Real-time signal detection for pulsars and radio transients using GPUs

W. Armour, M. Giles, A. Karastergiou and C. Williams.
University of Oxford.

15th July 2013

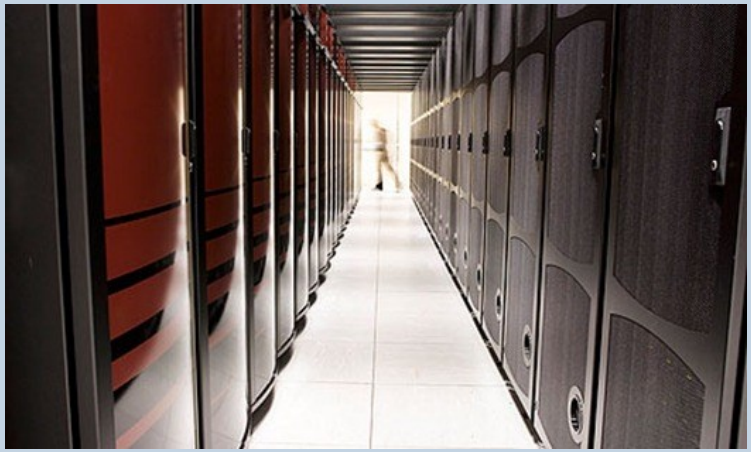
Background of GPUs

Why use GPUs ?

Influence and Take-up

TOP 500 – 5 out of top 20 utilise GPUs

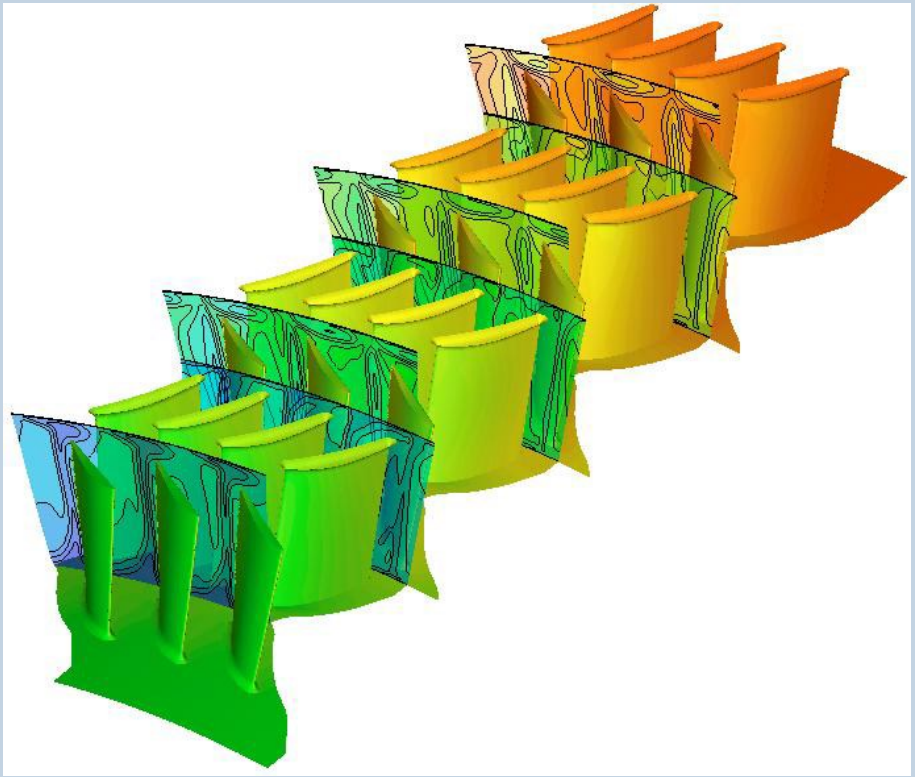
- Oak Ridge Titan - 27.1 petaflops
- Based on 560,640 Opteron cores and 261,632 K20x GPU cores
- Significant reductions in floor space and power consumption are achieved by using a CPU/GPU architecture.



Adopters of GPUs

GPUs are now used in many areas of Scientific Computing

- Engineering (CFD/CAD…)
- Computational Finance
- Medical Imaging
- Oil (Seismic Exploration)
- Bioinformatics
- Targeted Drug Discovery



Courtesy of Leigh Lapworth, Rolls-Royce plc.

Accessibility of GPUs (Development Ecosystem)

GPUs can be programmed in many different ways:

- MATLAB
- CUDA C/C++
- CUDA FORTRAN
- JCUDA (Java Bindings)
- pyCUDA

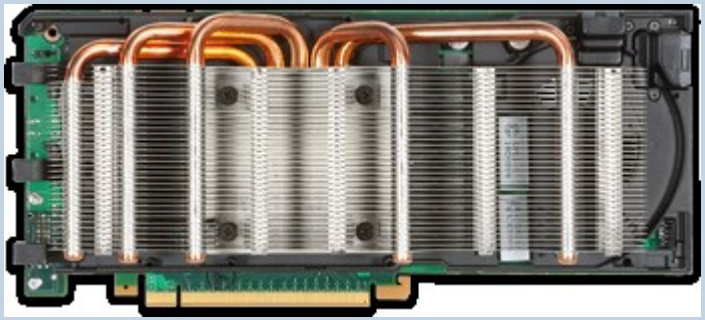
- Thrust
- cuBLAS
- cuSPARSE
- cuFFT
- Nsight
- Mathematica

- OpenCL
- OpenACC

Latest Kepler based cards

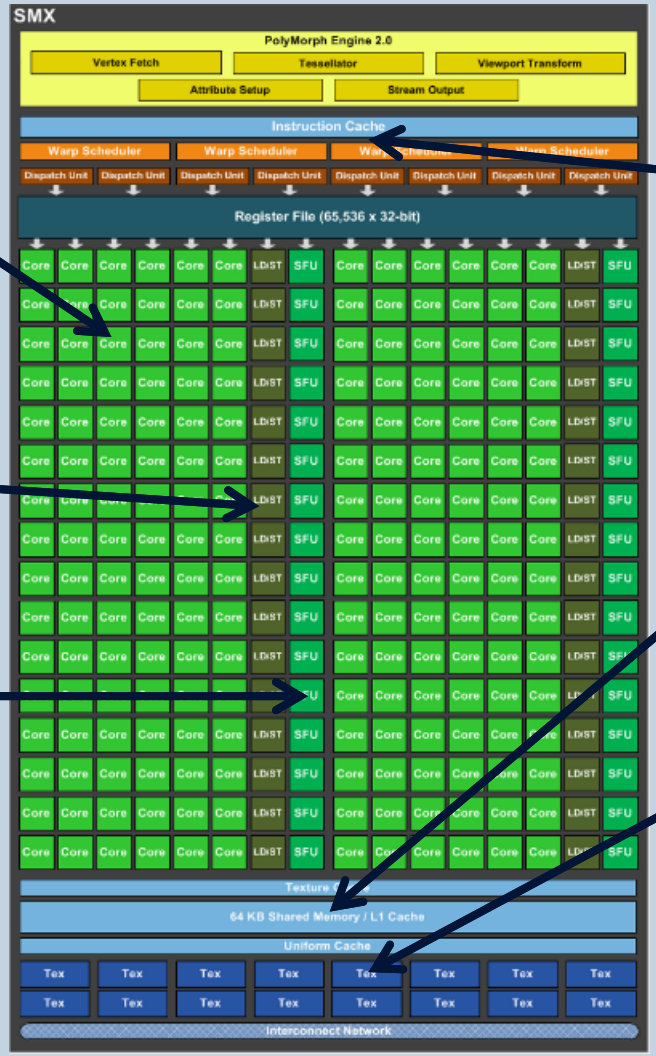
- GeForce GTX 690 – 2x GK104 GPUs
- 18.7 GFLOP / watt
- Peak performance ~ 5.6 TFLOP/S
- Price point ~ £1000

- K10 – 2x GK104 GPU 2x1536 cores.
- Peak performance 4.6 TFLOP/S
- 20.4 GFLOP / watt
- Price point ~ £3000 (?)



