

Event-driven computing

Andrew Brown
Southampton
adb@ecs.soton.ac.uk

David Thomas
Imperial College
d.thomas1@imperial.ac.uk

Simon Moore
Cambridge
simon.moore@cl.cam.ac.uk

Andrey Mokhov
Newcastle
andrey.mokhov@newcastle.ac.uk

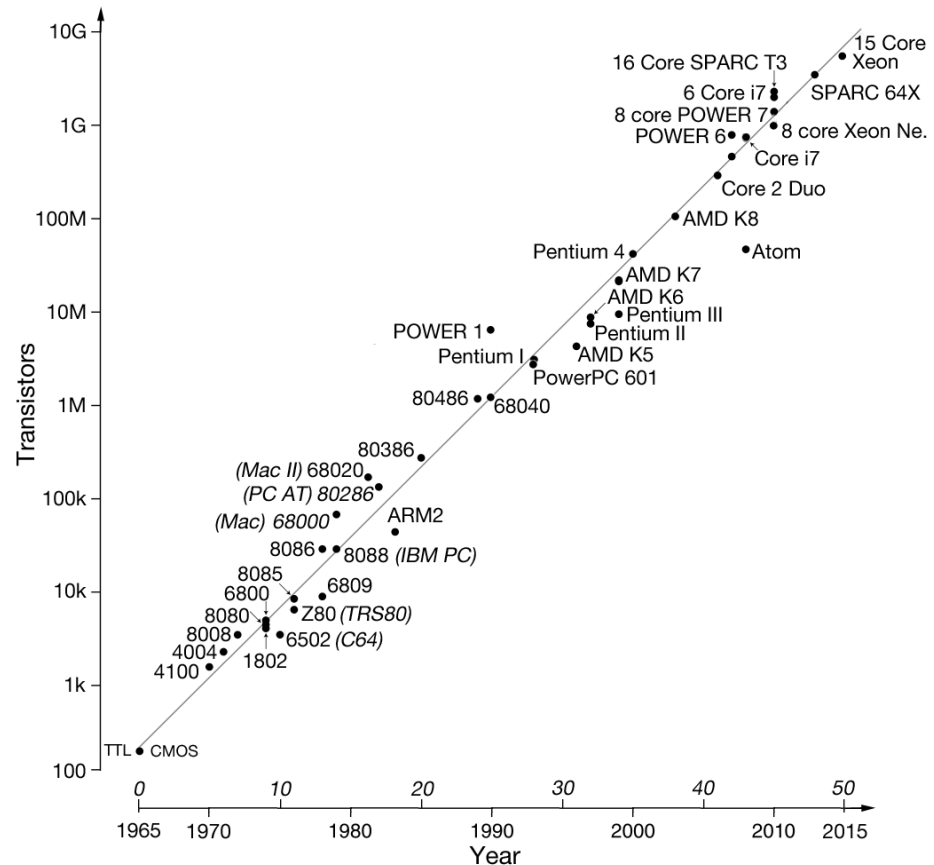


- POETS - the concept
- The realisation
- A panorama of applications
- Down amongst the hard sums
- A brief meander into reliability

Cores are the new transistor

–Silicon

- 10um .. 10 nm
- 2^{10} reduction in feature size
- 2^{20} increase in density
- ~20 process generations



The rise of the many-core exponent is just beginning

...has brought about massive changes:


- Hardware increasingly statically and dynamically unreliable
- Cores are free
- Communication costs $\sim 1000 \times$ compute costs

Simulation: The old way

- Build a model:
 - Probably some sort of graph
 - Local interactions
- Process the model:
 - Linearise and shove it through some (small) set of processors
- Big problems:
 - A lot of fetching
 - A lot of shoving
 - A lot of writing back

- Build a model:
 - Probably some sort of graph
 - Local interactions
- Process the model:
 - Distribute processors over the model
- Big problems:
 - No fetching
 - Wherever there's data, there's a processor
 - Little shoving
 - Things interact with their neighbours in parallel
 - No writing back
 - Data is stored locally to the generation site

*Interaction via
small, fast
messages*

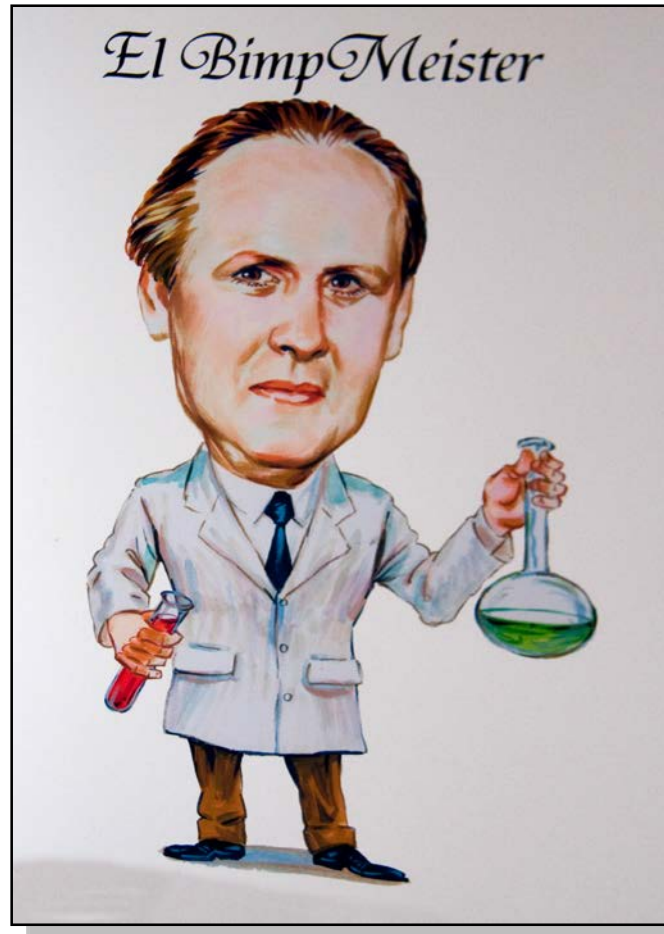


...needs to change in response to this plethora of cores:

- Software must cope with
 - Unknown and changing physical core topologies
 - Non-deterministic message passing
 - Localised data
- Dramatic performance improvements
 - For certain classes of problems
- Underlying *mathematics* still valid
 - Algorithmic embodiment different

