INFORMATIK

TECHNISCHE UNIVERSITÄT KAISERSLAUTERN

# Introduction to High Performance Computing

## Summer Term


Dr. habil. Josef Schüle

# memory system

- **volatile memory**

  Power required to maintain stored information. Stored data is lost very rapidly after power interruption.
  - main memory
  - caches
- **non-volatile memory**

  Stored information may be retrieved after a power cycle.
  - hard disk
  - flash memory

## Memory and times:

- **time to transport data**
  - speed of front side bus ( ~ 1 cycle)
- **time to react after incoming request**
  - memory latency (~ 2 cycles of DIMM)
- **power refresh of DRAM after read/write operation**
  - DRAM not accessible inbetween
  - (~ 40-60 cycles of DIMM)

# Memory and times:

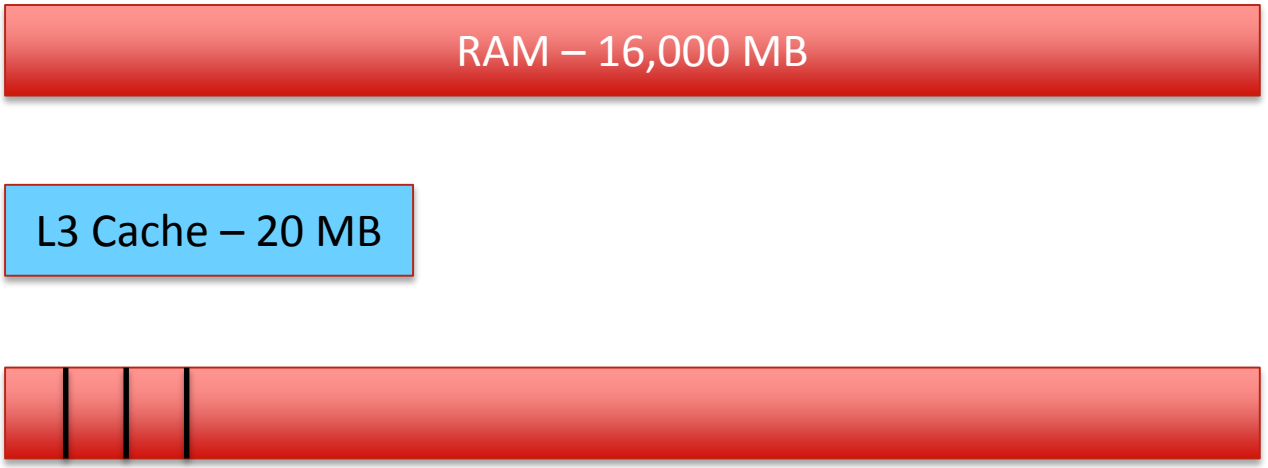
**Example: DDR3-1600 and 2,4 GHz for CPU, FSB with 64 bits bandwidth**

- **cycle time of DDR3-1600 is 800 MHz**
- **1st 64 bits of a vector x need ~42 cycles**
  **x[0], x[1]**
- **next 64 bits need ~2 cycles**
  **x[2], x[3]**
- **all subsequent 64 bits need ~2 cycles**
  **x[4] …..**

# Cache policies

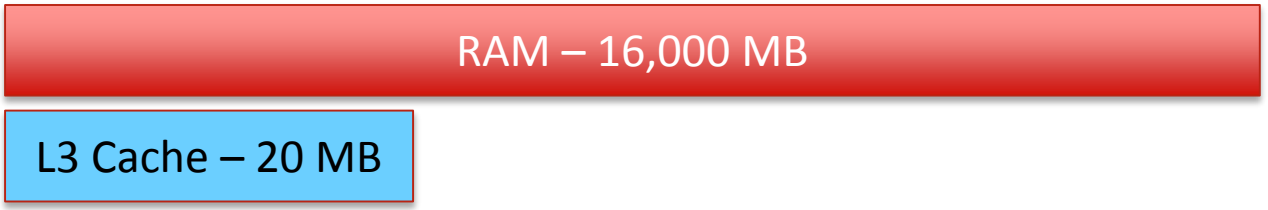
- **Replacement policies**
  - direct mapped
  - set associative
  - fully associative
- **Writing policies**
  - write back
  - write through
- **Levels, sizes, cache line sizes**

# direct mapped




- slices with 20 MB each
- Adress in cache = size of RAM modulo size of Cache
- Each memory adress has a single place to be put in cache.