Two Sandy Bridge Xeons (similar price) ~ 1 TFLOP/S

Kepler Streaming Multiprocessor (SMX) design



Compute cores

Schedulers

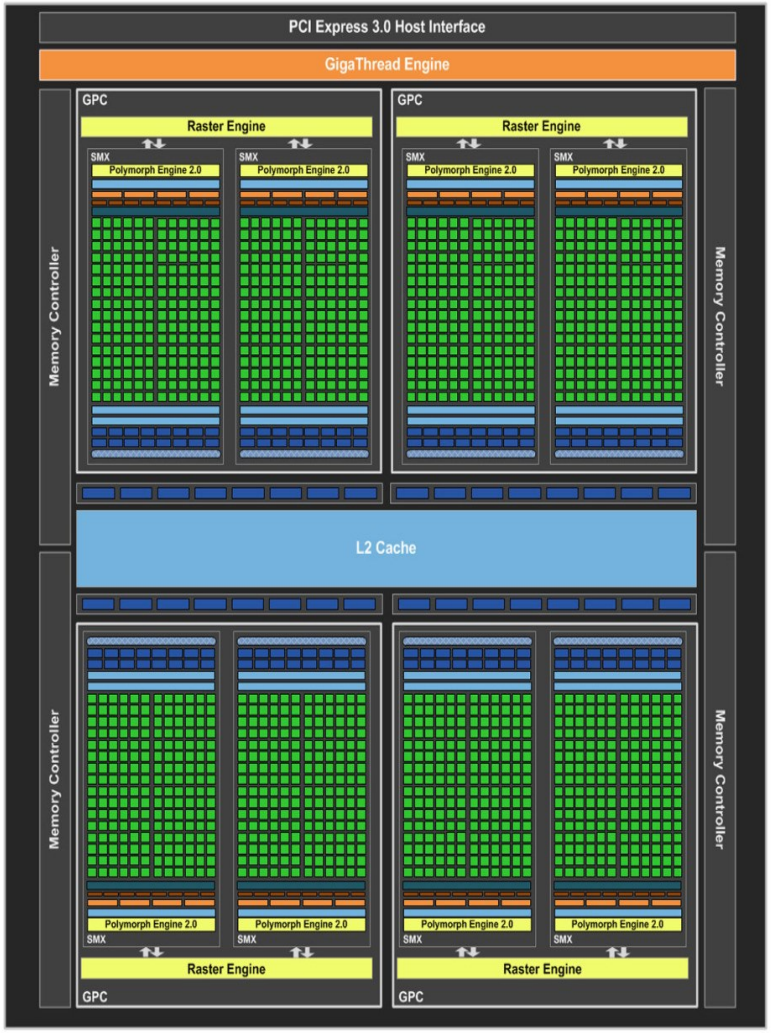
Load/Store

User managed cache

Special functions

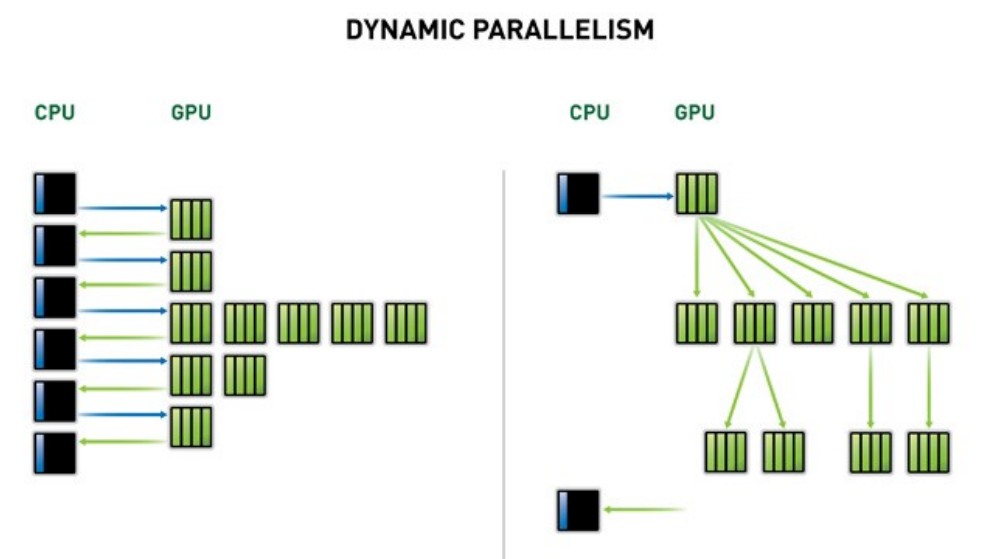
Texture units

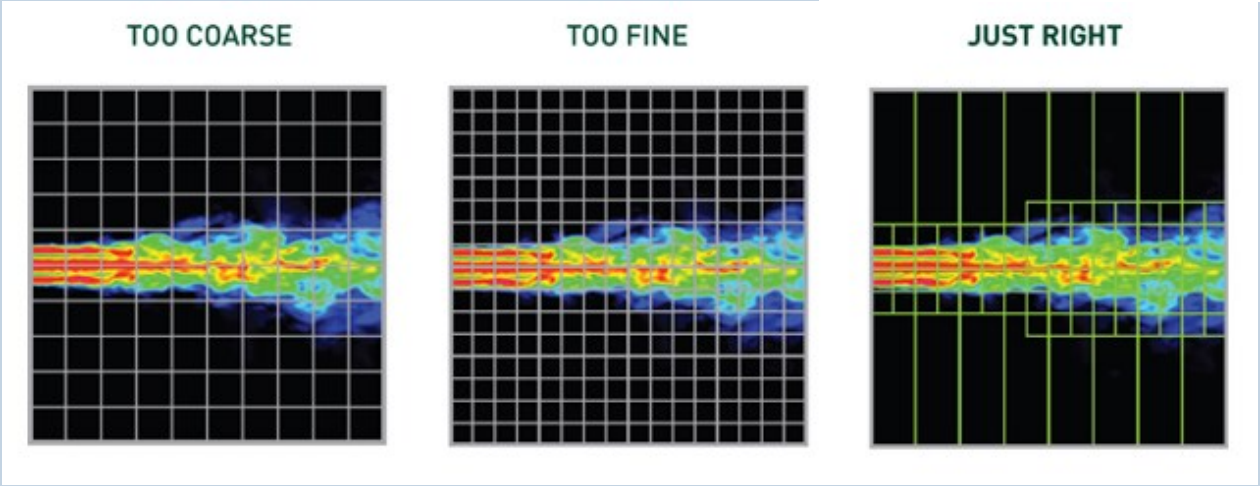
Kepler design



Latest Features - Dynamic Parallelism

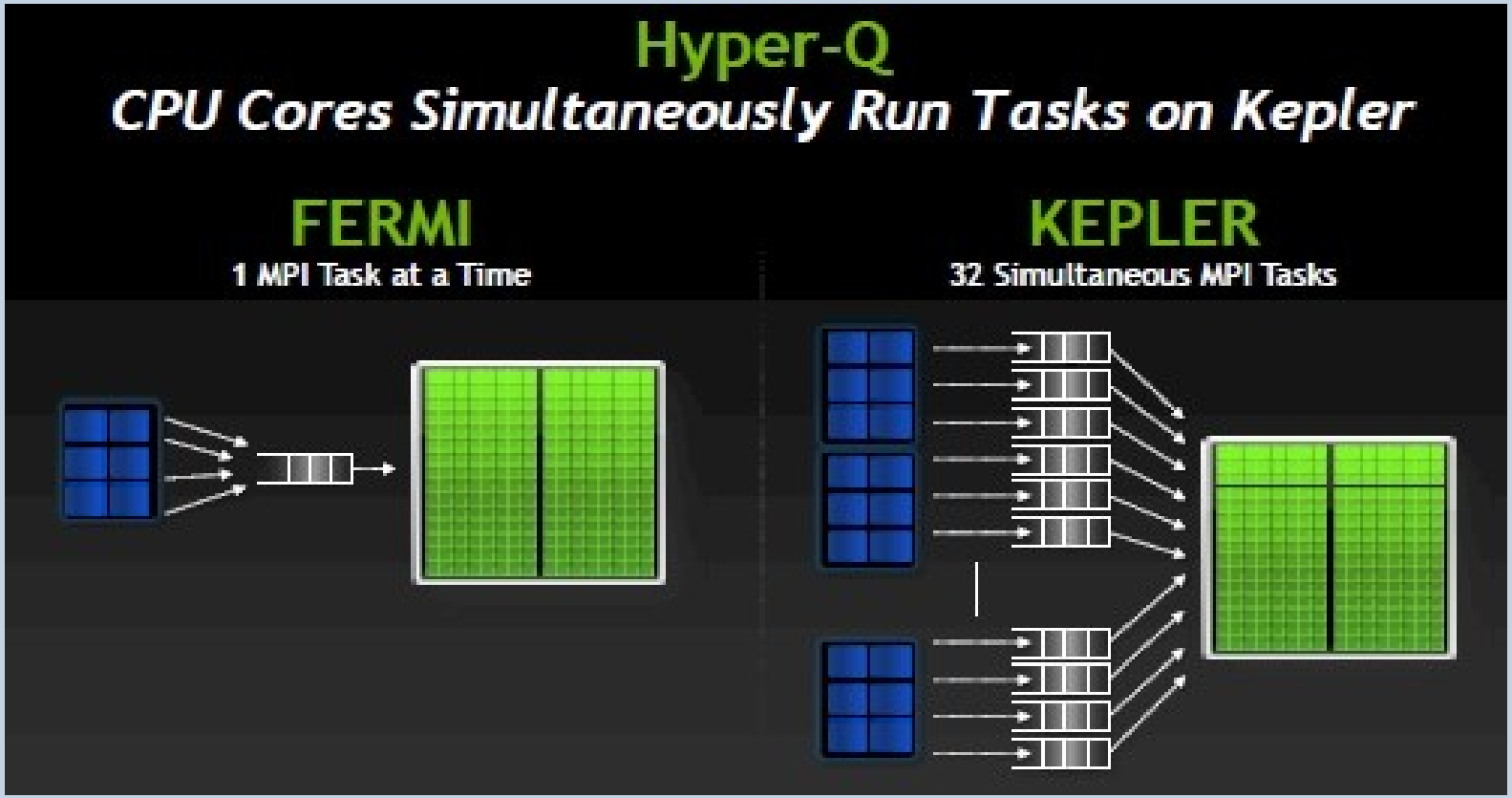
A GPU job can launch new GPU Jobs





Latest Features - Hyper-Q

Multiple CPU threads can run independent jobs on a single GPU at the same time

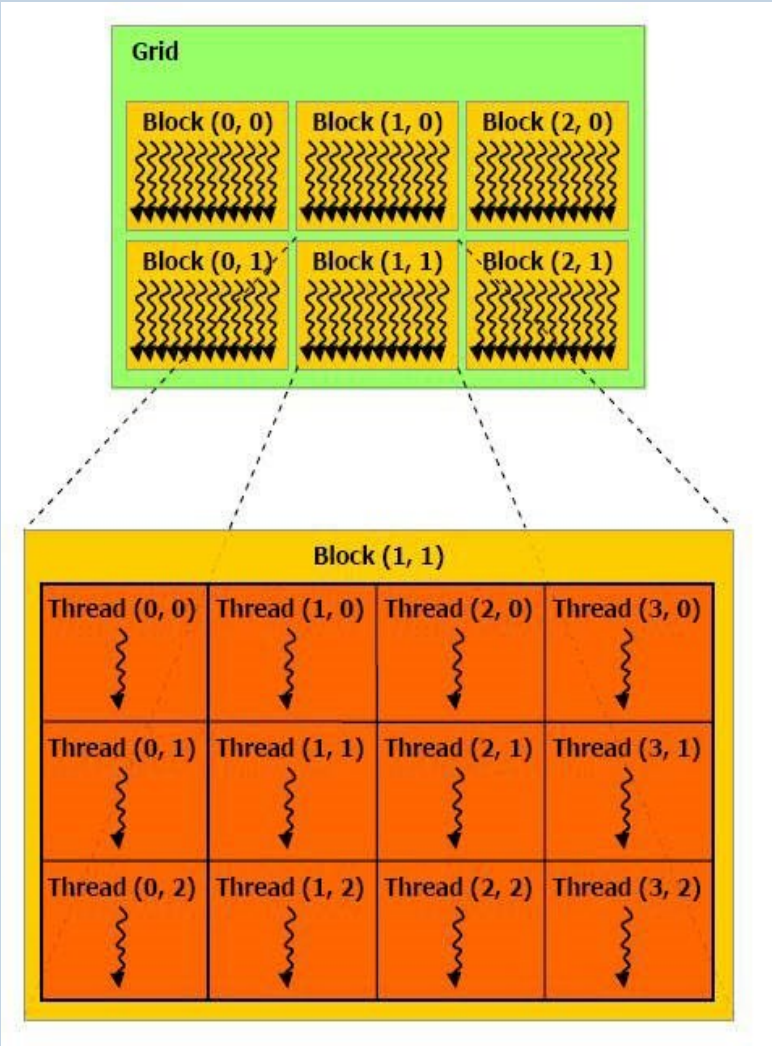


Programming Paradigm

Parallel granularity and data sharing.

- Each cuda core (SP) executes a sequential thread, in SIMT (Single Instruction, Multiple Thread) fashion - all cores in the same group execute the same instruction at the same time (like SIMD).
- Threads are executed in groups of 32 – a *warp*.
- To hide high memory latency, warps are executed in a time-multiplexed fashion - When one warp stalls on a memory operation, the multiprocessor selects another ready warp and switches to that one.

Parallel granularity and data sharing.



- Kernel launches a grid of (3,2) thread blocks…

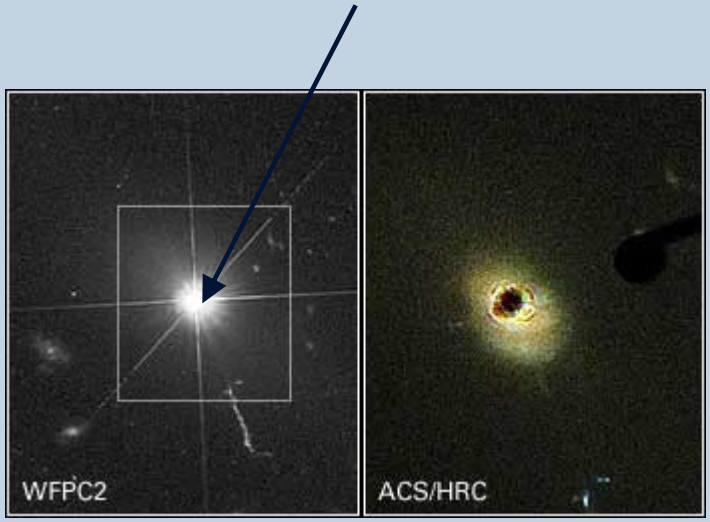
