DataFlow SuperComputing for ExaScale DM/FF: Revisiting the Algorithms

V. Milutinović, G. Rakocevic, S. Stojanović, and Z. Sustran

University of Belgrade

Oskar Mencer

Imperial College, London

Oliver Pell

Maxeler Technologies, London and Palo Alto

Michael Flynn

Stanford University, Palo Alto

Essence of the Paradigm:

For Big Data DM/FM algorithms
and for the same hardware price as before,
achieving:

a) speed-up, 20-200

b) monthly electricity bills, reduced 20 times

c) size, 20 times smaller

European Dimension of the Paradigm:

Absolutely all results achieved with:

a) all hardware produced in Europe, specifically UK
b) all software generated by programmers of EU and WB

ControlFlow vs. DataFlow:

ControlFlow:

- Top500 ranks using Linpack (Japanese K,…)

DataFlow:

- Coarse Grain (HEP) vs. Fine Grain (Maxeler)

Essence of the DataFlow Approach!

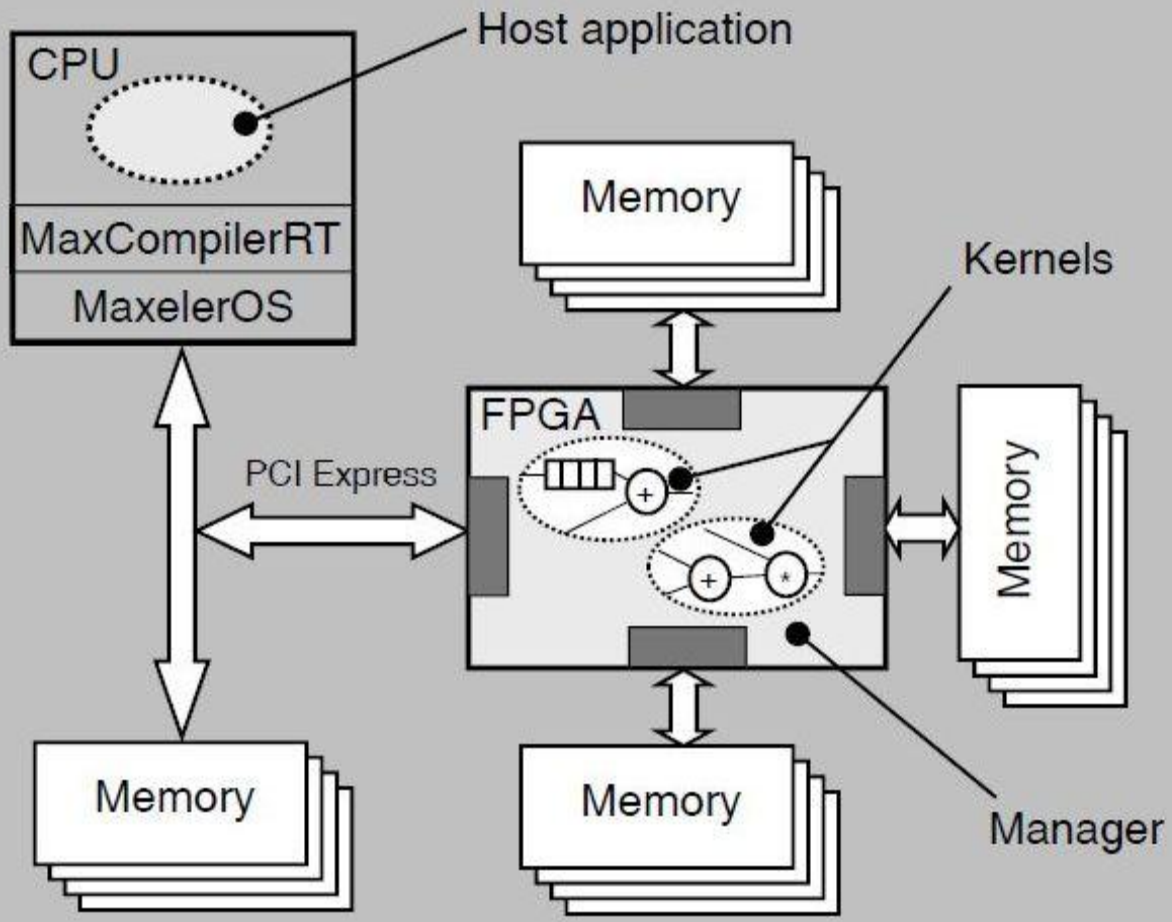
Compiling below the machine code level brings speedups;
also a smaller power, size, and cost.

The price to pay:
The machine is more difficult to program.

Consequently:
Ideal for WORM applications :)

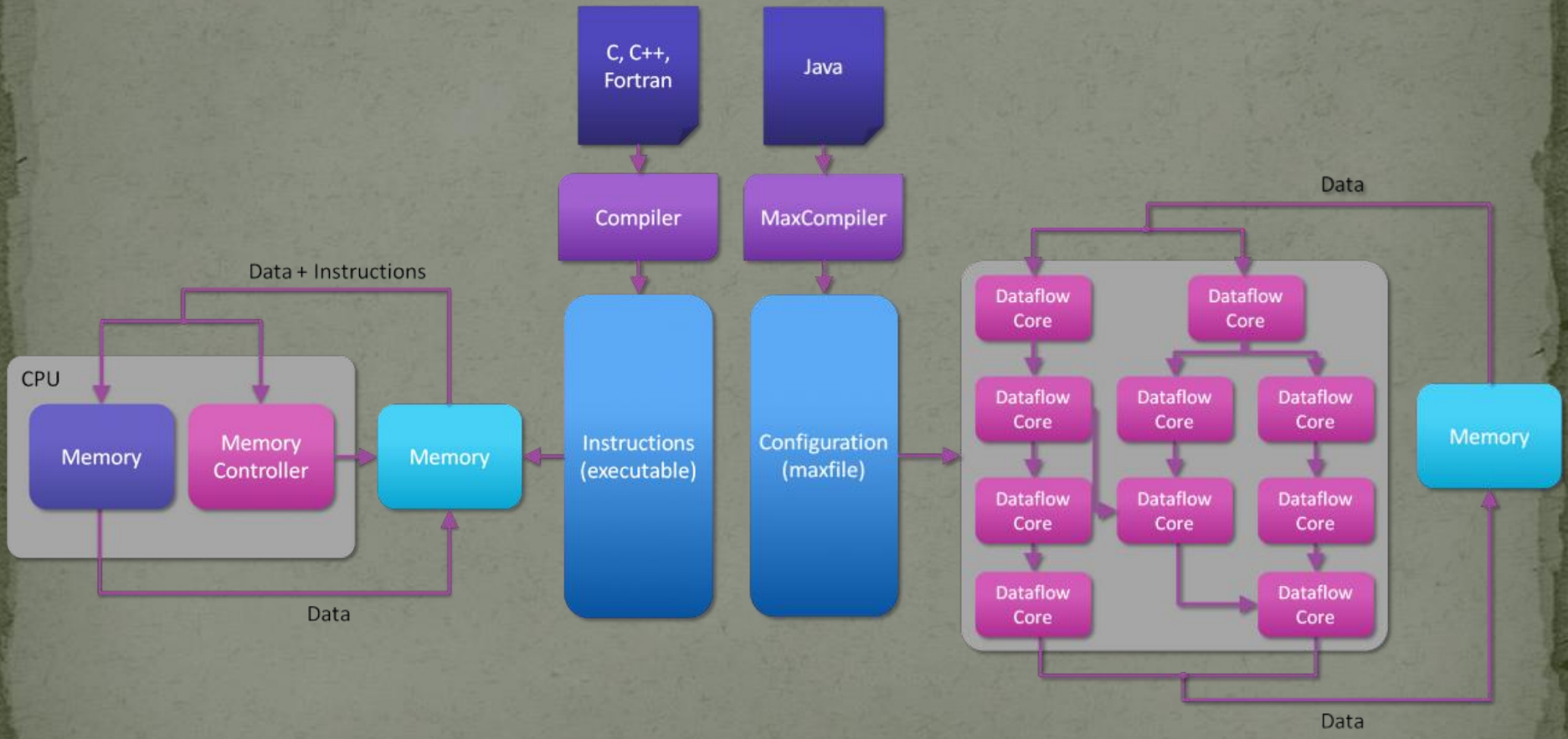
Examples using Maxeler:
GeoPhysics (20-40), Banking (200-1000, with JP Morgan 20%), M&C (New York City), Datamining (Google), ...

