Master of Computer Applications

CAPOL403R01: Computer Organization & Architecture

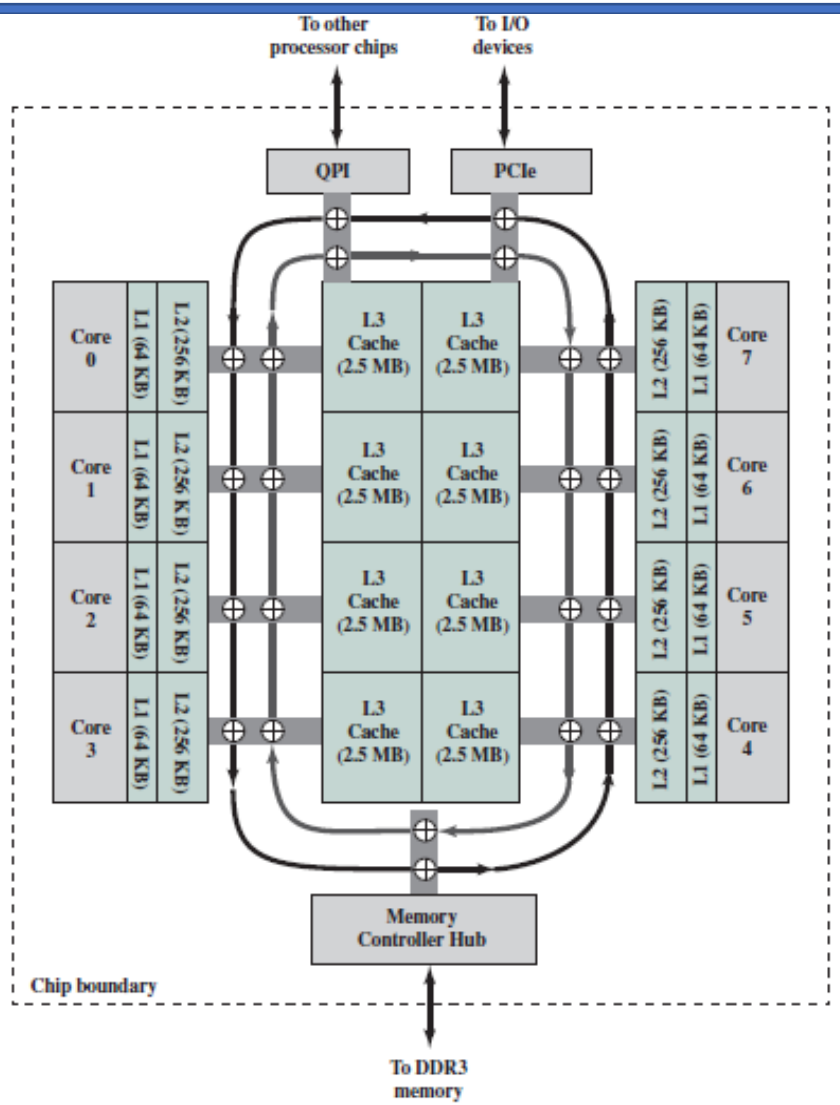
Unit V: Lecture 5
Direct Cache Access

Dr. D. MURALIDHARAN
School of Computing
SASTRA Deemed to be University

Why DCA?