Kernel<<< (3,2),(4,3) >>>(params)

- Each thread block consists of (4,3) unique threads.

Radio Transients and Pulsars

Radio Astronomy and Radio Transients

Quasars – Energetic region of a distant galactic core, surrounding a supermassive black hole



NASA and J. Bahcall (IAS)

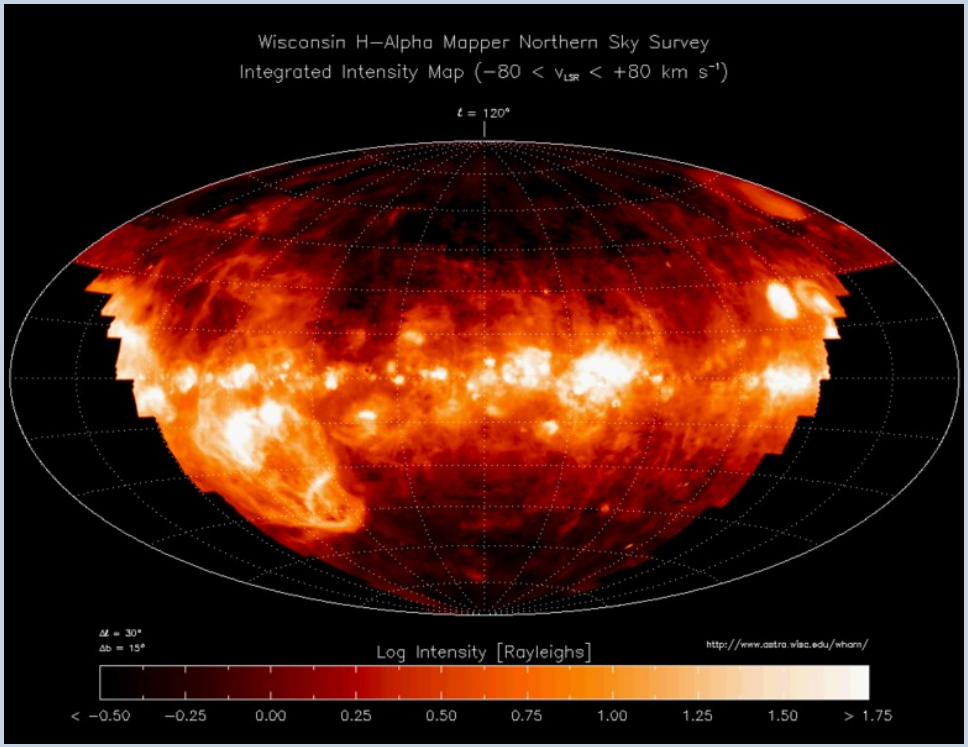
Hester et al.

Pulsars – Magnetized, rotating neutron stars. Emit synchrotron radiation from the poles, e.g. Crab Nebula

RRATS – Rotating Radio Transients. Short, bright irregular radio pulses. Discovered 2006

Dispersion of Radio waves by the ISM

The interstellar medium (ISM) is the matter that exists between stars in a galaxy.



Haffner et al. 2003

In warm regions of the ISM (~8000K) electrons are free and so can interact with and effect radio waves that pass through it.

The Dispersion Measure - DM

The time delay, $\Delta\tau$, between the detection of frequency f_{high} and f_{low} is given by:

$$\Delta\tau = C_{DM} \times DM \times \left(\frac{1}{f_{low}^2} - \frac{1}{f_{high}^2}\right)$$

Where C_{DM} is the dispersion constant. DM is the dispersion measure:

$$DM = \int_0^d n_e dl$$

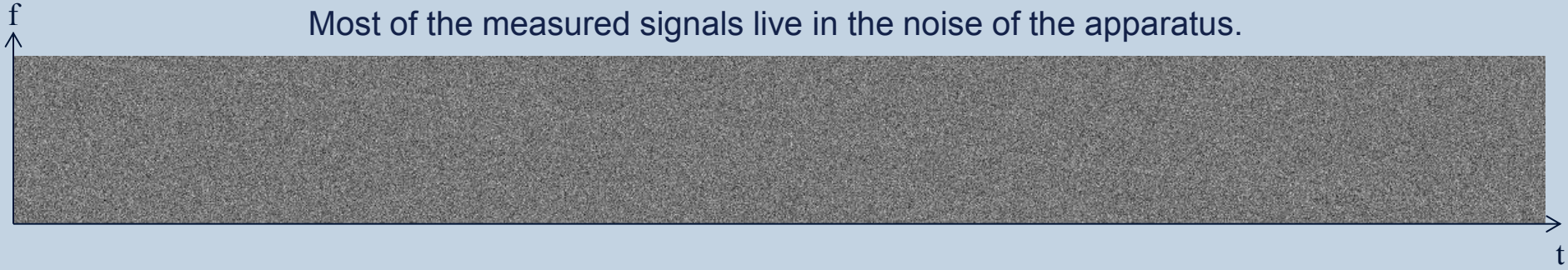
This is the free electron column density between the radio source and observer.