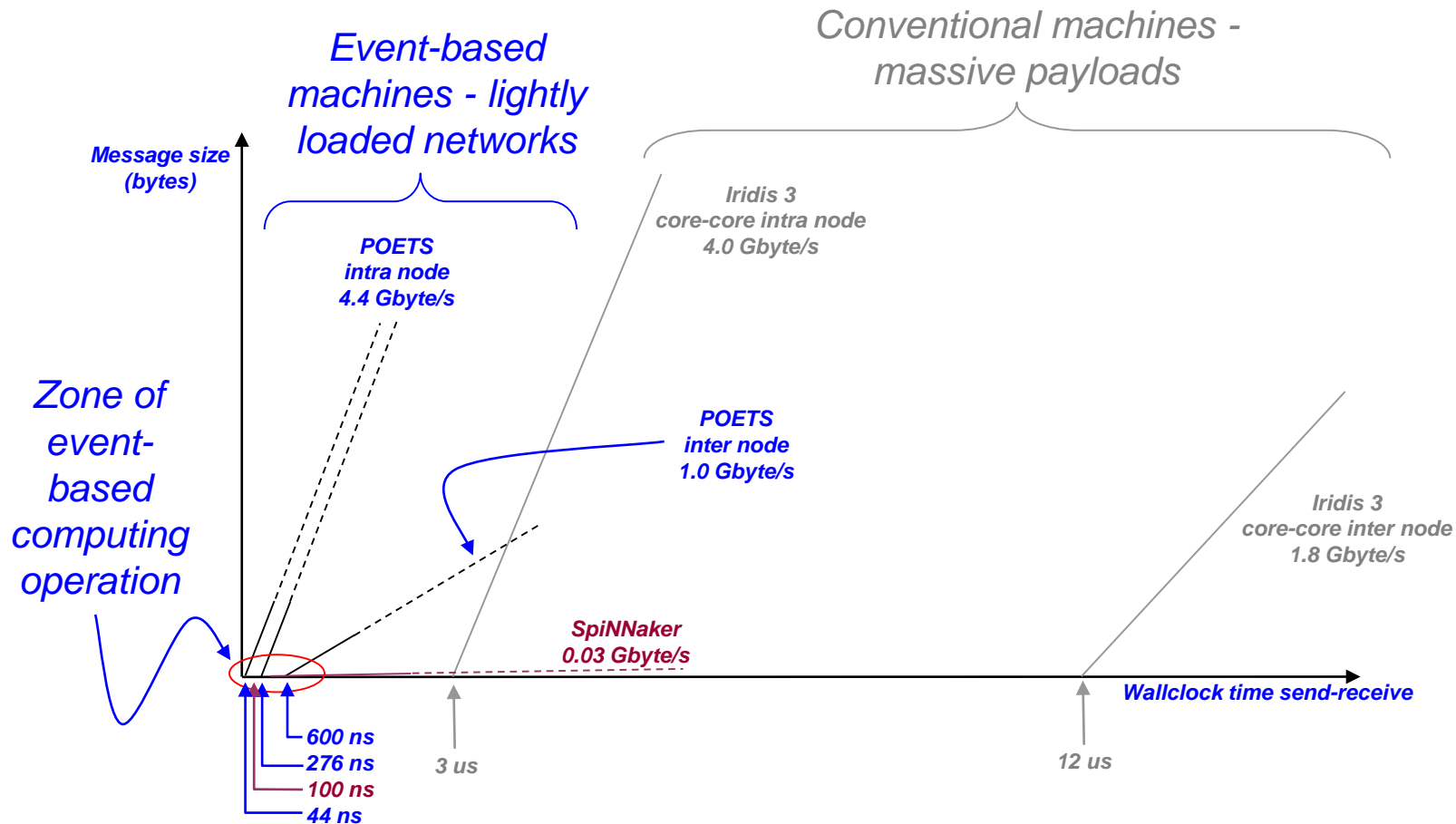
Inspired by SpiNNaker

Partially
Ordered
Event
Triggered
Systems



*We learnt
so much
from Steve*

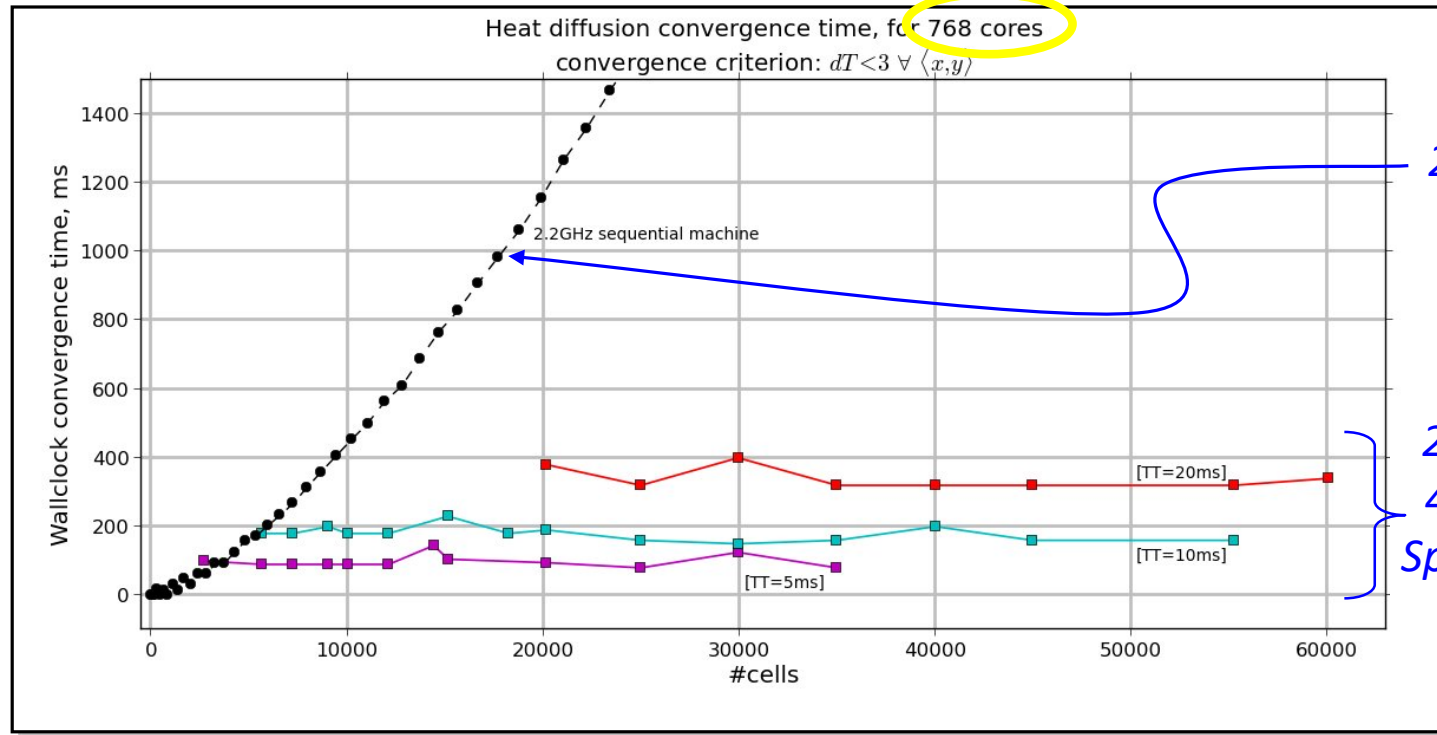
It's all about speed



The reward...

