

The logo for the FPGA High Performance Computing Alliance (FHPCA) is displayed in white text on a blue rectangular background. The letters 'FHPCA' are in a bold, sans-serif font, with a vertical line separating the 'F' and 'H'.

FPGA High Performance Computing Alliance

# Maxwell: a 64-FPGA Supercomputer

[www.fhpca.org](http://www.fhpca.org)

Dr Rob Baxter  
Software Development Group Manager, EPCC  
R.Baxter@epcc.ed.ac.uk  
+44 131 651 3579

# Outline

---

- The FHPCA
- Why build Maxwell?
- Hardware details
- Software environment
- Demo applications
  - easy
  - harder
  - very hard...
- Concluding thoughts

# Who are the FHPCA?



- The FPGA High-Performance Computing Alliance:
  - EPCC (lead partner)
  - Alpha Data Ltd
  - Nallatech Ltd
  - Xilinx Corporation
  - Institute for System Level Integration
  - Algotronix Ltd
- The Alliance is funded by
  - the partners themselves
  - Scottish Enterprise Priority Industries Team
  - Scottish Funding Council under the eDIKT2 SRDG

# Why build Maxwell?

- Maxwell was completed earlier this year...
  - but what is it for?
- FPGAs can be used to accelerate some computations
  - their use as coprocessors is well understood
- But, can FPGAs be used
  - as main processors?
  - in parallel arrays?
  - against real HPC applications?
- These are the questions we set ourselves
  - answers are at the end of the talk...

# Maxwell at Edinburgh's ACF



- Maxwell comprises
  - five IBM BladeCentre chassis
  - 32 IBM Intel Xeon Blades
  - 64 Xilinx Virtex-4 FPGAs
  - Dell Precision 670 headnode (4 GB memory, 1 TB local SATA)
- Each Blade
  - diskless 2.8 GHz Intel Xeon with 1 GB main memory
  - hosts two FPGAs through a PCI-X expansion module
- FPGAs mounted in two card types
  - Nallatech H101
  - Alpha Data ADM-XRC-4FX