We can measure $\Delta\tau$ and f and so can study DM

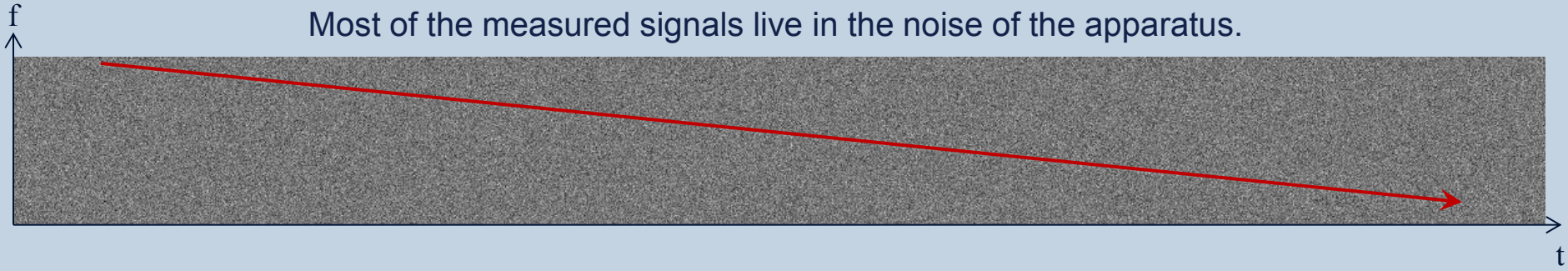
De-dispersion

Experimental data

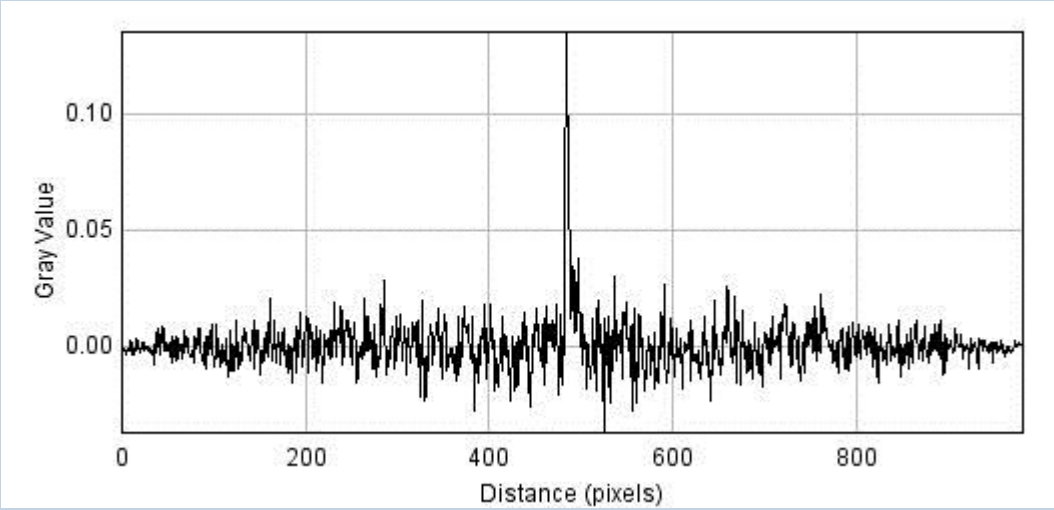
Most of the measured signals live in the noise of the apparatus.



Experimental data

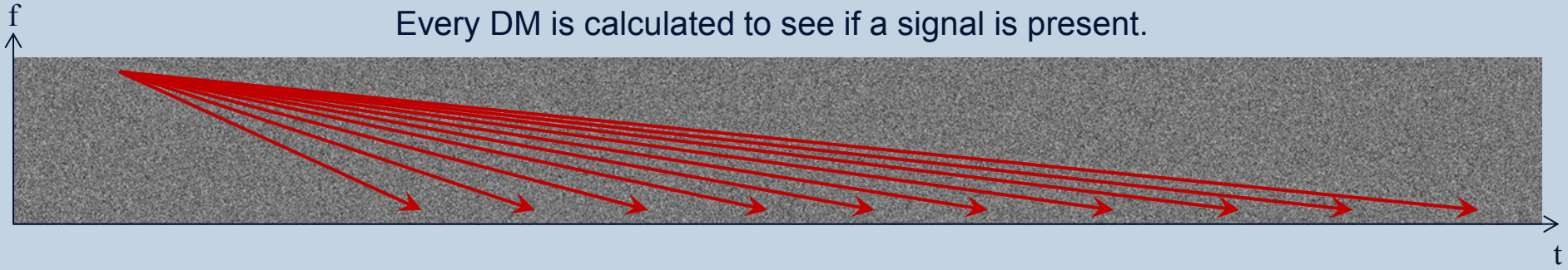


Hence frequency channels have to be “folded”



OXFORD e-Research CENTRE

Brute force algorithm

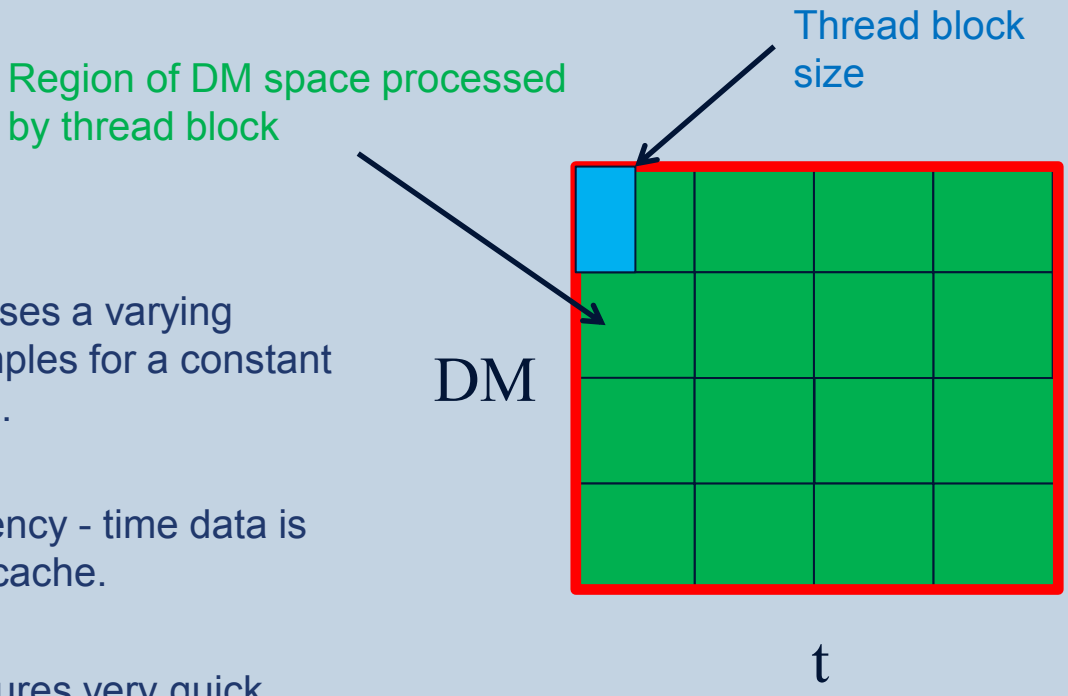


- In a blind search for a signal many different dispersion measures are calculated.
- This results in many data points in the (f,t) domain being used multiple times for different dispersion searches.
- This allows for data reuse in a GPU algorithm.

