332
Advanced Computer Architecture

Discussion questions for Knights Landing

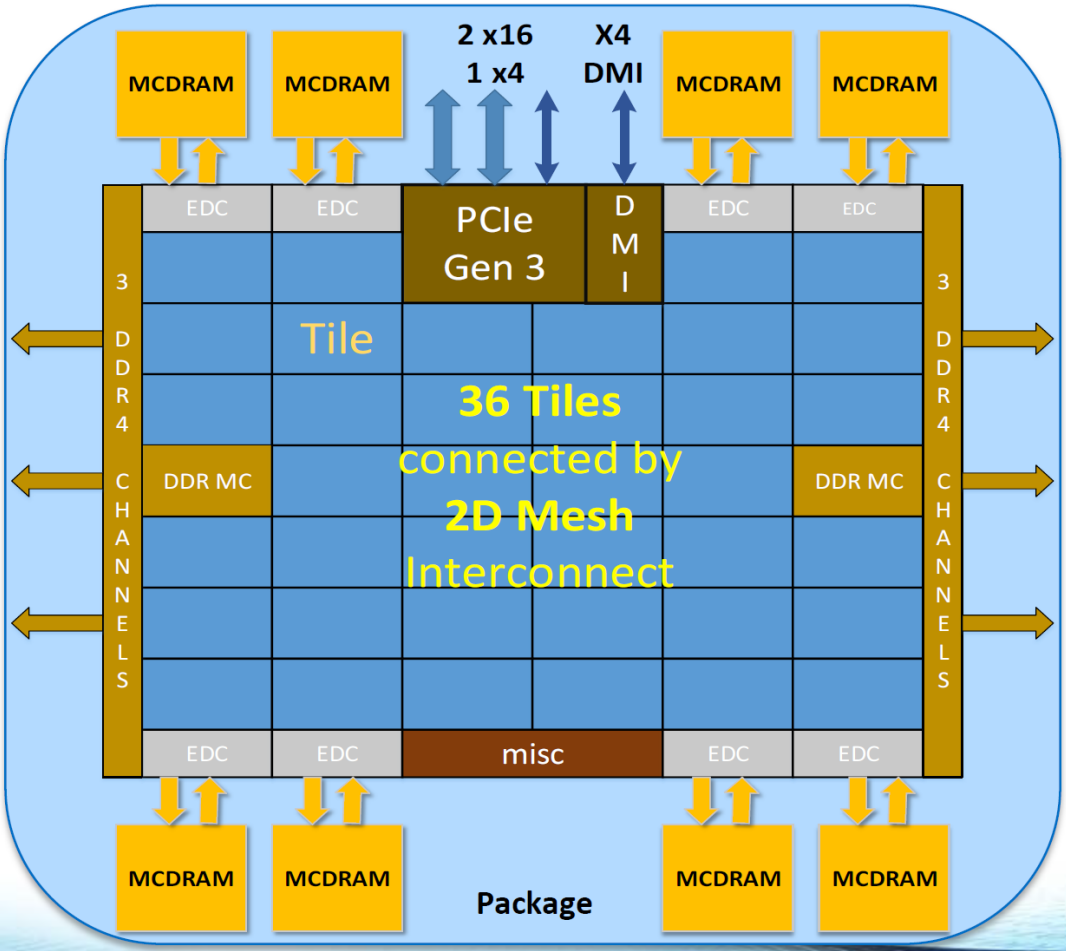
March 2017

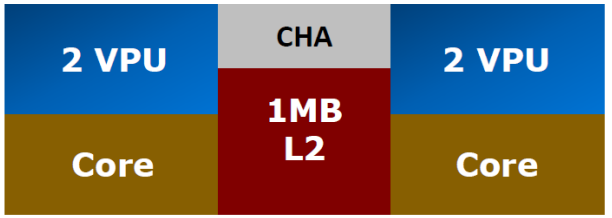
Paul H J Kelly

This presentation includes selected slides from Avinash Sodani's presentation at HotChips 27 (Aug 2015), "Knights Landing (KNL): 2nd Generation Intel® Xeon Phi™ Processor

http://www.hotchips.org/wp-content/uploads/hc_archives/hc27/HC27.25-Tuesday-Epub/HC27.25.70-Processors-Epub/HC27.25.710-Knights-Landing-Sodani-Intel.pdf

Knights Landing Overview





Chip: 36 Tiles interconnected by **2D Mesh**

Tile: 2 Cores + 2 VPU/core + 1 MB L2

Memory: MCDRAM: 16 GB on-package; High BW

DDR4: 6 channels @ 2400 up to 384GB

IO: 36 lanes PCIe Gen3. 4 lanes of DMI for chipset

Node: 1-Socket only

Fabric: Omni-Path on-package (not shown)