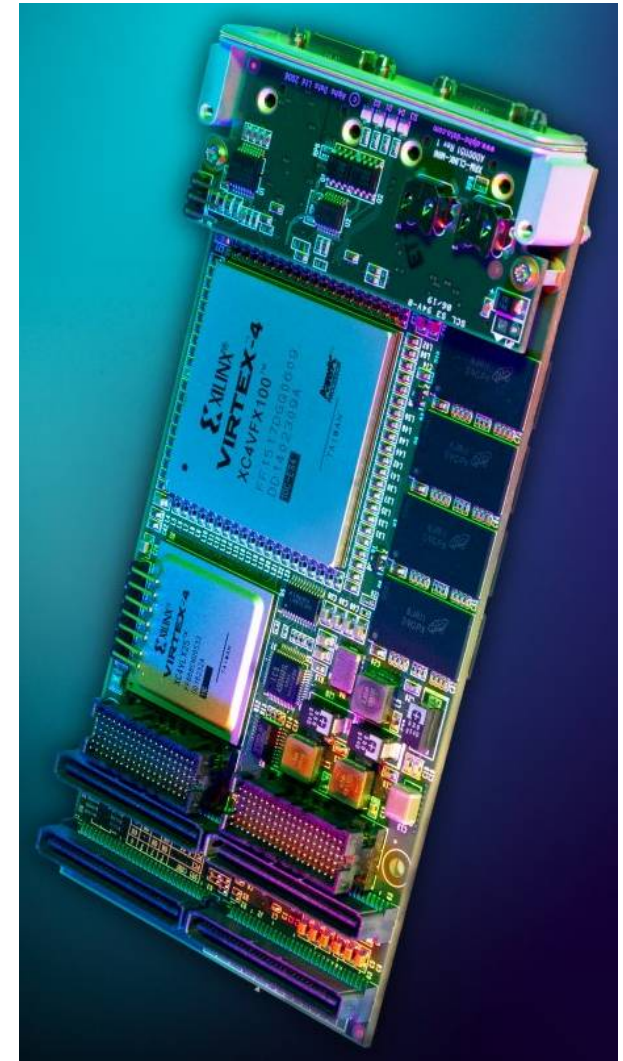
# Nallatech H101

- Xilinx V4LX160 main device
- 16 MB SRAM
  - 4x 4MB banks
  - 6.4 GB/s total bandwidth
- 512 MB SDRAM
  - 1x 512 bank
  - 3.2 GB/s total bandwidth
- V2Pro FX4 device for comms
  - 4x 2.5 Gb/s MGT ('RocketIO')



# Alpha Data ADM-XRC-4FX

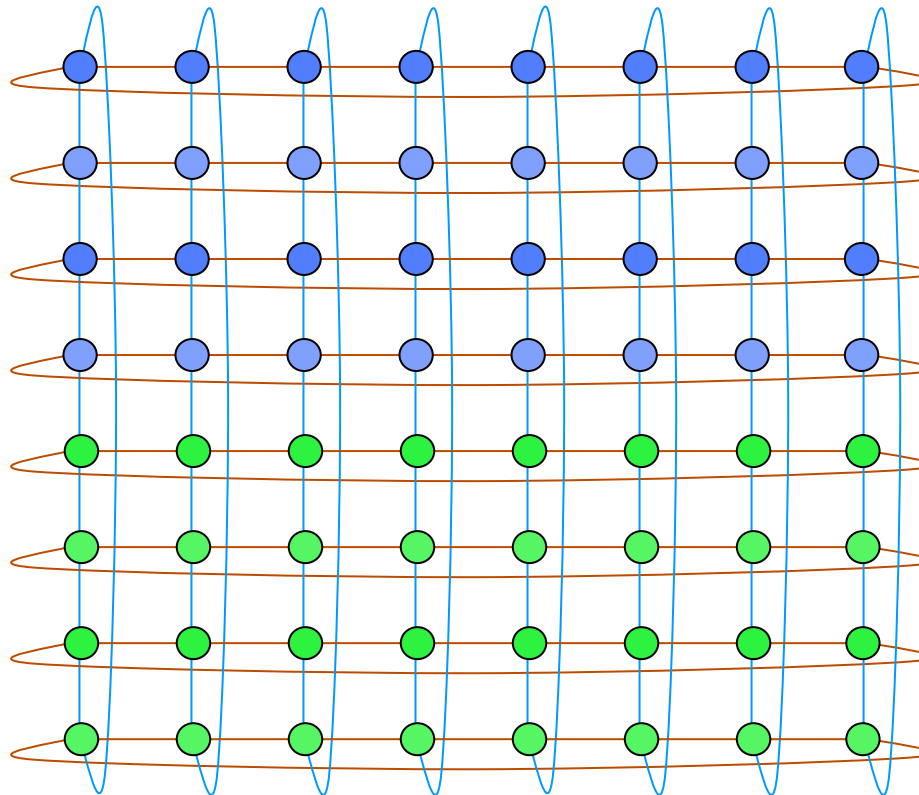
- Xilinx V4FX100 main device
- 16 MB SRAM
  - 4x 4MB banks
  - 6.4 GB/s total bandwidth
- 1,024 MB SDRAM
  - 4x 256MB banks
  - 8.4 GB/s total bandwidth
- Comms inherent in V4FX
  - 4x 3.125 Gb/s MGT ('RocketIO')



# Overall topology

- All 64 FPGAs are wired together directly
  - two-dimensional  $8 \times 8$  torus
  - this direct connection allows full distributed-memory parallel programming purely on the FPGAs
- The Xeons are connected over gigabit Ethernet
  - single 48-way Netgear switch
  - supports any inter-process communication that remains ‘above’ the FPGA level
- Thus two networks
  - all-to-all ‘software’ network
  - nearest-neighbour,  $8 \times 8$  , hardwired ‘FPGA’ network