All of this must happen in real-time i.e. The time taken to process all of our data must not exceed the time taken to collect it

Processing several DM's per thread

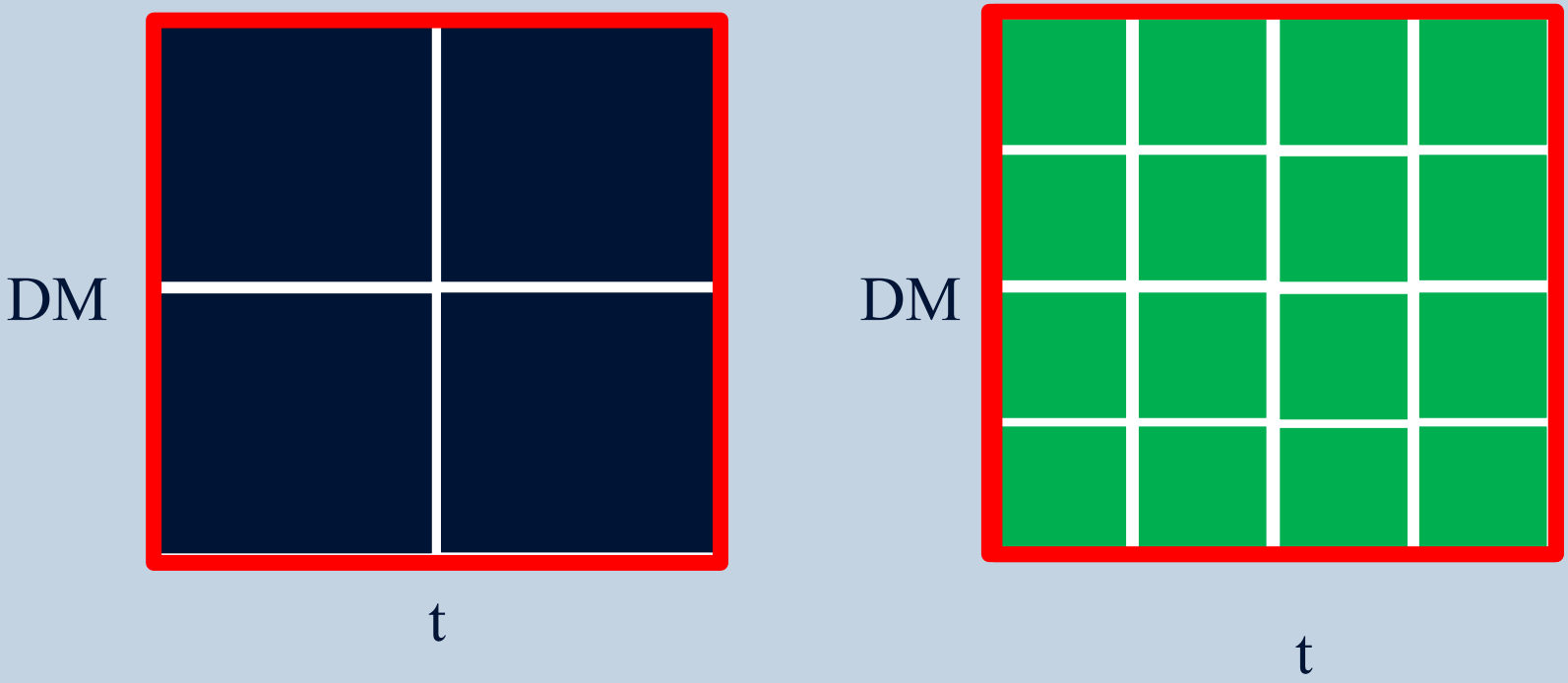
New Algorithm works in the DM - t space rather than frequency – time space.



- Each thread processes a varying number of time samples for a constant dispersion measure.
- This ensures frequency - time data is loaded into fast L1 cache.
- Using registers ensures very quick memory access.

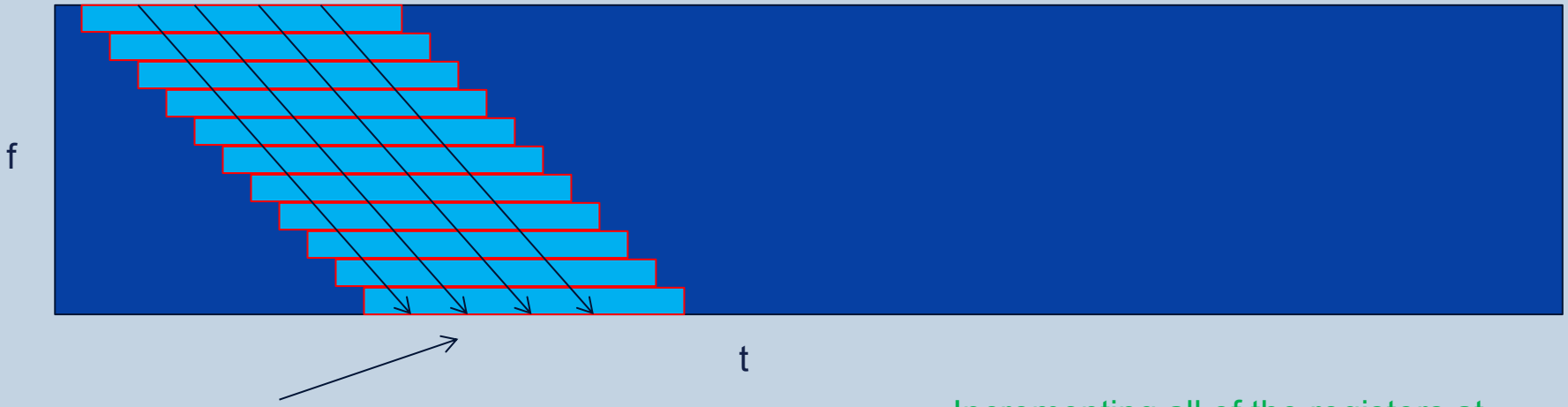
Optimising the parameterisation.

The GPU block size of the new algorithm can take on any size that is integer multiples of the size of a “data chunk”…



Exploiting the L1 cache / Shared Memory…

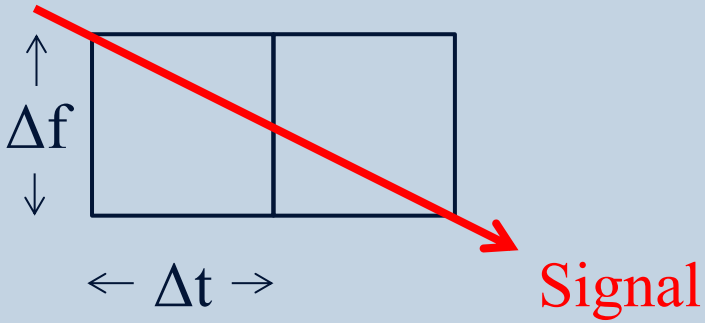
Each dispersion measure for a given frequency channel needs a shifted time value.



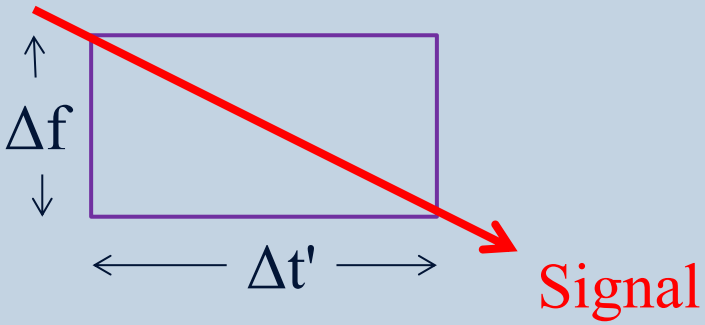
Constant DM's with varying time.

Incrementing all of the registers at every frequency step ensures a high data reuse of the stored frequency time data in the L1 cache or shared memory.

Time binning



When scattering and dispersion effects are high a radio signal can be spread over multiple time samples, all having the same frequency.



It makes sense to add the values of adjacent time data to increase the signal to noise. This reduces the amount of time samples to process at higher DMs (also increases the step size between DMs to achieve critical sampling)

Code Execution Path

Split DM Region (diagonal DM)

Code Execution Path

TIME

Split DM Region (diagonal DM)

De-disperse A

Code Execution Path

TIME

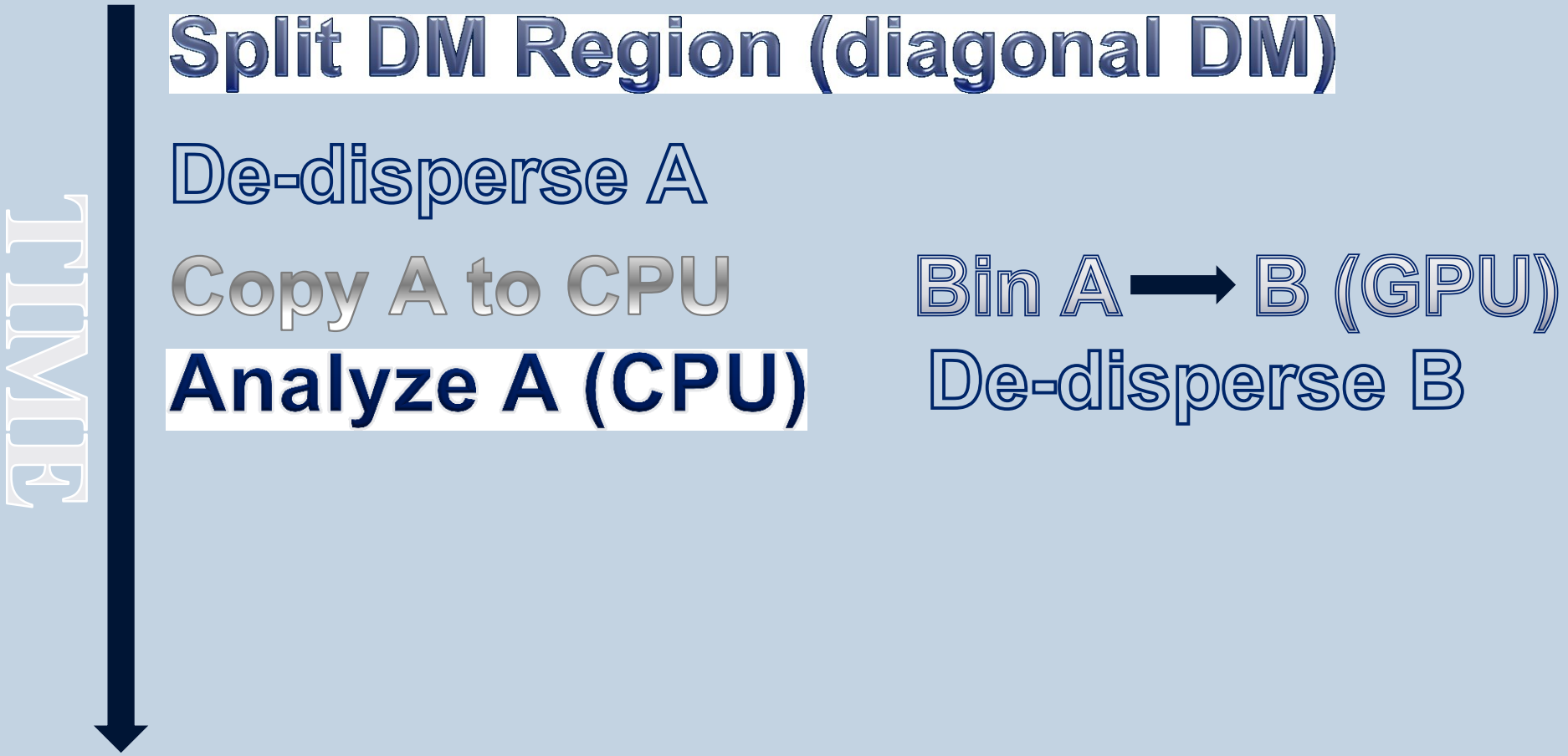
Split DM Region (diagonal DM)

