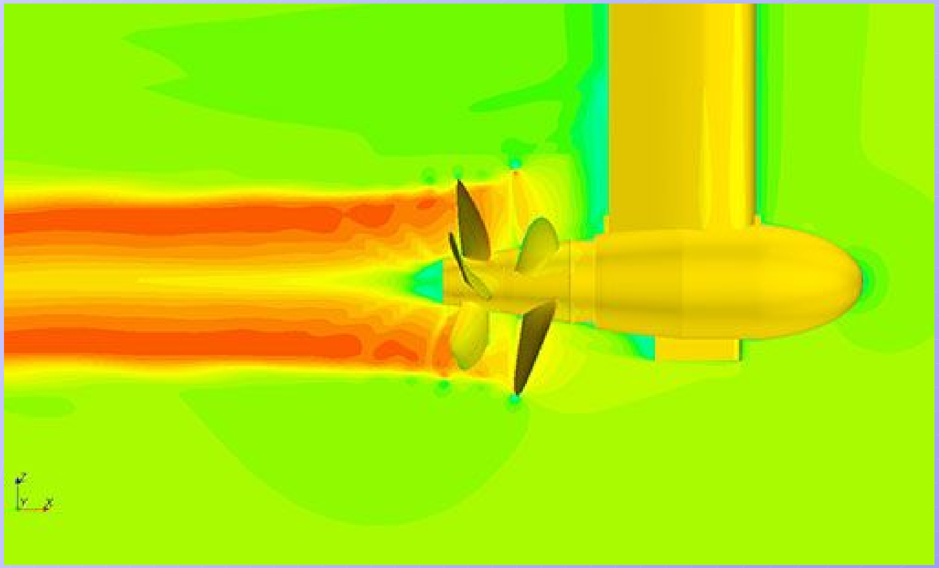
Master Thesis

Exploring the Epiphany manycore architecture
for the Lattice Boltzmann algorithm

Sebastian Raase

18th November, 2014

Preface



(source: www.sintef.no)

- This thesis is a cooperation between Volvo Penta AB and Högskolan i Halmstad.
- Volvo Penta designs and builds boat drive systems.

Motivation

- The Parallella system has been advertised on Kickstarter as “A Supercomputer For Everyone” – and succeeded!
- Computational Fluid Dynamics (CFD) is the largest user of high-performance computing in engineering.[citation needed]
- Connecting those might provide interesting insights about the architecture, and as far as I know, nobody did it before.

- (Of course, it might also have to do with me needing a master thesis to finish my degree, HH having access to the Parallella systems, and Volvo Penta being interested in CFD…)

I will talk about:

- Computational Fluid Dynamics
- Lattice Boltzmann algorithm
- Adapteva Epiphany and Parallella board
- Implementation
- Results
- Conclusion

Computational Fluid Dynamics

- uses numerical methods to analyze fluid flows
  → both gases and liquids are fluids
- widespread applications in aerodynamics, architecture, automotive, chemistry, meteorology, navy, …
- computationally very intensive
  → high-performance computing, parallelization, …
- focus on a single, particle-based algorithm
  → Lattice Boltzmann

Lattice Boltzmann algorithm (I)

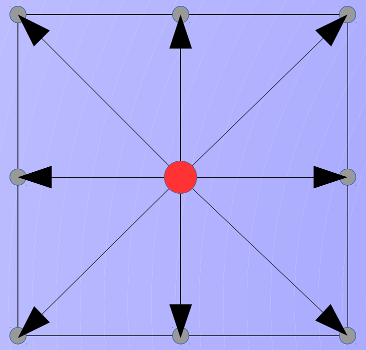
- based on Boltzmann equation, late 19th century:

$$\frac{\partial f}{\partial t} = \left.\frac{\partial f}{\partial t}\right|_{collision} + \left.\frac{\partial f}{\partial t}\right|_{diffusion} + \left.\frac{\partial f}{\partial t}\right|_{external}$$