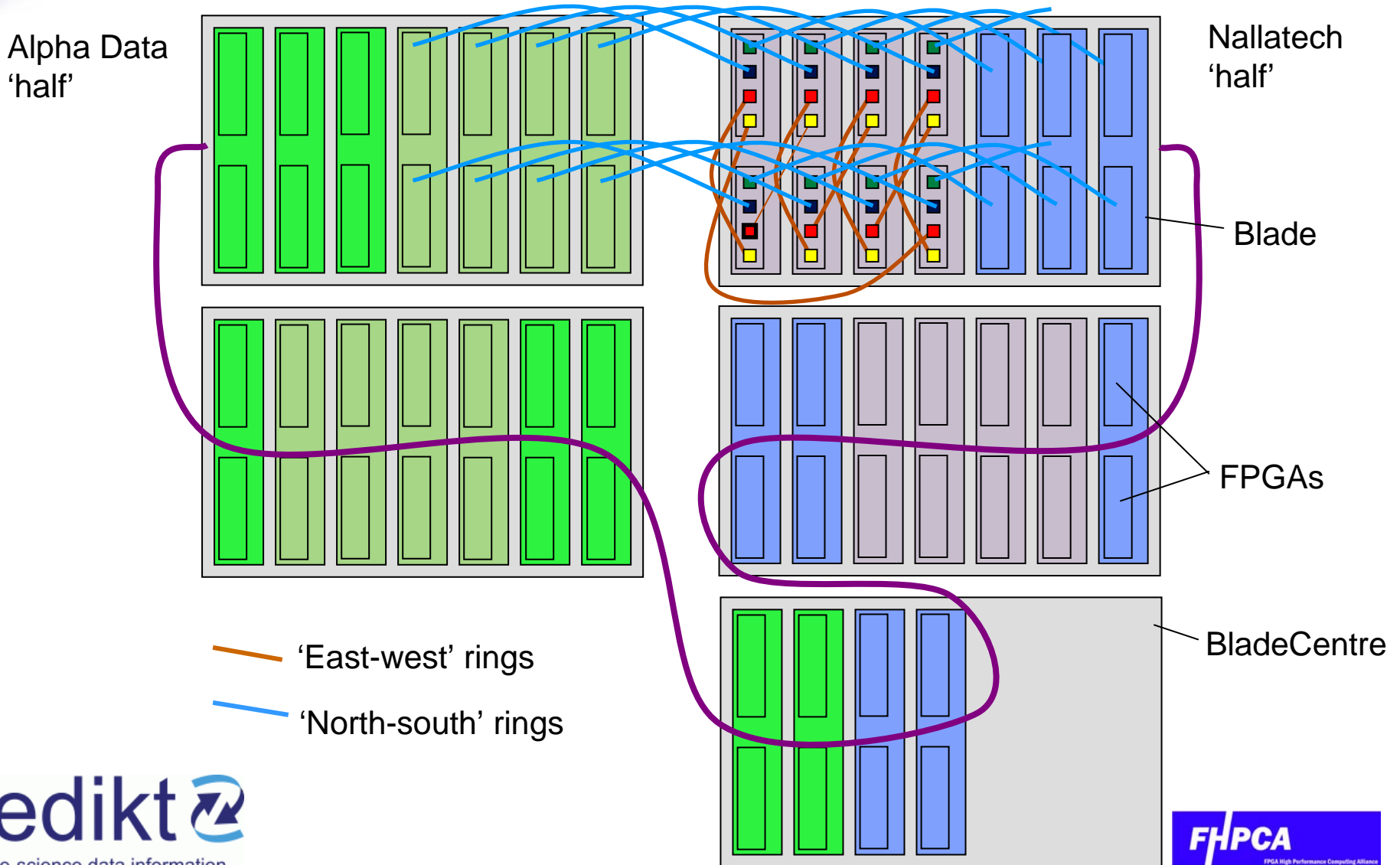
- FPGAs connected in a 2D torus of Rocket IO connections



