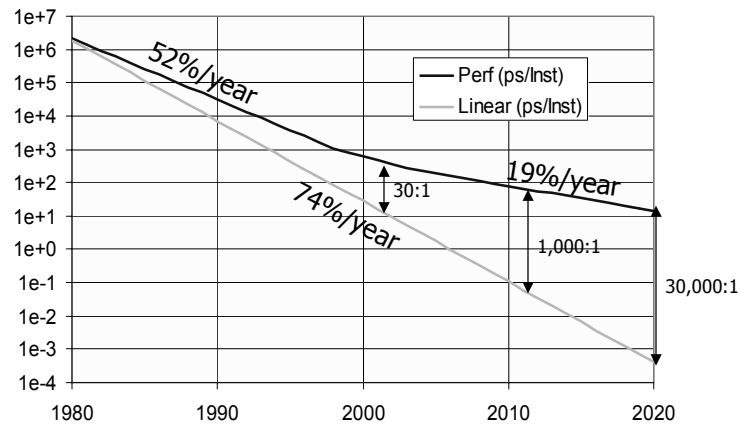
5

Conventional Processors No Longer Scale Performance by 50% each year



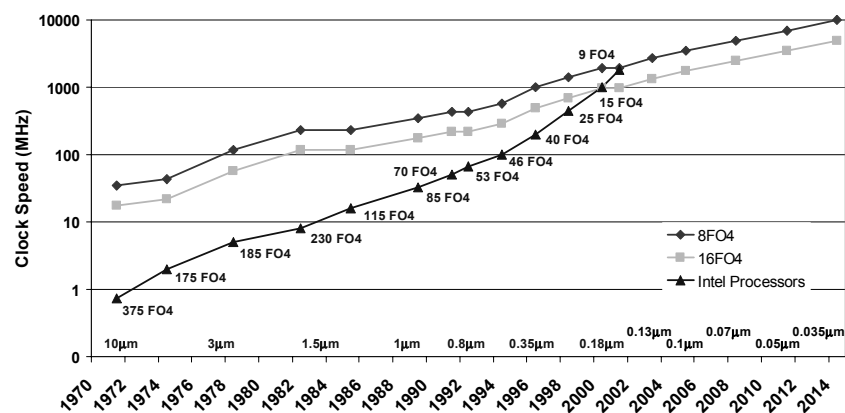
6

Future potential of novel architecture is large (1000 vs 30)

