MAXELER Technologies

SOFTWARE FOR CONFIGURABLE HARDWARE

MAXware*: acceleration in HPC*

R. Dimond, M. J. Flynn, O. Mencer and O. Pell
Maxeler Technologies
contact: flynn@maxeler.com

HPC: the case for accelerators

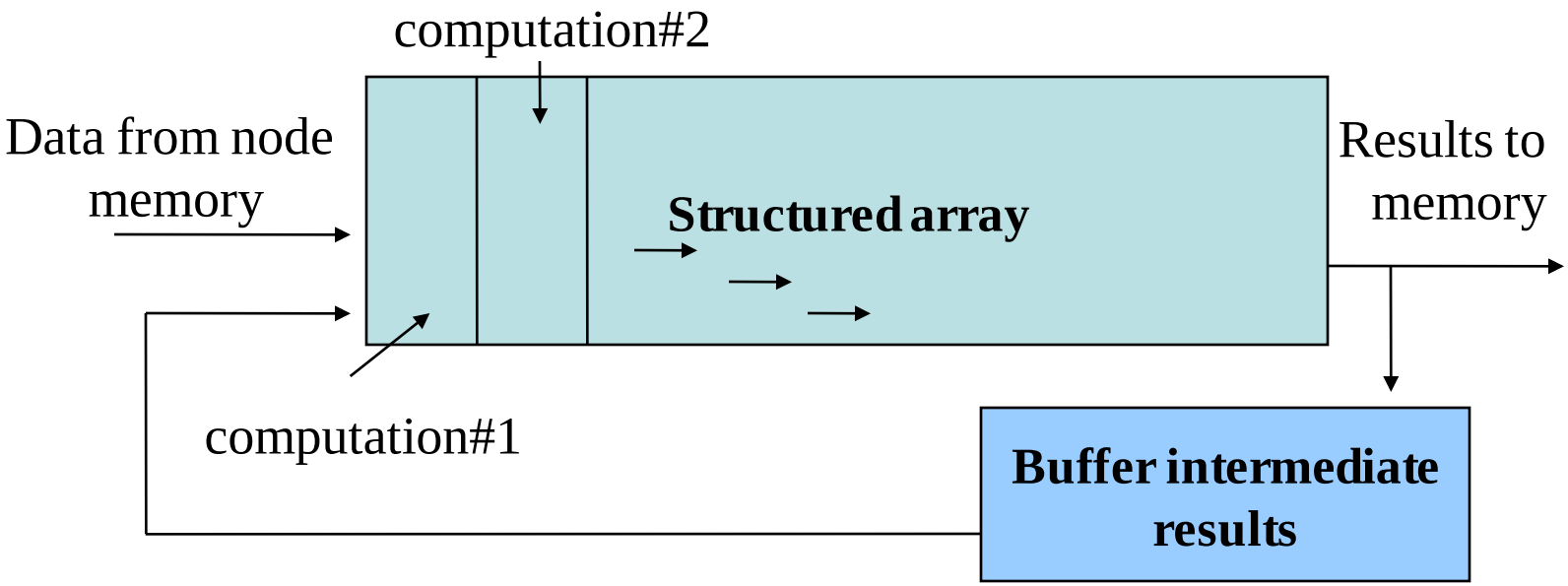
- Recent HPC systems using commodity processors optimize latency, not throughput.
- Multi core / multi thread is frequently memory hierarchy limited.
- Structured arrays (FPGAs, GPUs, hyper “core” or “cell” dies) offer complementary throughput acceleration; avoiding memory bottleneck.
- An array can provide 10-100x improvement in arithmetic (SIMD) or memory bandwidth (by streaming or MISD).

Design space for large applications

- Some distinctions
  - Memory limited vs. compute limited
  - Static vs. dynamic control structure
  - Homogeneous vs. core+accelerator at the node

Accelerate tasks by streaming

- MISD structured computation:
  streaming computations across a long array
  before storing results in memory.
  Can achieve 100x in improved use of memory.