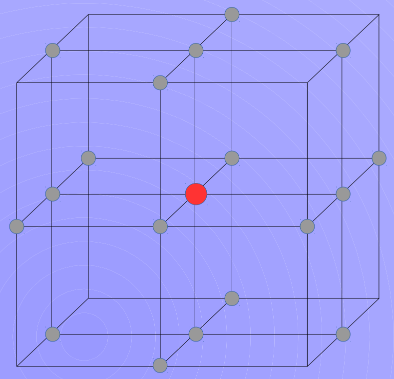
- f = f(**x**, **v**, t) describes the particle probability density in phase space (i.e. at specific position, velocity and time)
- collision term is particularly hard to solve
- *Particle distribution is only affected by collisions (particle-particle interactions), diffusion (particle movement), and external forces (environment), nothing else.*

Lattice Boltzmann algorithm (II)

- phase space f(**x**, **v**, t) is discretized (lattice models)
  → discrete positions, velocities and time (and angles)
- named D*m*Q*n* (m: dimensions, n: number of discrete velocities)
- focus on two models:



D2Q9 (single node)



D3Q19 (single node)

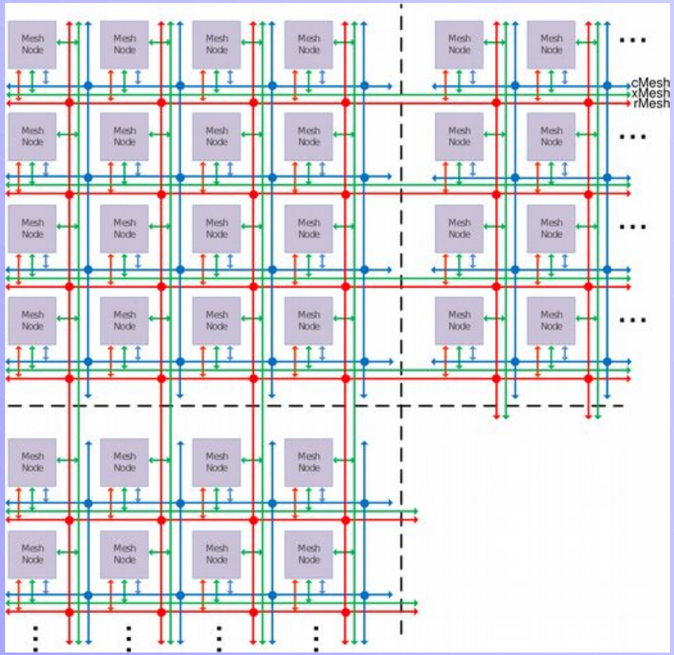
Adapteva Epiphany (I)

- two-dimensional mesh network-on-chip consisting of eCore processor nodes
- low power (16 cores @ 800 MHz < 1W)
- single shared, flat 32-bit address space
- 1 MiB address space per node, 64x64 (=4096) nodes maximum

row	column	local address

bit 31 25 19 bit 0

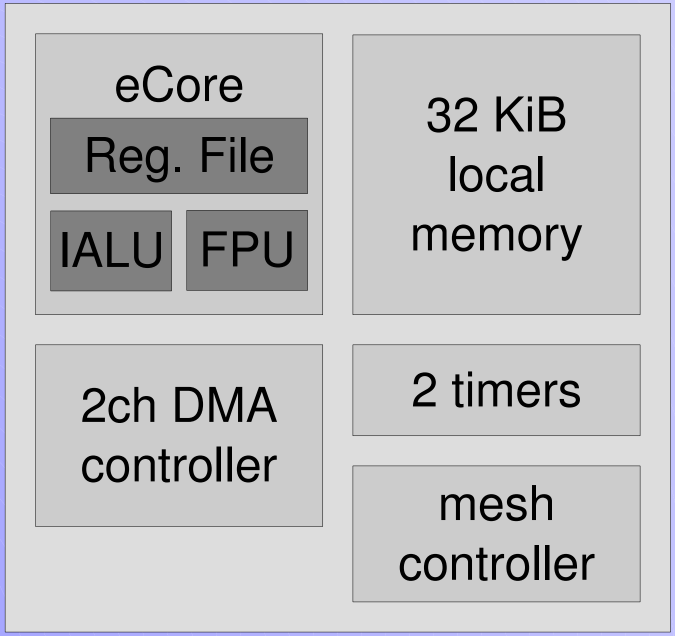
mesh address format



mesh structure
(source: Ep. Arch. Ref.