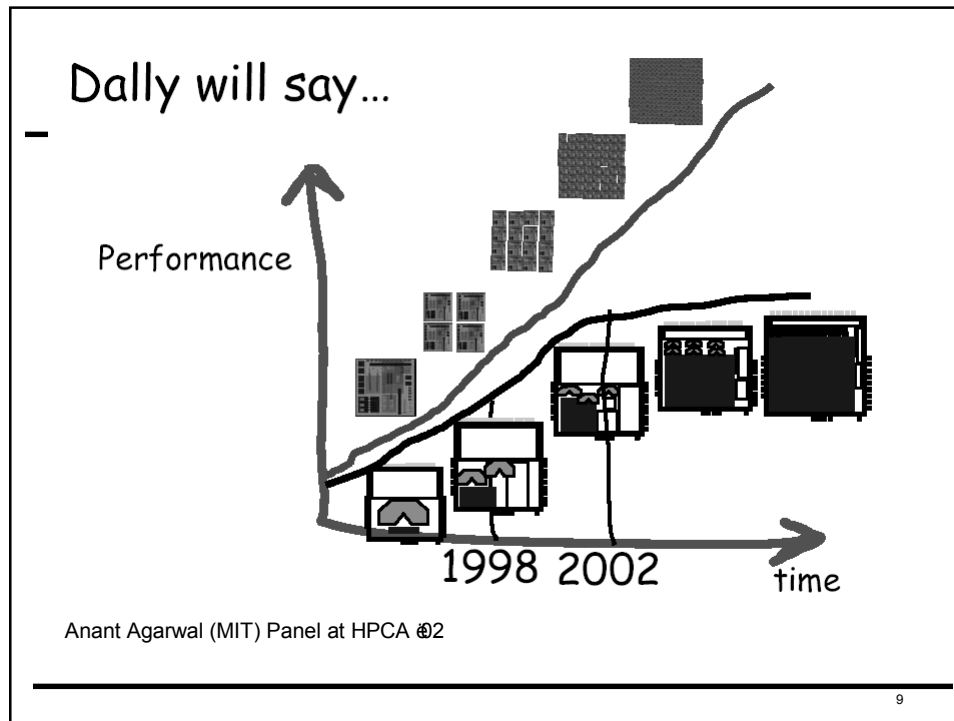


7

Clock Scaling: Historical and Projected



8



Pentium III vs. Pentium IV

	Pentium III	Pentium IV
Technology	180nm	180nm
Die Size	106mm ²	217mm ²
Transistor Count	24 million	42 million
# Grids	10 ⁸	2x10 ⁸
Pipeline Stages	10	20
Clock Rate	1GHz (15 FO4)	1.5GHz (10.4 FO4)
L1 D\$ Capacity	16KBytes	12KBytes
SpecInt2000	454	524
SpecInt/MHz	0.45	0.35

10

Outline

- ï Motivation
 - ñ We need low-power, programmable TeraOps
- ï The problem is **bandwidth**
 - ñ Growing gap between special-purpose and general-purpose hardware
 - ñ Its easy to make ALUs, hard to keep them *fed*
- ï A stream processor gives programmable bandwidth
 - ñ Streams expose locality and concurrency in the application
 - ñ A bandwidth hierarchy exploits this
- ï Imagine is a 20GFLOPS prototype stream processor
- ï Many opportunities to do better
 - ñ Scaling up
 - ñ Simplifying programming

11

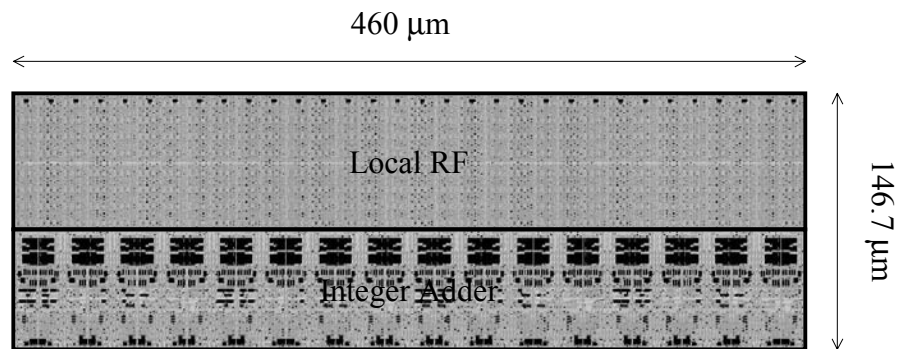
Motivation

- ï Some things I'd like to do with a few TeraOps
 - ñ Have a realistic face-to-face meeting with someone in Boston without riding an airplane
 - ï 4-8 cameras, extract depth, fit model, compress, render to several screens
 - ñ High-quality rendering at video rates
 - ï Ray tracing a 2K x 4K image with 10^5 objects at 60 frames/s

12

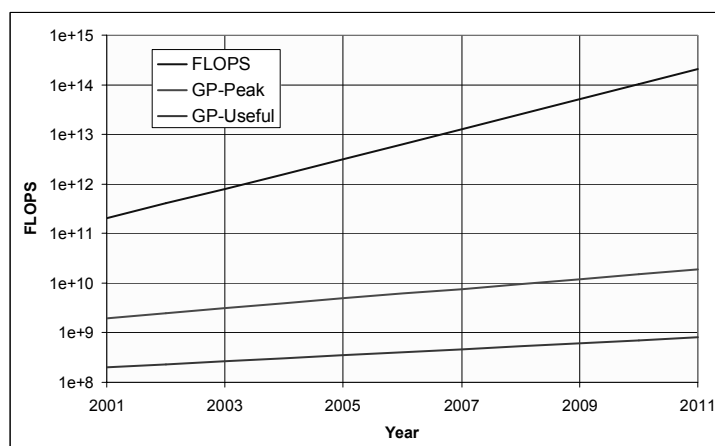
The good news ñ FLOPS are cheap, OPS are cheaper

- 32-bit FPU ñ 2GFLOPS/mm² ñ 400GFLOPS/chip
- 16-bit add ñ 40GOPS/mm² ñ 8TOPS/chip



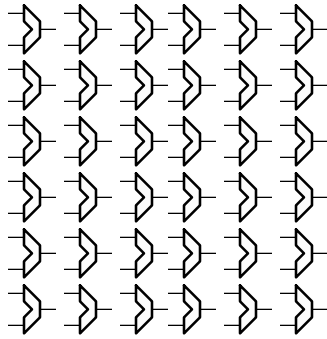
13

The bad news ñ General purpose processors can't harness this

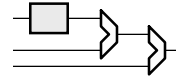


14

Why do Special-Purpose Processors Perform Well?