Vector Peak Perf: 3+TF DP and 6+TF SP Flops

Scalar Perf: ~3x over Knights Corner

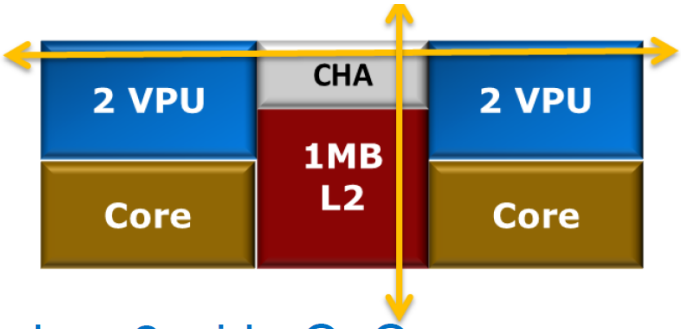
Streams Triad (GB/s): MCDRAM : 400+; DDR: 90+

Source Intel: All products, computer systems, dates and figures specified are preliminary based on current expectations, and are subject to change without notice. KNL data are preliminary based on current expectations and are subject to change without notice. 1Binary Compatible with Intel Xeon processors using Haswell Instruction Set (except TSX). 2Bandwidth numbers are based on STREAM-like memory access pattern when MCDRAM used as flat memory. Results have been estimated based on internal Intel analysis and are provided for informational purposes only. Any difference in system hardware or software design or configuration may affect actual performance.

KNL Tile:

2 Cores, each with 2 VPU

1M L2 shared between two Cores



Core: Changed from Knights Corner (KNC) to KNL. Based on 2-wide OoO Silvermont™ Microarchitecture, but with *many* changes for HPC.

4 thread/core. Deeper OoO. Better RAS. Higher bandwidth. Larger TLBs.

2 VPU: 2x AVX512 units. 32SP/16DP per unit. X87, SSE, AVX1, AVX2 and EMU

L2: 1MB 16-way. 1 Line Read and ½ Line Write per cycle. Coherent across all Tiles

CHA: **C**aching/**H**ome **A**gent. Distributed Tag Directory to keep L2s coherent. MESIF protocol. 2D-Mesh connections for Tile