- Dramatic performance improvements
 - For certain classes of problems

$$O(n^?) \rightarrow O(1)$$



2.2GHz single
core desktop

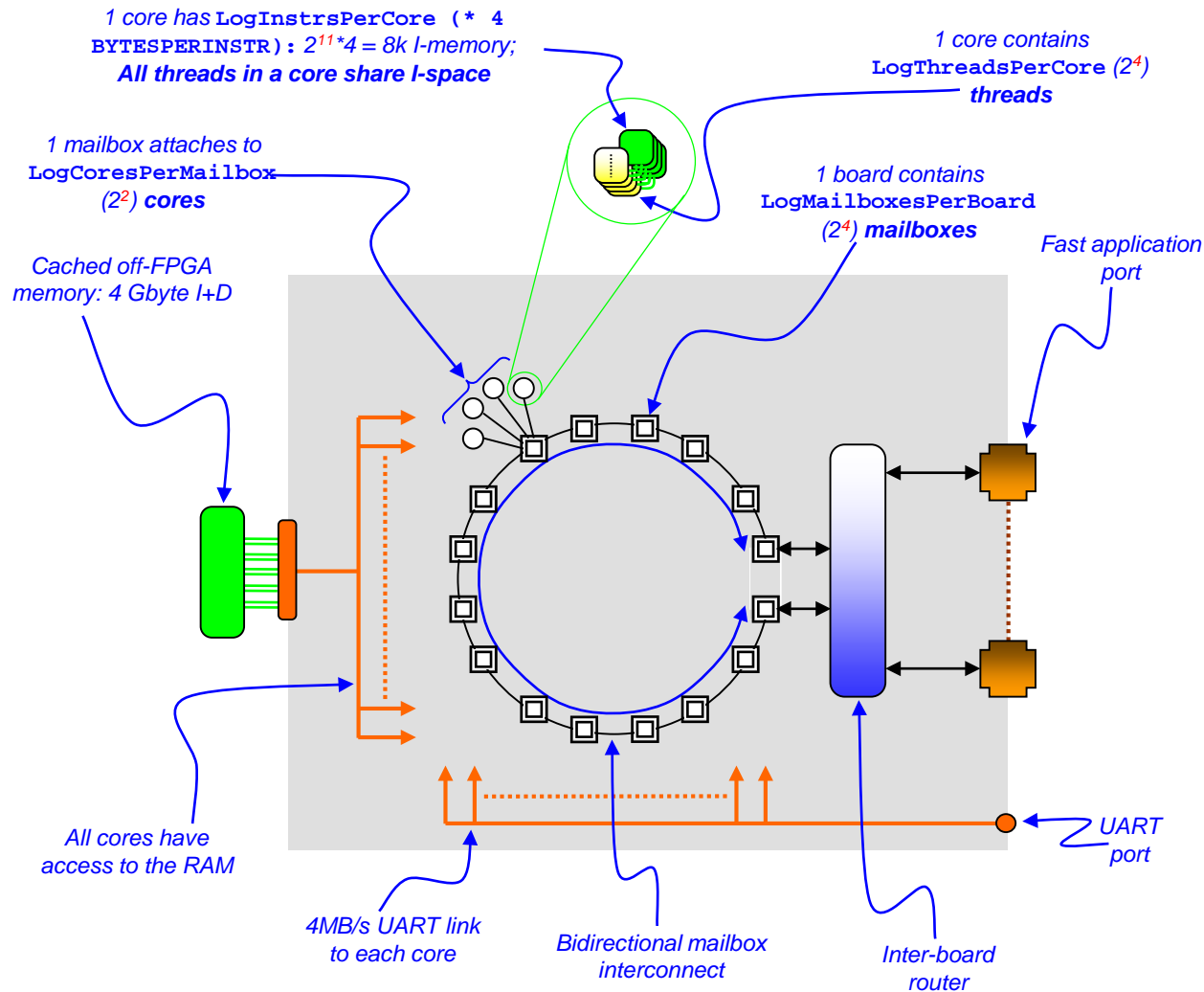
200MHz
48 node
SpiNNaker
engine

Imagine a system...

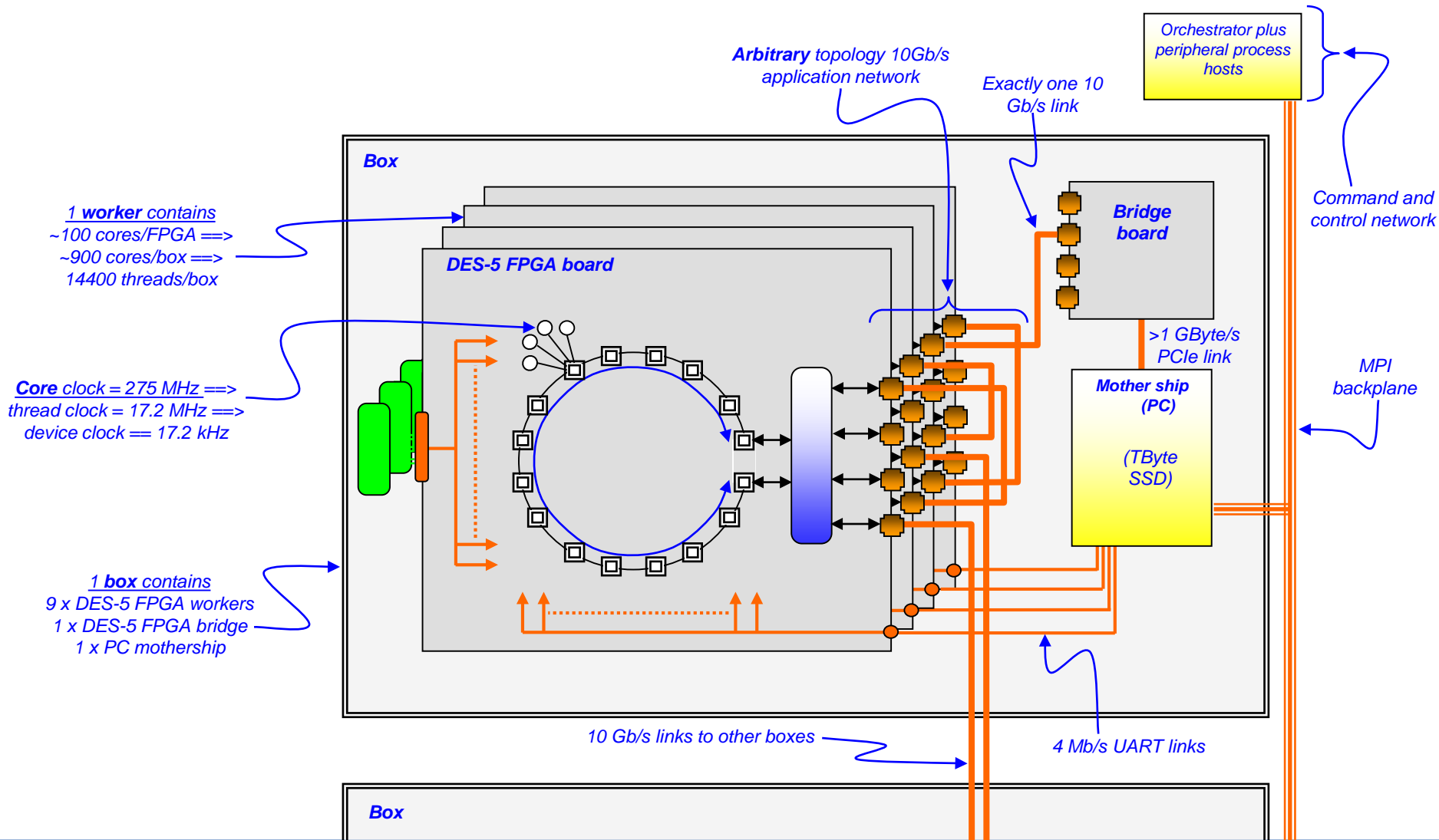
- Arbitrarily large number of cores
 - Cores are *free*
- Core-core communications *cheap*
 - We designed it that way
- *What* might we do with it?
- *How* might we do *anything* with it?