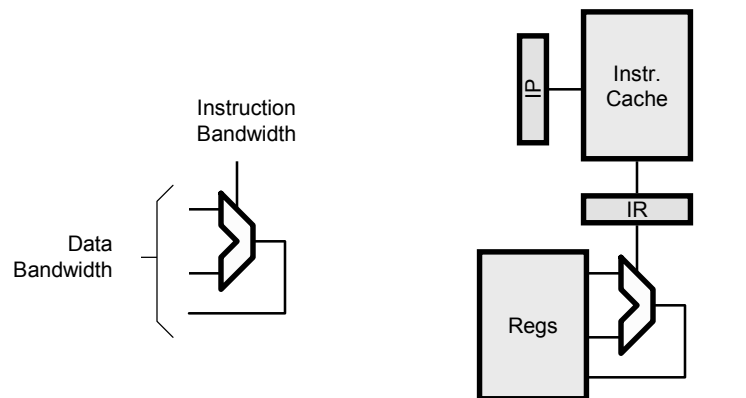
Lots (100s) of ALUs



Fed by dedicated wires/memories

15

Care and Feeding of ALUs



Feeding Structure Dwarfs ALU

16

Stream Programs make communication explicit

- ï This reduces energy and delay
- ï Energy is a matter of distance (interconnect)

Operation	Energy	
	(0.13um)	(0.05um)
32b ALU Operation	5pJ	0.3pJ
32b Register Read	10pJ	0.6pJ
Read 32b from 8KB RAM	50pJ	3pJ
Transfer 32b across chip (10mm)	100pJ	17pJ
Execute a uP instruction (SB-1)	1.1nJ	130pJ
Transfer 32b off chip (2.5G CML)	1.3nJ	400pJ
Transfer 32b off chip (200M HSTL)	1.9nJ	1.9nJ

300: 20: 1 off-chip to global to local ratio in 2002

1300: 56: 1 in 2010

17

Interconnect dominates delay

Operation	Delay	
	(0.13um)	(0.05um)
32b ALU Operation	650ps	250ps
32b Register Read	325ps	125ps
Read 32b from 8KB RAM	780ps	300ps
Transfer 32b across chip (10mm)	1400ps	2300ps
Transfer 32b across chip (20mm)	2800ps	4600ps

2: 1 global on-chip comm to operation delay
9: 1 in 2010

18

What is a Stream Program?

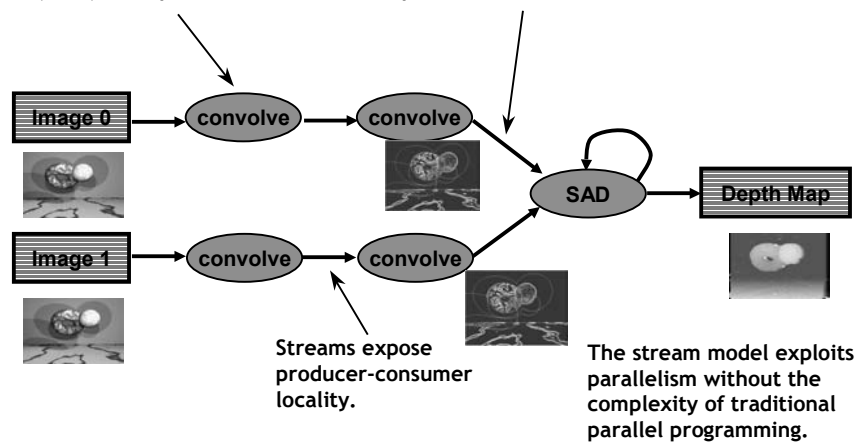
- ï A program organized as streams of records flowing through kernels
- ï Example, stereo depth extraction

19

Stereo Depth Extraction Stream Program

Kernels exploit both instruction (ILP) and data (SIMD) level parallelism.

Kernels can be partitioned across chips to exploit task parallelism.



20

Why Organize an Application This Way?