FPGAs in  
Nallatech  
hardware

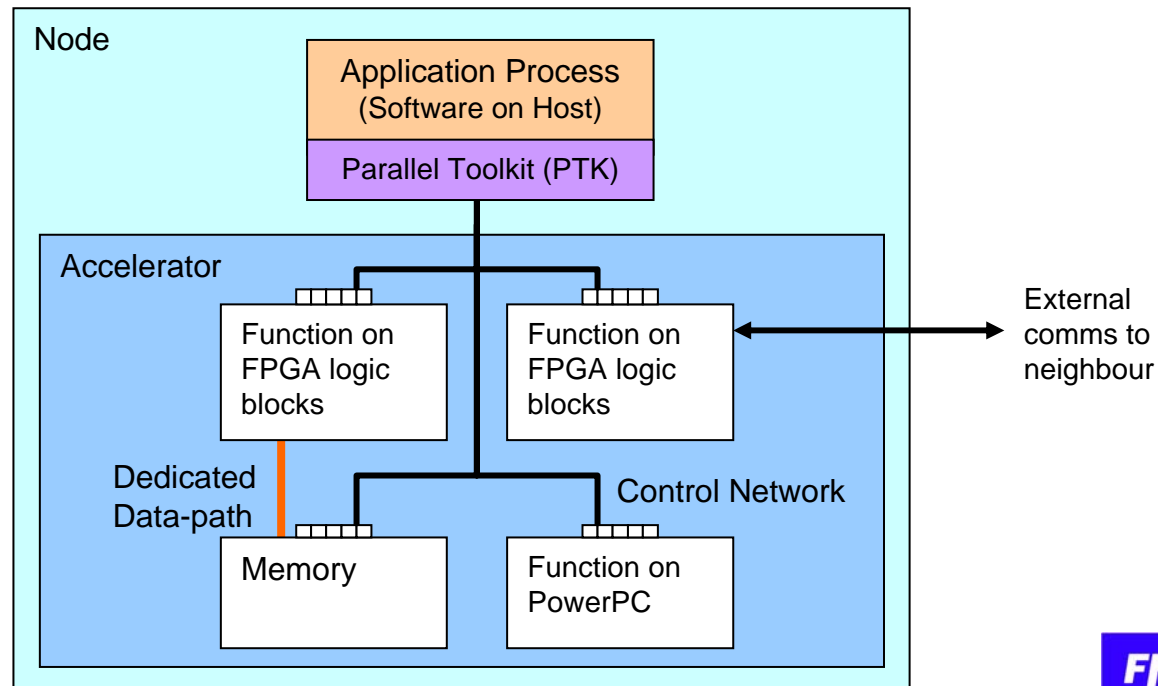
FPGAs in  
Alpha Data  
hardware

# Maxwell schematic



# Logical structure

- Logically, Maxwell can be regarded as a collection of nodes where a node is defined as:
  - a software process running on a host machine
  - plus some FPGA acceleration hardware



# Logical structure

- Typical configuration is 64 nodes
  - 64 software processes
  - each software process manages 1 FPGA
  - each Blade CPU hosts 2 software processes
- But could equally have 32 ‘fat nodes’
  - 32 software processes
  - each software process managing 2 FPGAs
  - each Blade CPU hosting 1 software process
- Logical structure is not set in stone
  - can be varied per application if beneficial

- Linux Red Hat-variant CentOS
- Standard GNU/Linux tools
- Sun Grid Engine (SGE) as the batch scheduling system
- MPI for inter-process communication
- Similar to other parallel clusters
- But Maxwell also has the FHPCA Parallel Toolkit (PTK)
  - a set of infrastructure and practices intended to address acceleration issues