- POETS - the concept
- The realisation
- A panorama of applications
- Down amongst the hard sums
- A brief meander into reliability

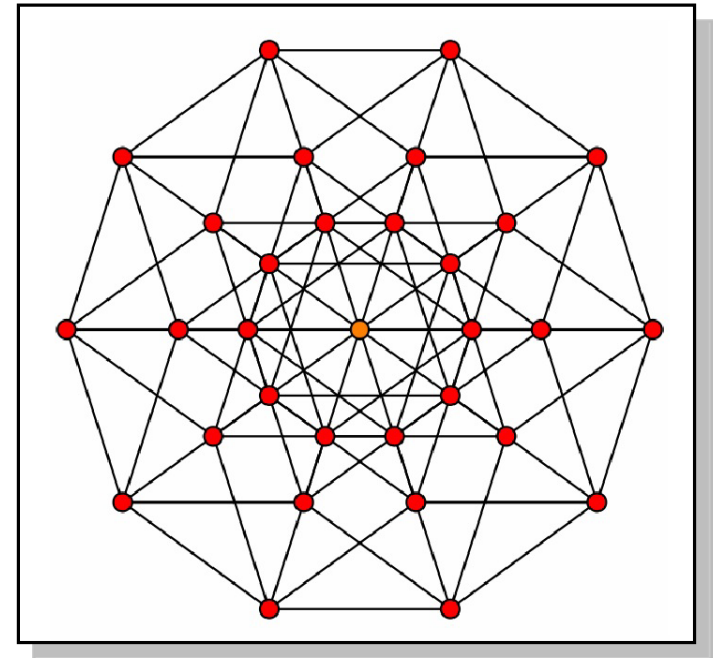
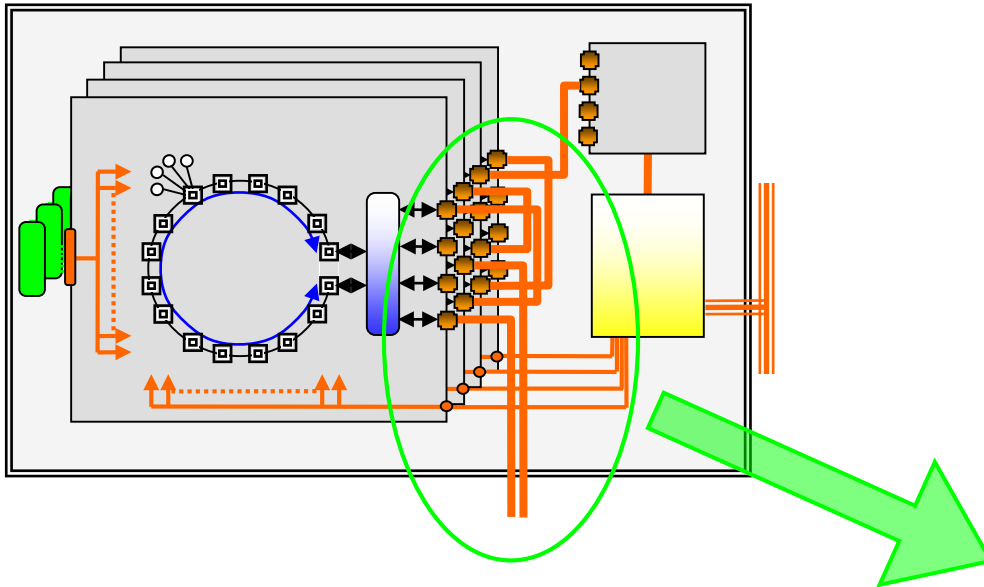
A POETS board