- ĩ Expose **parallelism** at three levels
 - ñ ILP within kernels
 - ñ DLP across stream elements
 - ñ TLP across sub-streams and across kernels
 - ñ Keeps ~~easy~~ parallelism easy
- ĩ Expose **locality** in two ways
 - ñ Within a kernel ñ kernel locality
 - ñ Between kernels ñ producer-consumer locality
 - ñ This locality can be exploited *independent* of spatial or temporal locality
 - ñ Put another way, stream programs make *communication explicit*

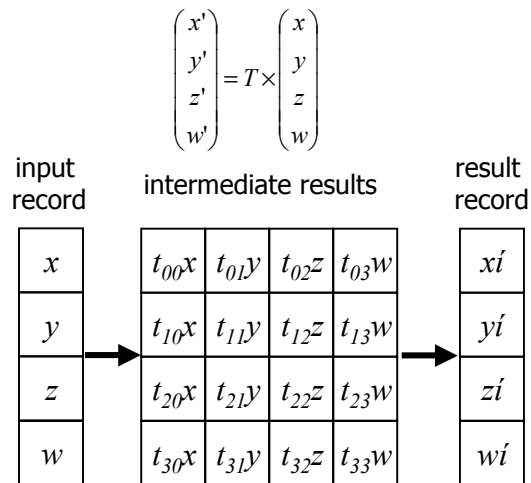
21

Streams expose *Kernel Locality* missed by **Vectors**

- | | |
|--|--|
| <ul style="list-style-type: none"> ĩ Streams <ul style="list-style-type: none"> ñ Traverse operations first <ul style="list-style-type: none"> ĩ All operations for one record, then next record ĩ Smaller working set of temporary values ñ Store and access whole records as a unit <ul style="list-style-type: none"> ĩ Spatial locality of memory references <ul style="list-style-type: none"> ñ e.g., get contiguous record on gather/scatter | <ul style="list-style-type: none"> ĩ Vectors <ul style="list-style-type: none"> ñ Traverse records first <ul style="list-style-type: none"> ĩ All records for one operation, then next operation ĩ Large set of temporary values ñ Group like-elements of records into vectors <ul style="list-style-type: none"> ĩ Read one word of each record at a time <ul style="list-style-type: none"> ñ No locality on gather/scatter |
|--|--|

22

Example ñ Vertex Transform

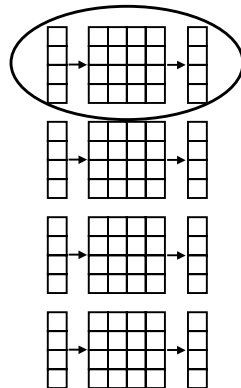


23

Vertex transform with Streams vs. Vectors

ĩ Small set of intermediate results

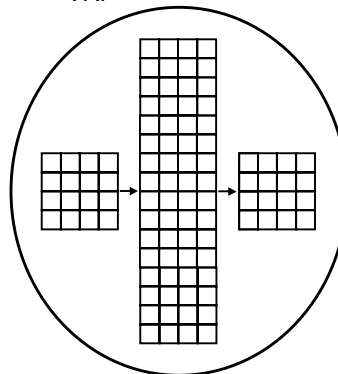
ñ enable small and fast LRFs



ĩ Large working set of intermediates

ñ VL times larger (e.g. 64x)

ñ Must use a large, slow global VRF



24

What class of applications can be written as stream programs?

- ĩ Media applications (signal, image, video, packet, and graphics processing) are naturally expressed in this style
- ĩ Scientific applications can be efficiently cast as stream programs
- ĩ Others?
 - ñ This is an open question
- ĩ Hypothesis
 - ñ Any application with a long run time (large operation count) has a great deal of parallelism and hence can be cast as a stream program.

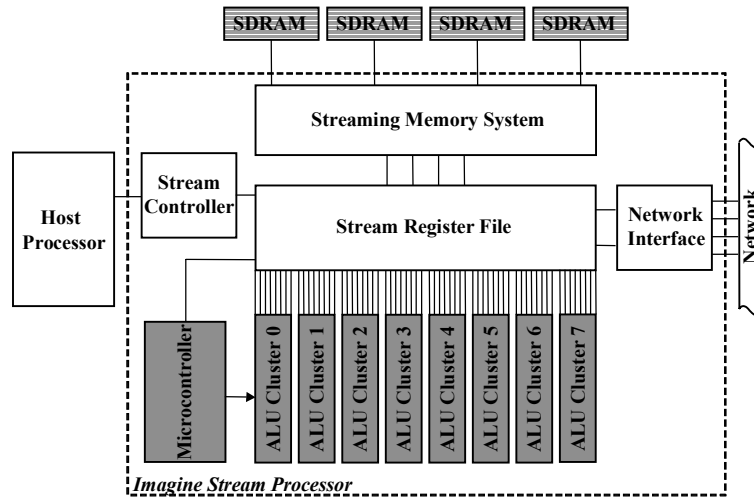
25

What is a Stream Processor?