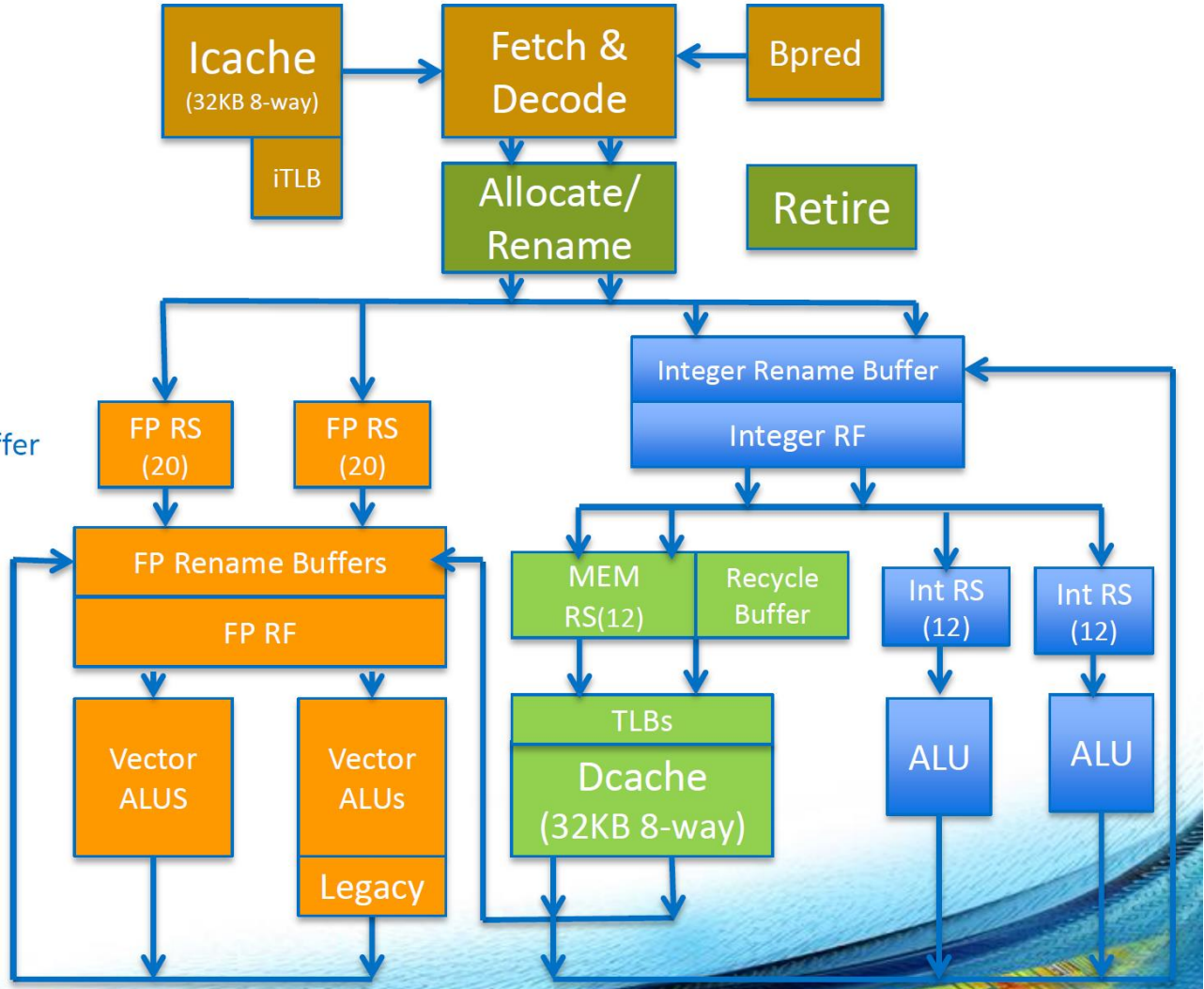
Core & VPU

- Out-of-order core w/ 4 SMT threads
- VPU tightly integrated with core pipeline

- 2-wide Decode/Rename/Retire
- ROB-based renaming. 72-entry ROB & Rename Buffers
- Up to 6-wide at execution
- Int and FP RS OoO.
- MEM RS inorder with OoO completion. Recycle Buffer holds memory ops waiting for completion.
- Int and Mem RS hold source data. FP RS does not.

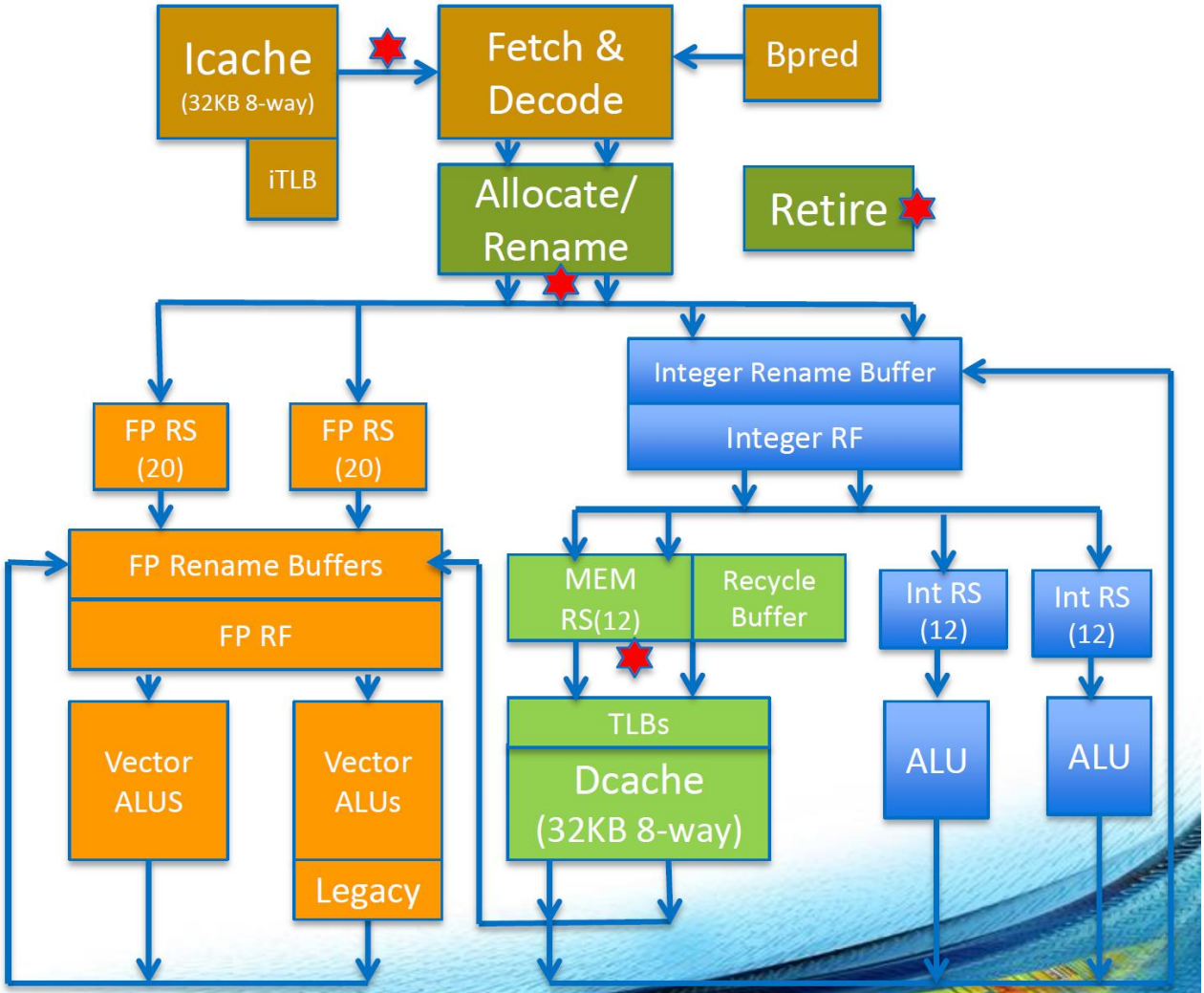
- 2x 64B Load & 1 64B Store ports in Dcache.
- 1st level uTLB: 64 entries
- 2nd level dTLB: 256 4K, 128 2M, 16 1G pages

- L1 Prefetcher (IPP) and L2 Prefetcher.
- 46/48 PA/VA bits
- Fast unaligned and cache-line split support.
- Fast Gather/Scatter support



