Australian National University

Acceleration of parallel algorithms using a heterogeneous computing system

Ra Inta,

Centre for Gravitational Physics,

The Australian National University

NIMS-SNU-Sogang Joint Workshop on Stellar Dynamics and Gravitational-wave Astrophysics, December 18, 2012

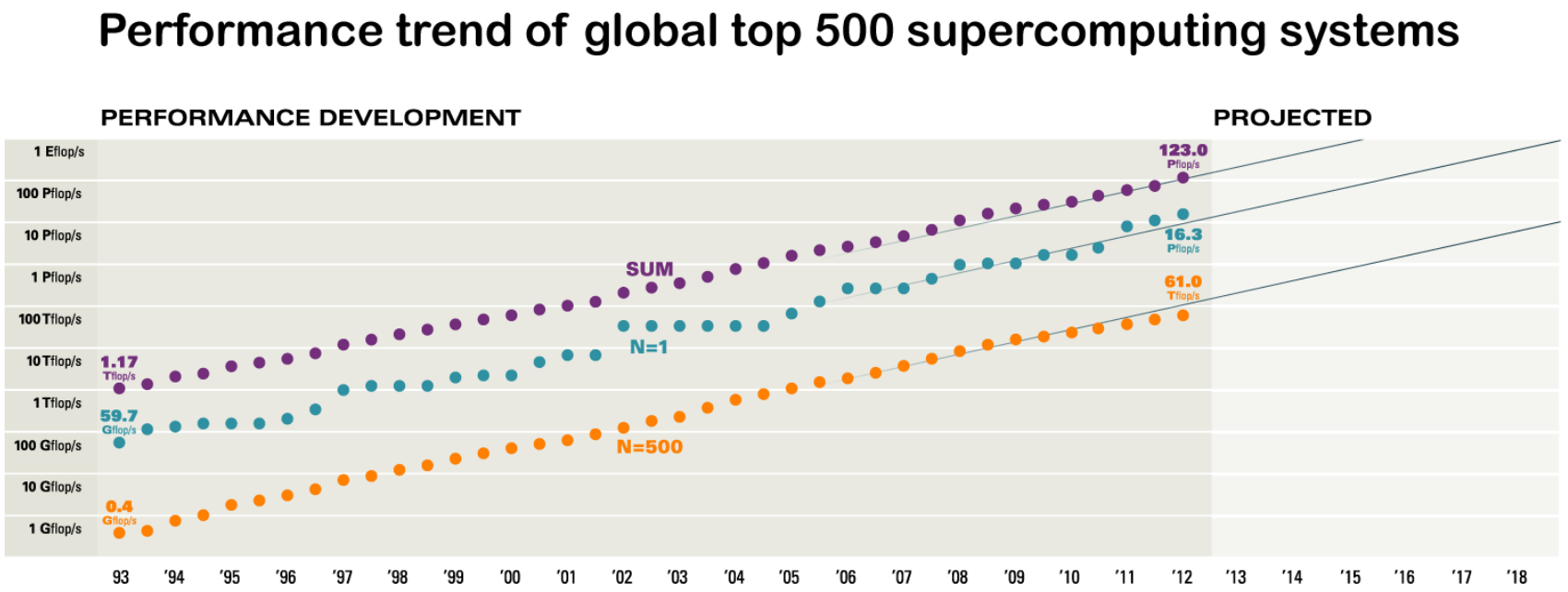
Overview

1. Why a new computing architecture?
2. Hardware accelerators
3. The Chimera project
4. Performance
5. Analysis via Berkeley's 'Thirteen Dwaves'
6. Issues and future work
7. Conclusion

Why do we need new computing architectures?

- We live in an increasingly data-driven age
- LHC experiment alone: data rate of 1 PB/s
- Many data-hungry projects coming on-line
- Computationally limited (most LIGO/Virgo CW searches have computation bounds)