Generic Acceleration Architecture



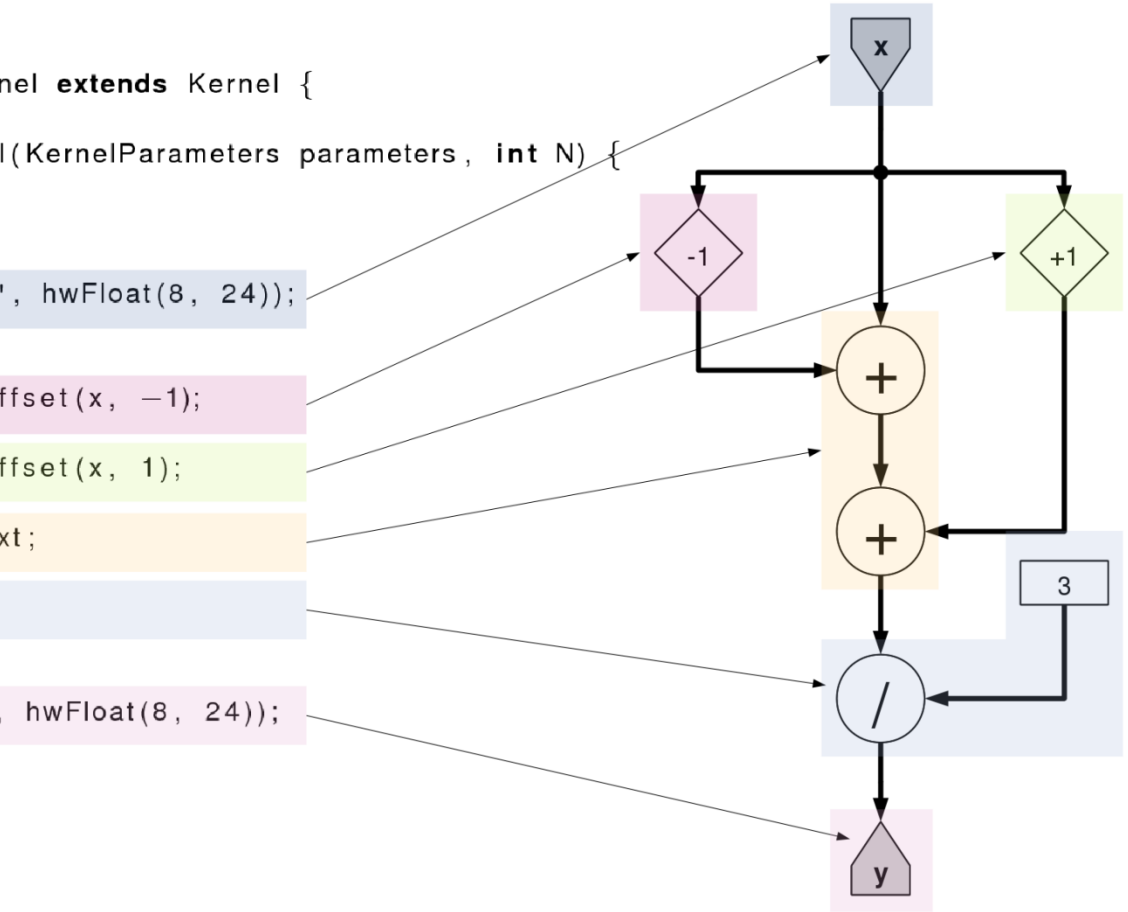
MAXELER Technologies
MAXIMUM PERFORMANCE COMPUTING

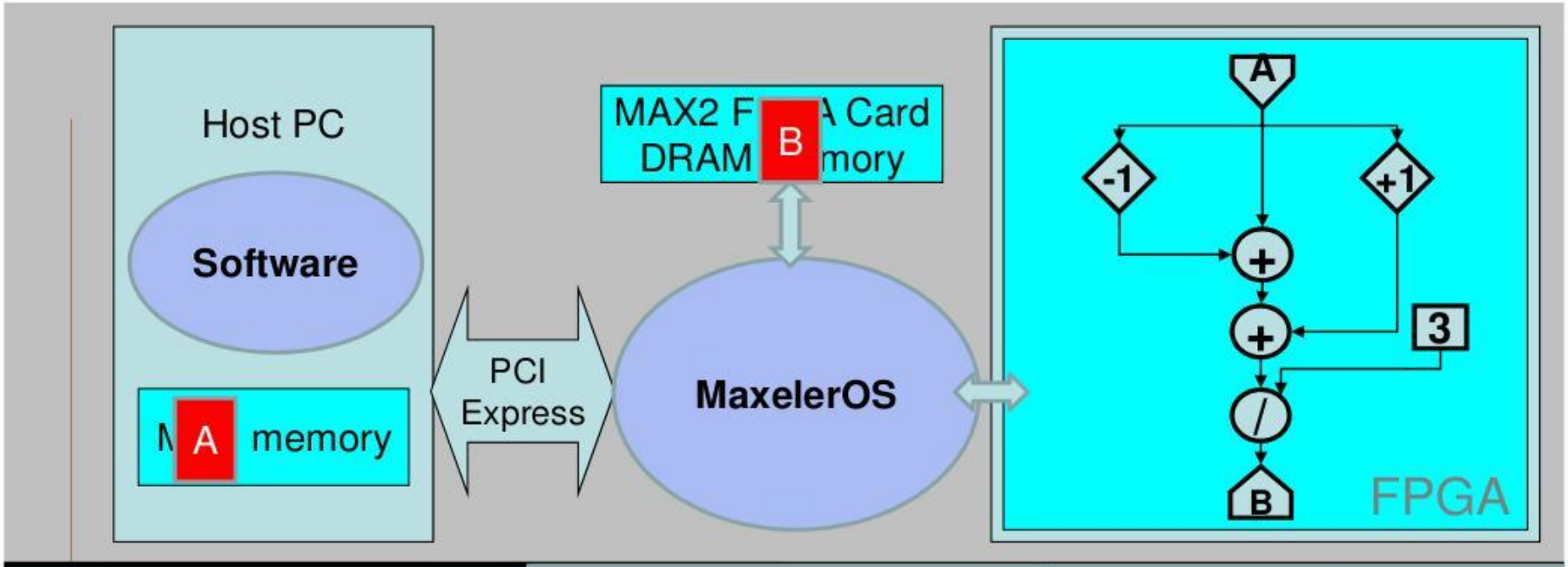
ControlFlow vs. DataFlow



DataFlow Programming (1C +5Java)

```
 7  public class MovingAverageKernel extends Kernel {
 8
 9      public MovingAverageKernel(KernelParameters parameters, int N) {
10          super(parameters);
11
12          // Input
13          HWVar x = io.input("x", hwFloat(8, 24));
14
15          // Data
16          HWVar prev = stream.offset(x, -1);
17
18          HWVar next = stream.offset(x, 1);
19
20          HWVar sum = prev+x+next;
21
22          HWVar result = sum/3;
23
24          // Output
25          io.output("y", result, hwFloat(8, 24));
26      }
27  }
```





Software

C

```
device = max_open_device(
         maxfile, "/dev/max0");
float A[SIZE];
...
stream_data(device, A);

for (int i=0; i<SIZE; ++i) {
  B[i] = ( A[i-1] + A[i] + A[i+1] )/3;
}

...
```

Manager

Java

```
Manager m = new
    Manager("Loop", MAX2);

m.kernel(mav_kernel,
      link("A", PCIE),
      link("B", DRAM(LINEAR));

m.build();
```

Kernel

MaxJava

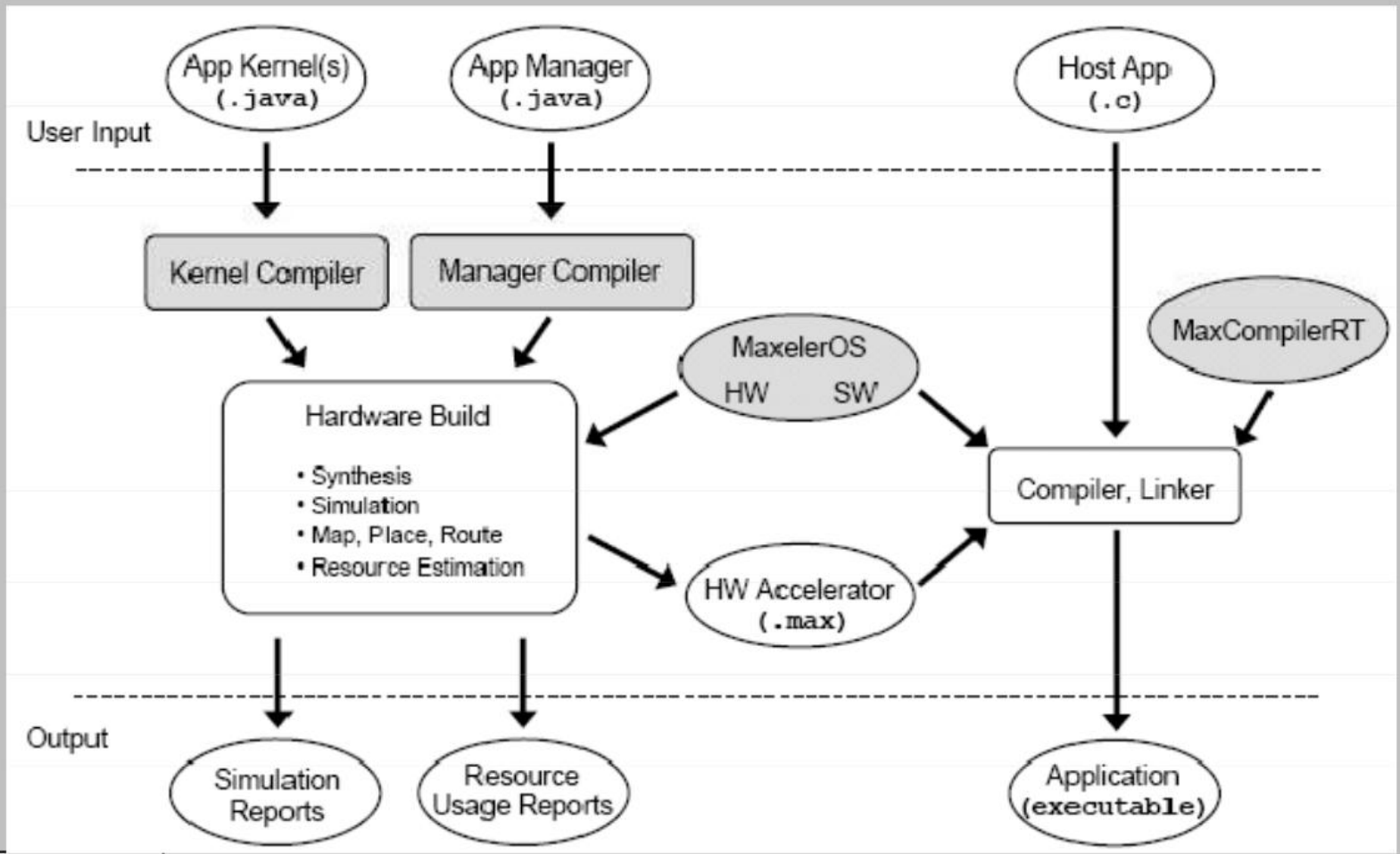
```
class mav_kernel
            extends kernel{

  input ("A",hwFloat (12 ,52) ) ;
  output ("B",hwFloat (12 ,52) ) ;

  A_prev=streamOffset(-1,A);
  A_next=streamOffset(1,A);

  B = (A_prev+A+A_next) / 3 ;
}
```