Threading

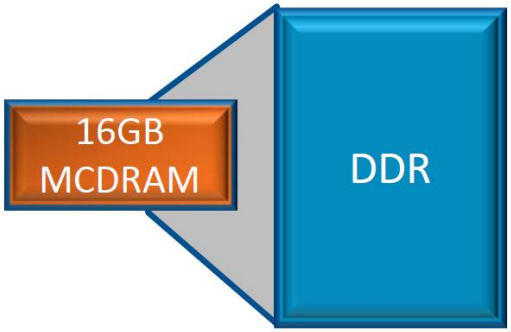
- 4 Threads per core. Simultaneous Multithreading.
- Core resources **shared** or **dynamically repartitioned** between active threads
  - ROB, Rename Buffers, RS: Dynamically partitioned
  - Caches, TLBs: Shared
  - E.g., 1 thread active → uses full resources of the core
- Several Thread Selection points in the pipeline. (✸)
  - Maximize throughput while being fair.
  - Account for available resources, stalls and forward progress



Memory Modes

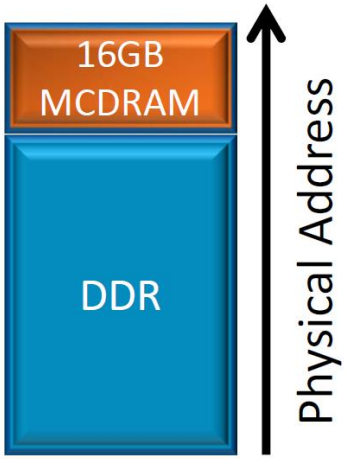
Three Modes. Selected at boot

Cache Mode



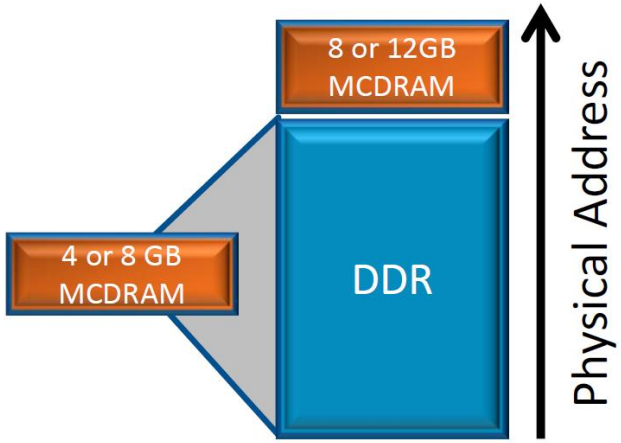
- SW-Transparent, Mem-side cache
- Direct mapped. 64B lines.
- Tags part of line
- Covers whole DDR range

Flat Mode



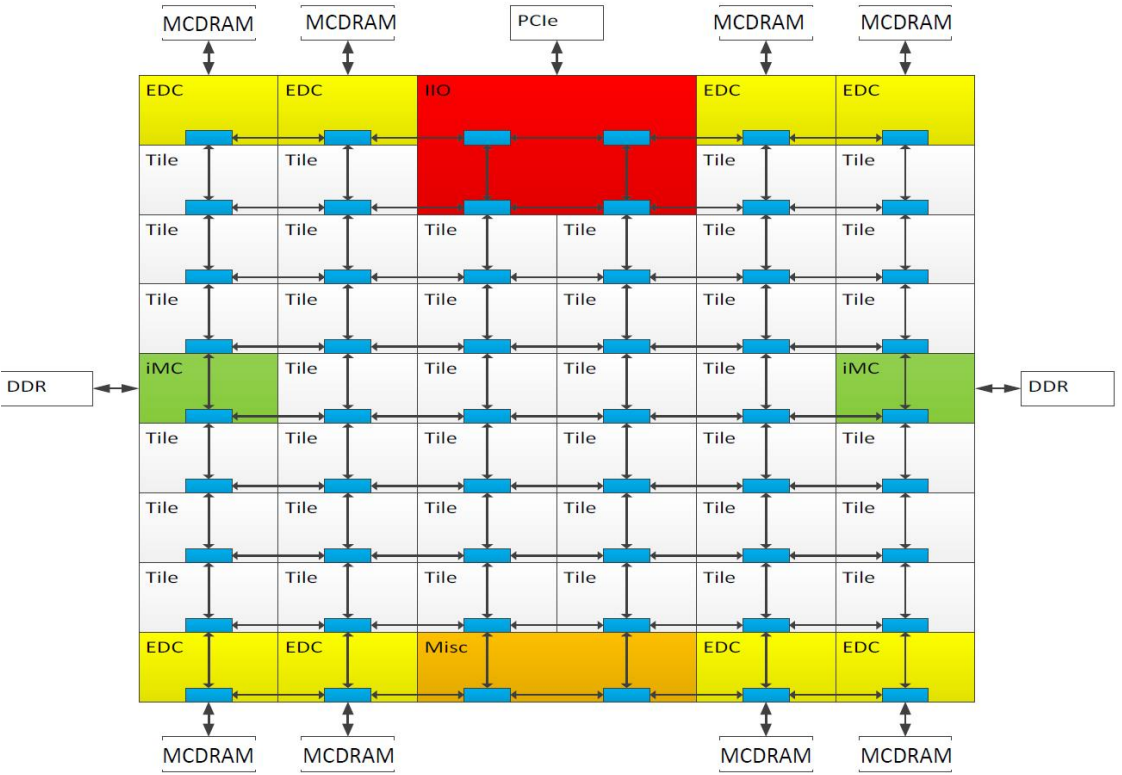
- MCDRAM as regular memory
- SW-Managed
- Same address space