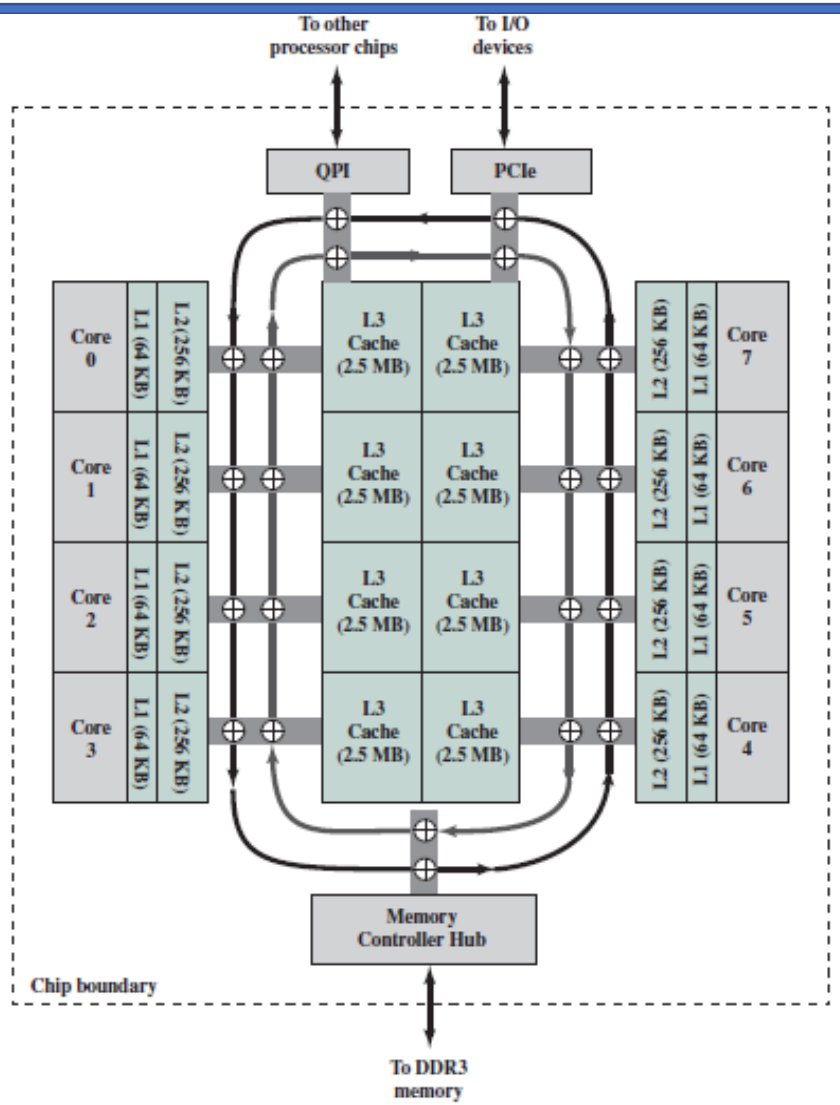
- DMA is unable to scale to meet the increased data rate demands
  - 100 Gbps Ethernet switches
  - Wi-Fi is in the Gigabit range
- DCA is considered as the solution
  - Enabling the I/O function to have direct access to the cache
  - Here, the cache refers to the last-level cache
  - Normally, this cache is shared by the cores

Intel Xeon Multicore Processor



- Many of the members of the Xeon family use a ring interconnect system
- The E5-2600/4600 can be configured with up to eight cores on a single chip
- Each core has dedicated L1 and L2 caches
- It has a 20MB shared L3 cache
- The L3 cache is divided into slices
  - one associated with each core
  - Each slice has its own cache pipeline
  - The requests can be sent in parallel to the slices.
- Each core can address the entire cache.

Intel Xeon Multicore Processor



- The ring is bidirectional.
- It interconnects cores, last-level cache, PCIe, and integrated memory controller (IMC).
- Ring logic
  - Ring agent: The components which are connected to the ring (QPI, PCI3, L3 and L2 cache)
  - Ring agents follow a distributed protocol
  - The protocol allotted time slots for the agent
  - The agent sends the data in its allotted time slot and in the direction of shorter distance of the destination
  - If more number of cores are added, multiple rings are used – each ring supports some number of cores

DMA use of the cache

- In traditional DMA operation, data are exchanged between main memory and an I/O device
- Consider Xeon uses DMA for memory output operation
  - An I/O driver running on a core would send an I/O command to the I/O controller
  - The command includes the location and size of the buffer in main memory
  - The I/O controller issues a read request
  - The request is routed to the memory controller hub (MCH)
  - MCH accesses the data on DDR3 memory
  - MCH puts it on the system ring for delivery to the I/O controller
    - one or more off-chip memory reads are required
    - The L3 cache is not involved in this transaction

DMA use of the cache

- Consider Xeon uses DMA for memory write operation
  - Data arrive from the I/O controller
  - Data are delivered over the system ring to the MCH
  - Data are written out to main memory
  - The MCH must also invalidate any L3 cache lines corresponding to the updated memory locations.
  - In this case, one or more off- chip memory writes are required.
  - If an application wants to access the new data, a main memory read is required

