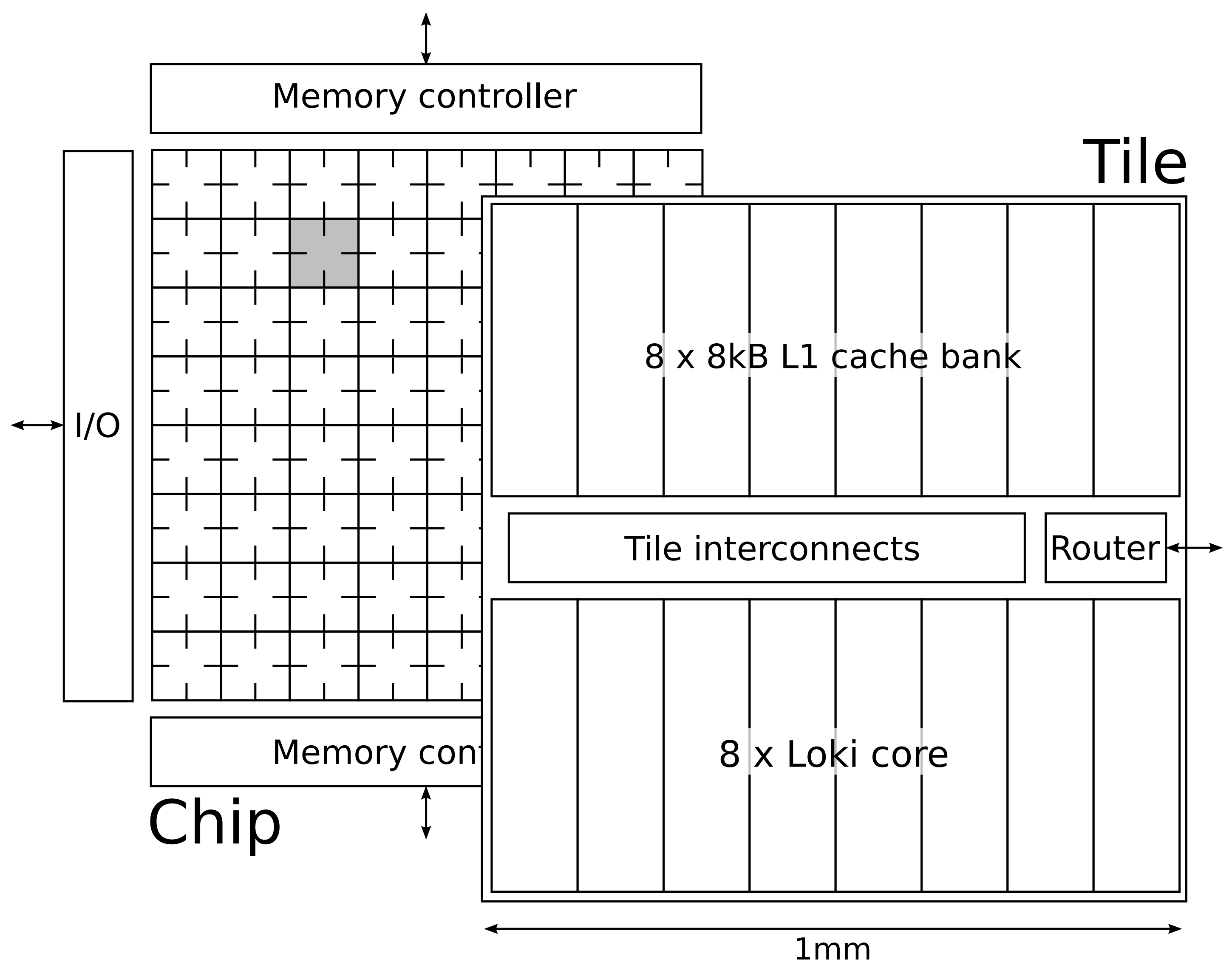
Exploiting tightly-coupled cores

Daniel Bates, Alex Bradbury,
Andreas Koltes, Robert Mullins

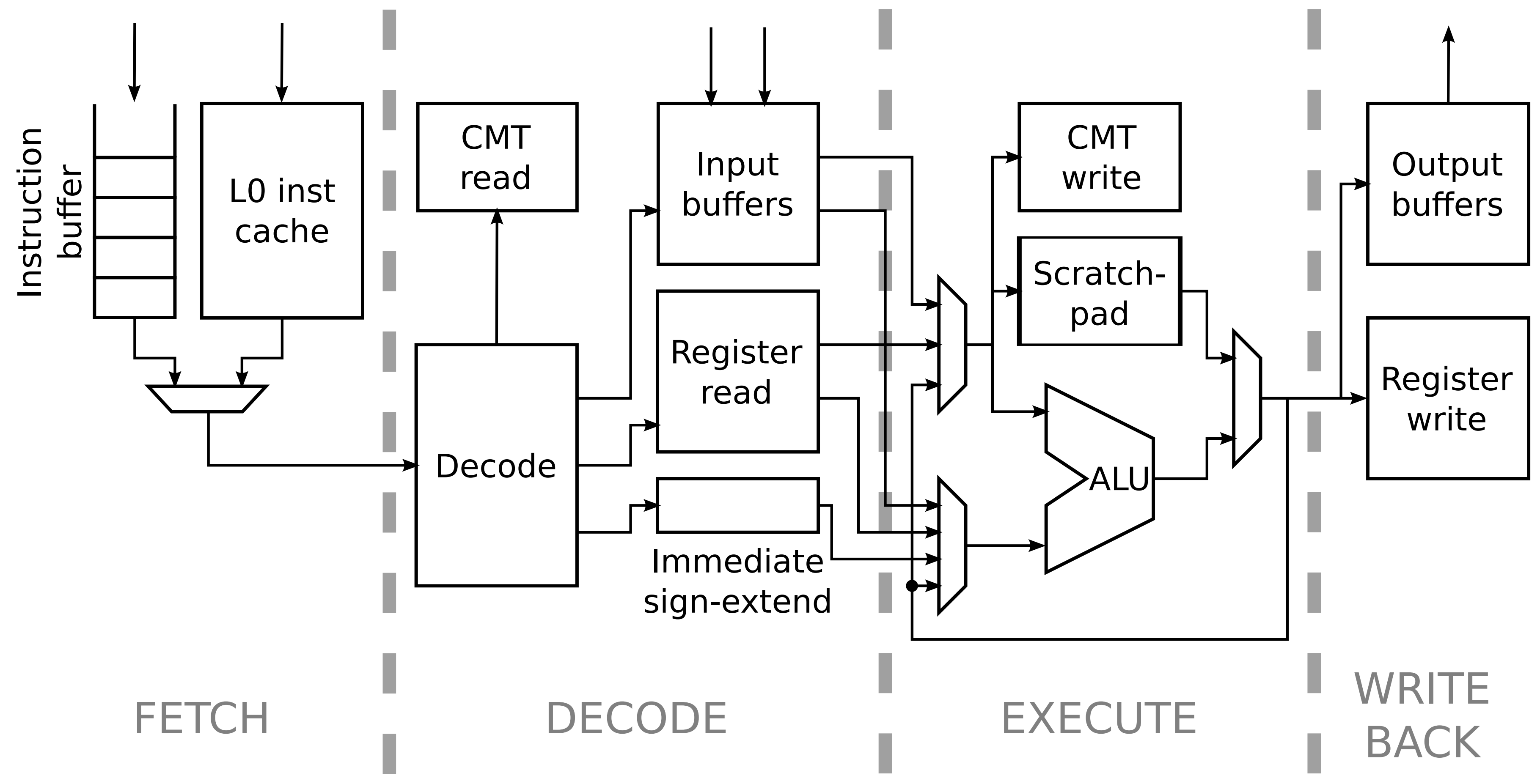
UNIVERSITY OF CAMBRIDGE
Computer Laboratory

Challenges: processor energy consumption limited by battery capacity; fault tolerance becoming necessary; design and validation costs rising rapidly; parallelism is difficult.

Solution: Loki, a flexible architecture capable of combining resources to create a *virtual processor* optimised for the task at hand. Efficient communication is key.