- ĩ A processor that is optimized to execute a stream program
- ĩ Features include
 - ñ Exploit parallelism
 - ĩ TLP with multiple processors
 - ĩ DLP with multiple clusters within each processor
 - ĩ ILP with multiple ALUs within each cluster
 - ñ Exploit locality with a bandwidth hierarchy
 - ĩ Kernel locality within each cluster
 - ĩ Producer-consumer locality within each processor
- ĩ Many different possible architectures

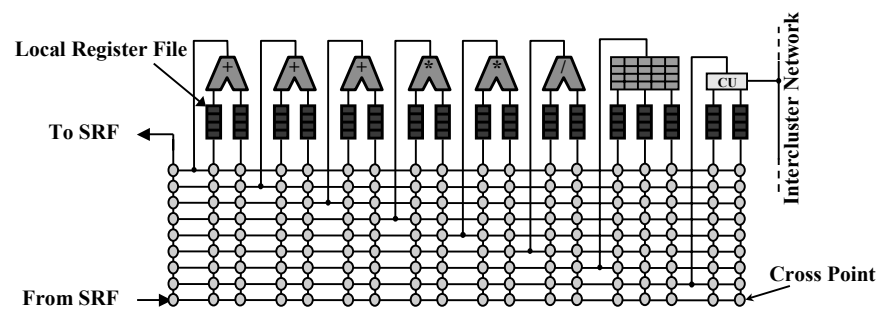
26

The Image Stream Processor



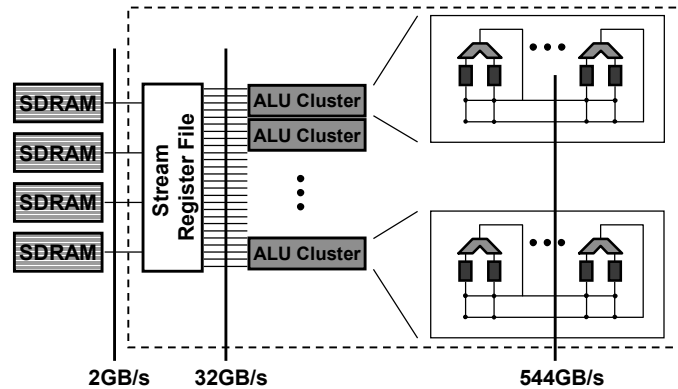
27

Arithmetic Clusters



28

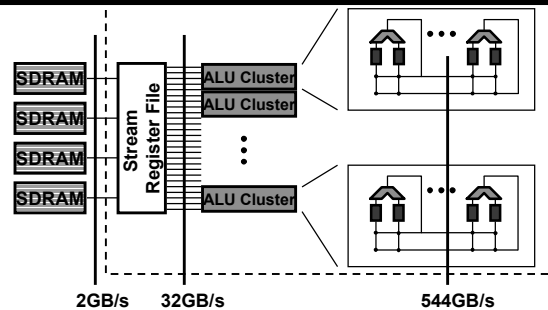
A Bandwidth Hierarchy exploits locality and concurrency



- VLIW clusters with shared control
- 41.2 32-bit floating-point operations per word of memory BW

29

A Bandwidth Hierarchy exploits kernel and producer-consumer locality



	Memory BW	Global RF BW	Local RF BW
Depth Extractor	0.80 GB/s	18.45 GB/s	210.85 GB/s
MPEG Encoder	0.47 GB/s	2.46 GB/s	121.05 GB/s
Polygon Rendering	0.78 GB/s	4.06 GB/s	102.46 GB/s
QR Decomposition	0.46 GB/s	3.67 GB/s	234.57 GB/s