# What is the Parallel Toolkit?

- The PTK is a set of practices and infrastructure intended to address identified acceleration issues e.g.
  - associating processes with FPGA resources
  - associating FPGAs with bitstreams
  - managing contention for FPGA resources within a process
  - managing code dependencies to facilitate re-use
- PTK infrastructure written mostly in C++
  - *bash* used for scripting tasks

# What's in the Parallel Toolkit? (1)

- Job configuration and launch
  - a repository of information used in configuring a given machine for a given application
  - a hardware-neutral, high-level way of configuring FPGAs
  - *ptkrun* script for launching accelerated applications
    - our equivalent of *mpirun* or *mpiexec*
- Accelerator classes
  - one Accelerator instance per process
  - models all the associated FPGA acceleration hardware
  - responsible for configuring any FPGA hardware
  - typically *application specific* but *hardware agnostic*

# What's in the Parallel Toolkit? (2)

- Component classes and Hard Data Structure classes
  - the “Abstract Interfaces”
  - model functional and data resources on the FPGA hardware
  - *hardware-neutral*, but by necessity *application-specific*
- Allocator classes
  - serve requests for Components and Hard Data Structures made by the application
- Runner classes
  - a level above the Components and Hard Data Structures
  - hides the need to request resources
  - presents simple-case interface to the application code above

- Monte Carlo simulation of stock option pricing
- Classic Black-Scholes model
  - $dS = S r dt + S \sigma \varepsilon \sqrt{dt}$
  - stock price  $S$ ; interest rate  $r$ ; time  $dt$ ; volatility  $\sigma$ , and Gaussian RN  $\varepsilon$
  - simple European options have closed-form solution
  - exotic Asian options need MC simulation
- Essentially an exercise in Gaussian RNG
- Simple core...
- ...and small data requirements

# Demo 2 – Facial Imaging

- Partner DI3D Ltd, medical imaging specialists
- 3 and 4D facial image reconstruction codes
- Pairwise merging and processing of images → 3D view
- Main aim is to batch process video images over 64 FPGAs
- Straightforward serial core...
  - c. 85% runtime on current data sizes
- ...and significant data requirements
  - images each 2-4 MB