DMA use of the cache

- As Xeon E5-2600/4600 has a large amount of L3 cache, the technique can be refined
  - The I/O controller issues a read request
  - The MCH first checks to see if the data are in the L3 cache
    - This is likely to be the case, if an application has recently written data into the memory block to be output.
  - In hit, the MCH directs data from the L3 cache to the I/O controller
  - As no main memory accesses are needed, the performance is enhanced
  - This technique is NOT referred as direct cache access

Data communication

- Network traffic is transmitted in the form of a sequence of protocol blocks
  - These blocks are called as packets or protocol data units
- Normally, Ethernet protocol is used as link level protocol
  - Each arriving and departing block of data consists of an Ethernet packet
  - Ethernet packets contain as payload the higher-level protocol packet
- TCP/IP protocols are, normally, used as higher level protocols
  - Internet protocol is operating on top of the Ethernet
  - Transmission Control Protocol is operating on top of the IP
  - The Ethernet payload consists of a block of data with a TCP header and an IP header

Data communication…

- The I/O controller or network interface controller (NIC) creates Ethernet packets for outgoing traffic
- The I/O controller strips off the Ethernet information and delivers the TCP/IP packet to the host CPU during incoming traffic
- For both outgoing and incoming traffic, the core, main memory, and cache are all involved

Cache-Related Performance Issues

- Consider a DMA scheme
  - An application wishes to transmit data
  - It places that data in an application-assigned buffer in main memory
  - The core transfers this to a system buffer in main memory
  - The core then creates the necessary TCP and IP headers
  - These are also buffered in system memory
  - The packet is then picked up via DMA for transfer via the NIC
  - For incoming traffic, similar transfers between system and application buffers are required
  - This transferring activity engages not only main memory but also the cache

Cache-Related Performance Issues

- Two factors in this scenario degrade performance
  1. the core consumes valuable clock cycles in copying data between system and application buffers
  2. The core loses time waiting on memory reads and writes as the CPU speed is higher compared to memory speed

- As the data and protocol headers are constantly changing, cache does not help to enhance the performance
  - To enhance the performance, the cache must constantly be updated

Cache write techniques

- Write through
  - All write operations are made to main memory as well as to the cache
  - Mani memory is always valid
- Write back
  - The write operations updates only the cache
  - When an update occurs, a dirty bit associated with the line is set
  - when a block is replaced, it is written back to main memory if and only if the dirty bit is set
- DDIO uses the write- back strategy in the L3 cache

Cache miss during write

- Write allocate:
  - The required line is loaded into the cache from main memory.
  - Then, the line in the cache is updated by the write operation.
  - This scheme is typically used with the write-back method.
- Non-write allocate:
  - The block is modified directly in main memory.
  - No change is made to the cache.
  - This scheme is typically used with the write-through method.

DDIO – Intel Xeon's memory write

- If there is a cache hit, only the cache line is updated
  - The Intel literature refers to this as **write update**
- If there is a cache miss, the write operation occurs to a line in the cache that will not be written back to main memory
  - Subsequent writes update the cache line
  - No reference to main memory
  - No future action that writes this data to main memory
  - The Intel literature refers to this as **write allocate**
    - **This is not same as the term used in cache literature**