Demand side…



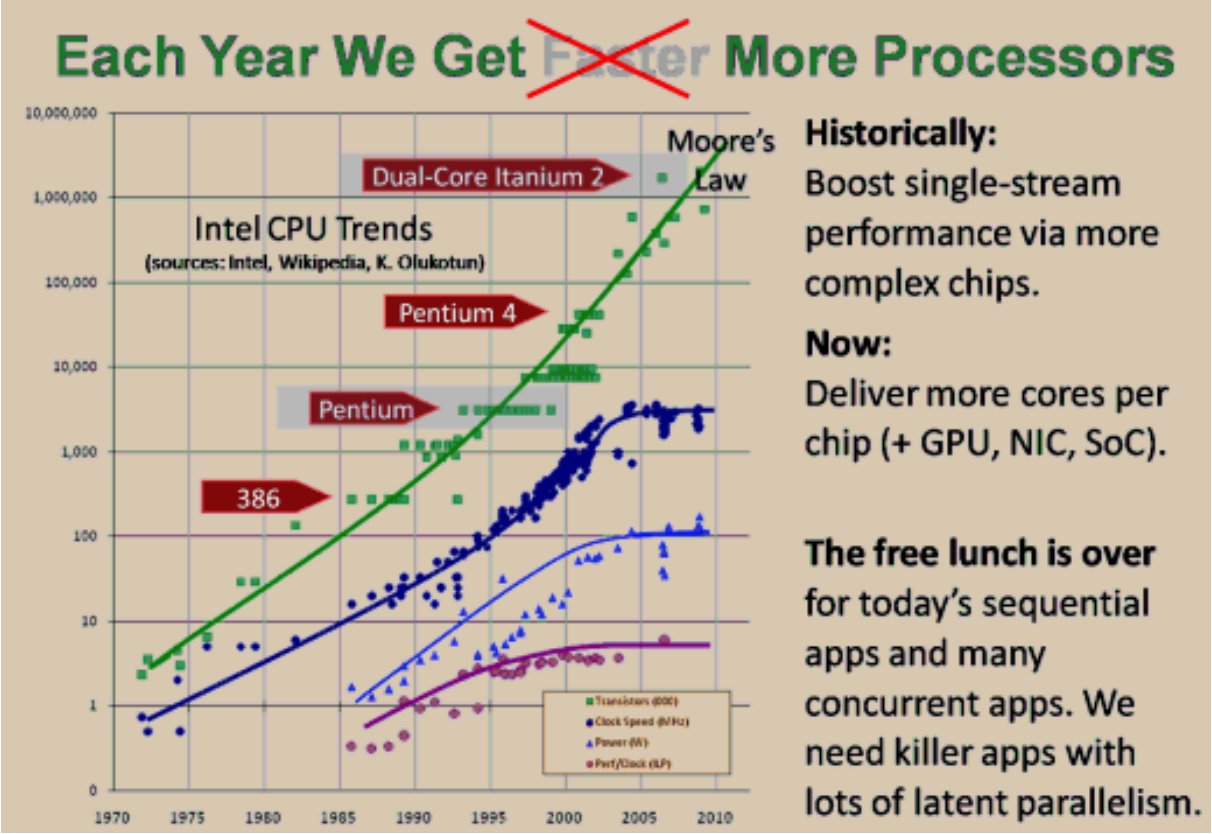
SKA

(adapted from www.top500.com , August 2012)

…supply side

- Intel discontinued 4GHz chip
- Speed walls
- Power walls
- Moore's Law

still holds, but for multicore

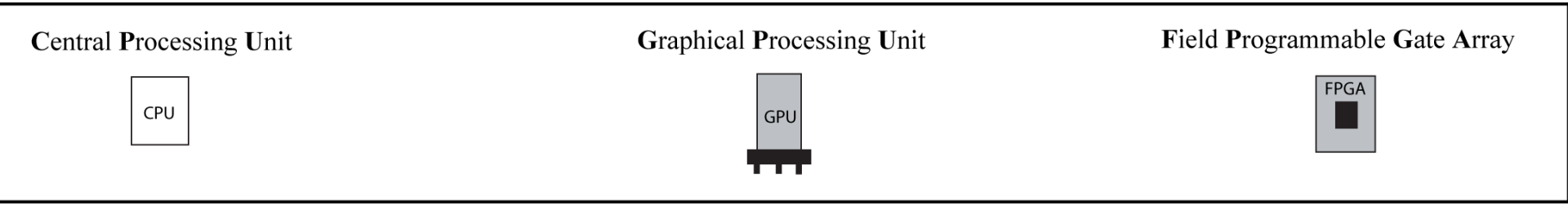


Time for a change in high performance computing?