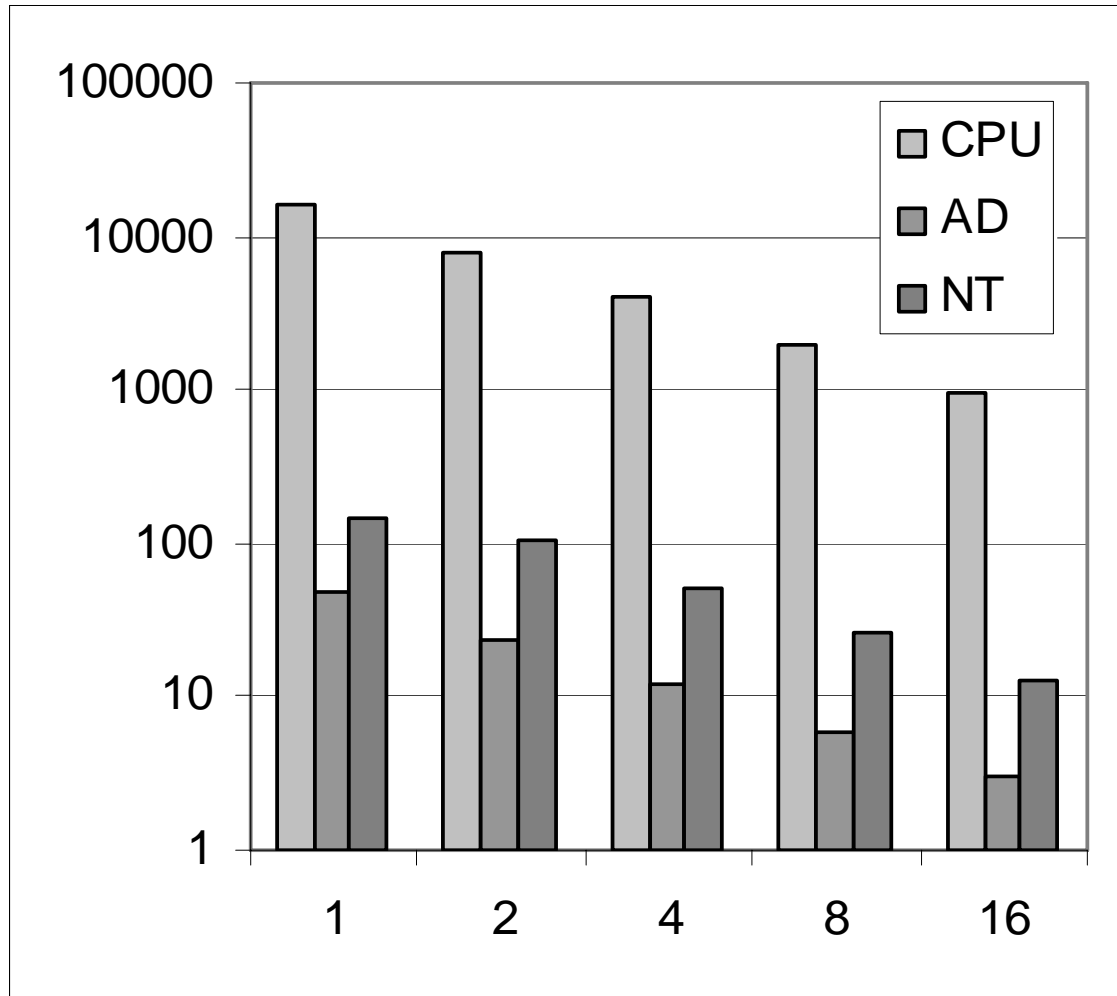
## Demo 3 – Oil & Gas

- Partner OHM plc, oil & gas services company
- 3D controlled source electromagnetics (CSEM) code
- Pretty typical physical simulation code
  - double precision
  - nine-point stencil (square nearest neighbour + corners)
  - logical regular mesh domain decomposed using MPI
- Has core parallel iterative solver...
  - c. 90% runtime for current data sizes
- ...and major data requirements
  - c. 500,000 point data sets and above