Hybrid Mode



- Part cache, Part memory
- 25% or 50% cache
- Benefits of both

KNL Mesh Interconnect



Mesh of Rings

- Every row and column is a (half) ring
- YX routing: Go in Y → Turn → Go in X
- Messages arbitrate at injection and on turn

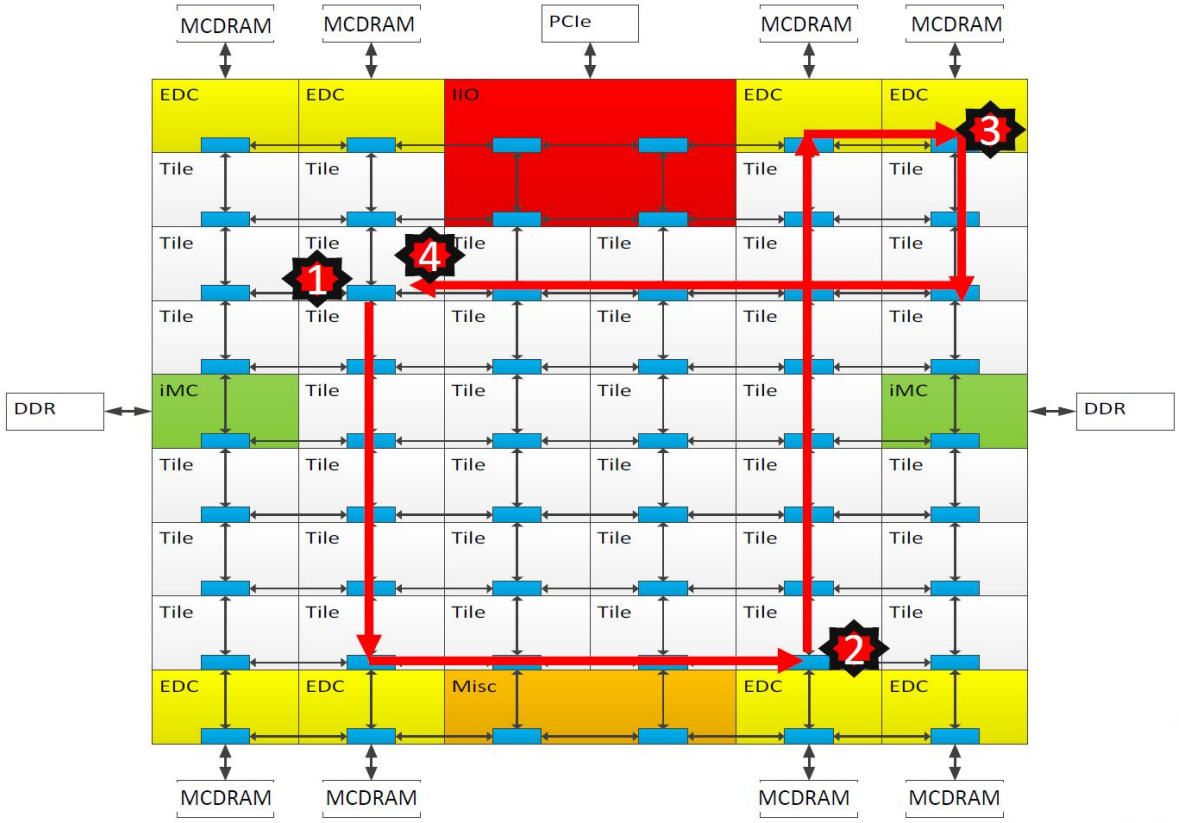
Cache Coherent Interconnect

- MESIF protocol (F = Forward)
- Distributed directory to filter snoops

Three Cluster Modes

(1) All-to-All (2) Quadrant (3) Sub-NUMA Clustering

Cluster Mode: All-to-All



Address uniformly hashed across all distributed directories

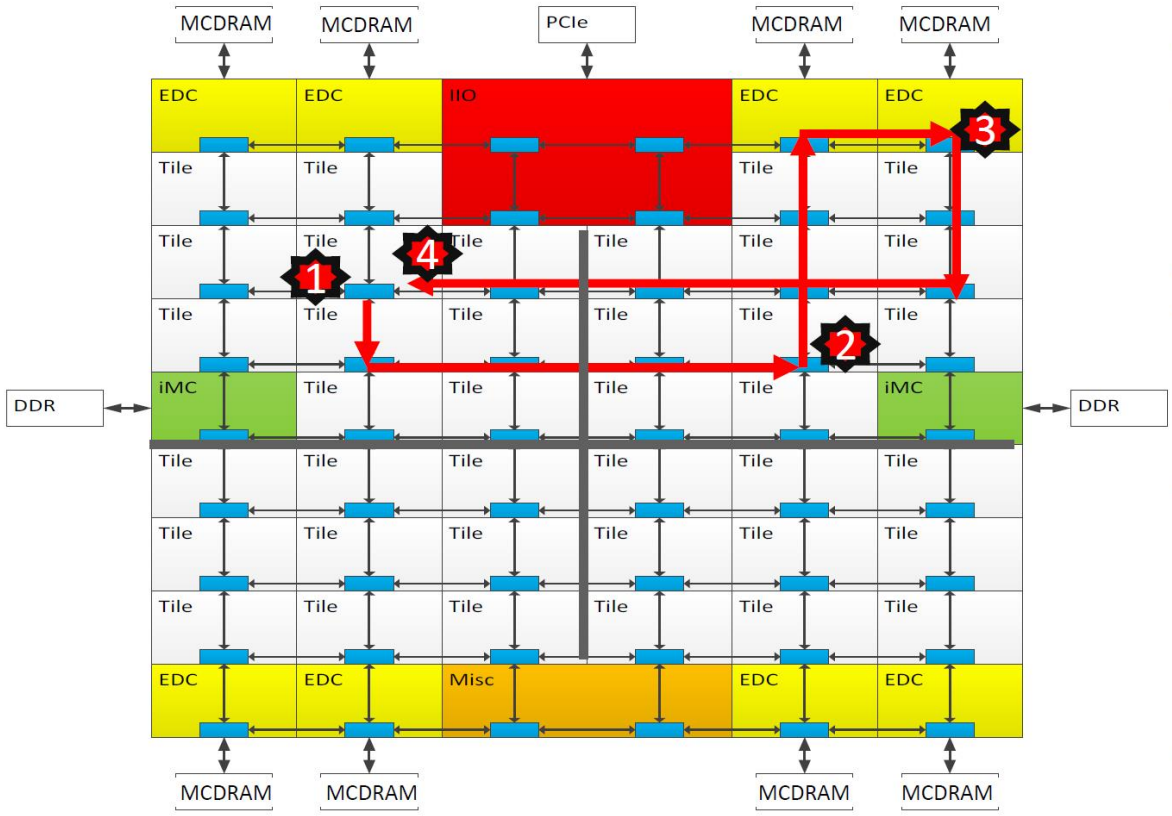