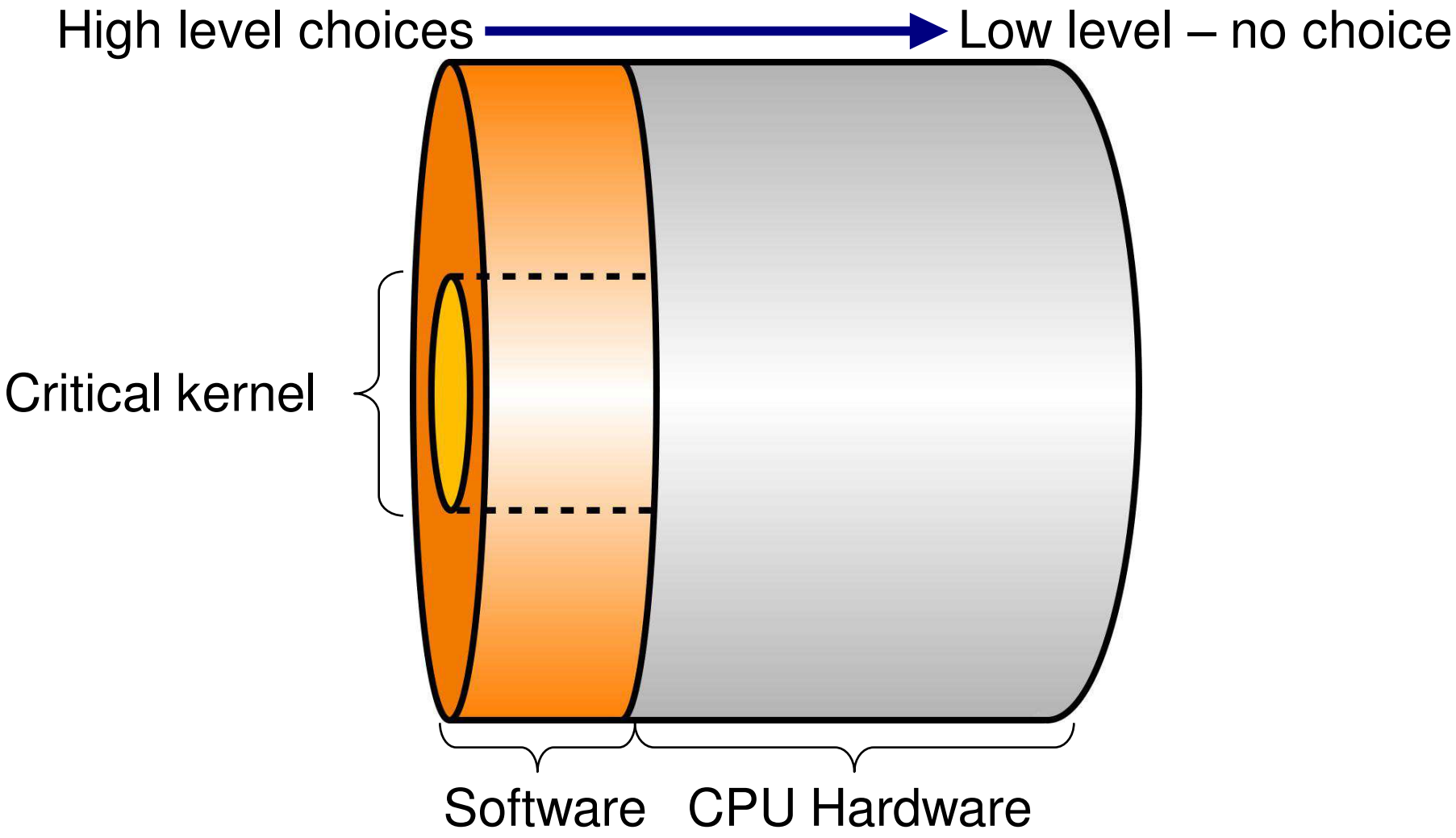
FPGA acceleration

- One tenth the frequency with 10^5 cells per die.
- Magnitude of parallelism overcomes frequency limitations.
- Stream data across large cell array,
  minimizing memory BW.
- Customized data structures
  e.g.17 bit FP; always just enough precision.
- A software (re)configurable technology
- Need an in-depth application study to realize acceleration;
  acceleration is *not* automatic;
  acceleration requires more programming effort.