# set associative

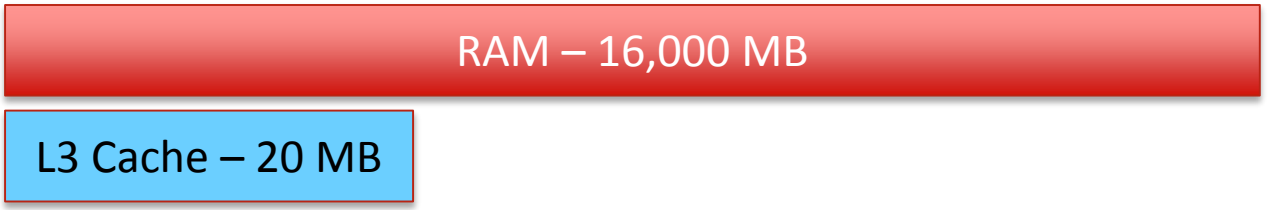



L3 consists of 20 sets each 1 MB in size

- slices with 1 MB each
- Adress in cache = size of RAM modulo size of cache set size
- Each memory adress has 20 places to be put in cache.
- Last recently is replaced.

/sys/devices/system/cpu/cpu0/cache/index3

# fully associative




- Adress in cache in no relation to RAM adress
- Each memory adress has many places to be put in cache.
- Last recently is replaced.

# replacement policies

- **direct mapped or set associative caches in practice**
- **fully associative has least bank conflicts and cache misses**
- **the more sets the better**

Elwe (head3):
L3: 20 sets consisting of 16384 lines with 64 bits width for a total of 20480K in size
L2: 8 sets, 512 lines with 64 bits (256K)
L1 – Instruction: 8 sets, 64 lines with 64 bits (32K)
L1 – Data: 8 sets, 64 lines with 64 bits (32K)