- Focus on throughput, rather than raw flop/s (e.g. Condor on LSC clusters)
- Focus on power efficiency (top500 ‘Green list’)

Hardware acceleration

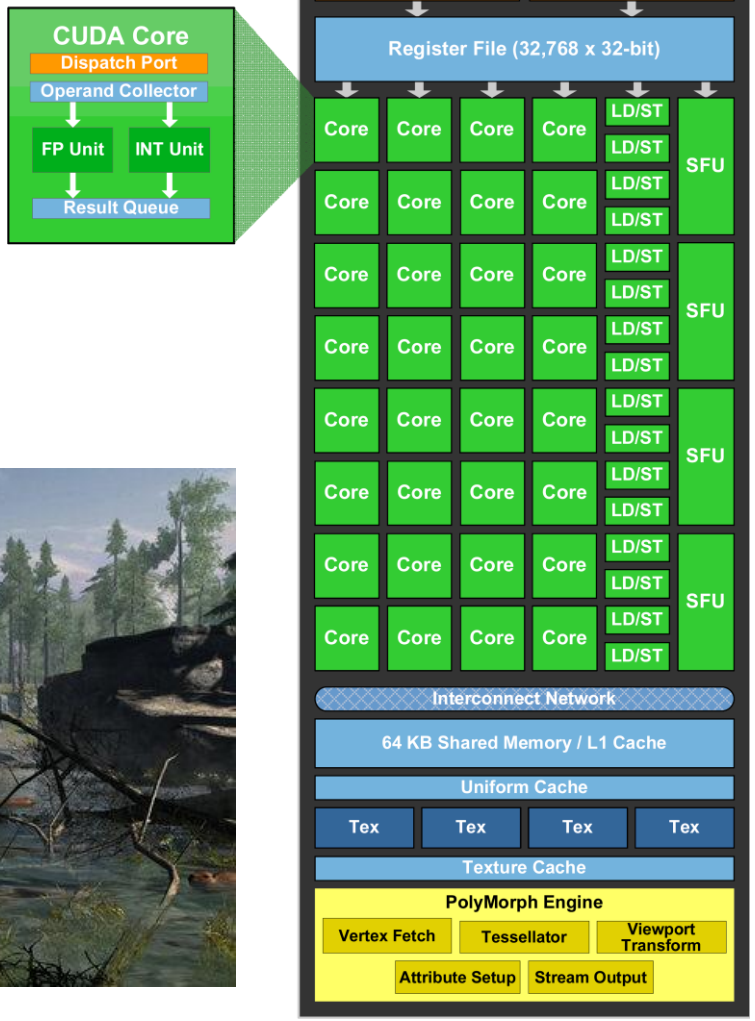
- Response to computational demand
- Two most common types: Graphical Processor Unit (GPU) and Field Programmable Gate Array (FPGA)



The Graphical Processor Unit

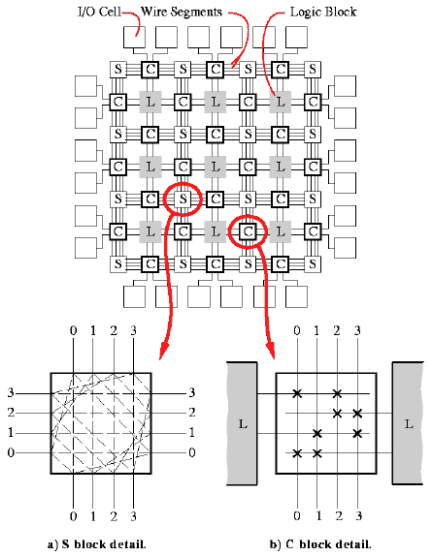
- Subsidised by gamers
- Most common accelerator
- Excellent support

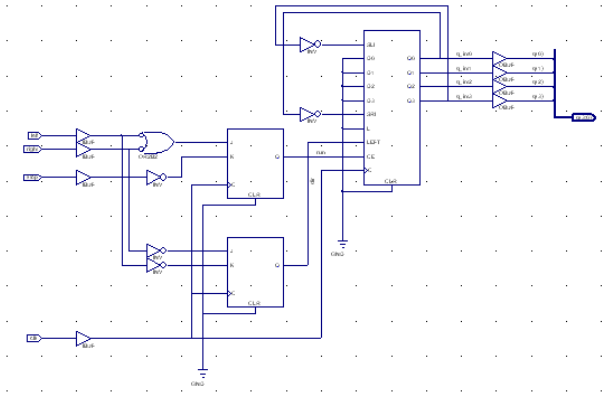
(nVidia)



The Field Programmable Gate Array

Programmable hardware (‘gateware’) replacement for application-specific integrated circuits (ASICs)





Cultural differences

- GPUs: software developers
- FPGAs: hardware, electronic engineers

FPGA development is a lot less like traditional microprocessor programming! (VHDL, Verilog)

Platform	Pros	Cons
CPU	Analysis ‘workhorse,’ multi-tasking	Power hungry, limited processing cores
GPU	Highly parallel, fairly simple interface (e.g. C for CUDA)	Highly rigid instruction set (don’t handle complex pipelines)
FPGA	Unrivalled flexibility and pipelining	Expensive outlay, specialised programming interface, prohibitive development time

Choosing the right hardware

Highly algorithm dependent, e.g.