Adapteva Epiphany (II)

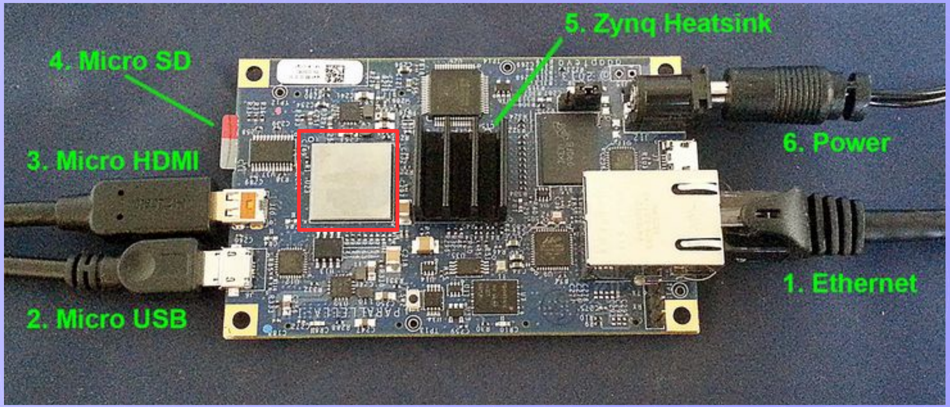
- eCores are 32-bit RISC processors with IALU (integer) and FPU (float) → single-precision FPU only
- only 32 KiB local memory per node, divided into independent 8 KiB-banks
- timers allow counting of events, allowing clock-cycle precise runtime measurements



mesh node

Parallella-16 board

- currently available “reference” platform for Epiphany arch
- Xilinx Zynq (1 GHz, dual-core ARM Cortex-A9) as host
- 16-core Epiphany E16G3 chip connected using FPGA logic
- 32 MiB of (Epiphany-)external shared memory



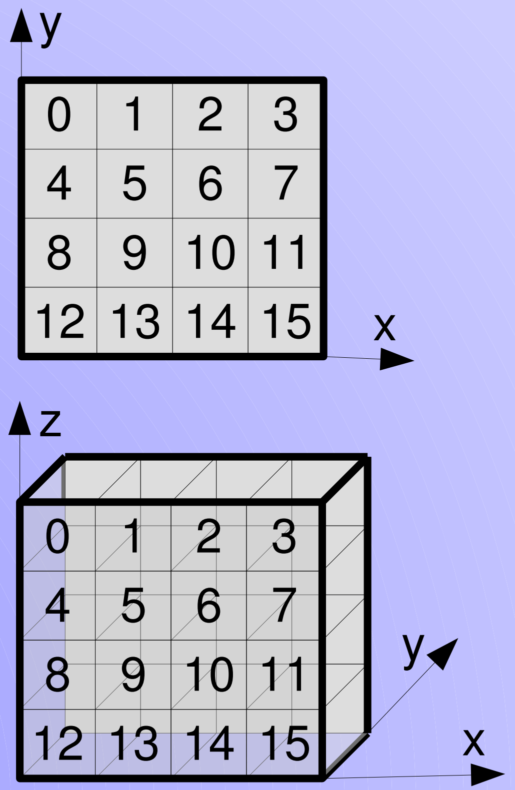
Parallella-16 board
Epiphany chip is marked red
(source: www.parallella.org)

Implementation (I)

- D2Q9 and D3Q19 implementations completely separate
- each implementation consists of two applications
- host application:
  - single-threaded ARM Linux application running on the Zynq
  - loads eCores with code and starts them
  - reads lattice data (results) from shared memory
  - creates density/velocity grayscale images and GIF animations
  - writes lattice data and time measurements to ASCII files

Implementation (II)

- Epiphany application:
  - single-threaded, but running on all active eCores simultaneously
  - works on a part of the lattice (*block*), which is always kept in local memory
  - after iteration, result may be copied to shared memory (→ to the host)
  - only next-neighbor communication (except for shared memory)
  - all cores run in lockstep, using barriers



blocking approaches
(bold: domain boundaries)

Results (I)

- very consistent results
- excellent scalability for the calculations (growing problem)
  - calculation times (almost) independent of number of cores
  - tiny 3% speed decrease* going from one to four active cores, but no further speed decrease (next-neighbor communication only)
- linear scalability for transmitting lattice to host
  - increased number of blocks (cores) → increased lattice size