A different programming model

- A cylindrical rather than a layered model suits static applications
- A streaming (MISD) computational model suits memory and (often) compute limited applications.

Cylindrical Model

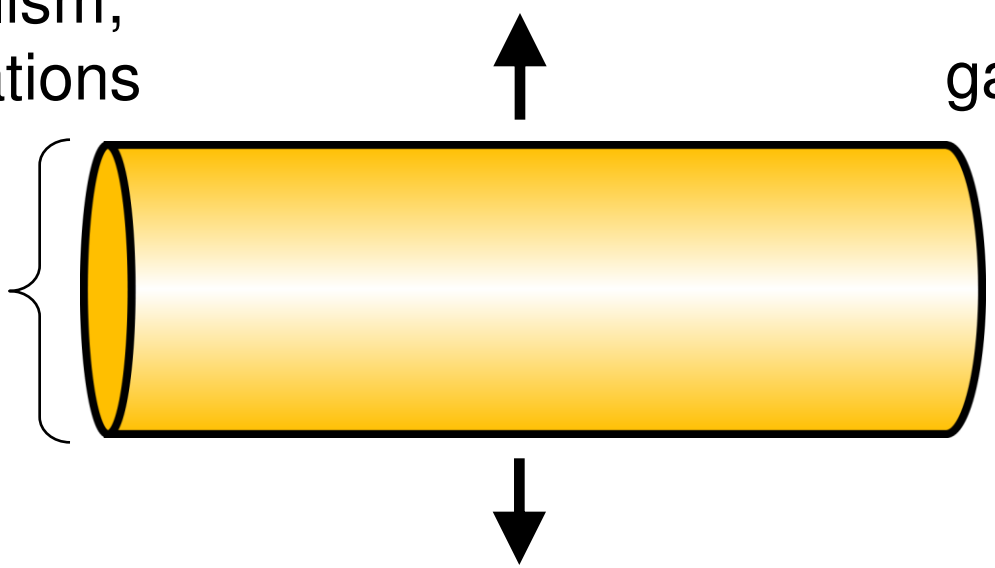


Cylindrical Model

High level choices
application choices,
algorithms,
types of parallelism,
data representations

optimize

Low level choices
communications,
routing,
timing,
gate delays



- First accelerate the (initially small) kernel
- Later extend kernel size

Cylindrical Model

- The programmer sees kernel application through a cylinder that spans high and low level design decisions
  - Applies to 1) large 2) static applications with 3) an identifiable critical section
  - Speedup is achieved by understanding the whole process; application down to the gates
  - After speeding up kernel; enlarge the application scope……Speedup requires *effort*

Speedup with cylindrical model

- Transform application to execute multiple simultaneous strips using DRAM “pipes”
- Stream computations through each pipe
- Strip size limited by FPGA area and DRAM bandwidth
  - Application specific data precision
    - multiplies FPGA area
    - multiplies DRAM bandwidth

MAXware: acceleration support

- Acceleration methodology
- Application tool set
- Acceleration hardware (MAX2 board)