- Dense linear algebra: GPU
- Logic-intensive operations: FPGA
- Pipelining: FPGA

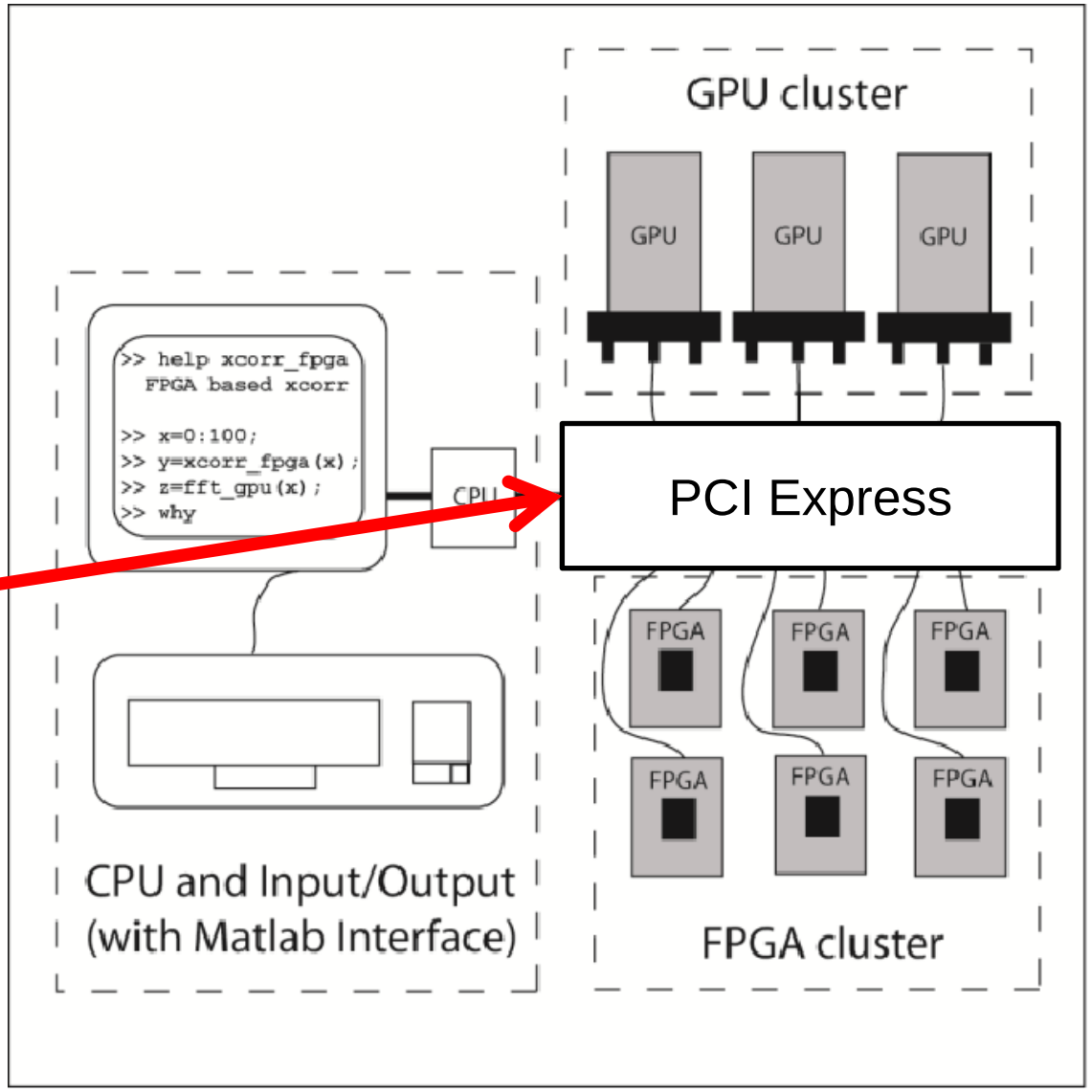
The ‘Chimera’ project

- Appropriately(?) combine benefits of three hardware classes
- Commercial, off the shelf (COTS) components
- Focus on high throughput, energy efficiency