(* 2D case, 24x24 blocksize, `-O3` optimization level)

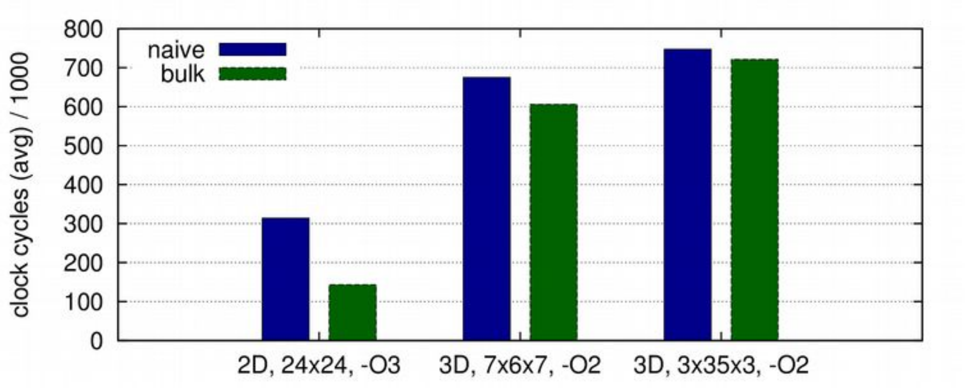
Results (II)

- good computational performance in 2D
  - 2.8 MLU/s* per core (45 MLU/s @ 16 cores)
  - in 2005, a single-core AMD Opteron was measured at 7 MLU/s, but in double precision
- much less impressive for 3D case
  - 0.34 MLU/s per core (5.4 MLU/s @ 16 cores)
  - in 2012, a single Nvidia Tesla achieved 650 MLU/s...
- comparison numbers were done on much larger lattices…

(* MLU/s: millions of lattice node updates per second)

Results (III)

- very small local memory, split into 8 KiB code / 24 KiB lattice
  - at most **682** (2D, ~26x26) or **323** (3D, ~7x6x7) nodes/core
  - bulk-based optimization ineffective in 3D (too few bulk nodes), but 2.2x speedup in 2D compared to naive approach
    → more with large blocks
  - maximum lattice size
    384 KiB @ 16 cores

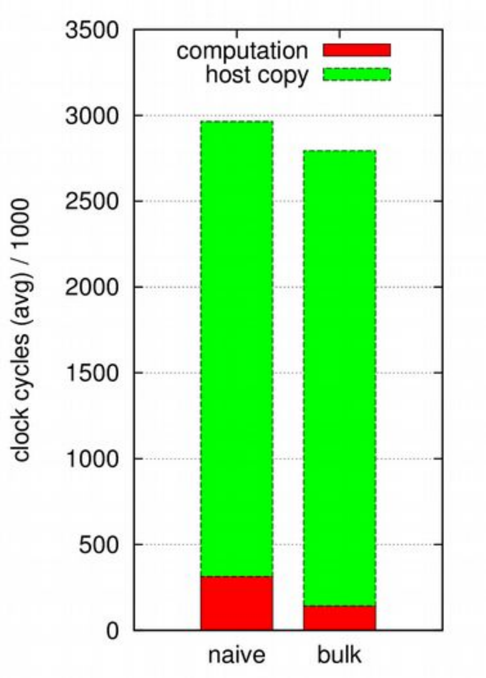


comparing naive / bulk-optimized algorithm

Results (IV)

- very small bandwidth to shared memory
  - measured 85 MiB/s
    (i.e. ~270 lattices/second @ 16-core)
  - theoretical maximum is 600 MiB/s, or
    200 MiB/s if non-optimal accesses*
    → not enough to stream lattice each iteration
  - no overlap possible between
    calculation and transmission…

(* but further limited by
current FPGA logic)



computation / host copy comparison
(2D, 24x24 block size, 16 cores)

Conclusion

- computations show excellent scalability, fair performance, and still room for optimization
- too small local memory, too little external bandwidth

  → currently *not suitable* for Lattice Boltzmann algorithm

- However:
  This work used the very first publicly available Epiphany chip.

The End.