MaxCompiler



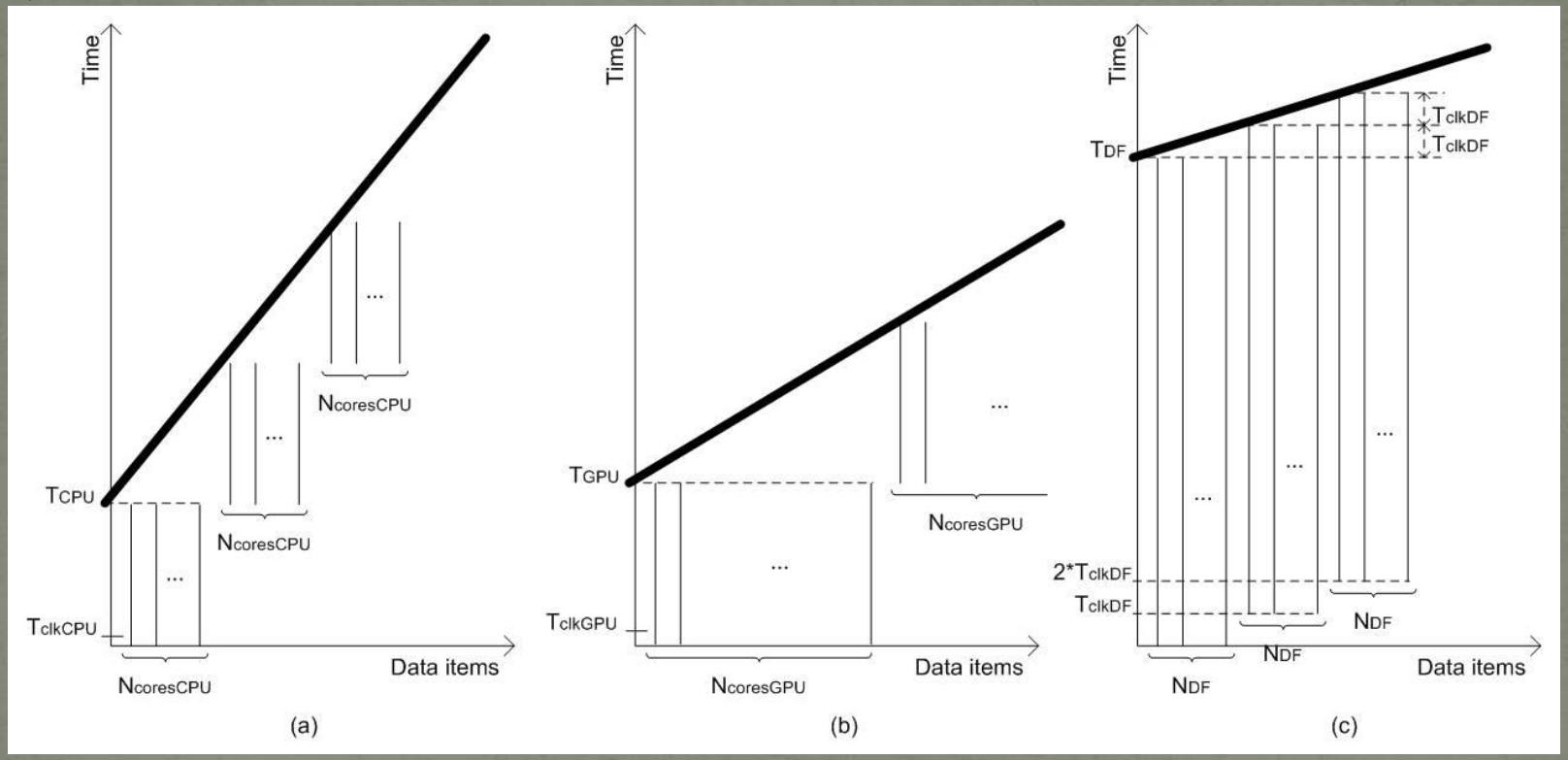
MAXELER Technologies
MAXIMUM PERFORMANCE COMPUTING

Essence: Our Paper in IEEE Computer

tCPU =
*N * NOPS * CCPU*TclkCPU*
/NcoresCPU

tGPU =
*N * NOPS * CGPU*TclkGPU /*
NcoresGPU

*tDF = NOPS * CDF * TclkDF +*
*(N – 1) * TclkDF / NDF*



Assumptions:

1. Software includes enough parallelism to keep all cores busy
2. The only limiting factor is the number of cores.

MultiCore

DualCore?

Which way are the horses going?

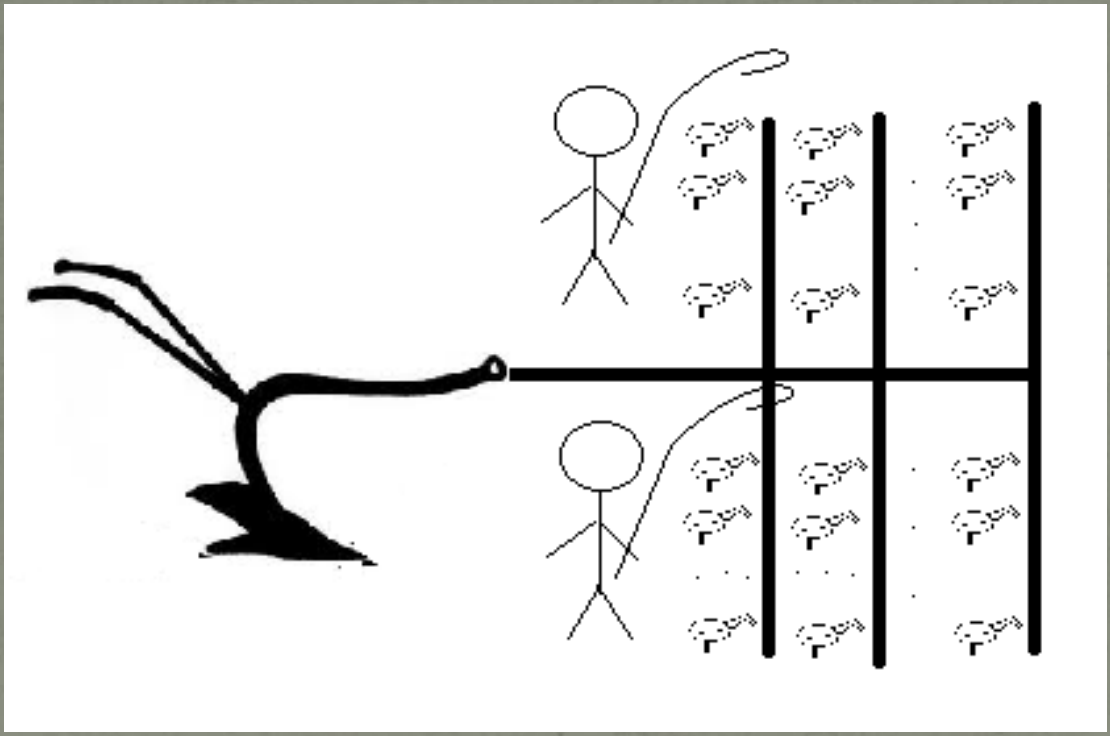
ManyCore

- Is it possible
  to use 2000 chicken instead of two horses?

?
==

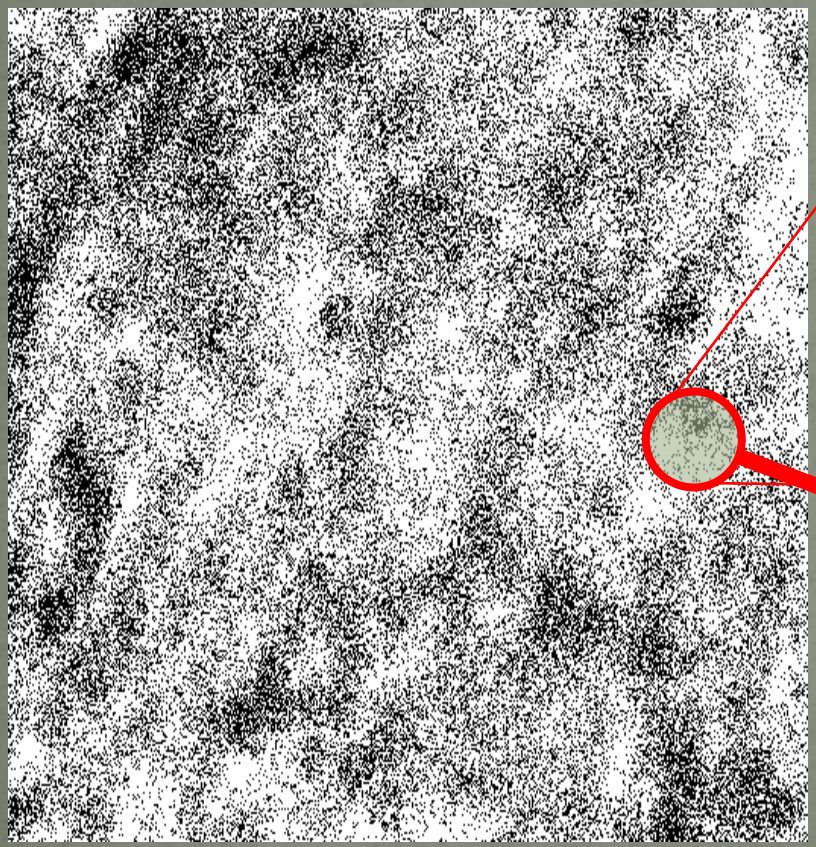
- What is better, real and anecdotic?