30

Stream Controller

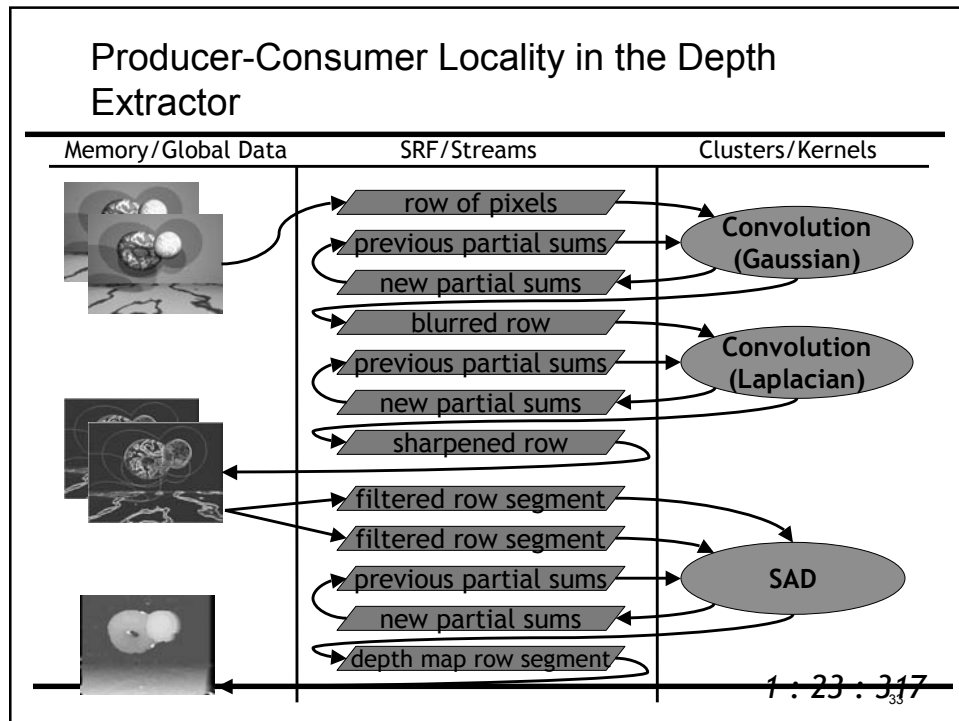
- ï 32 stream instructions that can be written to the stream controller by host processor
- ï Acts as a scoreboard
 - ñ Keeps track of dependencies using a bit field for each stream instruction