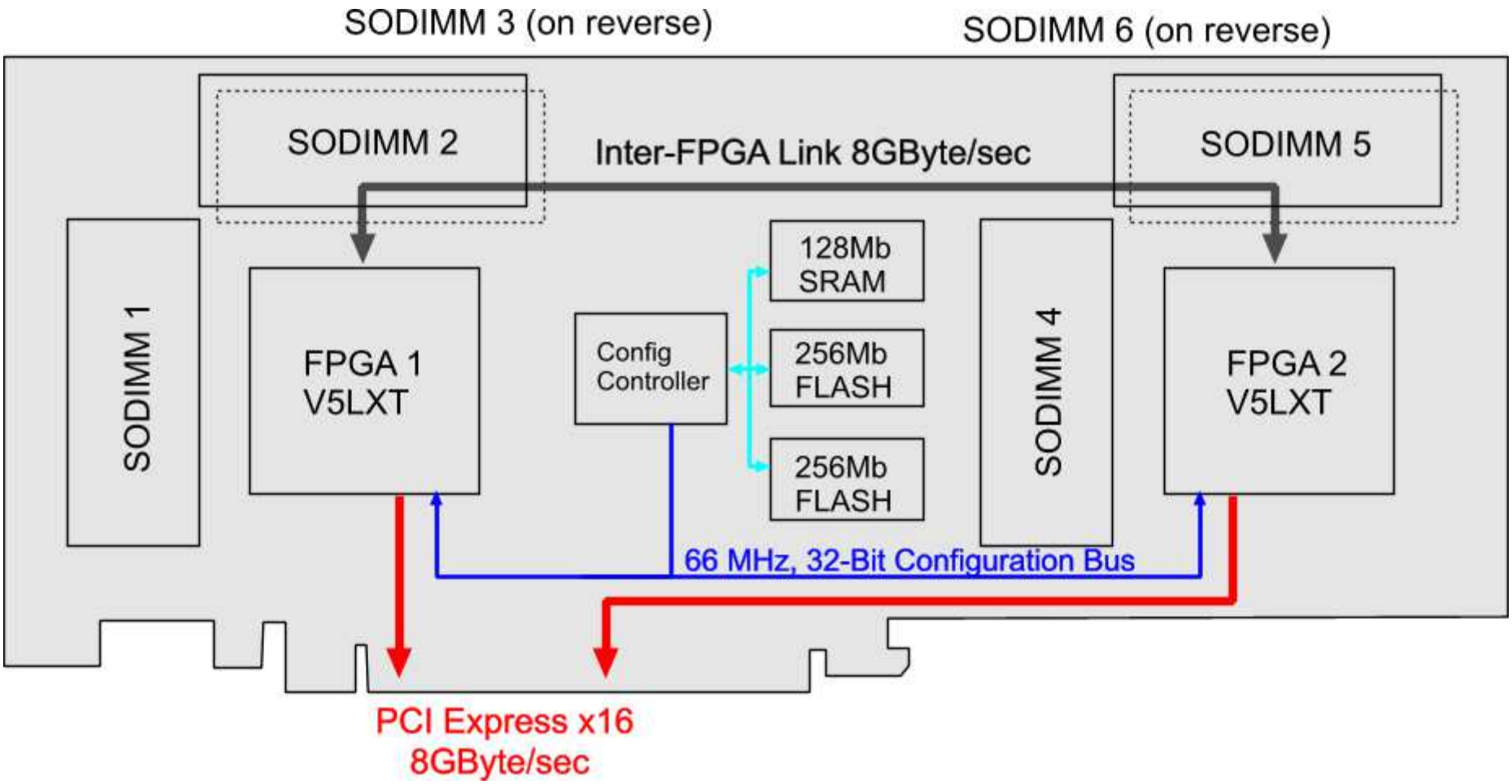
Acceleration methodology

- Four stages:
  - Analysis of program (static and dynamic)
  - Transformation: create dataflow graph, data layout and representation
  - Partitioning: optimize dataflow, data access and data representation options
  - Implementation: uses Maxeler stream compiler to configure application for FPGA, set run time interfaces

Maxeler tool set

- Parton profiles running application, finds hot spots and suitability for FPGA
  key metric: computations per memory access
- Performance modeler estimates execution time of alternative designs for FPGA; accurate to 10%
- MaxelerOS support for MAX2 board

MAX2 board


SODIMM 3 (on reverse)
SODIMM 6 (on reverse)
SODIMM 2
Inter-FPGA Link 8GByte/sec
SODIMM 5
SODIMM 1
FPGA 1
V5LXT
Config
Controller
128Mb
SRAM
256Mb
FLASH
256Mb
FLASH
SODIMM 4
FPGA 2
V5LXT
66 MHz, 32-Bit Configuration Bus
PCI Express x16
8GByte/sec