De-disperse A

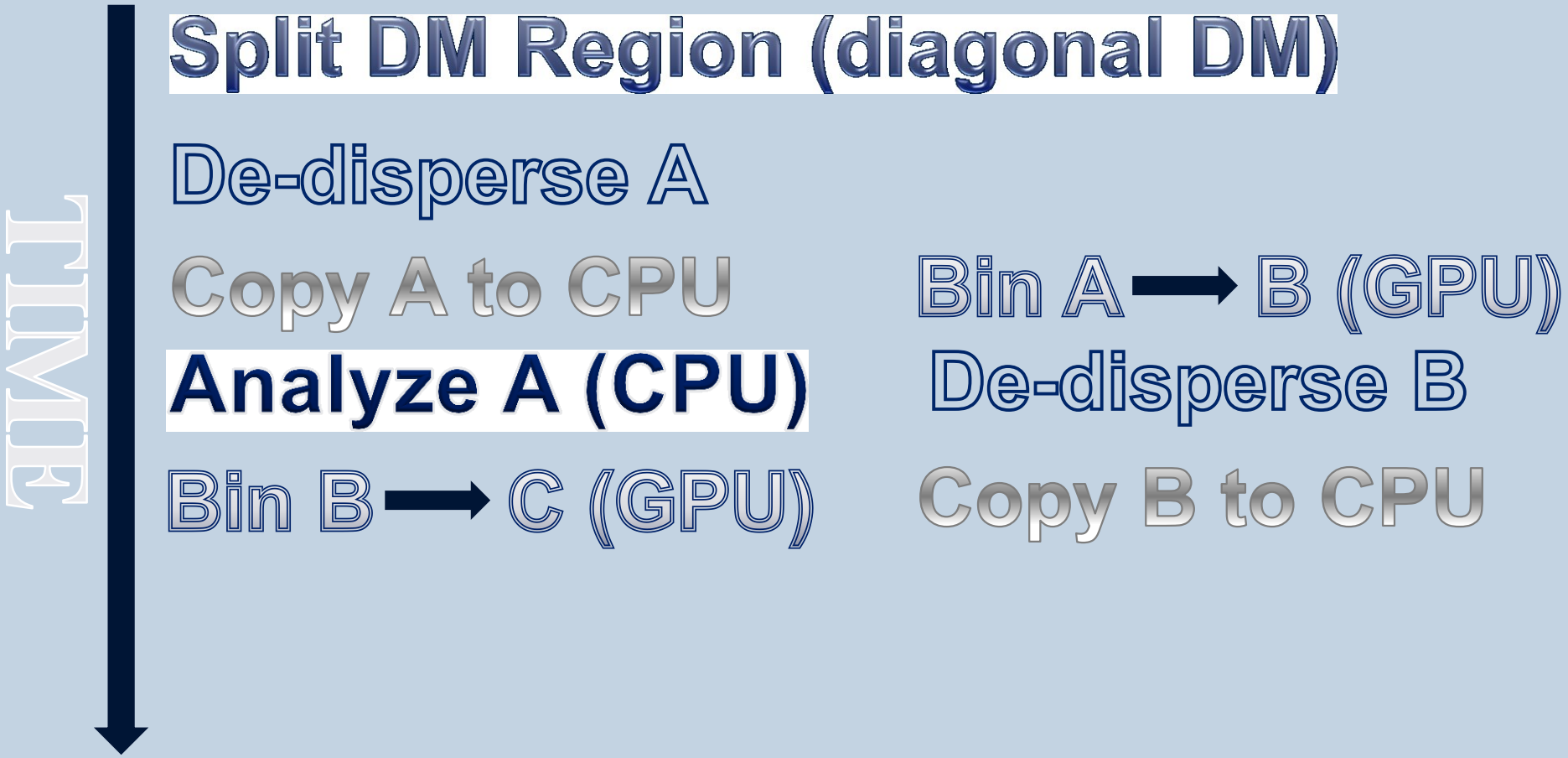
Copy A to CPU

Bin A → B (GPU)