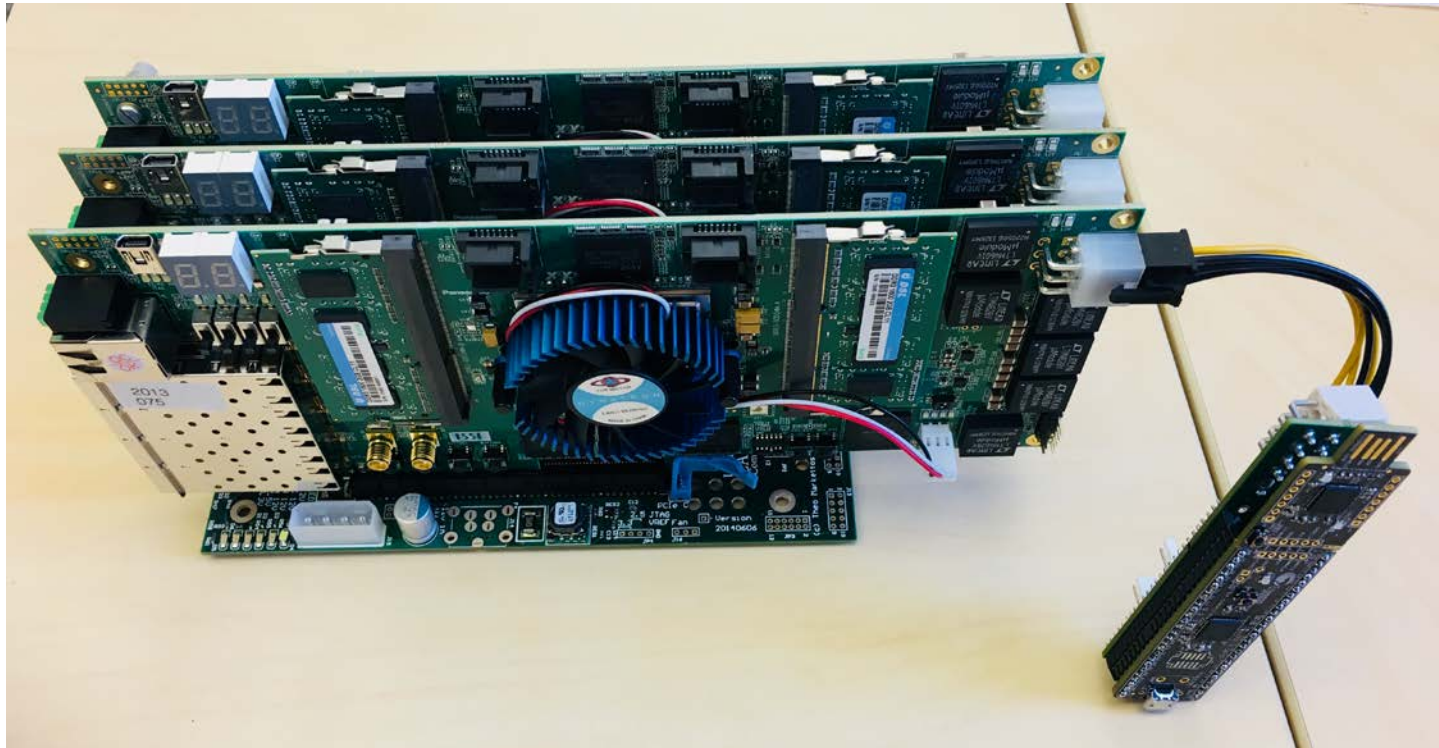
A box of boards



The penteract

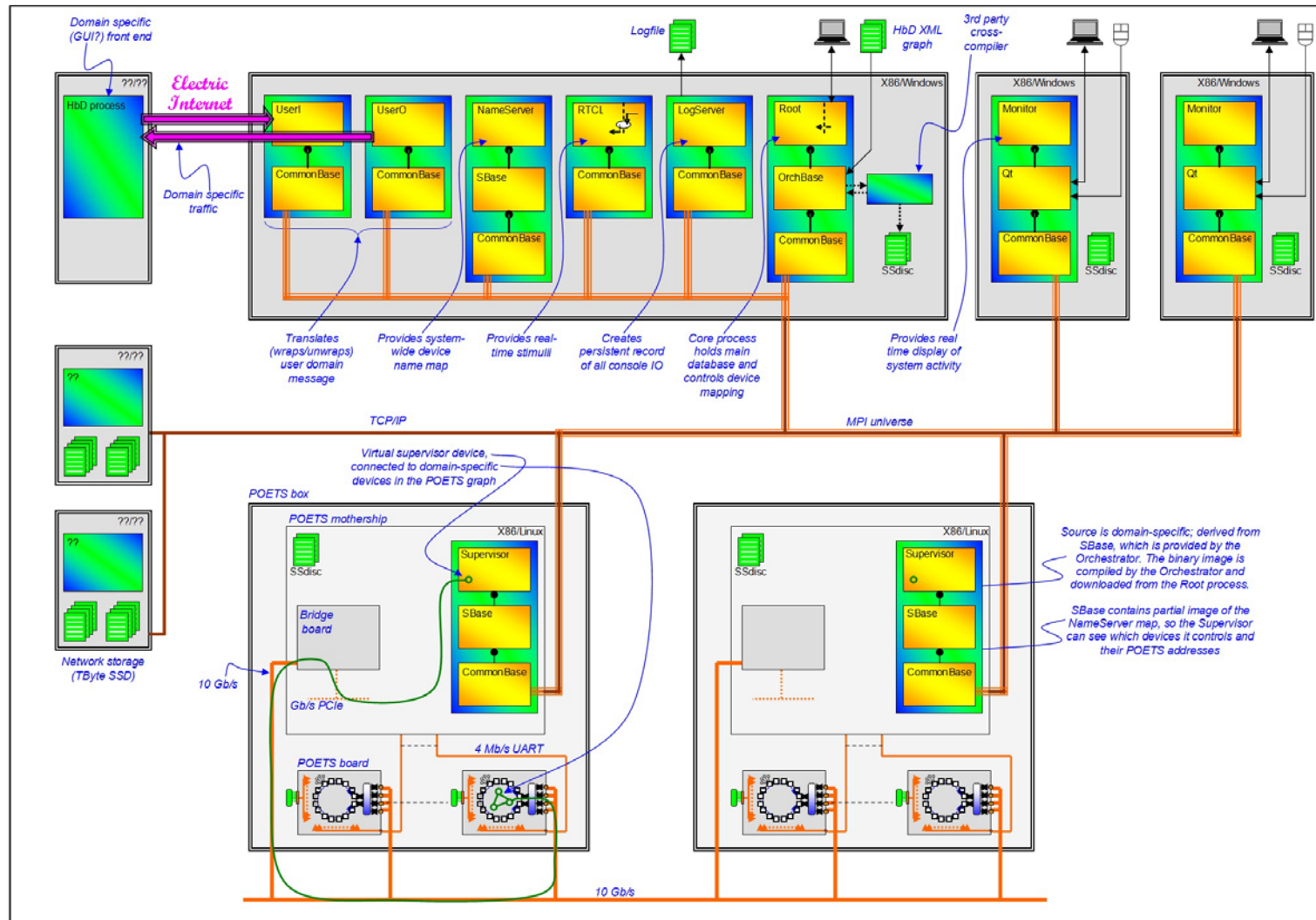


The reality



- Tinsel = multithreaded POETS processor
- 32-bit multithreaded RISC-V
- 64 cores x 16 threads/Tinsel = 1024 P-cores
- 16 caches, 16 mailboxes, 2 DDR3 controllers
- 40% of DE5-NET FPGA board
- Clocks at over ~300MHz (fast for a softcore)
- Passes RISC-V regression test suite

Hardware is nothing without software...



- POETS - the concept
- The realisation
- A panorama of applications
- Down amongst the hard sums
- A brief meander into reliability

What sort of problem?

Upper atmosphere
weather modelling

POETS 

Tomographic imaging

POETS 

- Industrial
- Production
- Mitigation

Terrestrial weather modelling

POETS 

Image-guided surgery

POETS 

Medical tomographic imaging

POETS 

Drug screening

POETS 

- Protein-protein interaction within a cell
 - Expressed as a graph (G_c)
- Drug effects
 - Expressed as a graph (G_d)



Continuous simulation

POETS 

Computational chemistry

POETS 

Neural synthesis:

POETS 

Particle-particle

POETS 

Tomographic imaging

POETS 

- Inverse field problems
 - Detection of submerged pseudo-cylindrical magnetohydrodynamic anomalies
- Equation set *secret*



Discrete simulation

POETS 

Genome reading

POETS 

- Part of DNA sequencing workflow when using certain next generation sequencing (NGS)

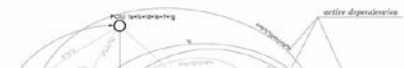
Gyrokinetic plasma

POETS 

And, of course.....

POETS 

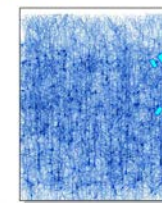
- Anything that can be expressed as a Petri Net



Neural simulation

POETS 

Flat data
– 10^9 neurons



SpNNektor



POETS



- Anything that can be transformed into
 - Large number of simple processes
 - Asynchronous short-range communication

- POETS - the concept
- The realisation
- A panorama of applications
- Down amongst the hard sums
- A brief meander into reliability

A real problem....

- Heat equation

$$\frac{\partial u}{\partial t} = D \frac{\partial^2 u}{\partial x^2}$$

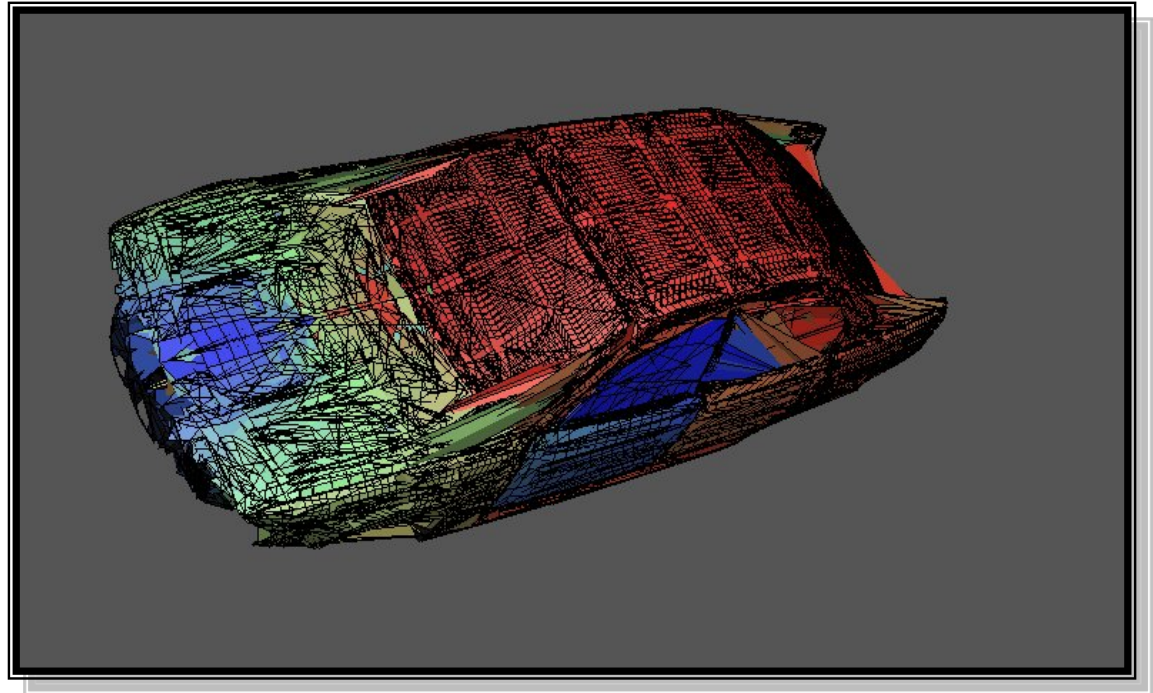
– becomes

nth timestep

$$u_i^{(n+1)} = u_i^{(n)} + D \frac{\partial t}{\partial x_i^2} (u_{i-1}^{(n)} - 2u_i^{(n)} + u_{i+1}^{(n)})$$

*dx_i are different because
the mesh is irregular*

simulated time $t = ndt$



Gauss-Seidl:

New value being calculated

New values used as soon as they are available

Previous value - no update yet available

$$\underline{x}^{(m+1)} = \underline{b} - \underline{L}\underline{x}^{(m+1)} - \underline{U}\underline{x}^{(m)}$$