MAX2 board

- 2 x Virtex-5 LX330T with triple channel interface to 24GB DDR2 memory with PCI Express x16 link to host.
- 8GB/s inter FPGA link;
  total off-chip BW of 47 GB/s
- MaxelerOS has Linux device drivers, manages all PCI & DRAM “plumbing”

Example: seismic data processing

- For O&G exploration: distribute grid of sensors over large area
- Sonic impulse the area and record reflections: frequency, amplitude, delay at each sensor
- (Sea based survey) use 30,000 sensors record data (120 db range) each sampled at more than 2kbps with new sonic impulse every 10 sec. Order of terabytes of data each day.

seismic data processing: there's a lot of data to be processed

- Data can be interpreted with frequency and amplitude indicates structure, delay indicates depth (z axis).
- Process data to determine location of structures of interest
- Many different ways to process
  - One option:
    simulate wave propagation from source to receivers and compare to recorded data

Computations per output point on Intel Xeon

	Cycles	FLOPs	Other Ops	L1 Cache Miss Rate	CPI
2nd order – X pass	11%	40.2	72.3	0.2%	0.6
2nd order – Y pass	15%	40.2	72.3	3.8%	0.8
2nd order – Z pass	21%	40.2	72.4	7.3%	1.1
Vector add	1%	1.0	9.0	1.3%	0.9
4th order – X pass	10%	40.2	64.7	0.2%	0.6
4th order – Y pass	15%	40.2	64.7	4.0%	0.9
4th order – Z pass	21%	39.6	65.3	7.8%	1.2
Update pressure	1%	1.9	9.9	1.0%	0.7
Boundary sponge	5%	0.8	4.0	5.6%	5.8