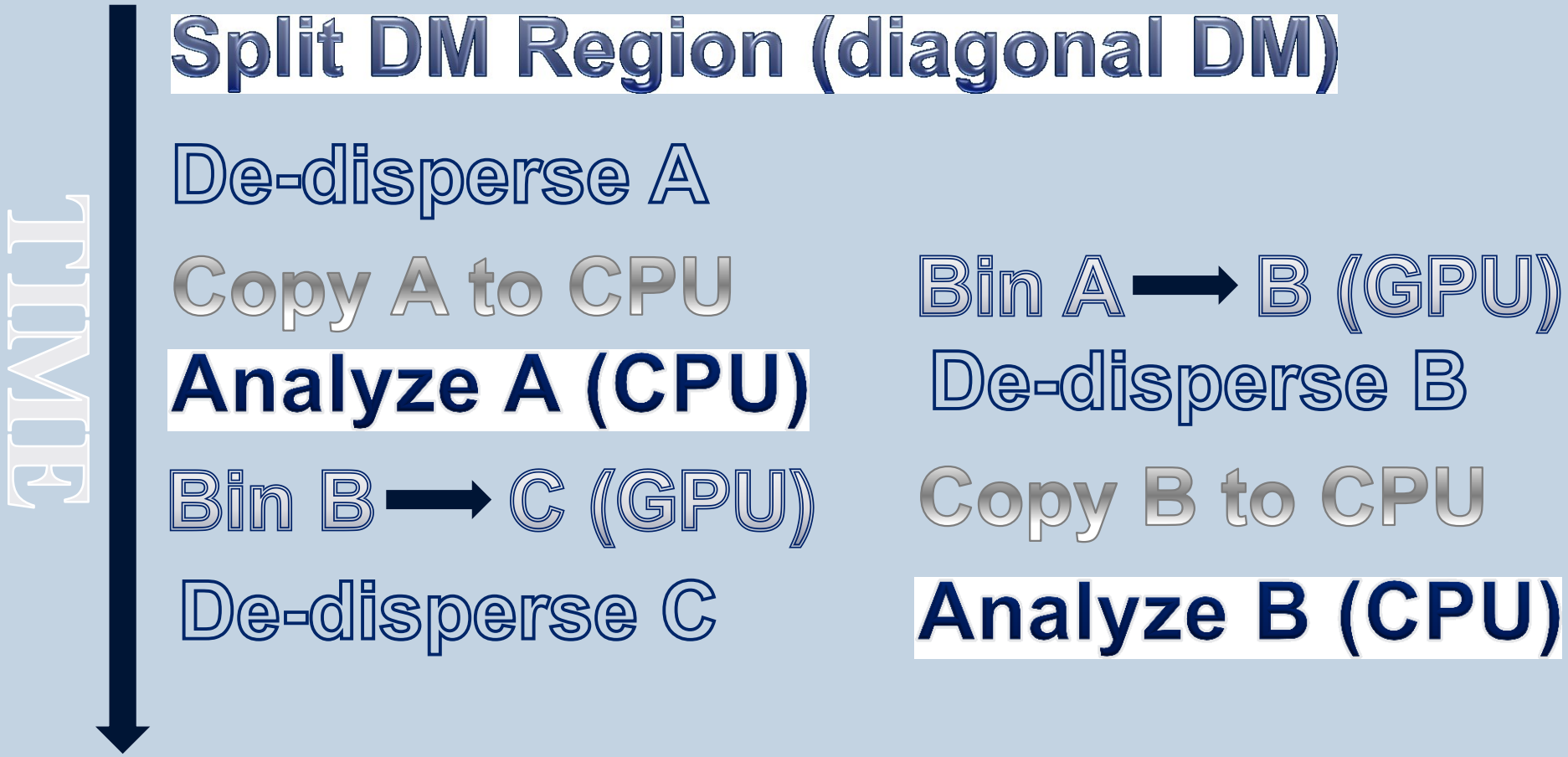
Code Execution Path



Code Execution Path



Code Execution Path



Code Execution Path

ODD/EVEN DOUBLE BUFFERED APPROACH

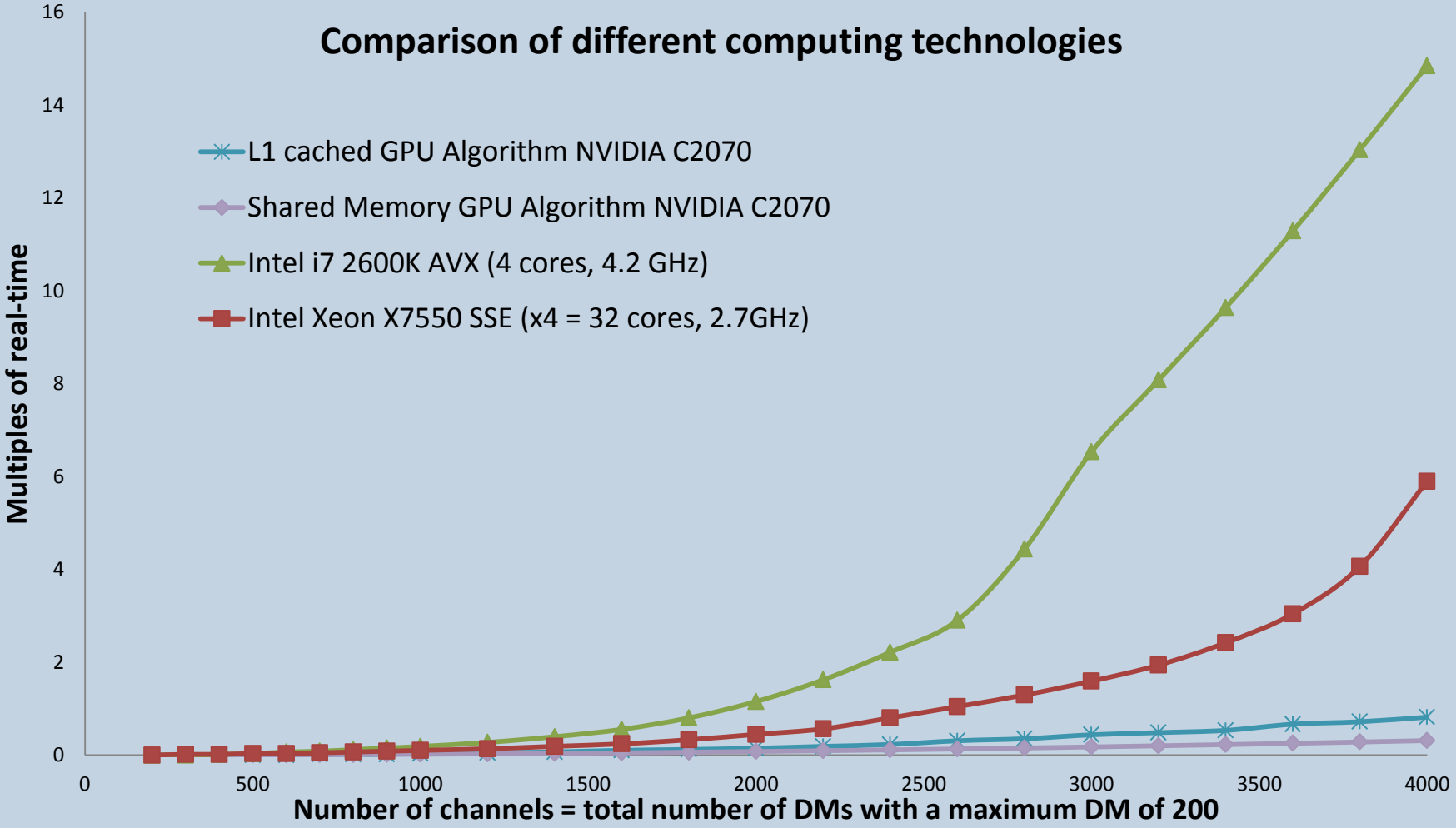
Allows for Multi-core (and Vector) CPU usage along with PCIe and GPU usage at the same time = High system utilisation

De-dispersion results…

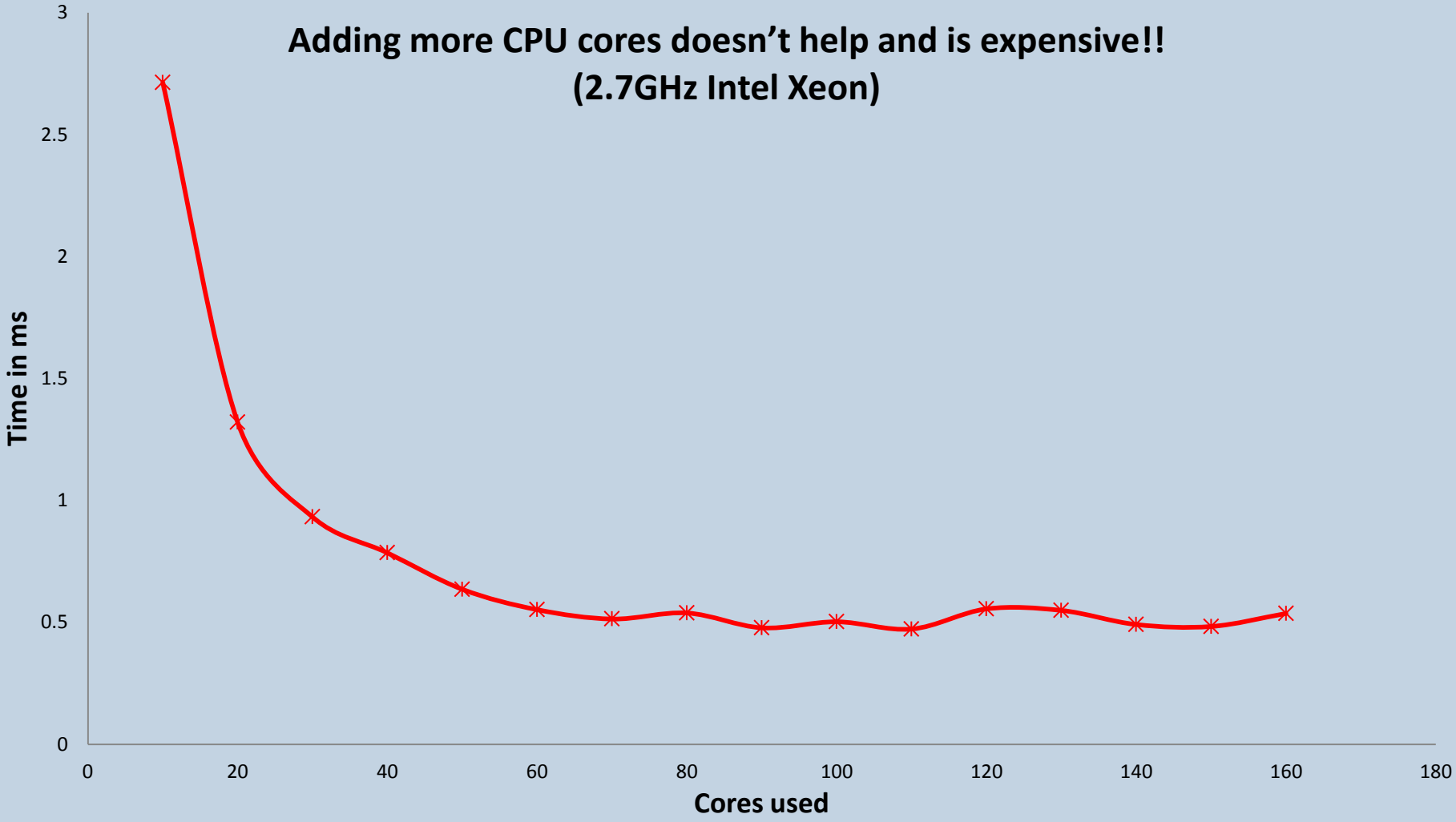
Results for LOFAR data

(SKA Pathfinder)