Convergence fast: *newest* values used as soon as they become available

Predicate tree thin - not parallelisable

Jacobi:

New value being calculated

Only old values used

$$\underline{x}^{(m+1)} = \underline{b} + (\underline{I} - \underline{A})\underline{x}^{(m)}$$

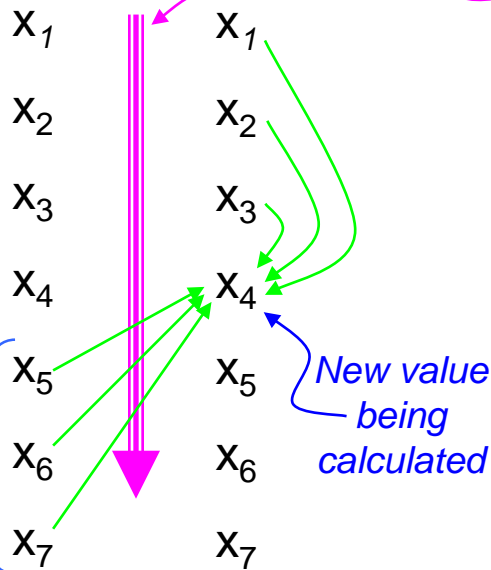
Convergence slow: *entire* old value set used to generate *entire* new value set

Predicate tree wide - easily parallelisable

*Solution trajectory (programmer-defined)
- sequential*

Gauss-Seidl:

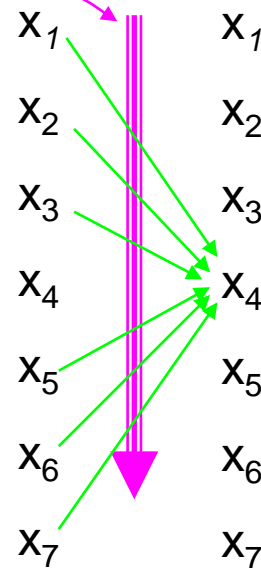
(m) (m+1)



*New values used as soon
as they are available*

Jacobi

(m) (m+1)

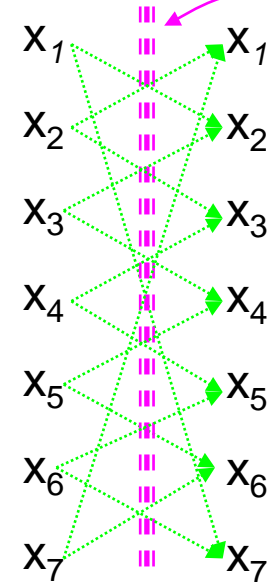


*Only old
values used*

*Solution trajectory (non-deterministic)
- massively parallel*

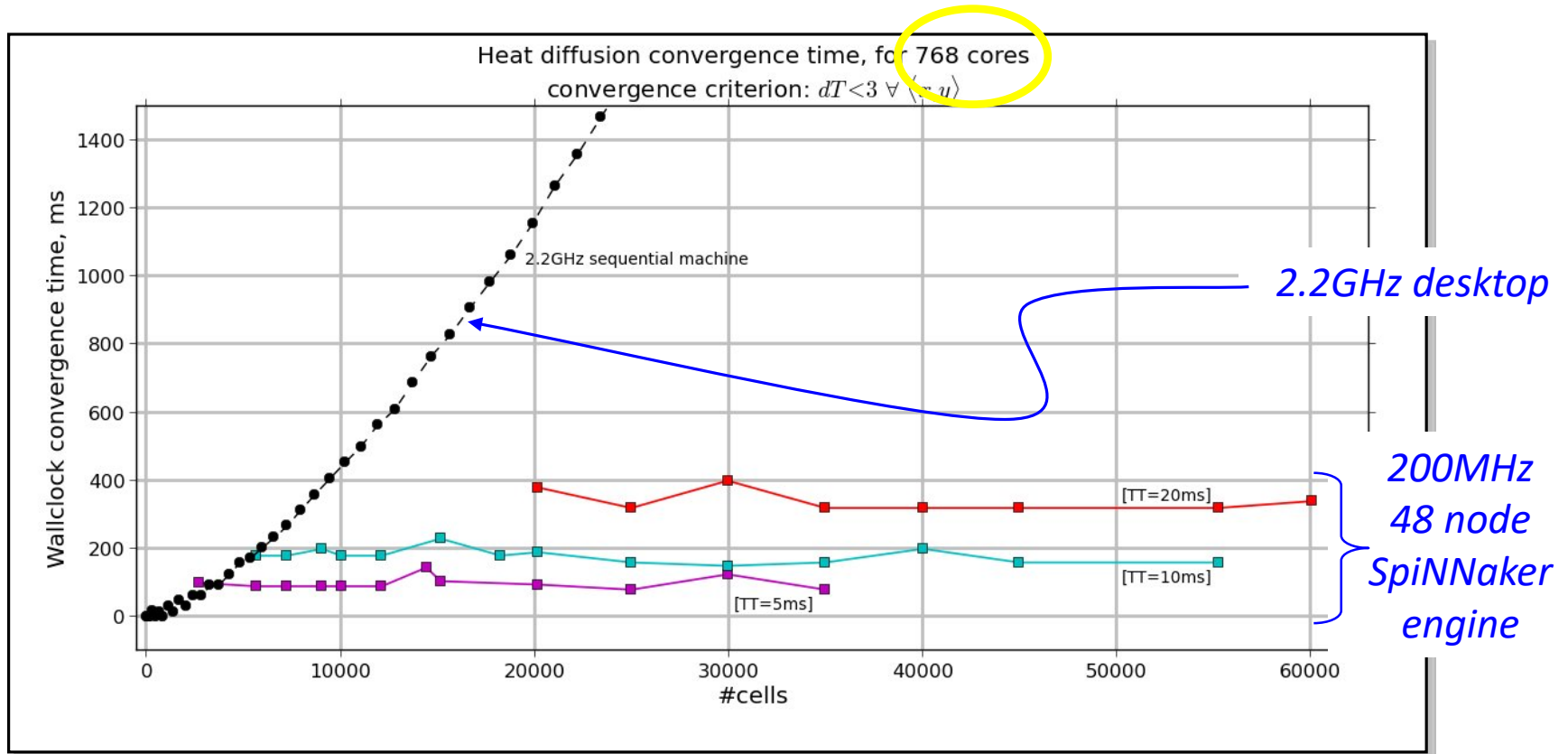
Event-driven

(m) (m+1)