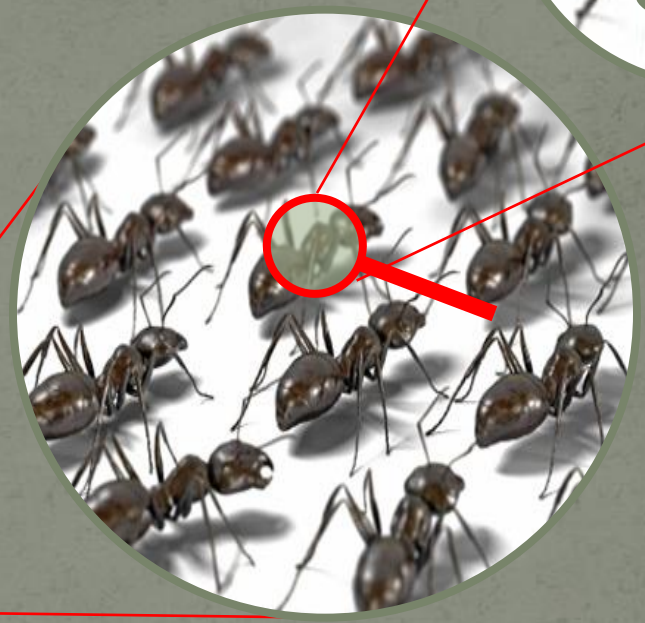
ManyCore



2 x 1000 chickens (CUDA and rCUDA)

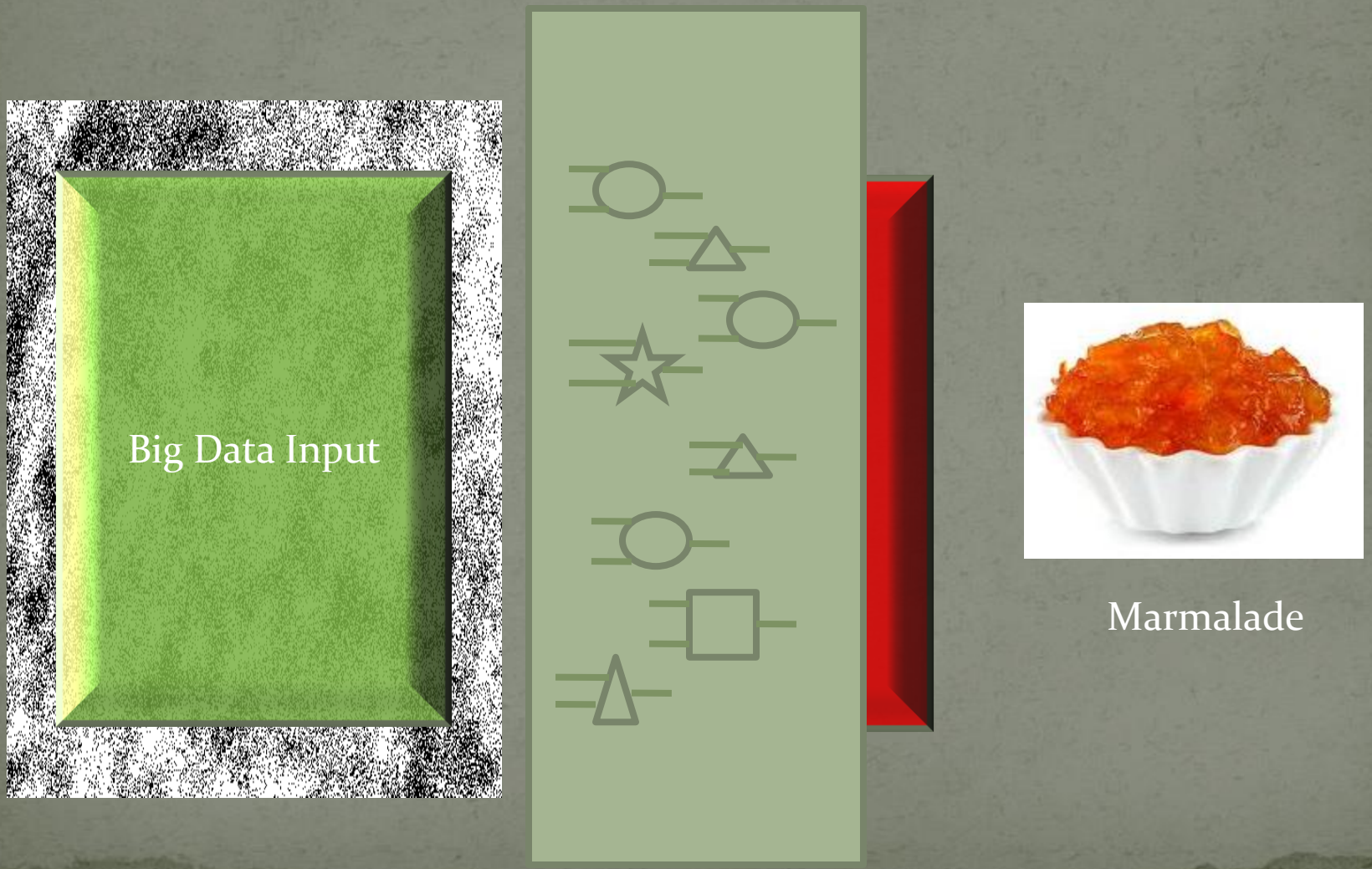
DataFlow





How about 2 000 000 ants?

DataFlow

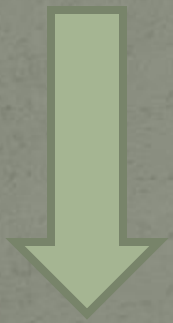


Marmalade

Why is DataFlow so Much Faster?

- Factor: 20 to 200

MultiCore/ManyCore



Machine Level Code

Dataflow

Gate Transfer Level

Why are Electricity Bills so Small?

- Factor: 20

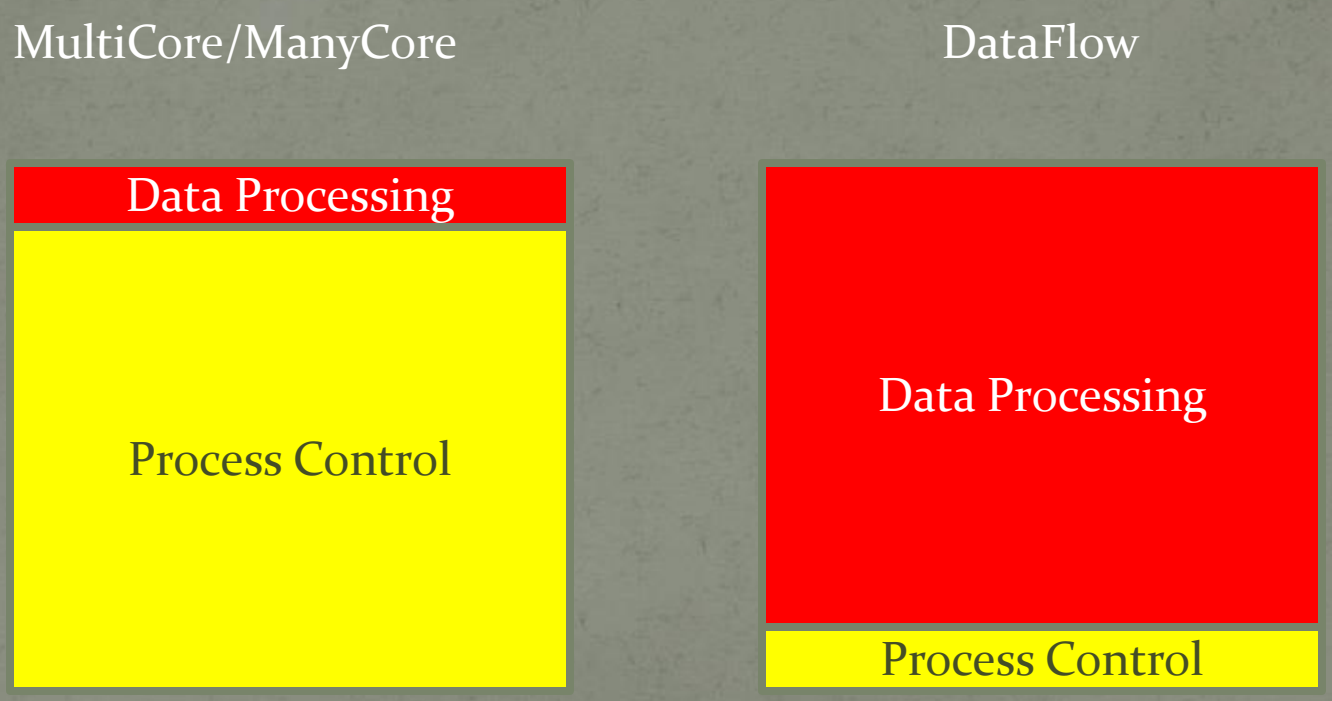
MultiCore/ManyCore

Dataflow

$$P = kfU^2$$

Why is the Cubic Foot so Small?

- Factor: 20



Required Programming Effort?