31

Pop Quiz

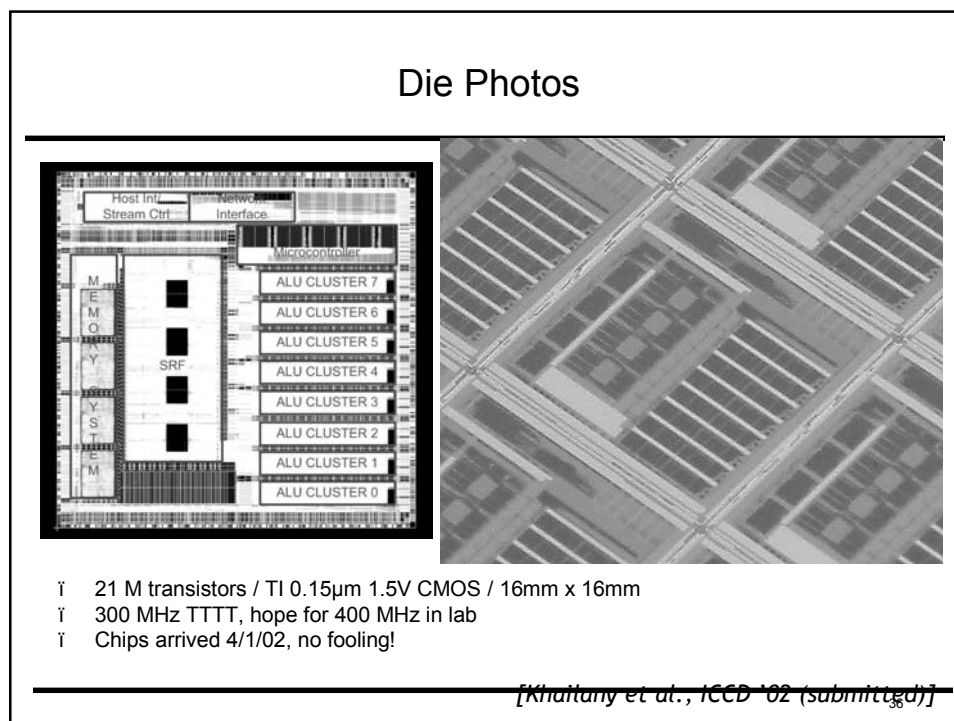
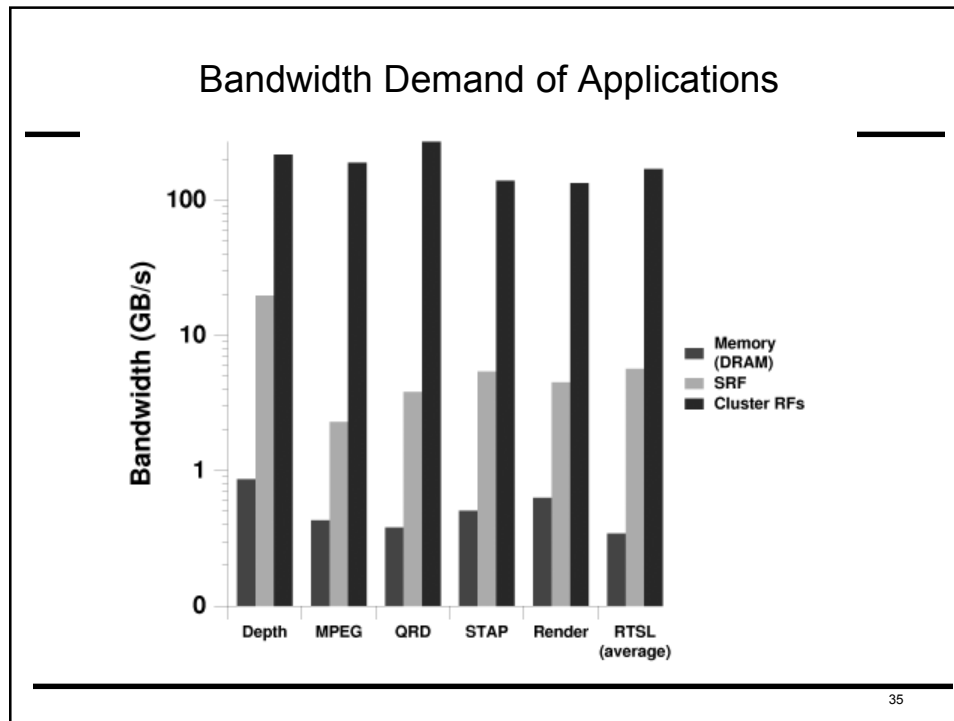
- ï Can you have different data types in a stream?
- ï How is memory access scheduling done?
 - ñ Do streams need to be contiguous in memory?
- ï In the convolve example, how is shared data of the partials managed?

32

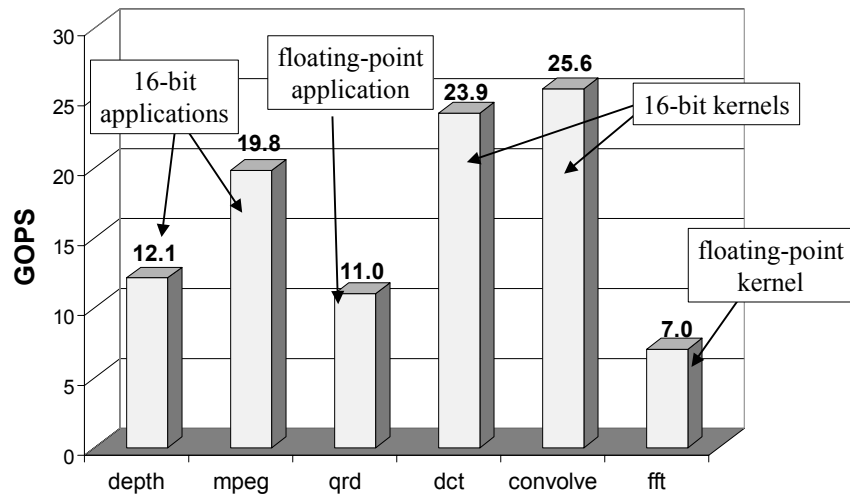


Imagine gives high performance with low power and flexible programming

- ĩ Matches capabilities of *communication-limited* technology to demands of signal and image processing applications
- ĩ Performance
 - ñ compound stream operations realize >10GOPS on key applications
 - ñ can be extended by partitioning an application across several Imagines (TFLOPS on a circuit board)
- ĩ Power
 - ñ three-level register hierarchy gives 2-10GOPS/W
- ĩ Flexibility
 - ñ programmed in iC1
 - ñ streaming model
 - ñ conditional stream operations enable applications like sort