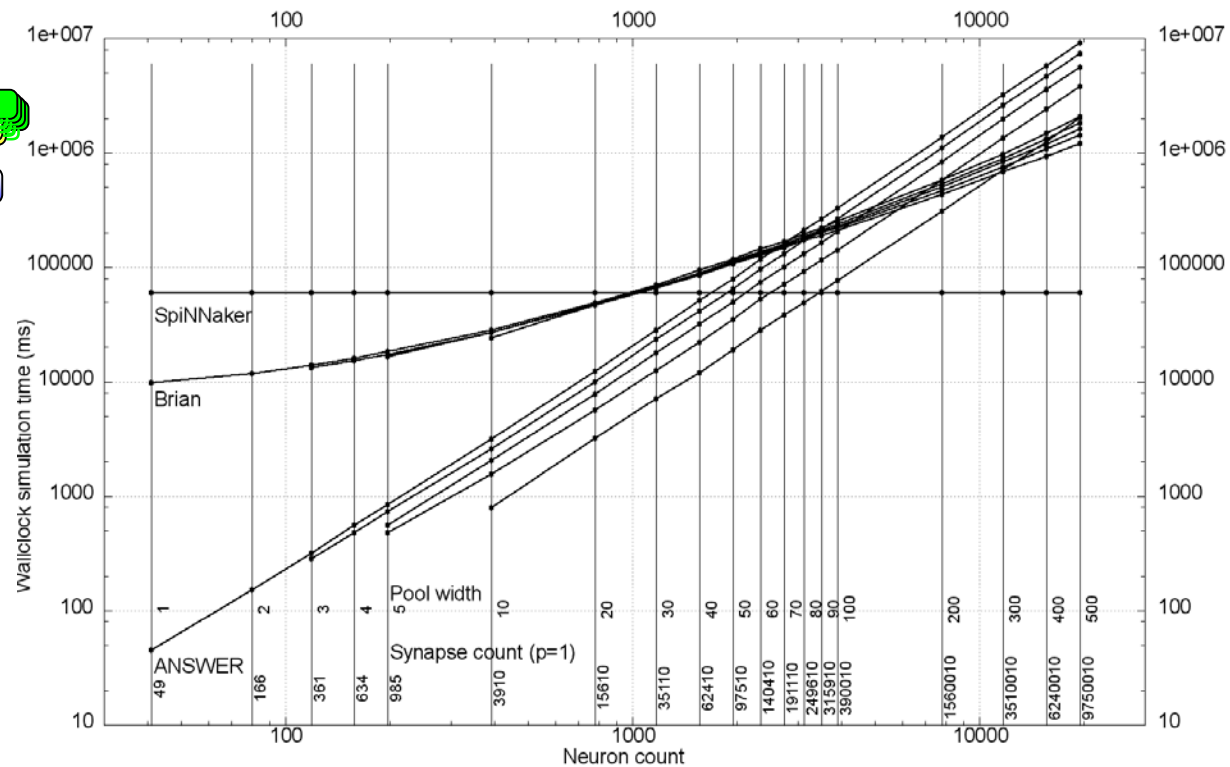
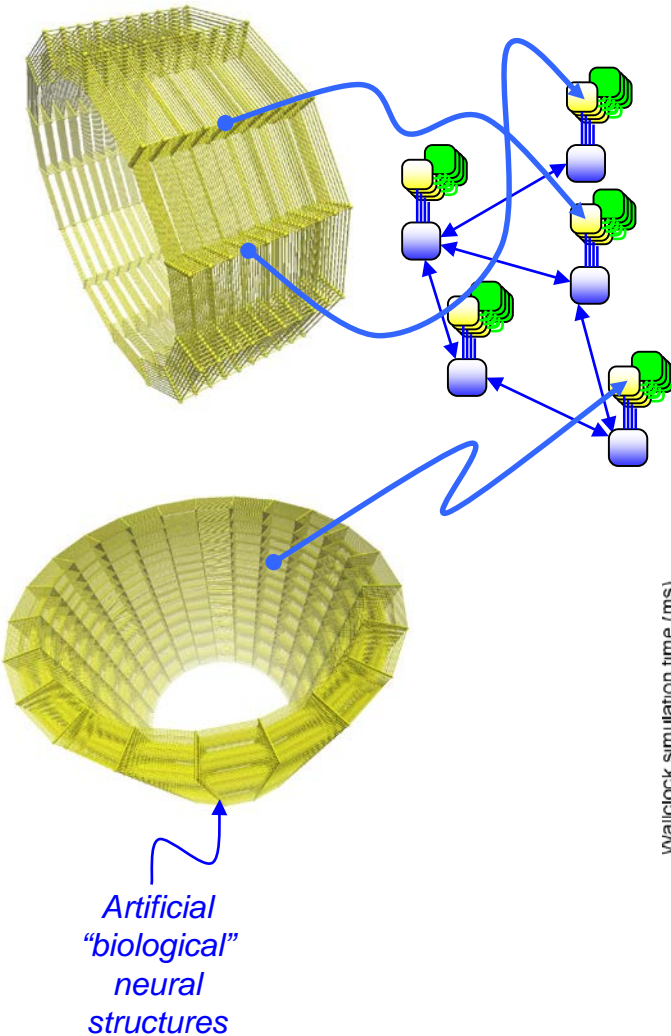


*Values updated randomly
IN PARALLEL*

Solution times

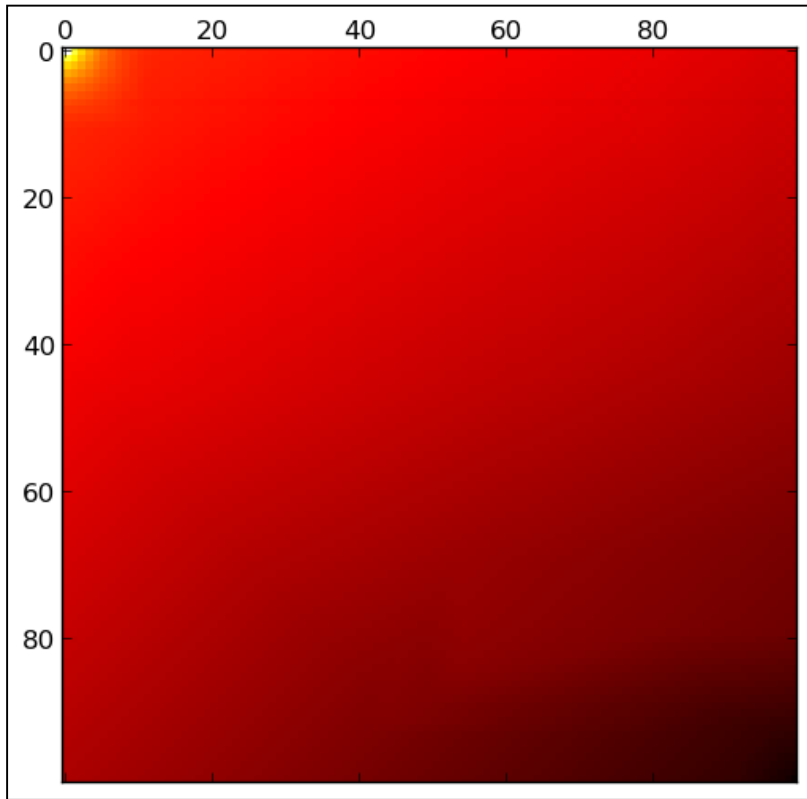


Synfire rings

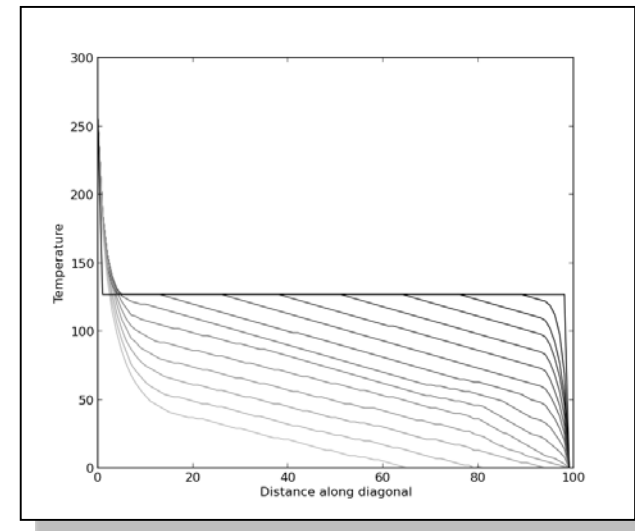


- POETS - the concept
- The realisation
- A panorama of applications
- Down amongst the hard sums
- A brief meander into reliability