No affinity between Tile, Directory and Memory

Most general mode. Lower performance than other modes.

Typical Read L2 miss

1. L2 miss encountered
2. Send request to the distributed directory
3. Miss in the directory. Forward to memory
4. Memory sends the data to the requestor

Cluster Mode: Quadrant



Chip divided into four virtual Quadrants

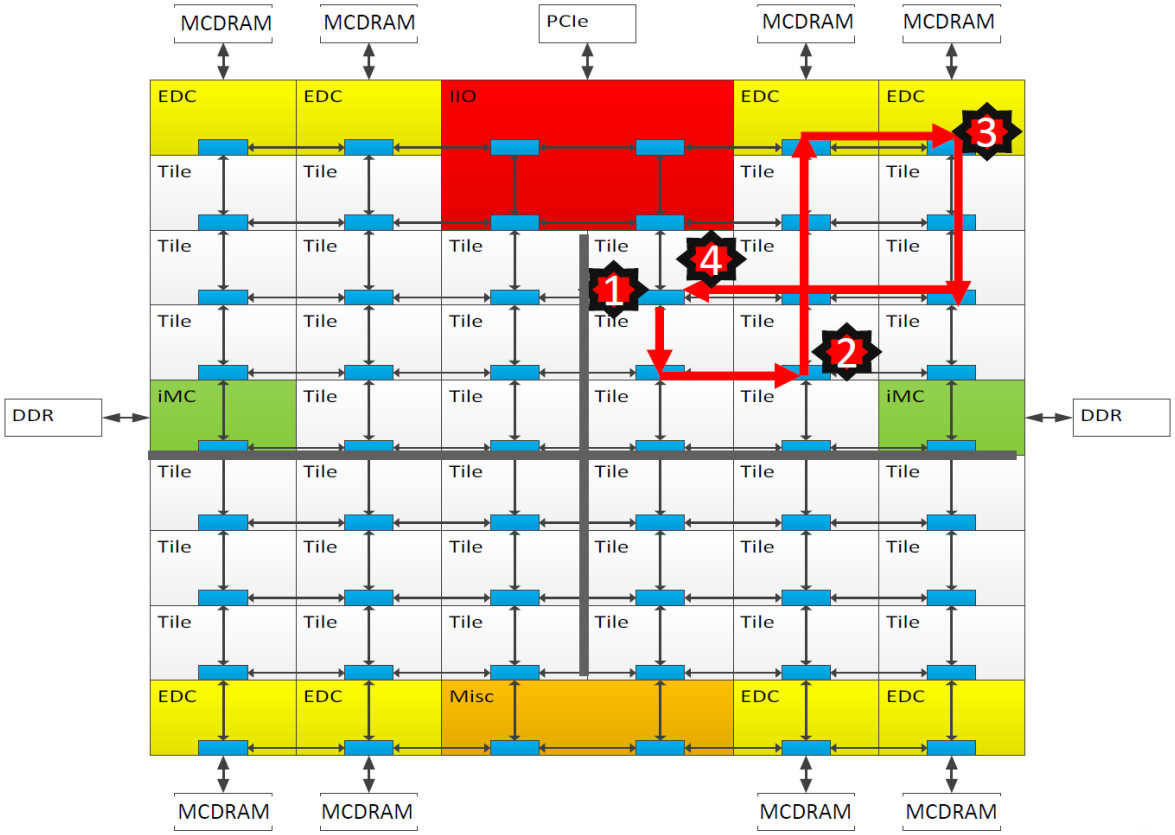
Address hashed to a Directory in the same quadrant as the Memory

Affinity between the Directory and Memory

Lower latency and higher BW than all-to-all. SW Transparent.