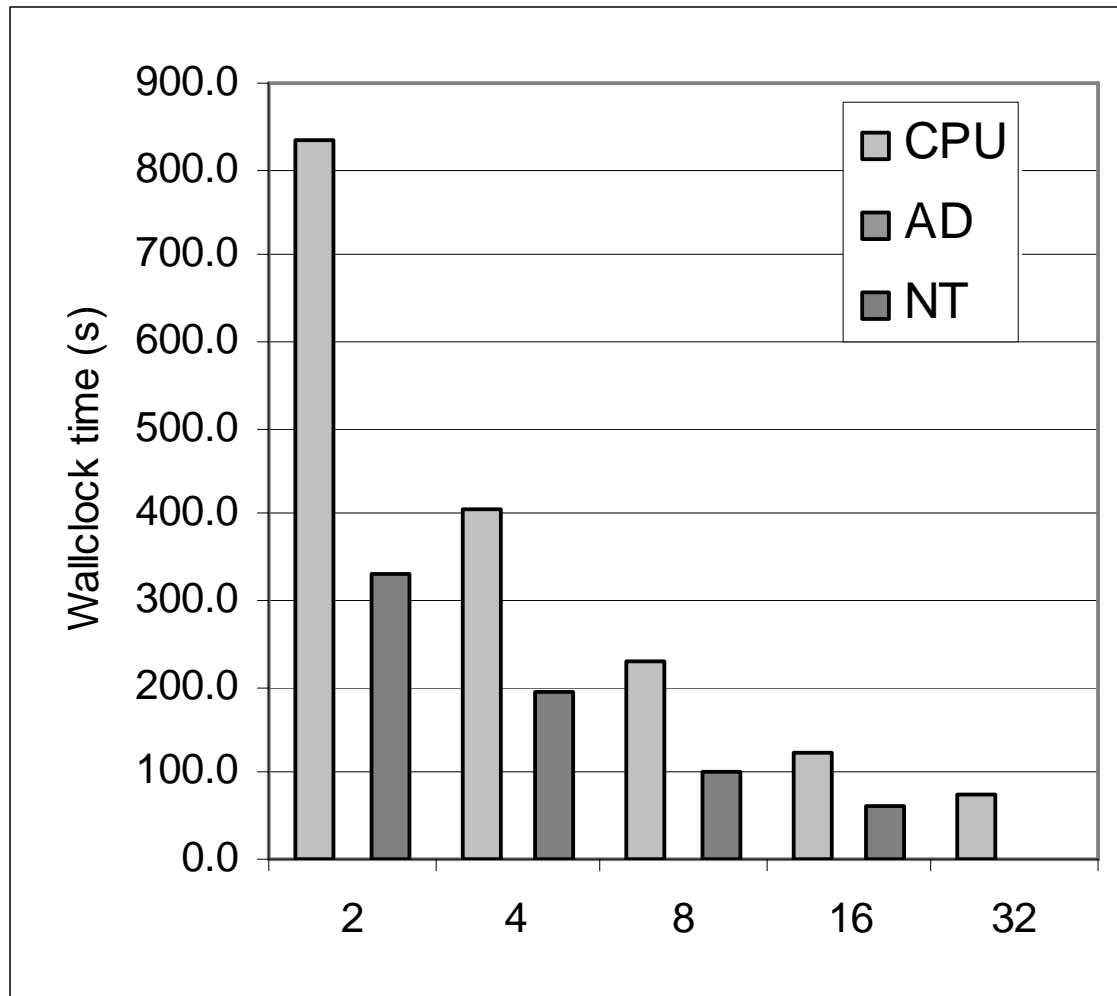
# Initial demo benchmarks

- Demo 1 – MCopt
  - runs 320× faster per FPGA
- Demo 2 – Facial Imaging
  - runs at sustained 2.5× faster per FPGA
- Demo 3 – CSEM
  - runs at sustained 5× faster per FPGA
  - scaling *still* not properly tested ☹
- Compared to software on Maxwell CPU (2.8 GHz Xeon)
  - IBM HS20 2.8GHz Xeon: SPECfp\_base2000 = 1559
  - cf. HS21XM 3.0GHz Xeon 2×2 core: SPECfp\_base2000 = 2636...
  - cf. Intel Core 2 2.13GHz: SPECfp\_base2000 = 2262...