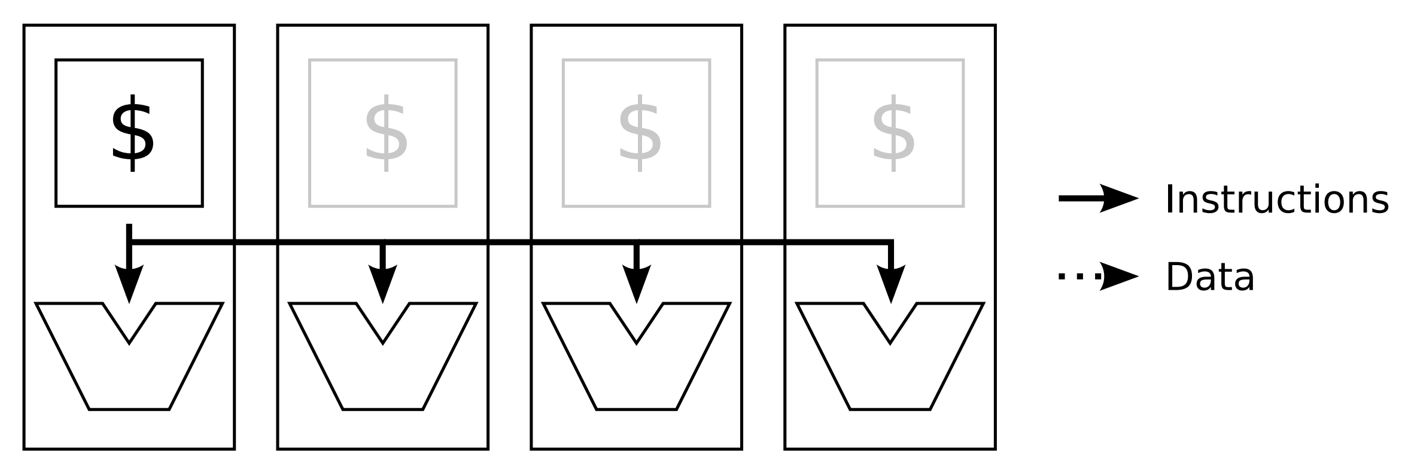
- Tiled, homogeneous architecture
- 100s-1000s of cores on a chip (at 40nm)
- All cores and memory banks have a channel map table (CMT) to hold network addresses



- Short, single-issue pipeline to reduce power
- Network connected directly to datapath
- Single-cycle latency between cores in same tile
- <10pJ per instruction possible (ARM1176: 140pJ)

These execution patterns (and more) can be combined arbitrarily to form an optimised virtual processor:

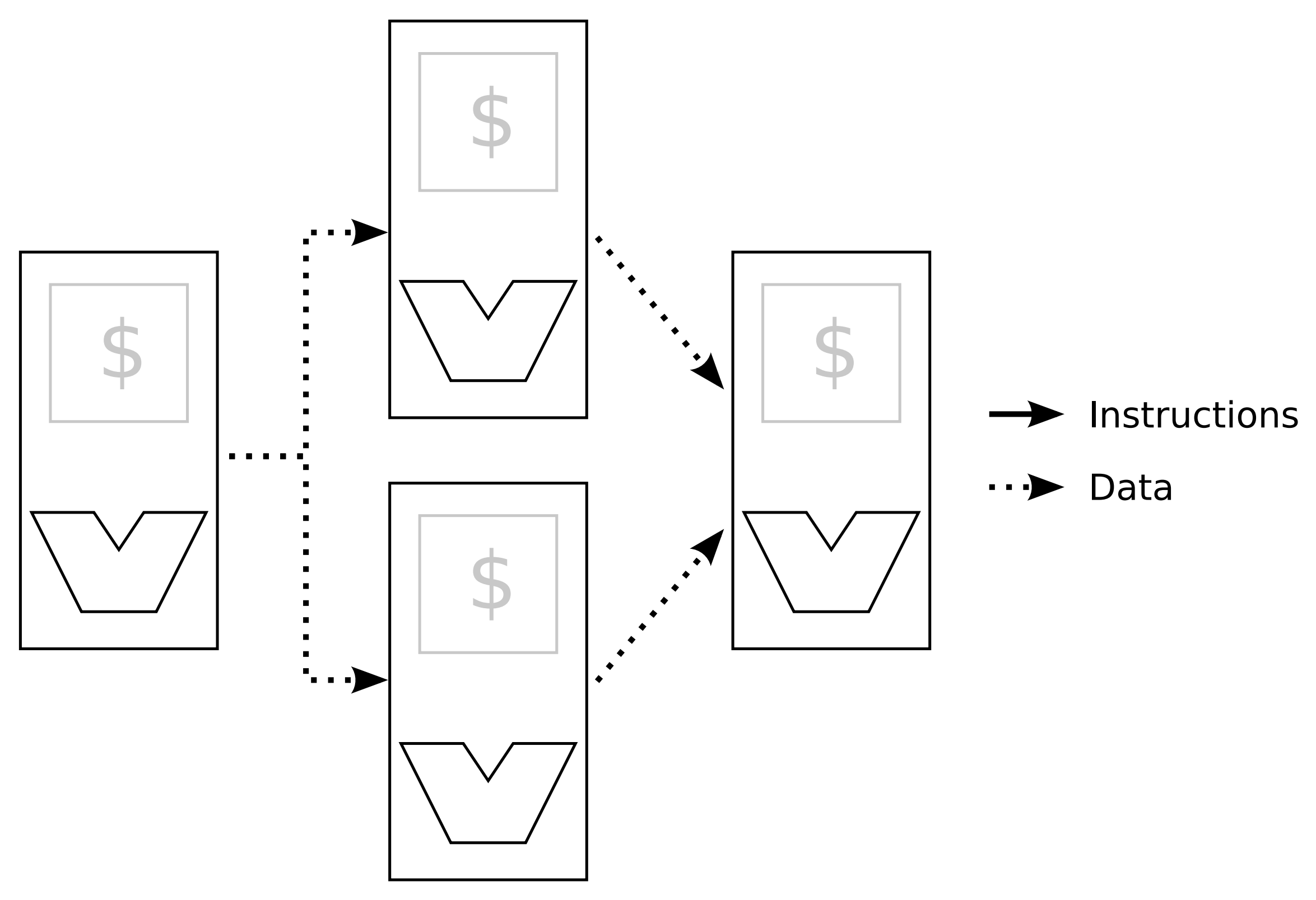
SIMD



- Only one core at a time fetches each instruction
- Multicast to all other cores
- Share cache capacity
- Optimisation: reserve one core to provide data common to all others

Typical example: Dijkstra - multiple paths	8 cores 4.4x speedup 0.8x energy

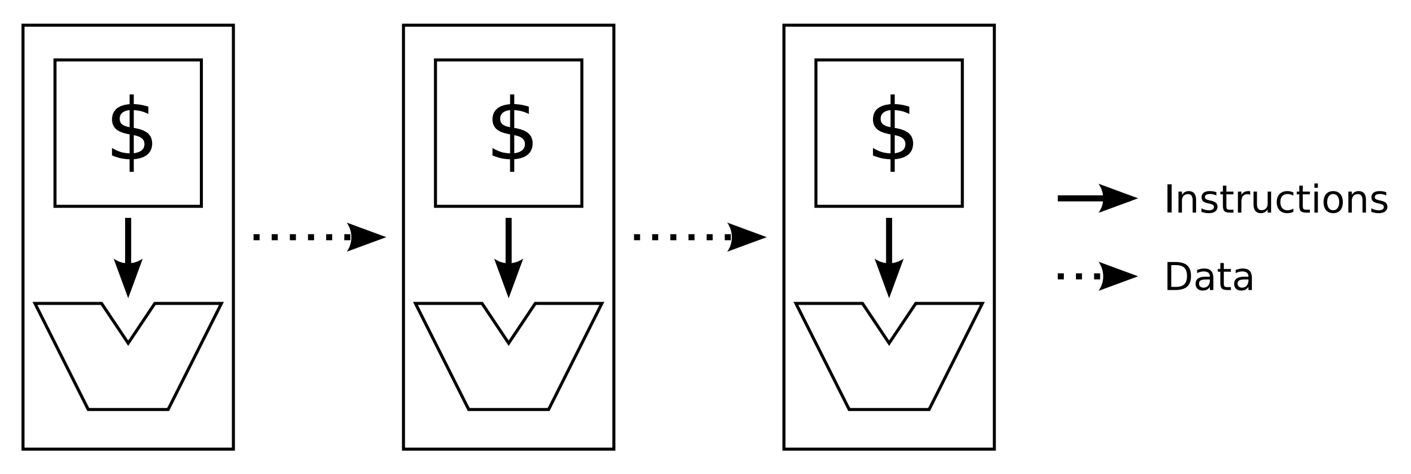
Dataflow



- Very few instructions per core
- Send/receive data over network
- With one instruction per core, don't need to re-issue each cycle
- Much of the pipeline is inactive, and can be clock-gated

Typical example: CRC	5 cores 4.7x speedup 0.6x energy

Task-level pipeline



- Perform a small block of work on input data before passing result on
- Specialise each core enough that its task can be cached well
- More core resources (e.g. registers) available for optimisations

Typical example: ADPCM compression	2 cores 2.5x speedup 0.3x energy

Daniel.Bates@cl.cam.ac.uk
Alex.Bradbury@cl.cam.ac.uk
Andreas.Koltes@cl.cam.ac.uk
Robert.Mullins@cl.cam.ac.uk

Computer Architecture Group
http://www.cl.cam.ac.uk/research/comparch