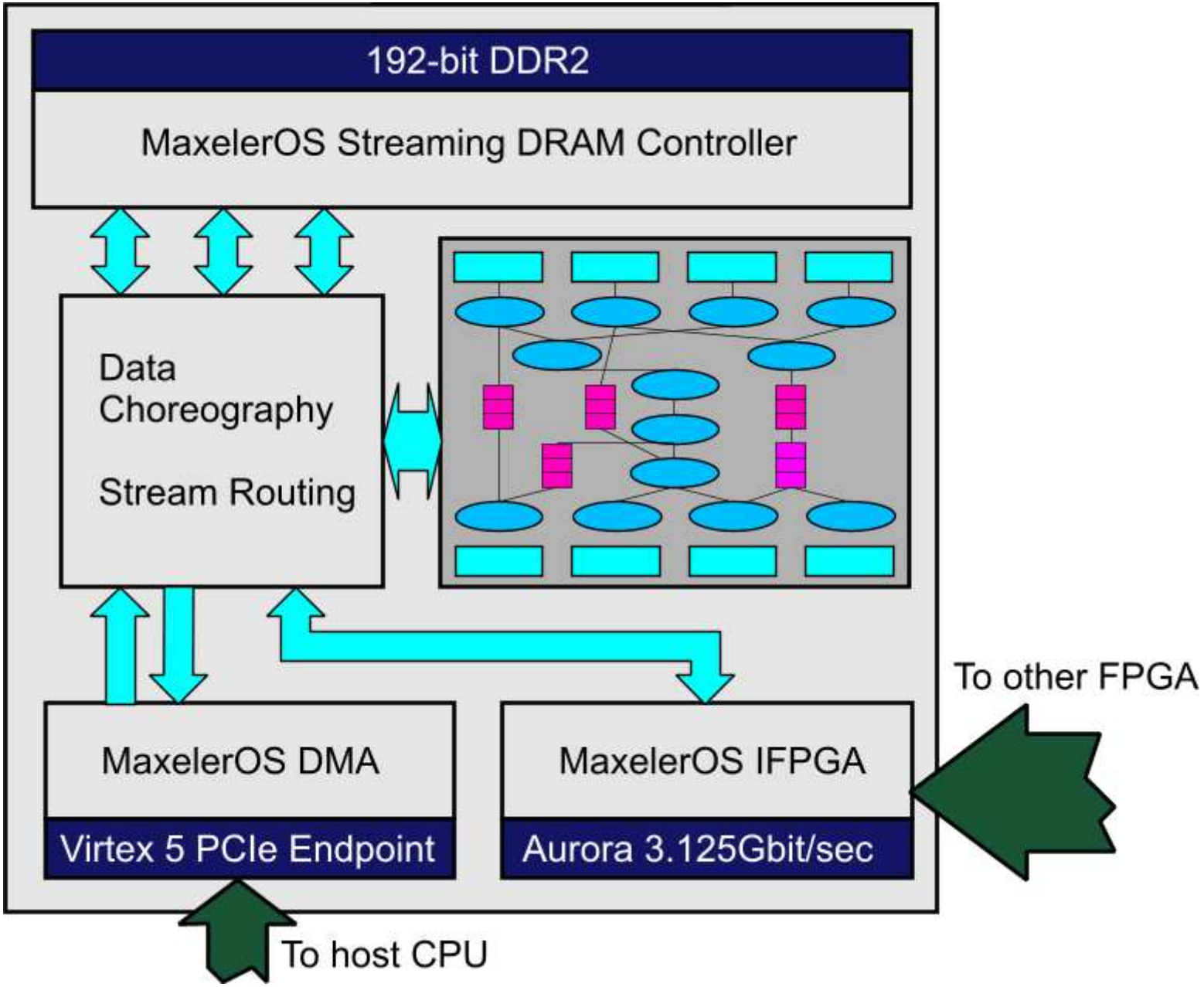
Computations

- On average about a data cache miss per 10 floating point ops.
- Xeon achieves about 1.0 cpi
- So Xeon has a 20x frequency starting advantage
  over an FPGA based computation
- BUT FPGA uses lots of parallelism
  to significant advantage

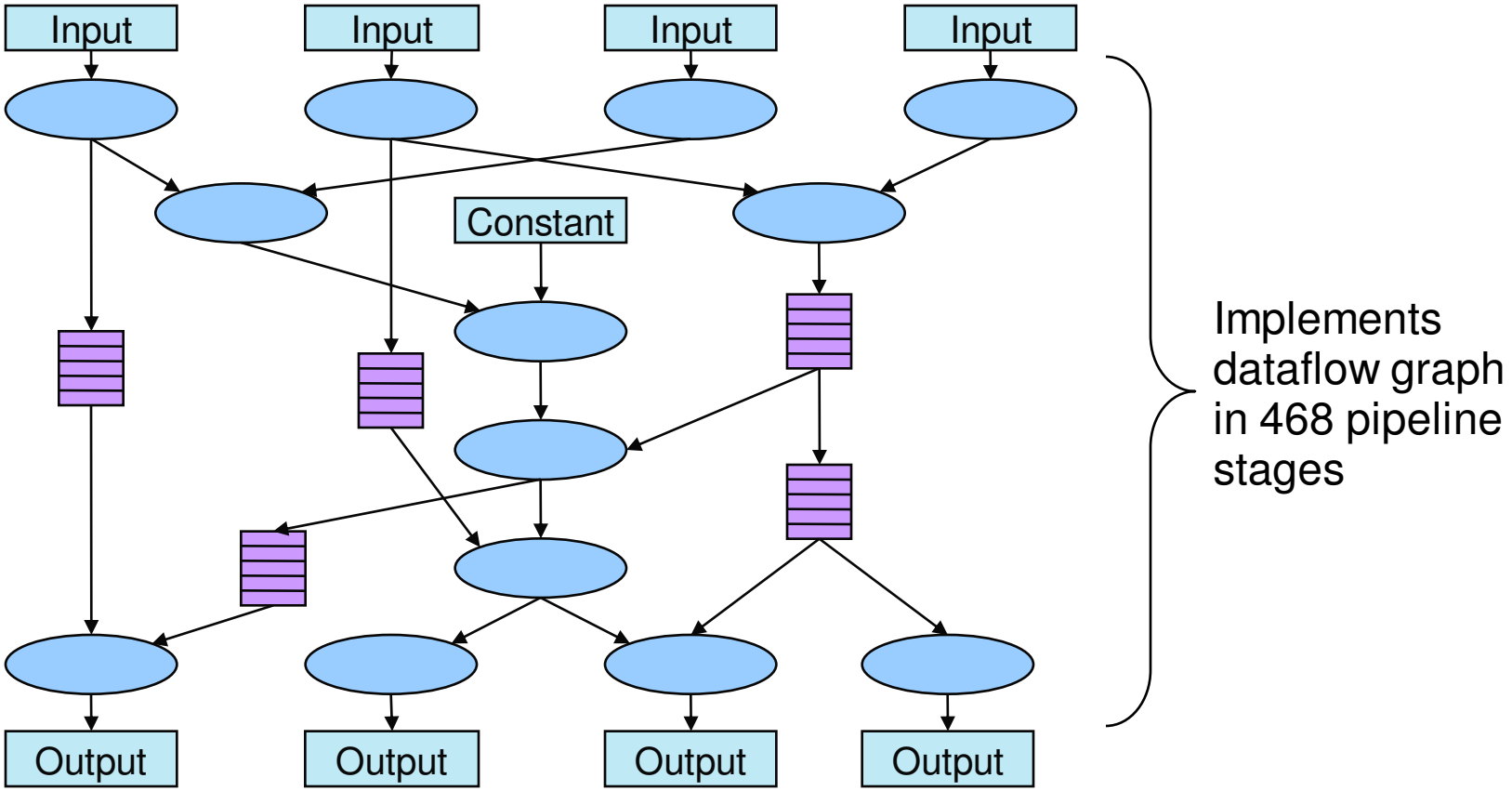
Streaming solution (FPGA)