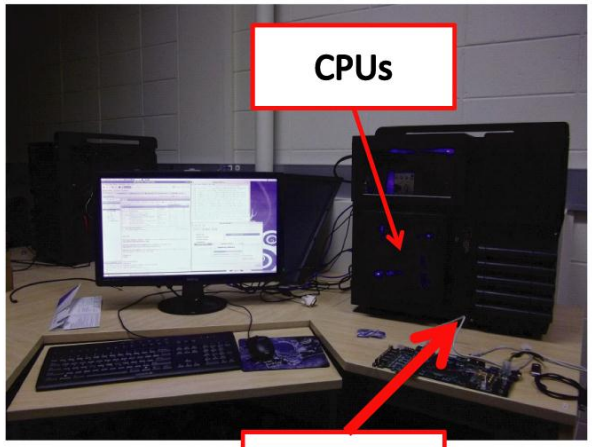
(Image: E. Koehn)

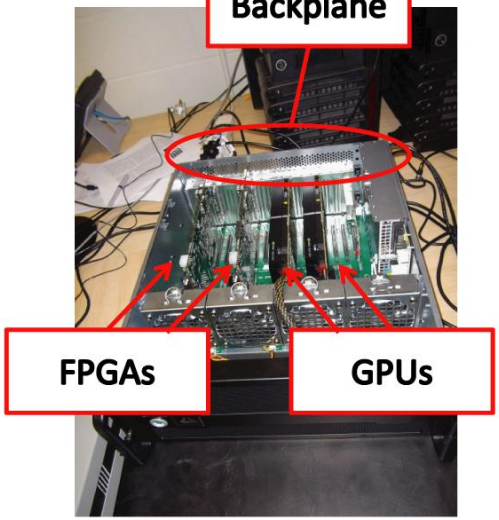
Initial design

COTS constraints



Subsystem	Vendor	Model
CPU	Intel	i7 Hexacore
GPGPU	nVidia	Tesla C2075
FPGA	Altera	Stratix-IV

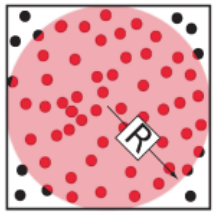




Photograph of an implementation of the system. Because of the three hardware classes, we named the system the 'Chimera,' after the mythical Greek beast with three different heads.

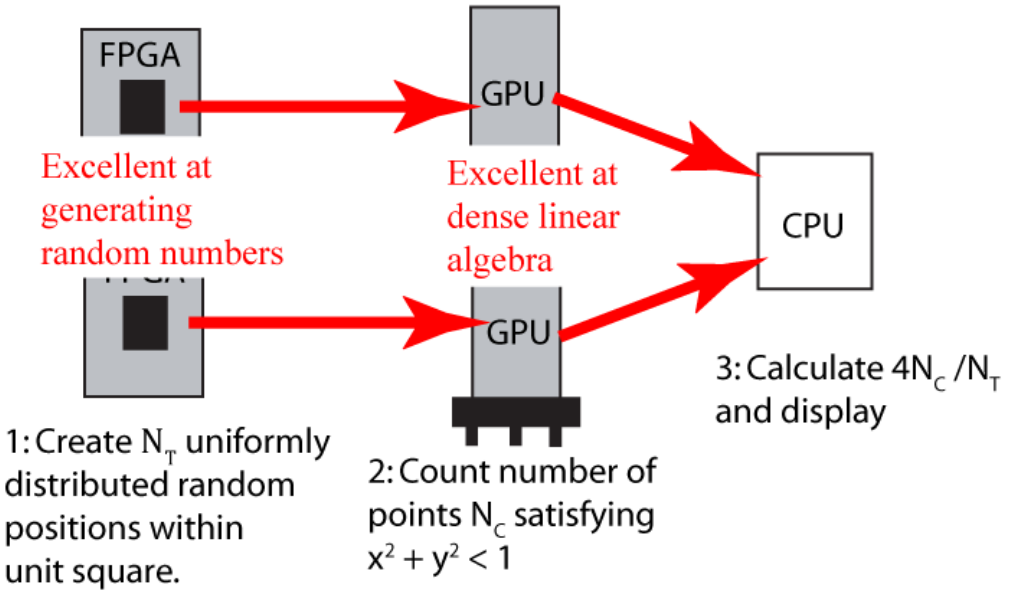
Performance

Monte Carlo calculation of π



$A_T = 4R^2$

$A_C / A_T = N_C / N_T = \pi/4$



(32-bit integer (x, y) pairs)

Throughput:

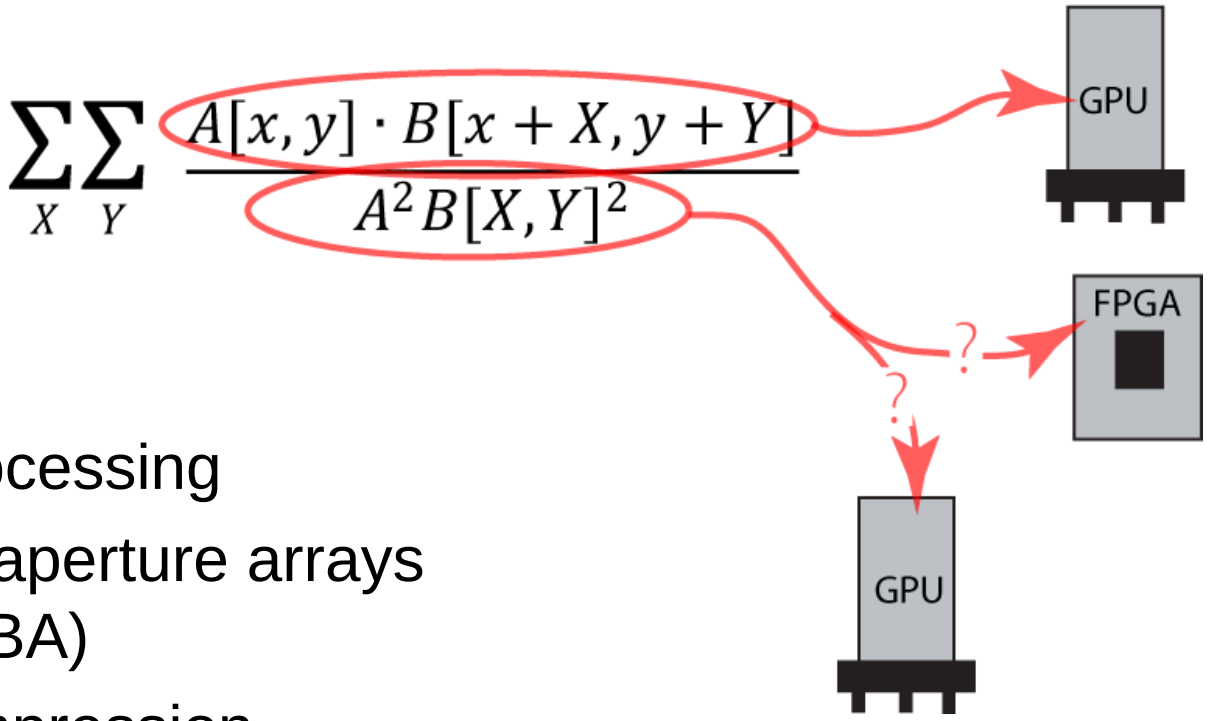
- FPGA alone: 24 $\times 10^9$ samp/s
- GPU alone: 2.1$\times 10^9$ samp/s
- CPU alone: 10 $\times 10^6$ samp/s

Surprising!

- FPGA pipeline was highly optimised
- Expect GPU to perform better with more complicated (floating point) integration problems
- Expect CPU to have even worse performance

Normalised cross-correlation

$$\sum_X \sum_Y \frac{A[x,y] \cdot B[x+X, y+Y]}{A^2 B[X,Y]^2}$$



- image processing
- synthetic aperture arrays (SKA, VLBA)
- video compression

Searched 1024×768 pixel image (16-bit greyscale) for a 8×8 template

- Numerator

GPU, 158 frame/s

- Denominator

GPU: 894 frame/s

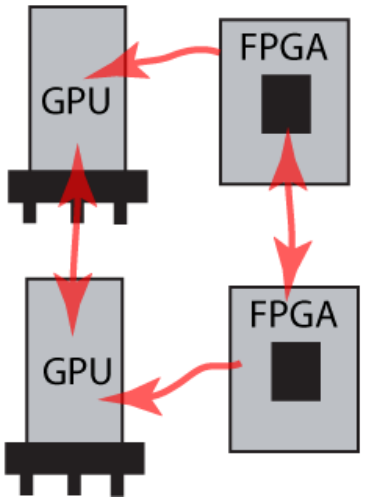
FPGA: 12,500 frame/s

So much for a *Graphical* Processor Unit!