# writing policies

- **write through**

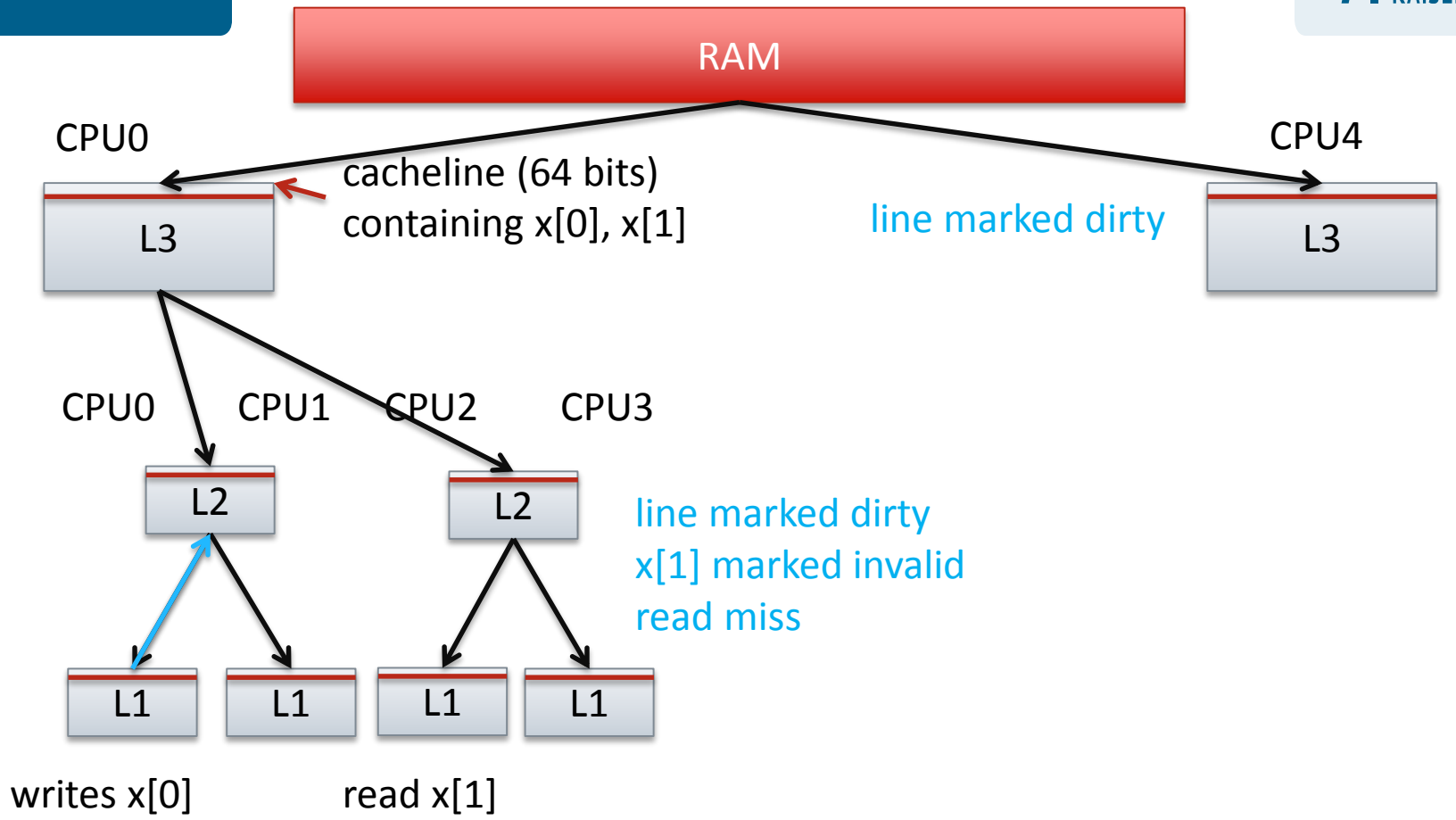

  - cache line is marked as invalid
  - consistency is simple

- **write back**


  - line may be written to RAM after serveral writes to it
  - consistency is violated

# cache coherency

- only one valid copy of a memory location does exist
- cache consistency: This copy is located in main memory

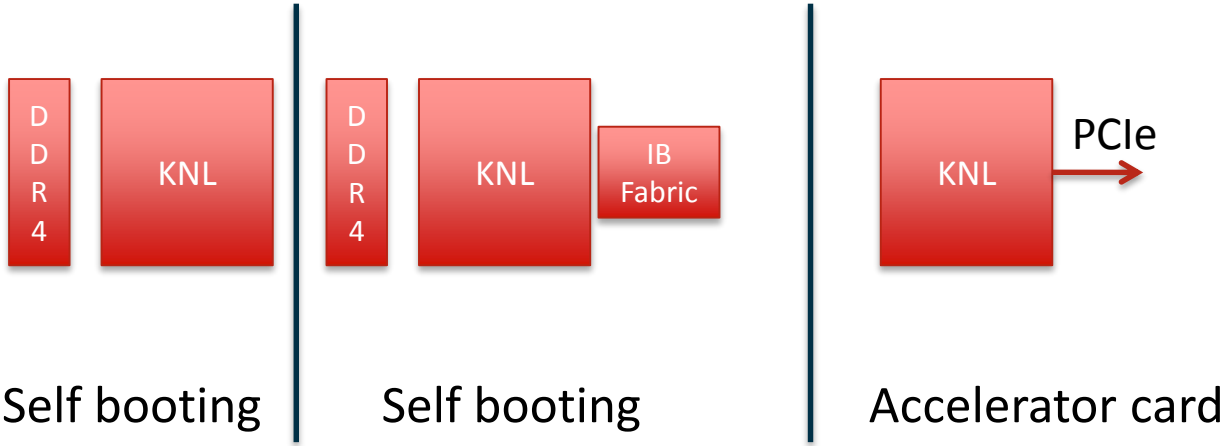

RAM
CPU0
CPU4
cacheline (64 bits)
containing x[0], x[1]
line marked dirty
L3
L3
CPU0
CPU1
CPU2
CPU3
L2
L2
line marked dirty
x[1] marked invalid
read miss
L1
L1
L1
L1
writes x[0]
read x[1]