- Convolve 4 input points (strips) simultaneously (per FPGA); buffer intermediate results; forward 4 outputs to next pipeline stage.
- Continue (streaming) pipelining until the silicon runs out (468 stages)
- Size the Floating Point so that there is *just* enough range & precision

O. Pell, T. Nemeth, J. Stefani and R. Ergas. *Design Space Analysis for the Acoustic Wave Equation Implementation on FPGA Circuits.* European Association of Geoscientists and Engineers (EAGE) Conference, Rome, June 2008.


192-bit DDR2
MaxelerOS Streaming DRAM Controller
Data
Choreography
Stream Routing
MaxelerOS DMA
Virtex 5 PCIe Endpoint
MaxelerOS IFPGA
Aurora 3.125Gbit/sec
To other FPGA
To host CPU

4 input x 4 output points; no cache misses; no memory accesses



Achieving Speedup > 100x

- Stream the computation in 468 stage pipeline
- Execute 8 points simultaneously
- Eliminate cache misses, eliminate overhead operations (load, store, branch..)
- So 8x (points processed) x 468 stages x 2 (overhead ops)/ 20 = 374 (max Sp possible)
- Operate at one-twentieth the frequency; reduce power and space.

So how typical is this?

- Other application areas (ongoing studies)
  - Financial simulation: Monte Carlo based derivative pricing
  - Image processing
  - CFD
- Kernels, referenced to a Xeon
  - Gaussian Random Number Generation 300x
  - 3D finite difference 160x
  - seismic offset gathers 170x

So how can an emulation (FPGA) be better than the x86 processor(s)?

- Lack of robustness in streaming hardware (spanning area, time, power)
- Lack of robust parallel software methodology/ tools
- Effort (and support tools) enable large dataflows with excellent A x T x P
- Effort (and support tools) provide significant application Speedup (*there's no free lunch in parallel programming*)

Summary

MAXELER Technologies
SOFTWARE FOR CONFIGURABLE HARDWARE

- Acceleration based speedup using structured arrays.
- More attention to memory and/or arithmetic: data representation, streaming, and RAM.
- Lower power with non aggressive frequency use.
- Programming uses cylindrical model; but speedup requires lots of low level program optimization. Good tools are golden.