1) L2 miss, 2) Directory access, 3) Memory access, 4) Data return

Cluster Mode: Sub-NUMA Clustering (SNC)



1) L2 miss, 2) Directory access, 3) Memory access, 4) Data return

Each Quadrant (Cluster) exposed as a separate NUMA domain to OS.

Looks analogous to 4-Socket Xeon

Affinity between Tile, Directory and Memory

Local communication. Lowest latency of all modes.

SW needs to NUMA optimize to get benefit.

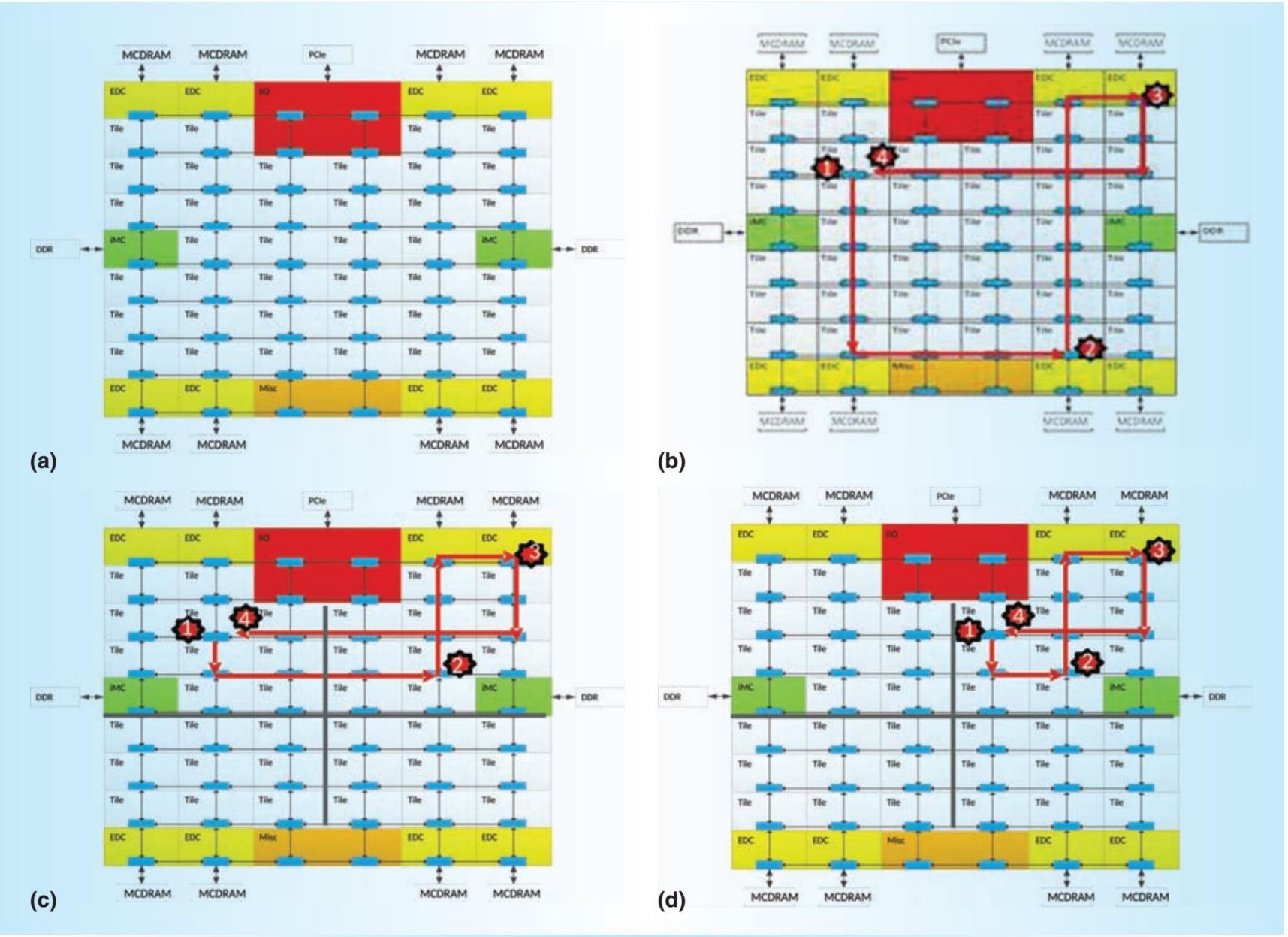


Figure 6. Example of an L2 miss flow: (1) tile with L2 miss encountered, (2) request sent to the tile with tag directory, (3) request forwarded to memory after miss in tag directory, (4) data read from memory and sent to the requester. (a) Mesh architecture, (b) all-to-all cluster mode, (c) quadrant cluster mode, and (d) sub-NUMA (nonuniform memory access) clustering (SNC) cluster mode.