# caviats with shared memory

- caches help to read data
  - once read into cache
  - multiple times read from cache
- caches complicate writing of data
  - either (expensive) direct update of memory
  - or (expensive) invalidation and read misses if cache line is required by several CPUs
- caches require a cache coherency protocol
- optimal data layout:
  - read neighbouring data
  - write distant data

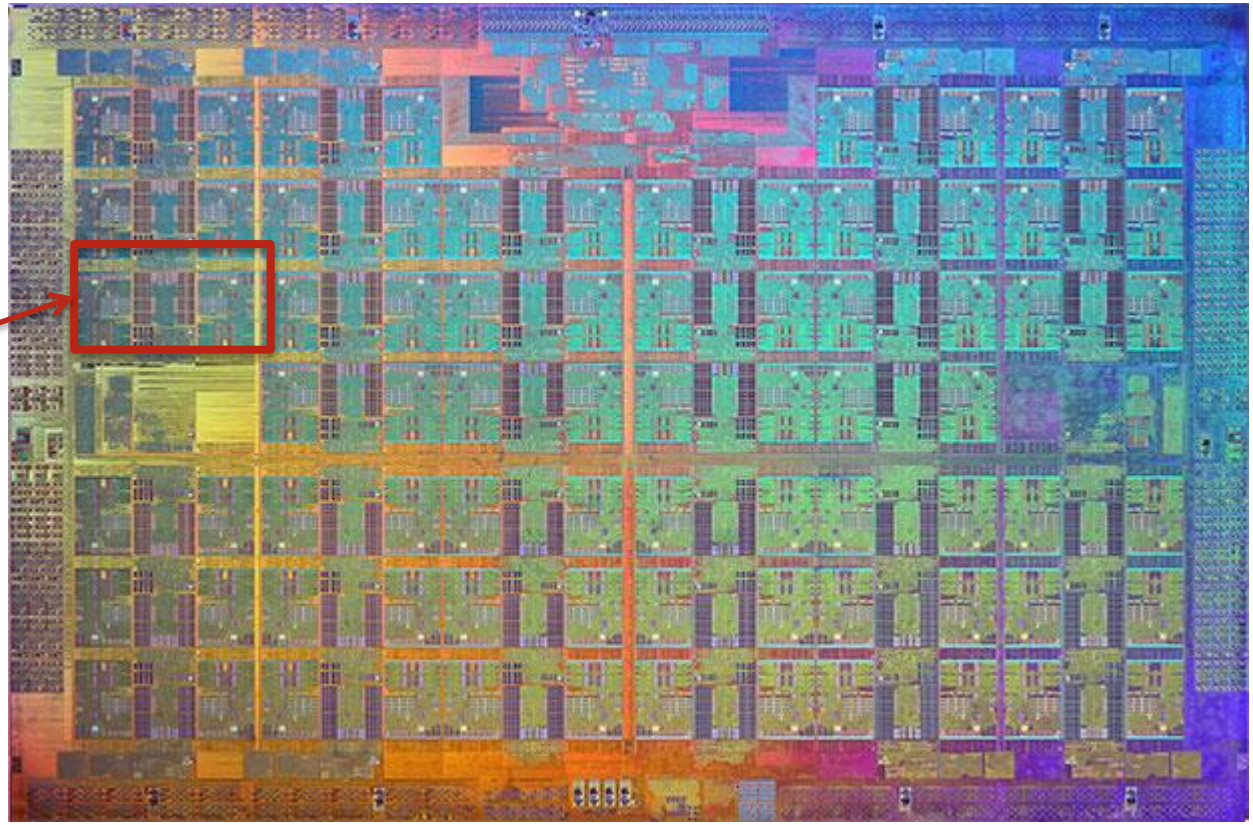
# Knights Landing

Flavours:



# Knights Landing

Building block



Source: Intel

# Knights Landing


Source: Intel

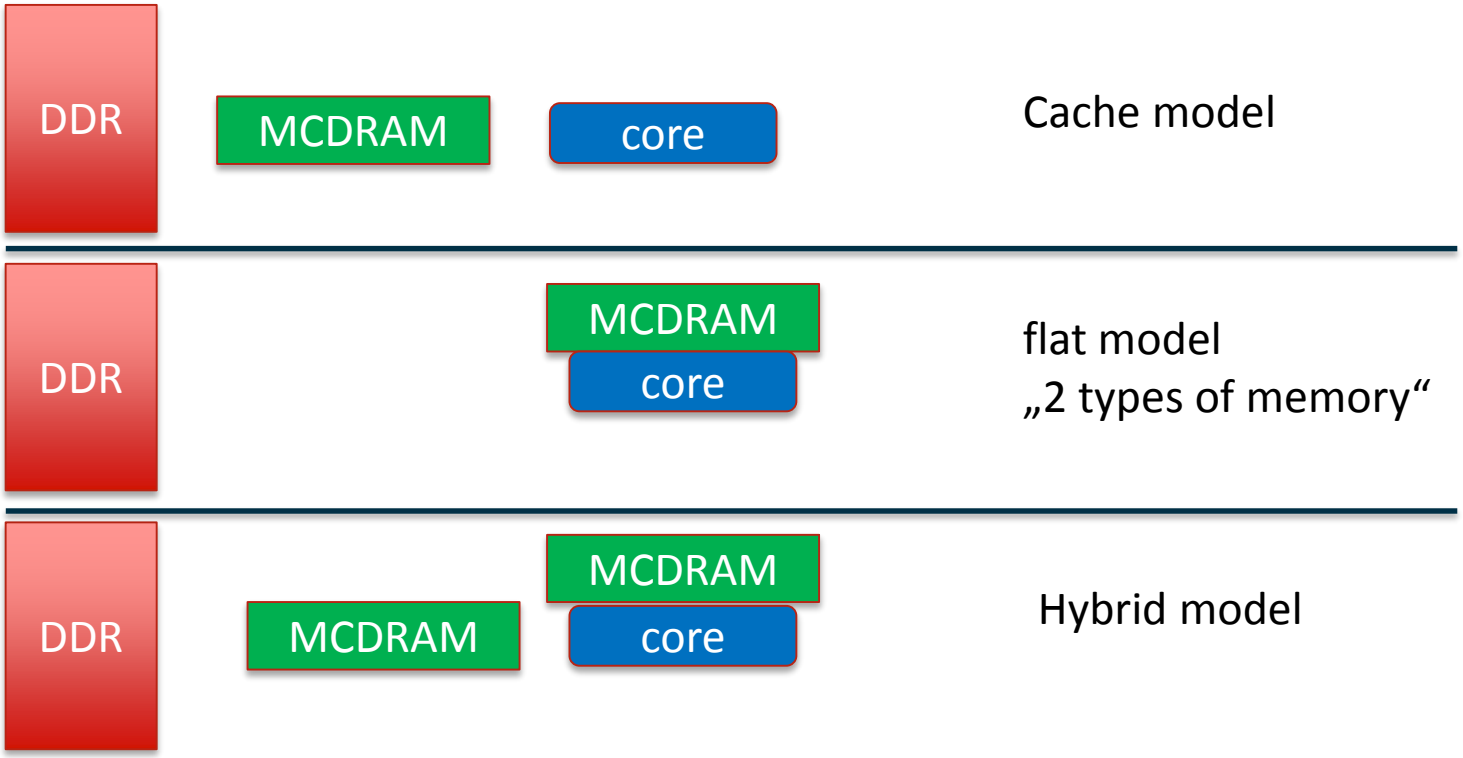
Each core with two AVX512 – each16 floats or 8 double (VPU)
L2 with 16 lines (32 bit wide)

Question:

- How do 16 lines transport data to 1 VPU?
- How do 16 lines transport data to 2VPU2?

# Knights Landing

## Local memory modes (16 GB MCDRAM):



# Knights Landing - MCDRAM

- multi channel provide more bandwidth than DDR (400+ GB/s, latencies are a bit higher)
- physically: DRAM in 3D
- less power consumption

# Thanks for your attention