Best combination: GPU + FPGA

Other algorithms

- Data analysis
- Interface GPUs with experimental FPGA DAQ systems (GMT, quantum cryptography)



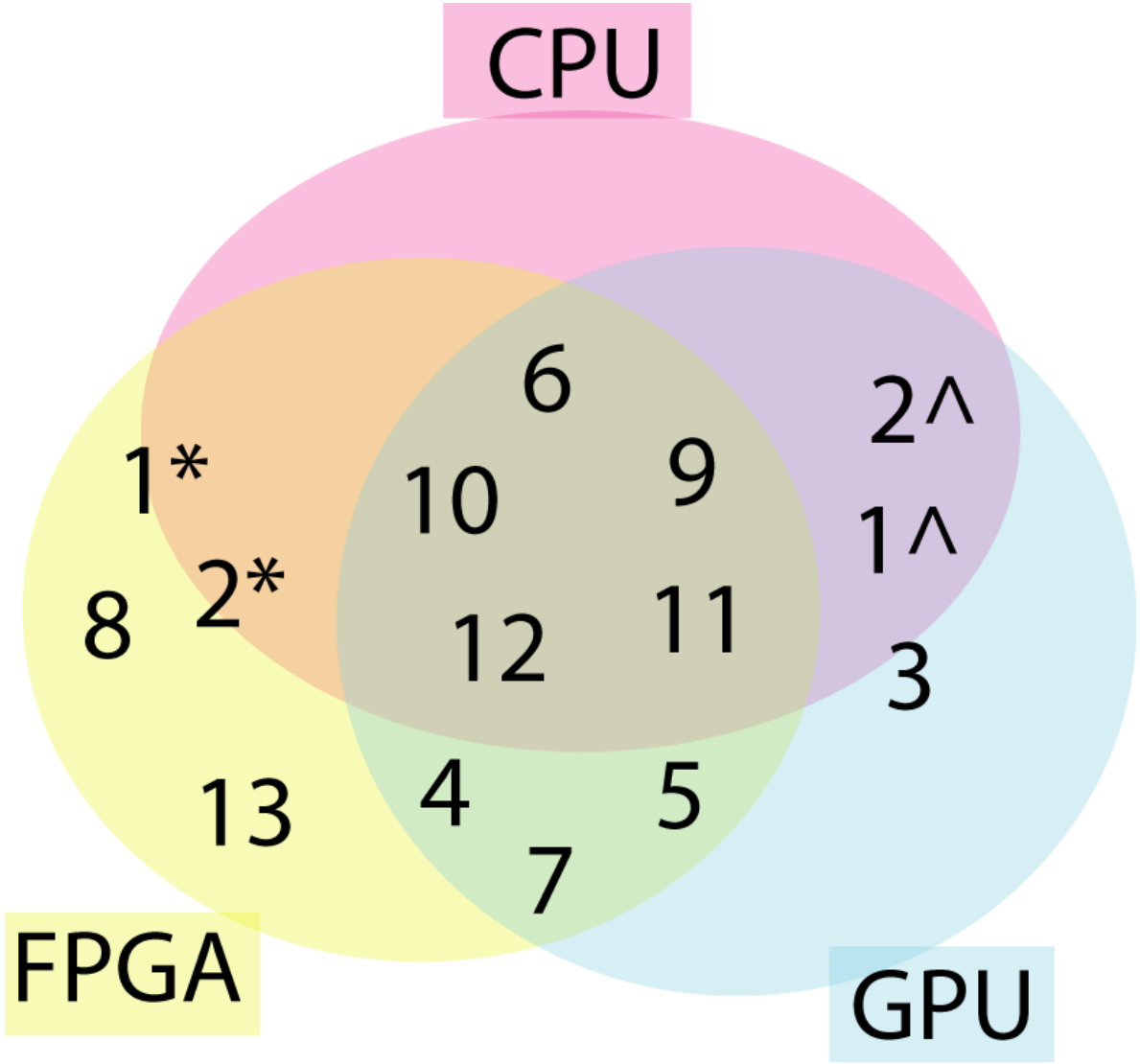
R. Inta, D. J. Bowman, and S. M. Scott, “The ‘Chimera’: An Off-The-Shelf CPU/GPGPU/FPGA Hybrid Computing Platform,” *Int. J. Reconfigurable Comp.*, 2012(241439), 10 pp. (2012)

The Thirteen Dwarves of Berkeley

- Phillip Colella identified 7 classes of parallel algorithms (2004)
- A dwarf: “an algorithmic method encapsulating a pattern of computation and/or communication”
- UC Berkeley CS Dept. increased to 13

K. Asanovic and U C Berkeley Computer Science Department, “The landscape of parallel computing research: a view from Berkeley,” *Tech. Rep. UCB/EECS-2006-183*, UC Berkeley (2005)