Results…

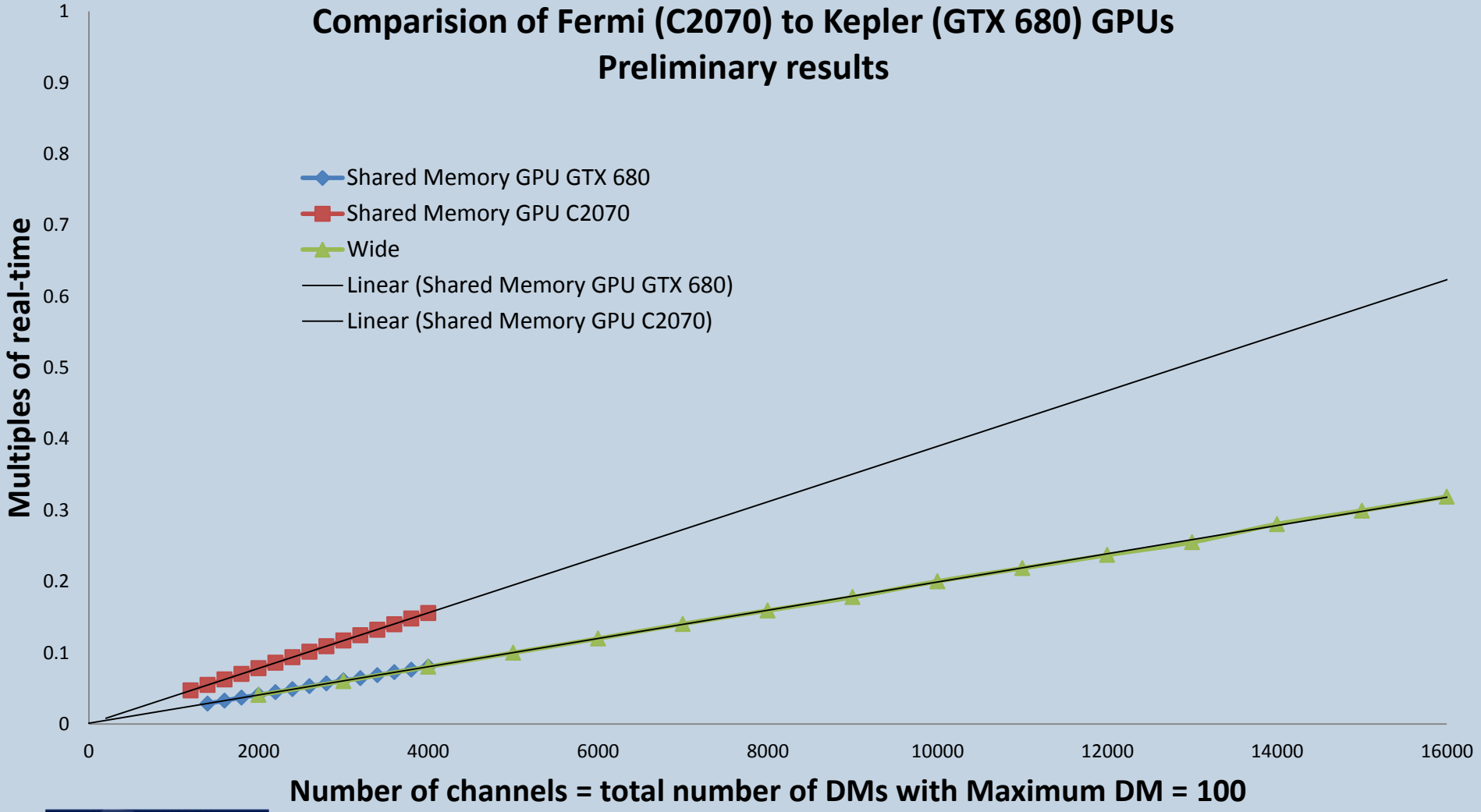

Comparison of different computing technologies
L1 cached GPU Algorithm NVIDIA C2070
Shared Memory GPU Algorithm NVIDIA C2070
Intel i7 2600K AVX (4 cores, 4.2 GHz)
Intel Xeon X7550 SSE (x4 = 32 cores, 2.7GHz)
Multiples of real-time
0
2
4
6
8
10
12
14
16
0
500
1000
1500
2000
2500
3000
3500
4000
Number of channels = total number of DMs with a maximum DM of 200

Results…



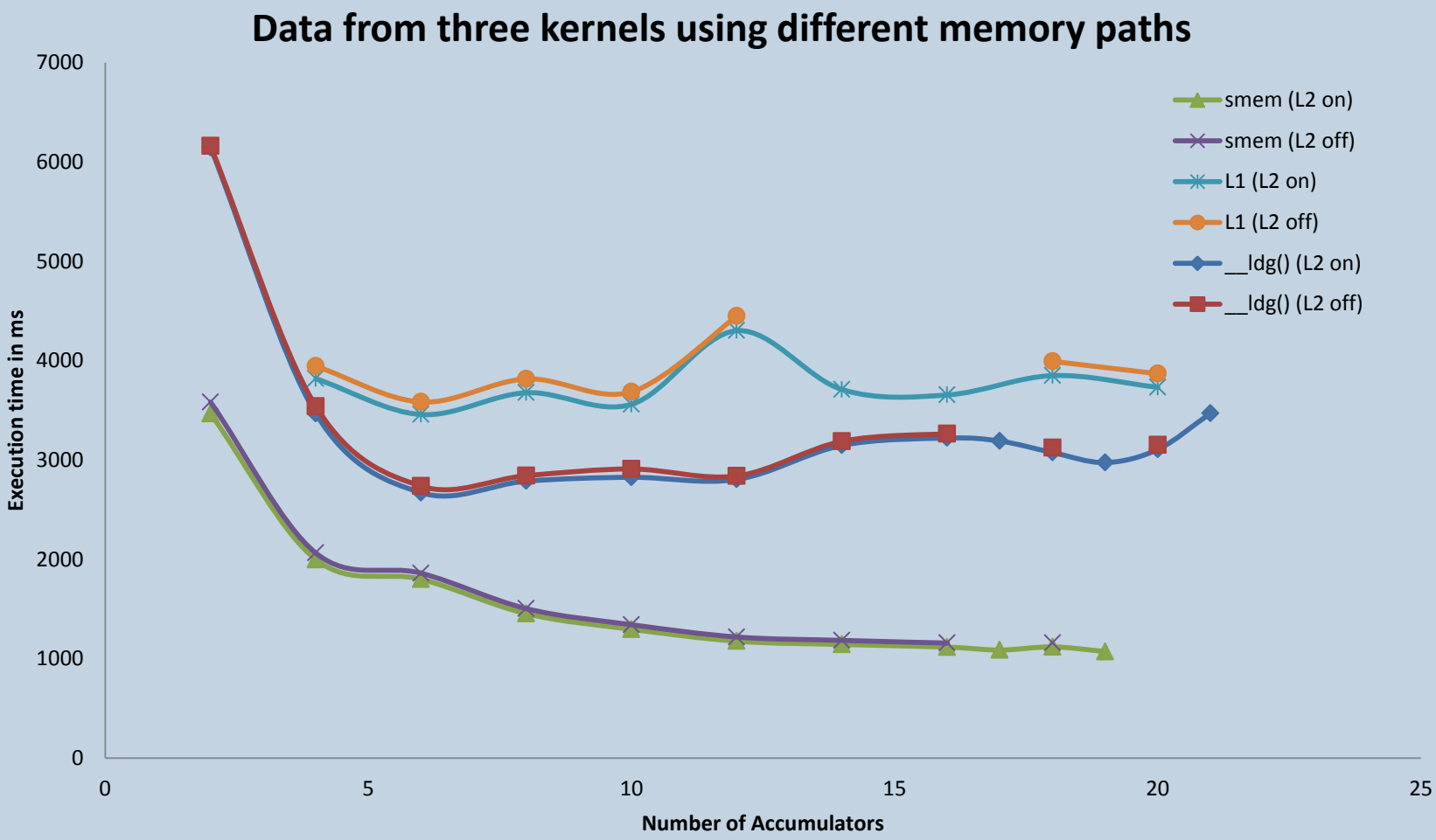
OXFORD e-Research CENTRE

Results…



OXFORD e-Research CENTRE

Results…


Data from three kernels using different memory paths
Execution time in ms
7000
6000
5000
4000
3000
2000
1000
0
0
5
10
15
20
25
Number of Accumulators
smem (L2 on)
smem (L2 off)
L1 (L2 on)
L1 (L2 off)
__ldg() (L2 on)
__ldg() (L2 off)

Conclusions

- GPU wins hands-down. At the moment (and for the foreseeable future)!

Shared Memory Algorithm achieves 85% of peak performance.

- OpenCL Algorithm, Dan Curran / Simon McIntosh-Smith (Bristol): Initial results are currently 2x slower than NVIDIA CUDA Code but improving!

- NVIDIA promised a 3x performance increase moving from Fermi to Kepler (GPU generations). **We achieved a respectable 2.6x** and would expect this again moving from Kepler to Volta in 2015 - 2016.

A single Kepler K10 can process a 12Gb LOFAR data stream in real-time using our latest algorithm.

Acknowledgments and Collaborators

GPU de-dispersion : http://www.oerc.ox.ac.uk/research/wes

ARTEMIS : http://www.oerc.ox.ac.uk/research/artemis

University of Oxford

Mike Giles (Maths) – Cuda, GPU algorithms.

Aris Karastergiou (Physics) – ARTEMIS, Astrophysics, Experimental Work.

Kimon Zagkouris (Physics) – Astrophysics, Experimental Work.

Chris Williams (OeRC) – RFI Clipper, Data pipeline.

Ben Mort (OeRC) – Data Pipeline, pelican.

Fred Dulwich (OeRC) – Data Pipeline, pelican.

Stef Salvini (OeRC) – Data Pipeline, pelican.

Steve Roberts (Engineering) – Signal processing/detection algorithms.

University of Bristol

Dan Curran (Electrical Engineering) – OpenCL work.

Simon McIntosh Smith (Electrical Engineering) – OpenCL work.

University of Malta

Alessio Magro – MDSM