- MultiCore:
  - Explain what to do, to the driver
  - Caches, instruction buffers, and predictors needed
- ManyCore:
  - Explain what to do, to many sub-drivers
  - Reduced caches and instruction buffers needed
- DataFlow:
  - Make a field of processing gates: 1C+nJava+4Java
  - No caches, etc. (300 students/year: BGD, BCN, LjU, ICL,…)

Required Debug Effort?

- MultiCore:
  - Business as usual
- ManyCore:
  - More difficult
- DataFlow:
  - Much more difficult
  - Debugging both, application and configuration code

Required Compilation Effort?

- MultiCore/ManyCore:
  - Several minutes
- DataFlow:
  - Several hours for the real hardware
  - Fortunately, only several minutes for the simulator
  - The simulator supports
    both the large JPMorgan machine
    as well as the smallest “University Support” machine
- Good news:
  - Tabula@2GHz

Now the Fun Part

Required Space?

- MultiCore:
  - Horse stable

- ManyCore:
  - Chicken house

- DataFlow:
  - Ant hole

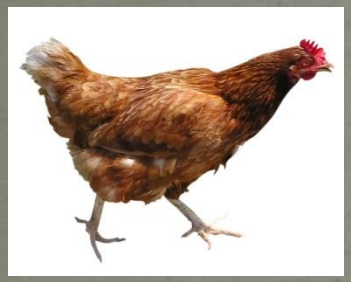
Required Energy?

- MultiCore:
  - Haystack

- ManyCore:
  - Cornbits

- DataFlow:
  - Crumbs

Why Faster?

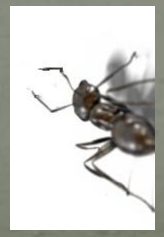


Small Data: Toy Benchmarks (e.g., Linpack)

Why Faster?

Medium Data
(benchmarks
favorising Nvidia,
compared to Intel,...)

Why Faster?



DataFlow for ExaScale DM

- Revisiting the Top 500 SuperComputer Benchmarks
  - Our paper in Communications of the ACM
- Revisiting all major Big Data DM algorithms
  - Massive static parallelism at low clock frequencies
- Concurrency and communication
  - Concurrency between millions of tiny cores difficult, "jitter" between cores will harm performance at synchronization points
- Reliability and fault tolerance
  - 10-100x fewer nodes, failures much less often
- Memory bandwidth and FLOP/byte ratio
  - Optimize **data choreography, data movement**, and the algorithmic computation

Maxeler Hardware

MPC C Series

CPUs plus DFEs
Intel Xeon CPU cores and up to 4 DFEs with 192GB of RAM

MPC X Series

DFEs shared over Infiniband
Up to 8 DFEs with 384GB of RAM and dynamic allocation of DFEs to CPU servers

MPC N Series

Low latency connectivity
Intel Xeon CPUs and 1-2 DFEs with up to six 10Gbit Ethernet connections

MaxWorkstation
Desktop development system

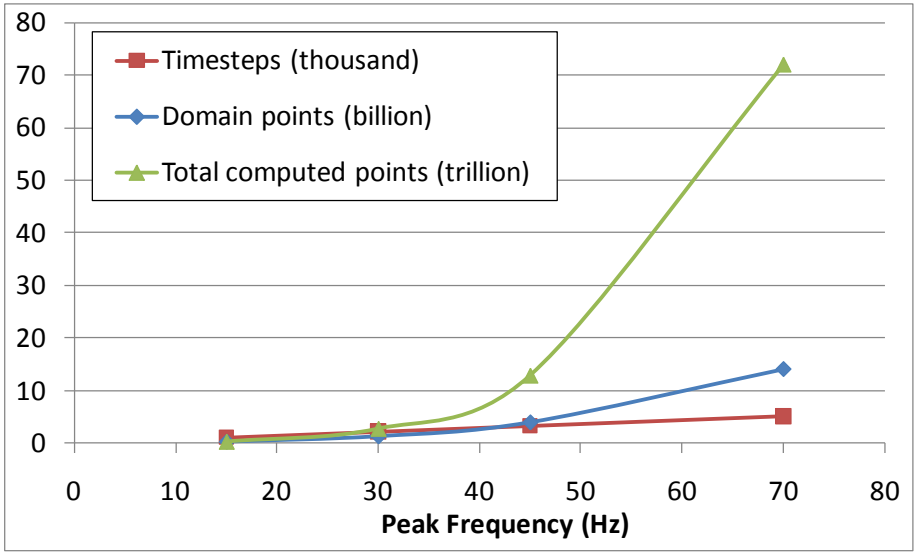
MaxCloud
On-demand scalable accelerated compute resource, hosted in London

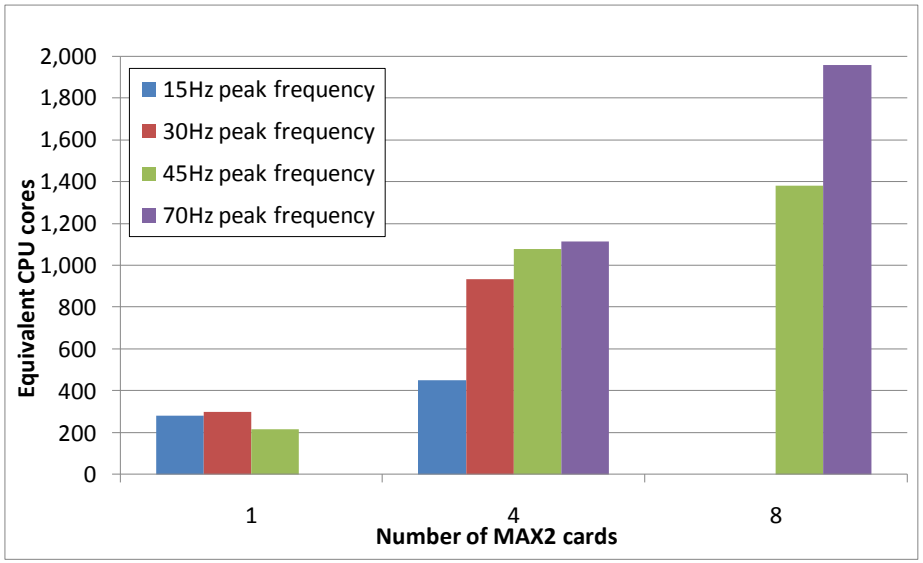
Major Classes of Algorithms, from the Computational Perspective

1. Coarse grained, stateful: Business
   - CPU requires DFE for minutes or hours
2. Fine grained, transactional with shared database: DM
   - CPU utilizes DFE for ms to s
   - Many short computations, accessing common database data
3. Fine grained, stateless transactional: Science (FF)
   - CPU requires DFE for ms to s
   - Many short computations

Coarse Grained: Modeling

- Long runtime, but:
- Memory requirements change dramatically based on modelled frequency
- Number of DFEs allocated to a CPU process can be easily varied to increase available memory
- Streaming compression
- Boundary data exchanged over chassis MaxRing



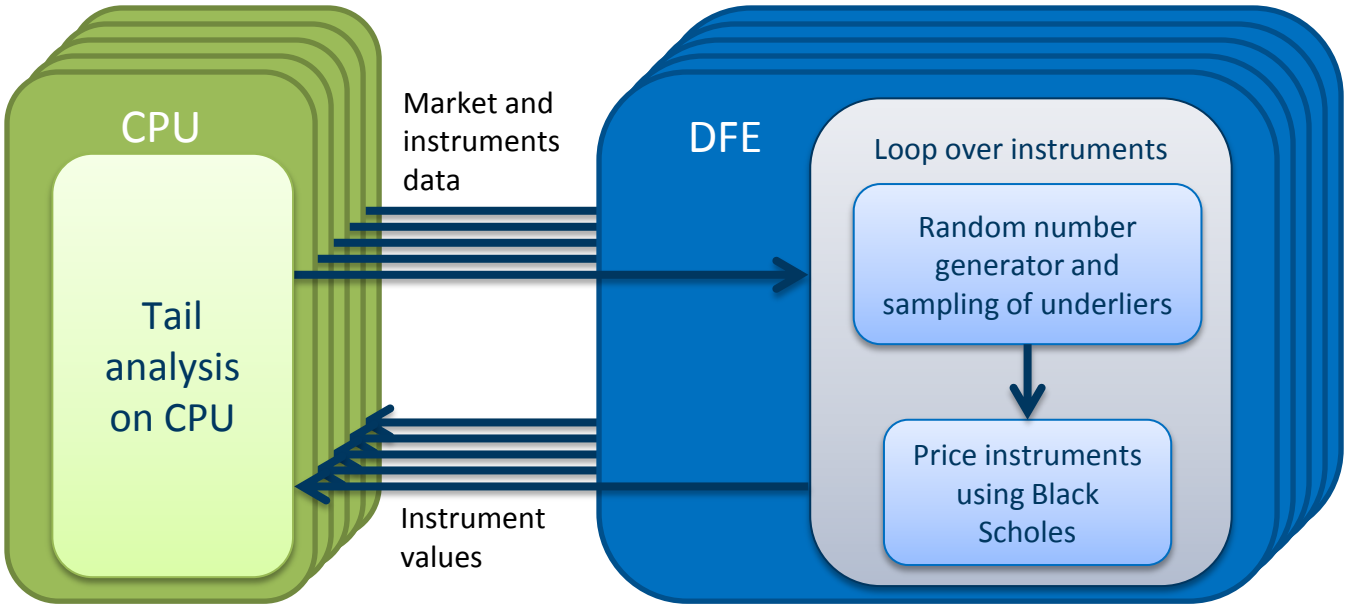


Fine Grained, Shared Data: Monitoring

- DFE DRAM contains the database to be searched
- CPUs issue transactions *find(x, db)*
- Complex search function
  - Text search against documents
  - Shortest distance to coordinate (multi-dimensional)
  - Smith Waterman sequence alignment for genomes
- Any CPU runs on any DFE
  that has been loaded with the database
  - MaxelerOS may add or remove DFEs
    from the processing group to balance system demands
  - New DFEs must be loaded with the search DB before use