Reliable computing on unreliable substrates



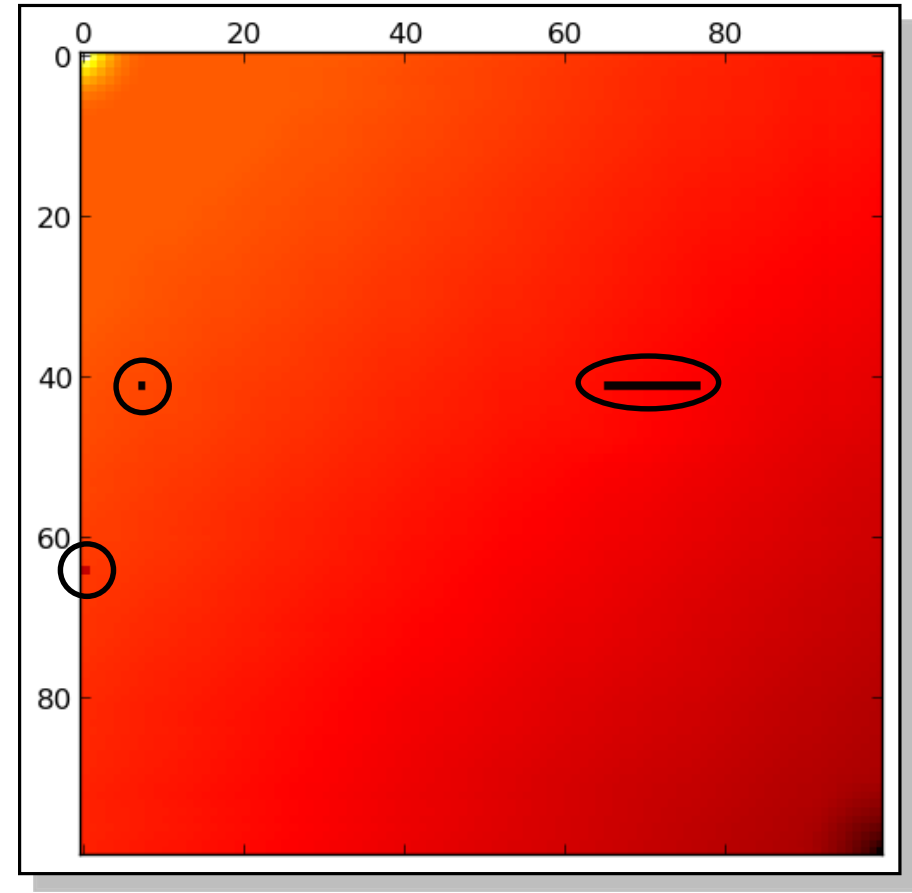
Finite difference heat diffusion
canonical 2D square grid:



Diagonal temperature profile vs iteration

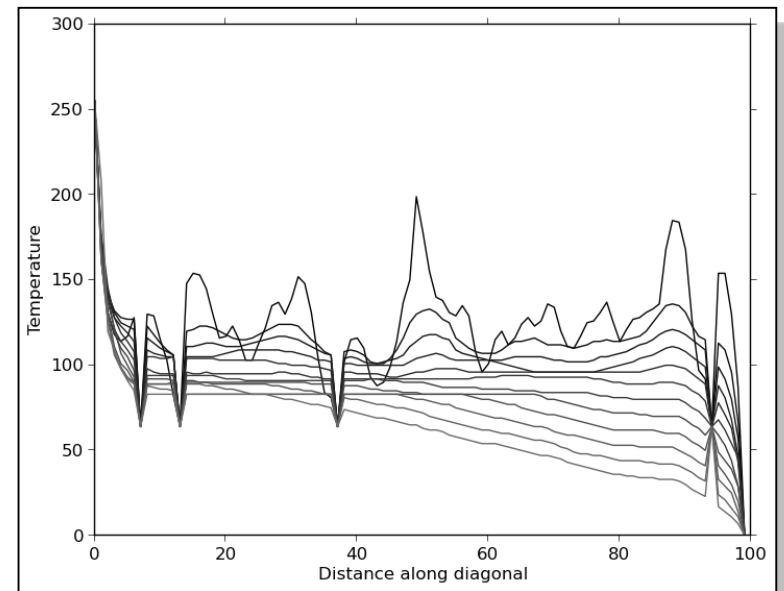
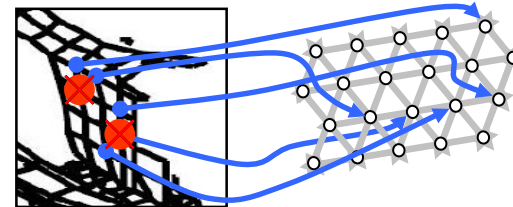
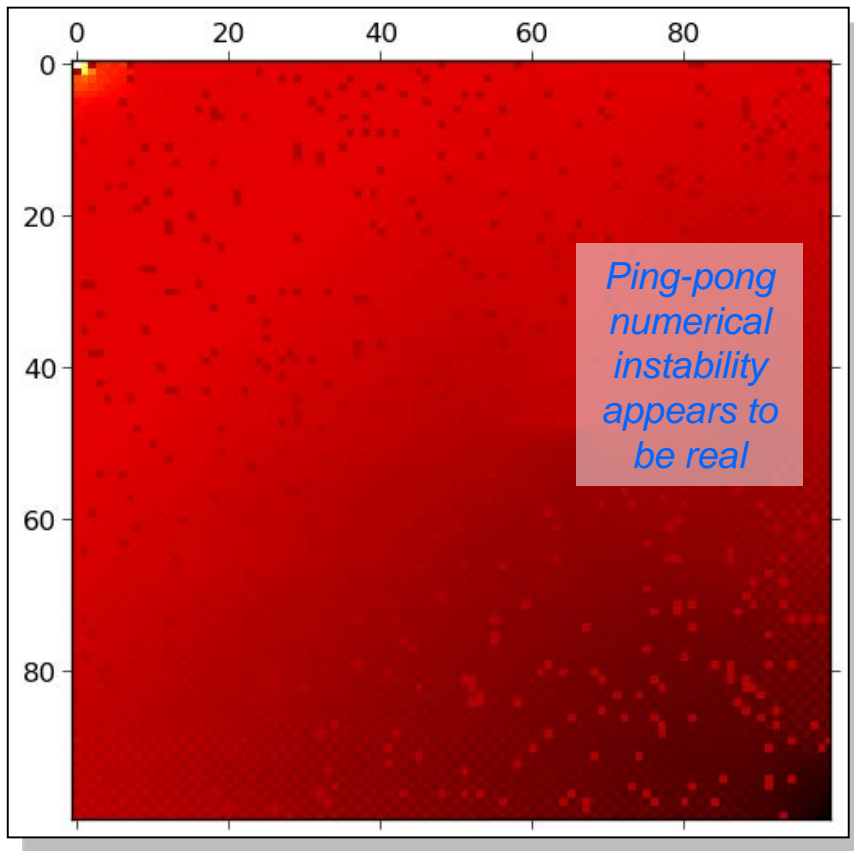
When things break:

- System with 1000000 cores
- Unrealistic to expect 100% uptime
- What happens when cores die?
- *Algorithm* 'self-heals' around unresponsive core



5% device death

Results degrade gracefully



Diagonal temperature profile vs iteration

On the nature and consequence of failures

- Generic electronic system:
 - Stuck-at, bridging, crosstalk....
 - Arbitrary effects propagate - usually disabling
- Event-based architectures:
 - Interconnect complex
 - Core/node fails *silently*
 - Affected mesh sites disappear from *model*
 - Solution *algorithm* unaffected

By 2025...

A desktop machine ...

- a *personal* computer -

... will have 25000 cores

Tree

'control_5a_run_5_stuff\IP_0000_F_Blob.frm' u 3:4:5

+