	Dwarf	Examples/Applications
1	Dense Matrix	Linear algebra (dense matrices)
2	Sparse Matrix	Linear algebra (sparse matrices)
3	Spectral	FFT-based methods
4	N-Body	Particle-particle interactions
5	Structured Grid	Fluid dynamics, meteorology
6	Unstructured Grid	Adaptive mesh FEM
7	MapReduce	Monte Carlo integration
8	Combinational Logic	Logic gates (e.g. Toffoli gates)
9	Graph traversal	Searching, selection
10	Dynamic Programming	‘Tower of Hanoi’ problem
11	Backtrack/Branch-and-Bound	Global optimization
12	Graphical Models	Probabilistic networks
13	Finite State Machine	TTL counter

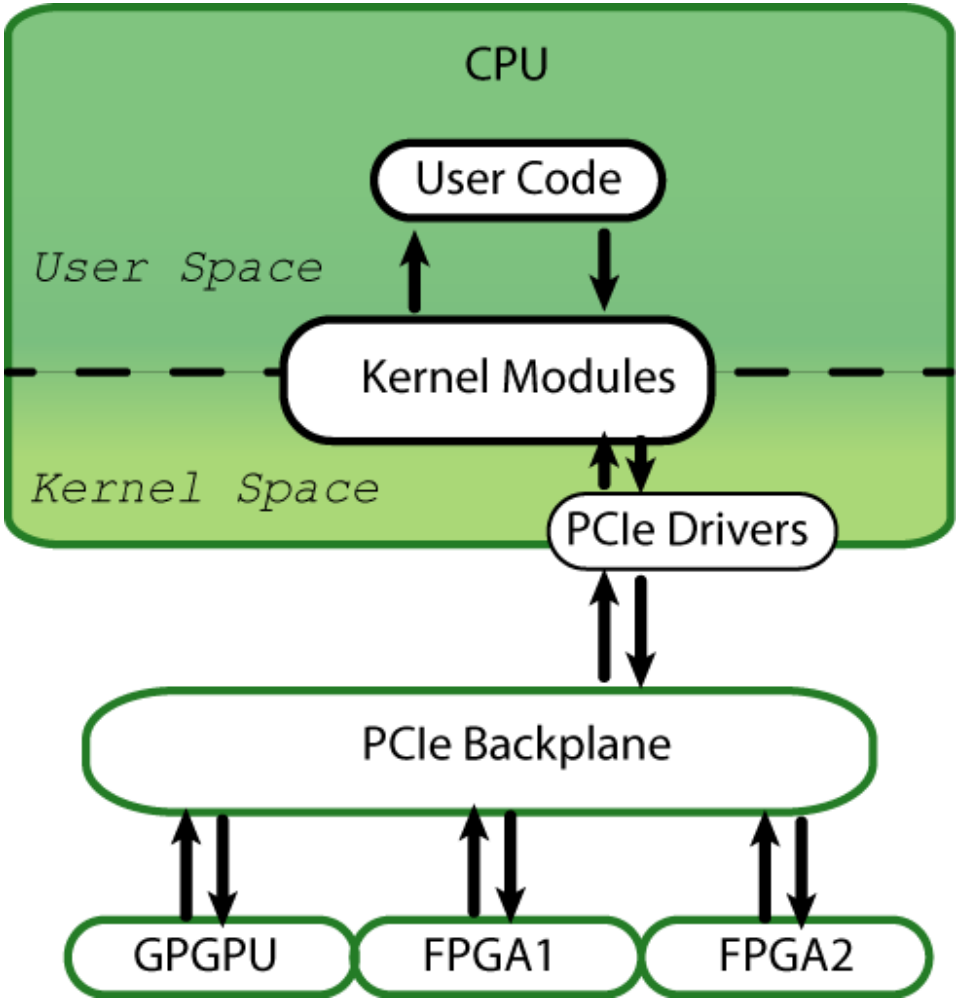


*Fixed point
^Floating point

Issues/future work

- Kernel module development
- Scalability is problem-size dependent
- FPGA development is labour-intensive
- Pipeline development
- GW data analysis!

Issues/future work



Issues/future work

- Kernel module development
- Scalability is problem-size dependent
- FPGA development is labour-intensive
- Pipeline development
- GW data analysis!

Conclusion

- We are in a transition period for HPC architecture
- Can exploit highly heterogeneous, COTS, hardware accelerators for certain algorithms
- Powerful and energy efficient
- Requires further development

More information:
www.anu.edu.au/physics/cgp/Research/chimera.html

Thanks to NIMS and Seoul National University!

- Dr Sang Hoon Oh
- Dr John J. Oh

Thanks for listening!

Ra.Inta@anu.edu.au