# MCOpt performance (log scale)

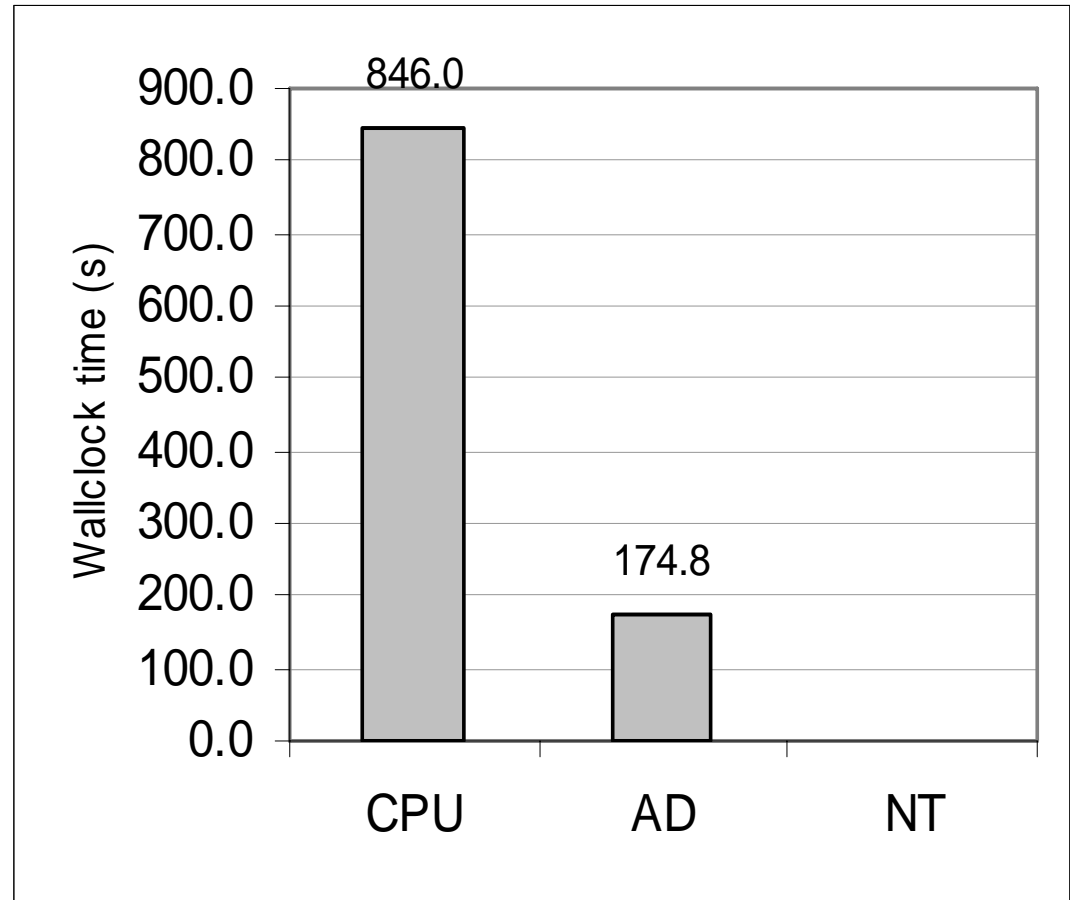


# DI3D performance



# OHM3D performance

- 8 node ring only
- Challenges remain in running different decompositions



- So, can FPGAs be used as main processors?
  - yes: you can fit a lot of logic on a V4100/160
  - but: complexity & compilation overhead makes development slow
- in parallel arrays?
  - yes: RocketIO is a good connection technology
  - but: some form of all-to-all routing is desirable
- against real HPC applications?
  - yes: where the numeric kernel is compact & well-defined
  - but: memory bandwidth limitations are still critical

# Top three challenges for FPGAs in HPC



## 1. Development costs

- cost of an *efficient* port of a major code is still too high
- four or five times longer than producing optimised software

## 2. Memory bandwidth

- HPC generally suffers lack of memory bandwidth
- FPGAs exacerbate this with more compute for same DDR cost

## 3. Amdahl's Law

- HPC performance demands parallel scalability
- accelerating small cores is not enough
- contrast OpenMP vs MPI

# Top three solutions (TODO)

## 1. Better tooling, more standardisation

- need common C dialects
- need standard APIs while avoiding library call overheads
  - eg. no calls to FPGA BLAS over PCI vs through L1 cache

## 2. Better FPGA-memory connectivity; better memory chips!

- multi-banked memory chips needed
- and not just by HPRC!

## 3. Parallel FPGAs

- FPGA-to-FPGA connectivity essential to keep up with the Joneses
- regard FPGAs as main compute platform, not just accelerator

# Next steps for FHPCA



- Ongoing industrial and academic collaboration programmes
  - we welcome academic collaborators and can pay travel costs for them to visit Edinburgh to work with us
  - you can also apply for development time on the supercomputer through the Technology Translator
- Proposal to EU FP7 to enhance programmability
  - with Xilinx, Nallatech, Alpha Data, ZIB, U. Ferrara, QUB
- Ongoing EPCC work supported by eDIKT2 and HPCx
- <http://www.fhpca.org>