Fine Grained, Stateless: The BSOP Control

- Analyse > 1,000,000 scenarios
- Many CPU processes run on many DFEs
  - Each transaction executes on *any* DFE in the assigned group atomically
- ~50x MPC-X vs. multi-core x86 node

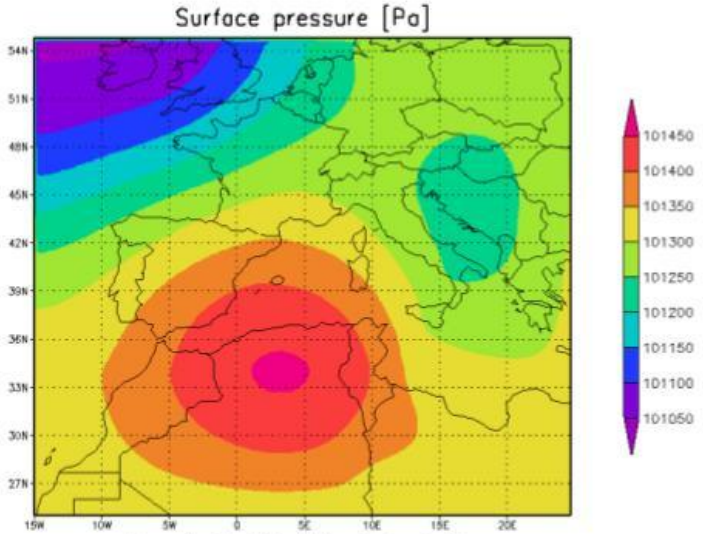


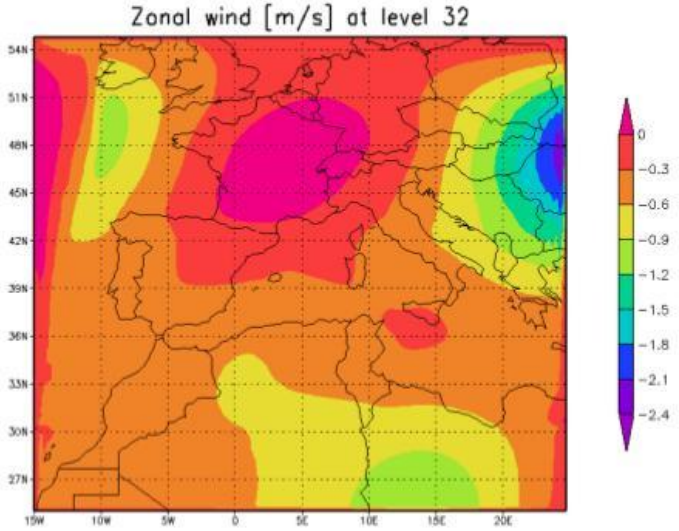
Selected FF Examples

More Results

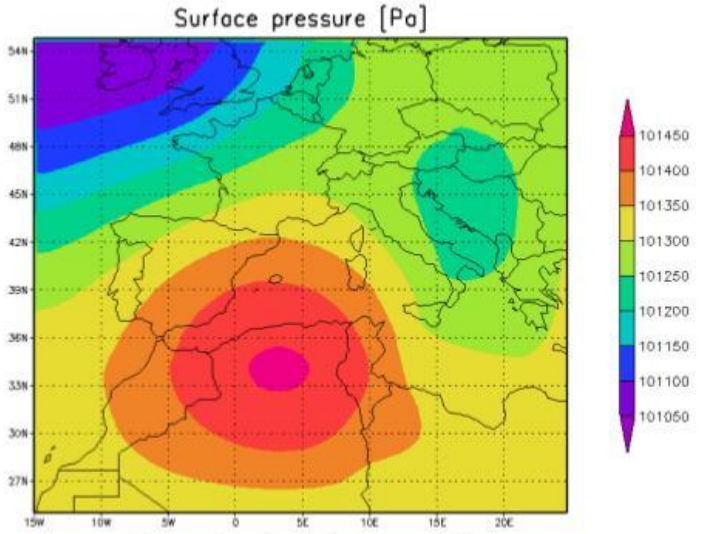
Problem size: (Longitude) 13,600 Km x (Latitude) 3330 Km
Simulation of baroclinic instability after 2 hrs (500 time steps)