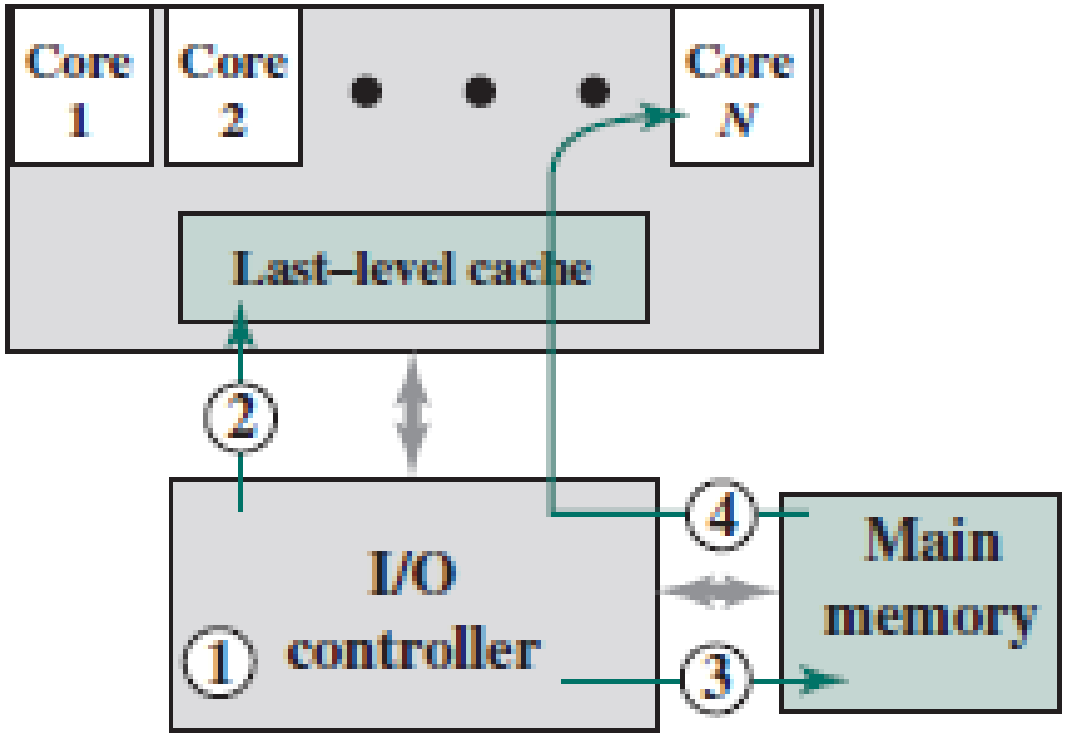
Direct Data IO (DDIO)

- For the specific function of protocol processing, the packet and packet descriptor information are accessed only once in the system buffer by the core.
  - For incoming packets, the core reads the data from the buffer and transfers the packet payload to an application buffer.
    - It has no need to access that data in the system buffer again.
  - Similarly, for outgoing packets, the core has placed the data in the system buffer
    - It has no need to access that data again.
  - Consider the I/O system were equipped with the capability of directly accessing the cache, both for input and output operations.
  - Then it would be possible to use the last-level cache instead of the main memory to buffer packets and descriptors of incoming and outgoing packets

DDIO – Memory input operation

DMA

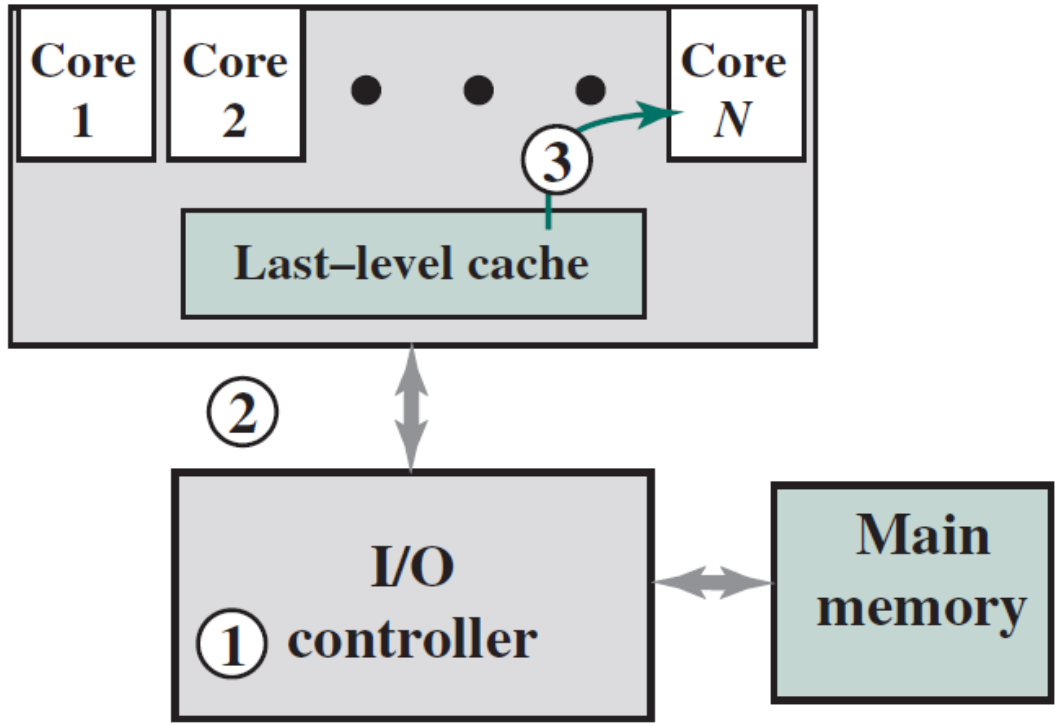
1. The NIC initiates a memory write
2. Then the NIC invalidates the cache lines corresponding to the system buffer
3. Next, the DMA operation is performed, depositing the packet directly into main memory
4. Finally, after the appropriate core receives a DMA interrupt signal, the core can read the packet data from memory through the cache