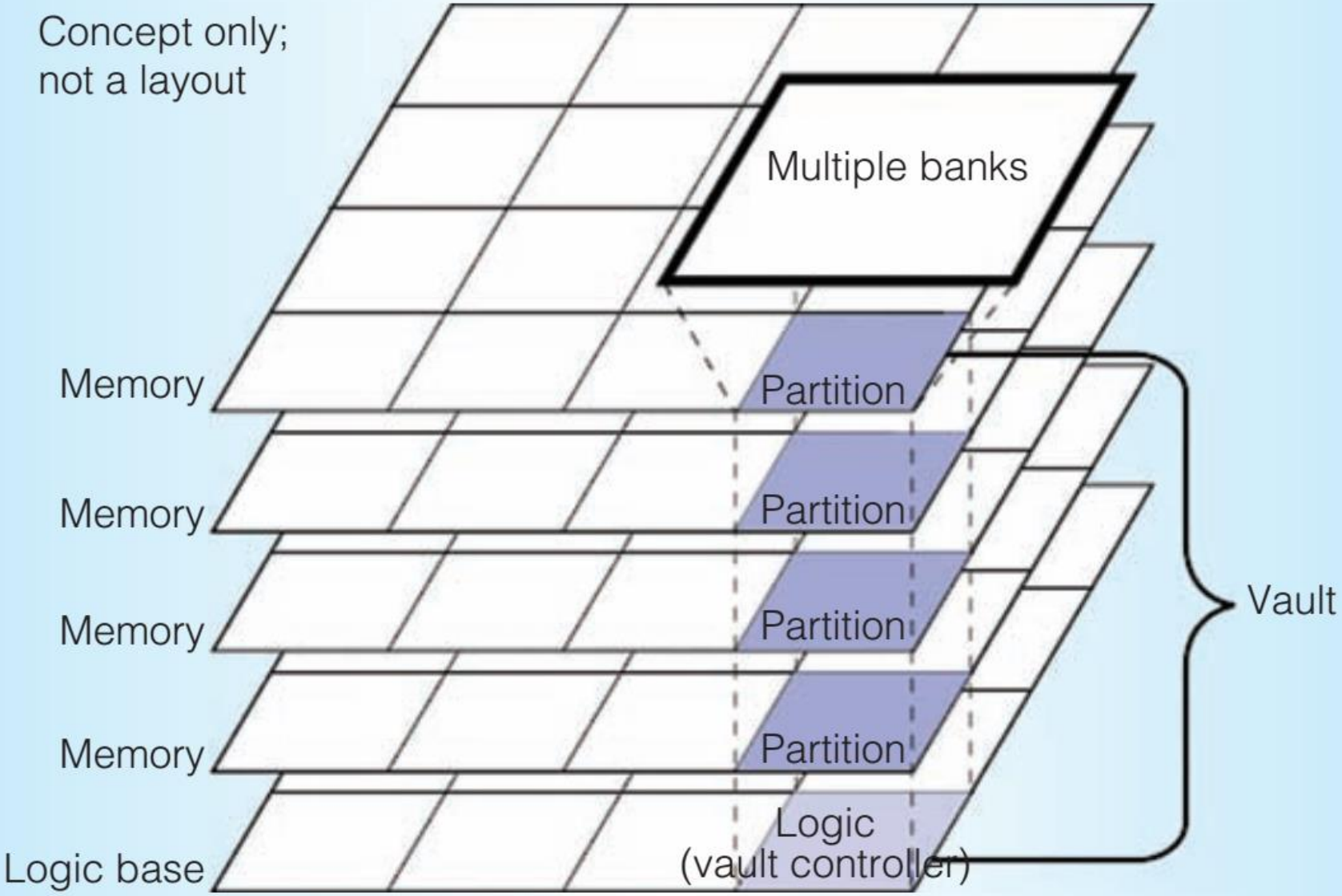


Figure 7. MCDRAM architecture. It shows a conceptual picture of DRAM dies stacked over a logic die and their organization.

- **Various questions around Fig 3, along the lines of**
  - **where is the physical register file,**

- **Various questions around Fig 3, along the lines of**
  - **why are rename buffers in front of reservation stations for integers, but behind for FP?**

- **Various questions around Fig 3, along the lines of**
  - **Is there a “common data bus” as in Tomasulo?**

- **Various questions around Fig 3, along the lines of**
  - **When are branch mispredictions discovered/initiated?**

- **What is the rationale for using MCDRAM in a cache mode – when would you do this, and why would it help?**

- **What is the distributed tag directory for, and why is it called a “tag” directory – isn’t it just a directory in the classical ccNUMA sense?**

- **What OS mechanism is needed to support explicit MCDRAM allocation?**

- **In what sense is the MCDRAM “multichannel”?**

- **How is the L2 hardware prefetcher keyed – using the requesting instruction's PC? But can L2 even see the PC?**