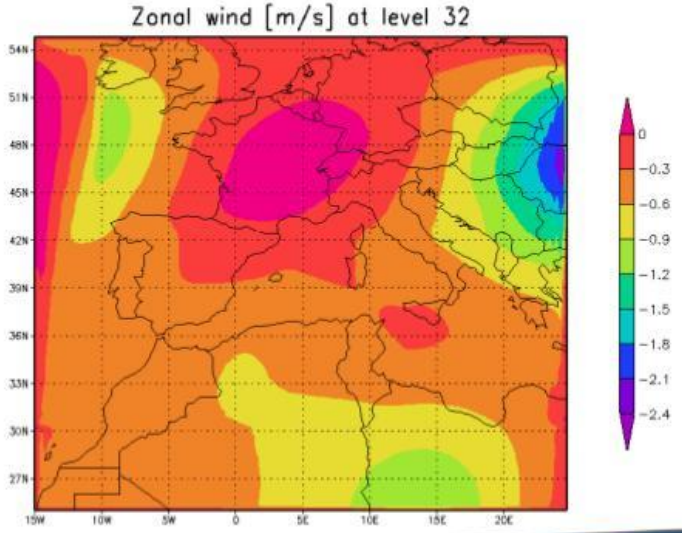
CPU





Dataflow

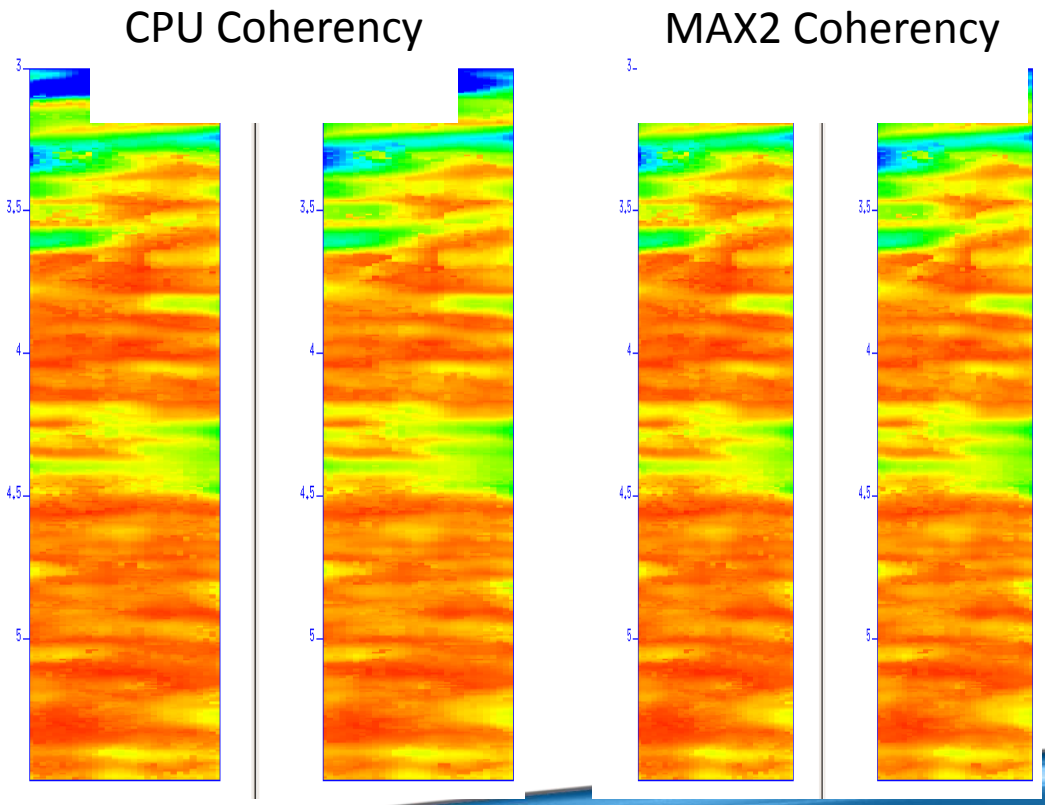




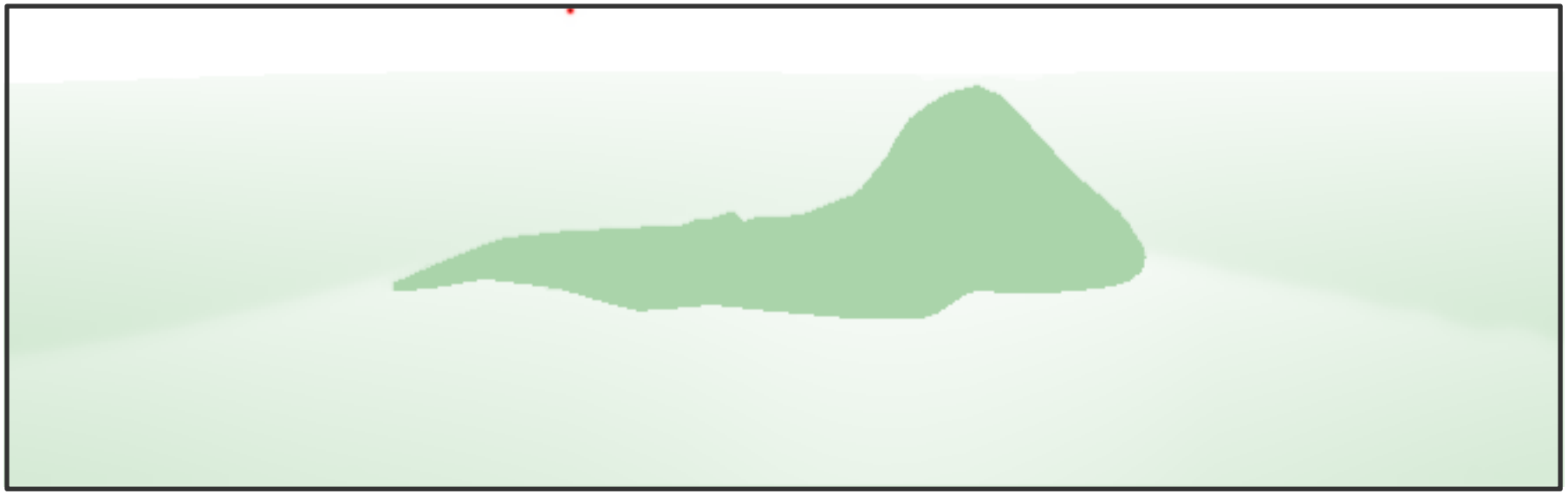
MAXELER Technologies

The CRS Results

- Performance of one MAX2 card vs. 1 CPU core
  - Land case (8 params), speedup of 230x
  - Marine case (6 params), speedup of 190x



Seismic Imaging



- Running on MaxNode servers
  - 8 parallel compute pipelines per chip
  - 150MHz => low power consumption!
  - 30x faster than microprocessors

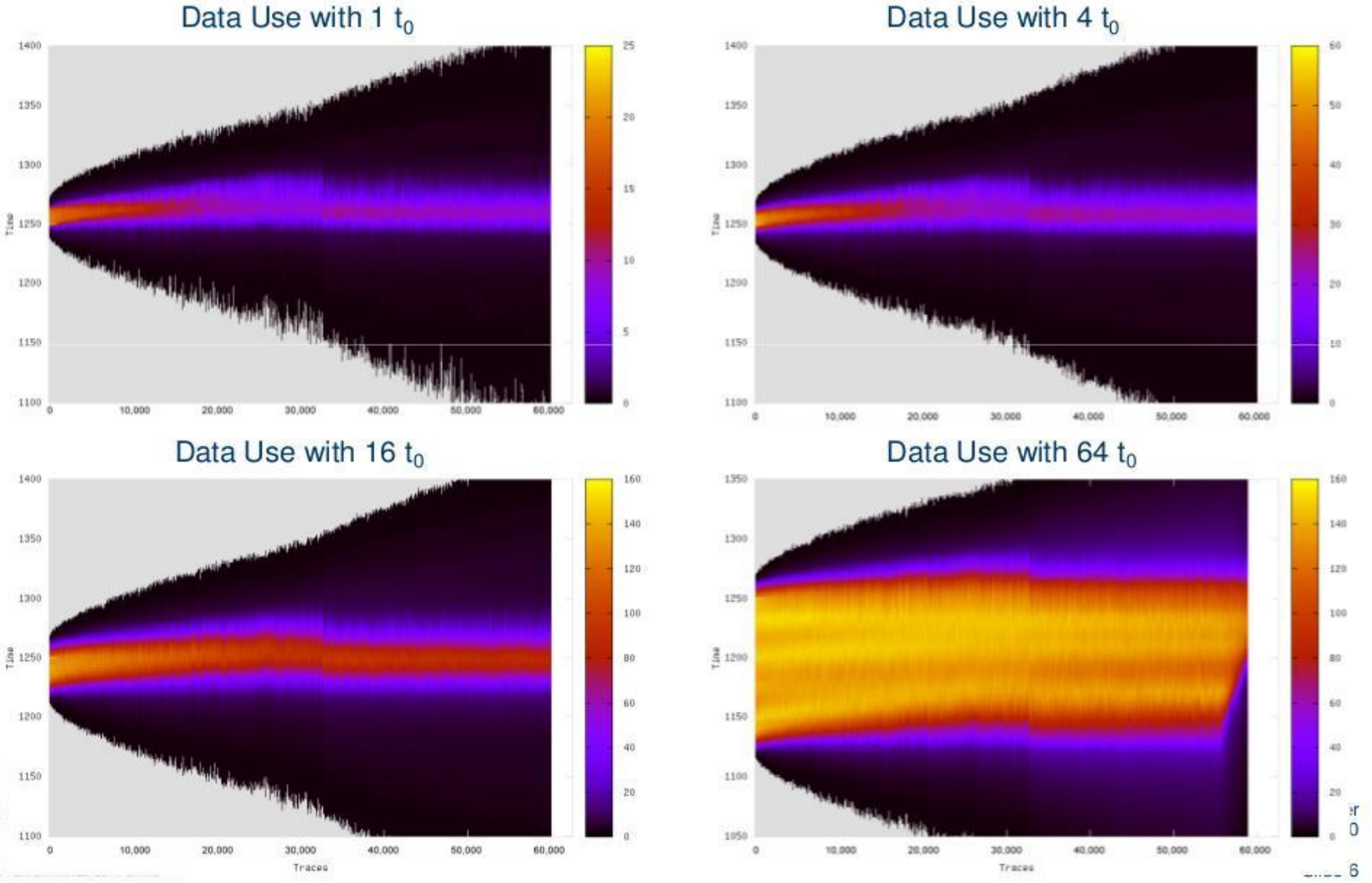
An Implementation of the Acoustic Wave Equation on FPGAs

T. Nemeth[†], J. Stefani[†], W. Liu[†], R. Dimond[‡], O. Pell[‡], R.Ergas[§]

[†]Chevron, [‡]Maxeler, [§]Formerly Chevron, SEG 2008

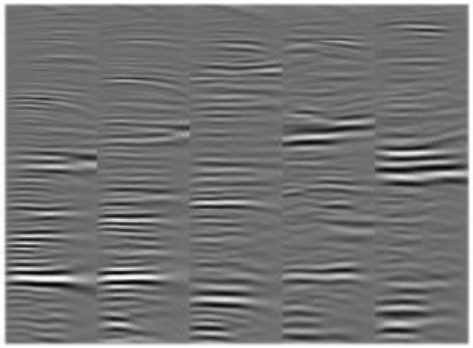
ENI-AGIP Seismic Trace App: Conjugate Gradient Optimization

100 MAX2 cards delivering performance of 21,800 CPU cores[EAGE2010]



48x Speedup of Angle Gathers

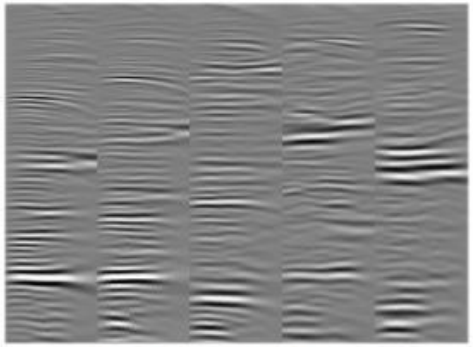
with Stanford Center for Earth and Environmental Sciences *)



Angle gathers from CPU computed subsurface offsets

- Can be dominant cost in shot profile migration
- Cross-correlating two fields by various shifts:

$$I(h,x,z) = \sum_{s} \sum_{w} S(x-h,z,w,s) \cdot G^{*}(x+h,z,w,s)$$



Angle gathers from FPGA computed subsurface offsets

SPEEDUP RESULTS FROM CUSTOM TRACE MEMORY SYSTEM:

- **Trace = Unit of Transfer**
- **Buffers Prefetch Right Traces in Advance**

*) Case Study: Subsurface Offset Gathers, presented at SEG 2008

Trace Stacking: Speed-up 217

- DM for Monitoring and Control in Seismic processing
- Velocity independent / data driven method to obtain a stack of traces, based on 8 parameters
  - Search for every sample of each output trace