DDIO – Memory input operation

DCA

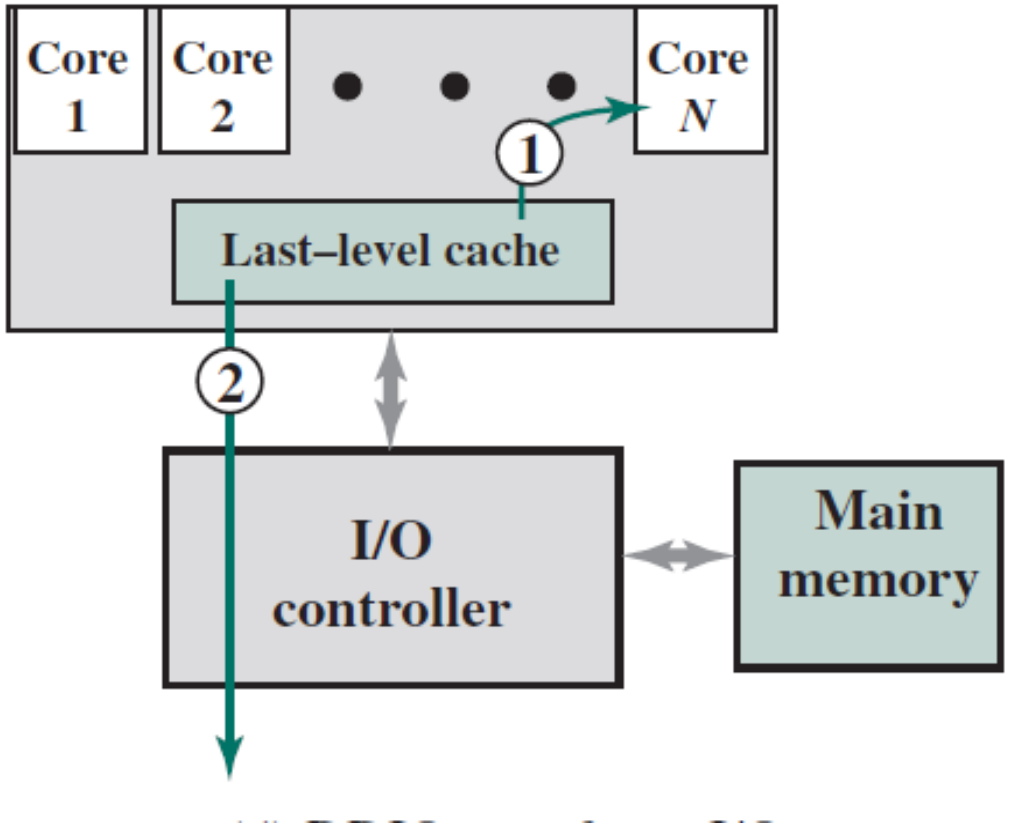
1. The NIC initiates a memory write
2. Then the NIC invalidates the cache lines corresponding to the system buffer and it deposits the data in the cache
3. Finally, after the appropriate core receives a DCA interrupt signal, the core can read the packet data from the cache



DDIO – Memory output operation

DCA

- The TCP/IP protocol handler creates the packet to be transmitted and stores it in allocated space in the L3 cache
  - But not in main memory
- The read operation initiated by the NIC is satisfied by data from the cache



Thank you