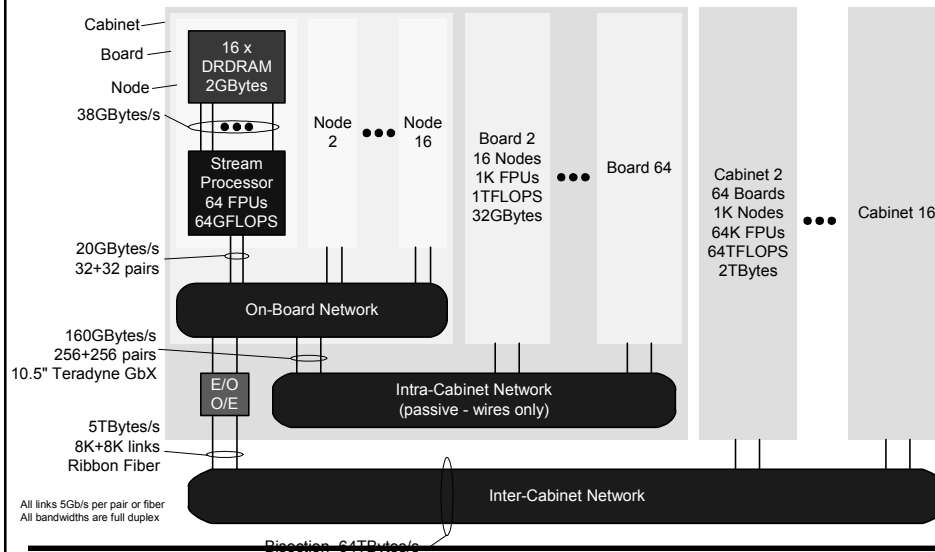


Performance demonstrated on signal and image processing



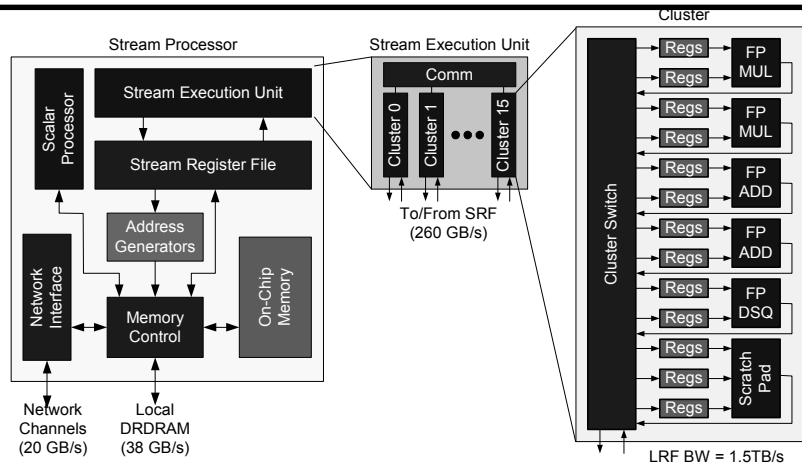
37

Architecture of a Streaming Supercomputer



38

Streaming processor



39

Rough per-node budget

Item	Cost	Per Node
Processor chip	200	200
Router chip	200	50
Memory chip	20	320
Board/Backplane	3000	188
Cabinet	50000	49
Power	1	50
Per-Node Cost		976
\$/GFLOPS (64/node)		15
\$/M-GUPS (250/node)		4

Preliminary numbers, parts cost only, no I/O included.

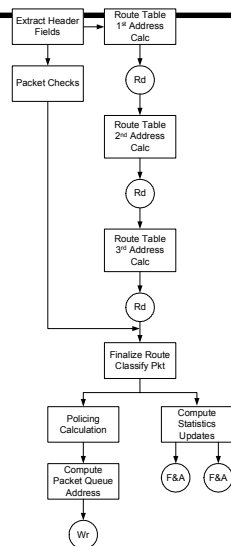
40

Many open problems

- | | |
|--|---|
| <ul style="list-style-type: none"> ĩ A small sampling ĩ Software <ul style="list-style-type: none"> ñ Program transformation ñ Program mapping ñ Bandwidth optimization ñ Conditionals ñ Irregular data structures | <ul style="list-style-type: none"> ĩ Hardware <ul style="list-style-type: none"> ñ Alternative stream models ñ Register organization ñ Bandwidth hierarchies ñ Memory organization ñ Short stream issues ñ ISA design ñ Cluster organization ñ Processor organization |
|--|---|

41

Example, Ingress Packet Functions



42