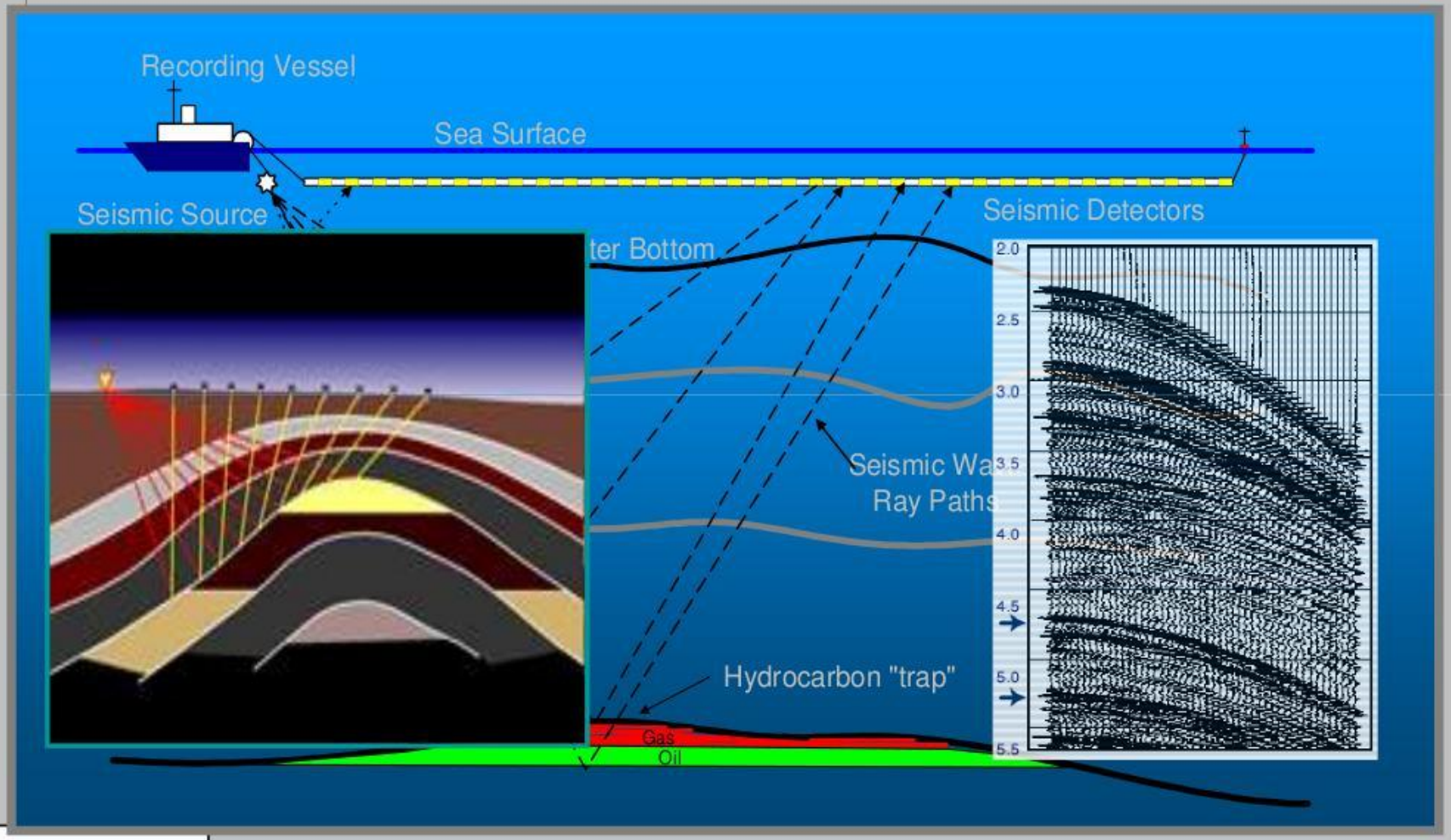
$$t_{hyp}^2 = \left(t_0 + \frac{2}{v_0} \mathbf{w}^T \mathbf{m} \right)^2 + \frac{2t_0}{v_0} \left(\mathbf{m}^T \mathbf{H}_{zy} \mathbf{K}_N \mathbf{H}_{zy}^T \mathbf{m} + \mathbf{h}^T \mathbf{H}_{zy} \mathbf{K}_{NIP} \mathbf{H}_{zy}^T \mathbf{h} \right)$$

2 parameters (emergence angle & azimuth)

3 Normal Wave front parameters ($K_{N,11}$; $K_{N,12}$; K_{N22})

3 NIP Wave front parameters ($K_{Nip,11}$; $K_{Nip,12}$; K_{Nip22})

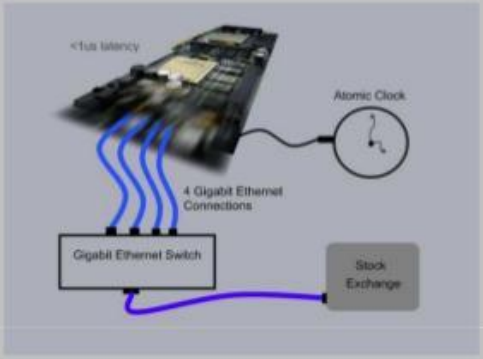
Seismic Data Acquisition



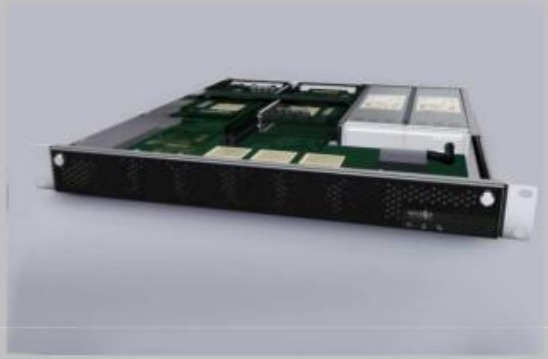
Courtesy of Schlumberger

Maxeler Technologies

MaxCard
e.g. HFT Solution

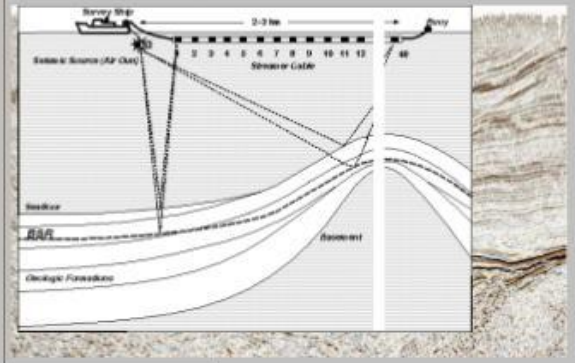


MaxBox
4 MaxCards in a 1U box

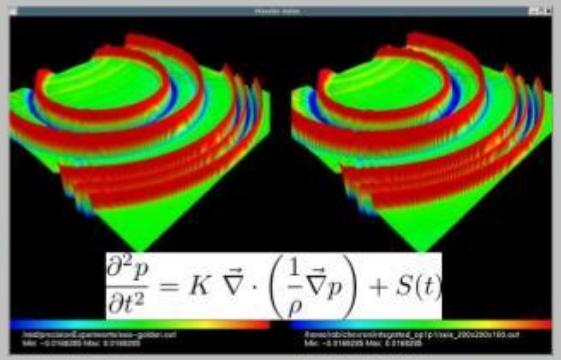


MaxRack
Storage, Network and Compute

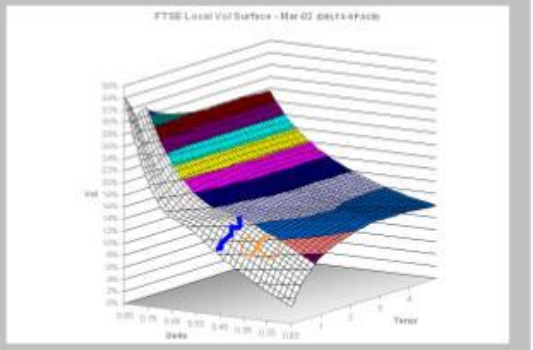
Real-time trace processing



Finite Difference (with Chevron)



Local Vol Approximation



MAXELER Technologies
MAXIMUM PERFORMANCE COMPUTING

Acceleration is Hard

Conclusion: Nota Bene

This is about algorithmic changes,
to maximize
the algorithm to architecture match:

Data choreography
and
process modifications

The winning paradigm
of Big Data DM/FF?

The TriPeak

BSC
+ Imperial College
+ Maxeler
+ Belgrade

The TriPeak

MontBlanc = A ManyCore (NVidia) + a MultiCore (ARM)
Maxeler = A FineGrain DataFlow (FPGA)

How about a happy marriage?
MontBlanc (ompSS) and Maxeler (an accelerator)

In each happy marriage,
it is known who does what :)

The Big Data DM algorithms:
What part goes to MontBlanc and what to Maxeler?

Core of the Symbiotic Success

An intelligent DM algorithmic scheduler,
partially implemented for compile time,
and partially for run time.

At compile time:
Checking what part of code fits where
(MontBllanc or Maxeler): LoC 1M vs 2K vs 20K

At run time:
Rechecking the compile time decision,
based on the current data values.

Maxeler: Teaching (Google: prof vm)

TEACHING, VLSI, PowerPoints, Maxeler:

Maxeler Veljko Explanations, August 2012
Maxeler Veljko Anegdotic,
Maxeler Oskar Talk, August 2012
Maxeler Forbes Article
Flyer by JP Morgan
Flyer by Maxeler HPC
Tutorial Slides by Sasha and Veljko: Practice (Current Update)
Paper, unconditionally accepted for Advances in Computers by Elsevier
Paper, unconditionally accepted for Communications of the ACM
Tutorial Slides by Oskar: Theory (7 parts)
Slides by Jacob, New York
Slides by Jacob, Alabama
Slides by Sasha: Practice (Current Update)
Maxeler in Meteorology
Maxeler in Mathematics
Examples generated in Belgrade

THE COURSE ALSO INCLUDES DARPA METHODOLOGY FOR MICROPROCESSOR DESIGN,
with an example

Maxeler: Research (Google: good method)

Structure of a Typical Research Paper: Scenario #1
Curve A: MultiCore of approximately the same PurchasePrice
Curve B: Maxeler after a direct algorithm migration
Curve C: Maxeler after algorithmic modifications
Curve D: Maxeler after data choreography

Structure of a Typical Research Paper: Scenario #2
CurveSet A: Comparison of Algorithms on a MultiCore
CurveSet B: Comparison of Algorithms on a ManyCore
CurveSet C: Comparison on Maxeler, after a direct algorithm migration
CurveSet D: Comparison on Maxeler, after algorithmic modifications
CurveSet E: Comparison on Maxeler, after data choreography

Maxeler: Topics (Google: HiPeac Berlin)

SRB (TR):
KG: Blood Flow
NS: Combinatorial Math
BG1: MiSANU Math
BG2: Meteos Meteorology
BG3: Physics
BG4: Physics
(reusability with MPI/OpenMP vs effort to accelerate)

FP7 (Call 11):
ICL, UK,
BSC, Spain,
QPLAN, Greece,
ETF, Serbia,
University of Siena, Italy,
IJS, Slovenia, …

Q&A

vm@etf.rs