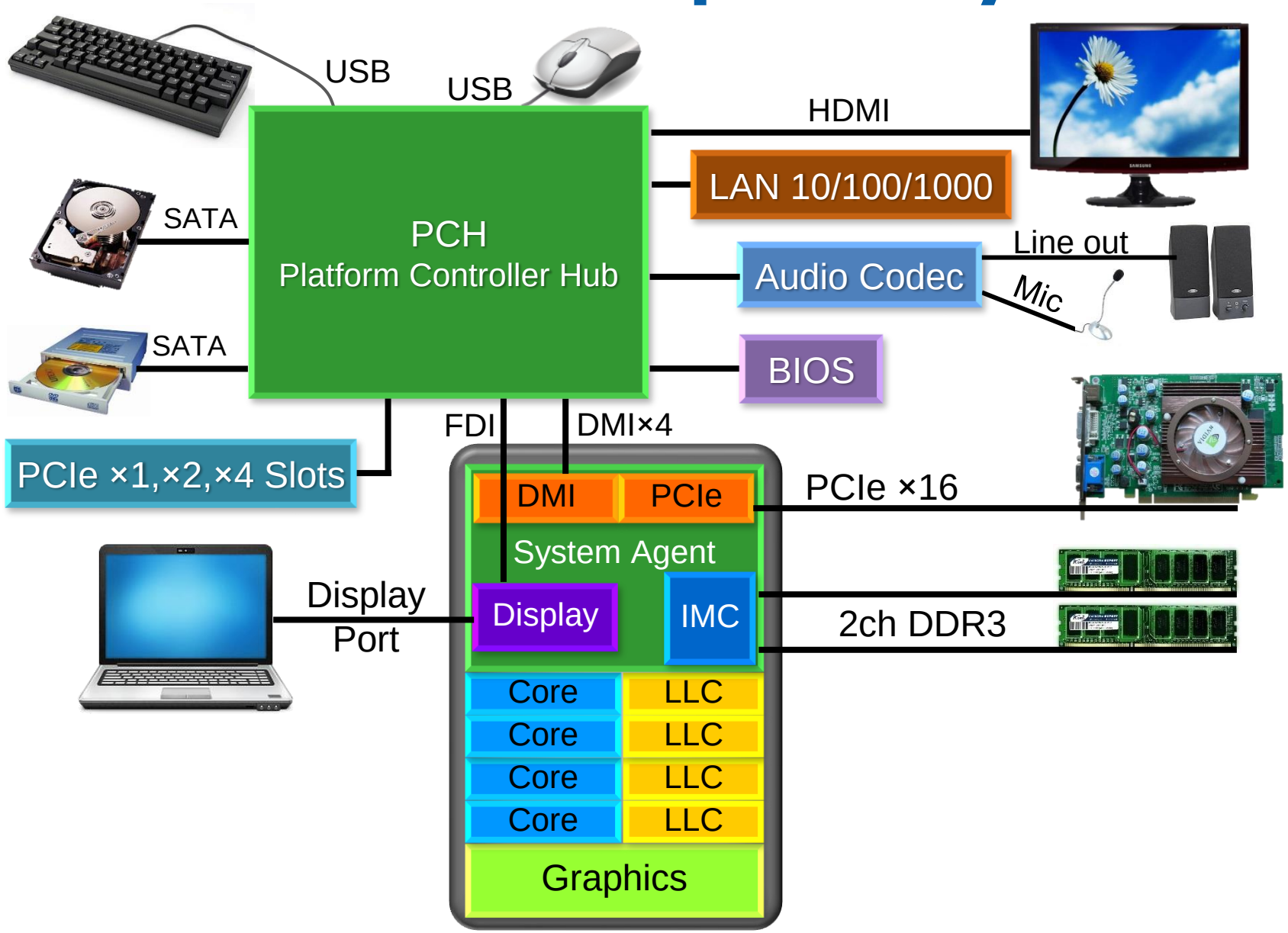
Computer Structure

SOC – System on a Chip

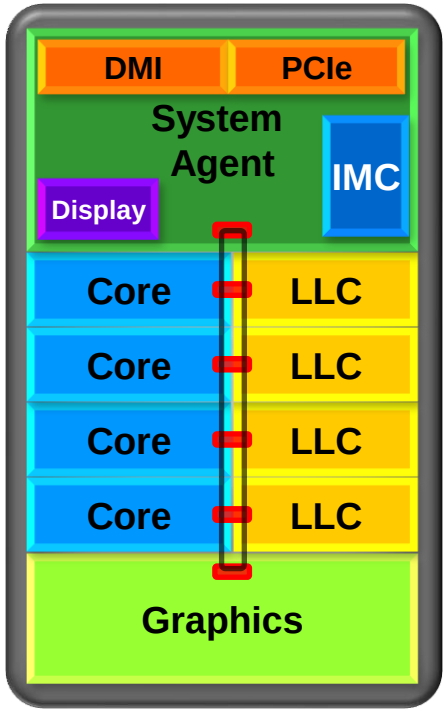
Lihu Rappoport And Adi Yoaz

Personal Computer System



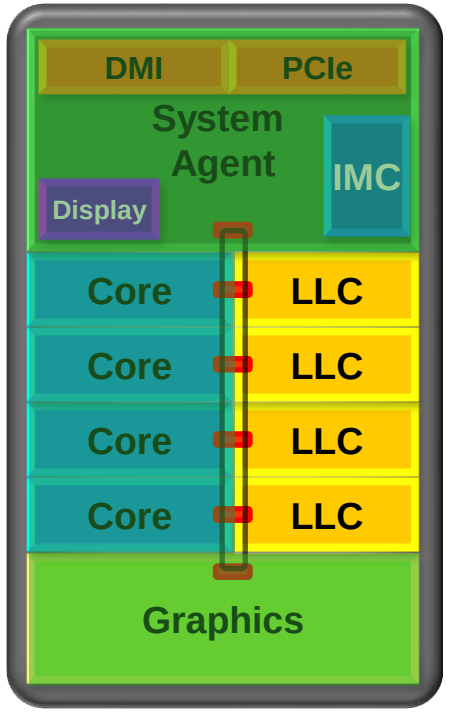
SOC – System On a Chip

- **Client SOC includes**
  - The System Agent, which handles Memory and I/O
    - PCI Express controller (for external GFX), Integrated Memory Controller, Display Engine
    - DMI – Direct Media Interface – connection to Platform Controller Hub (PHC)
  - Cores, where each core contains its own
    - L1 Data cache
    - L1 instruction cache
    - L2 cache (used for both data and instructions), also called Mid Level Cache (MLC)
  - L3 cache, also called Last Level Cache (LLC), split into slices
    - All slices accessed by all cores – used for load distribution
  - Internal Graphics Unit
  - High bandwidth ring bus
    - Connects between the Cores, LLC slices, Graphics, and System Agent



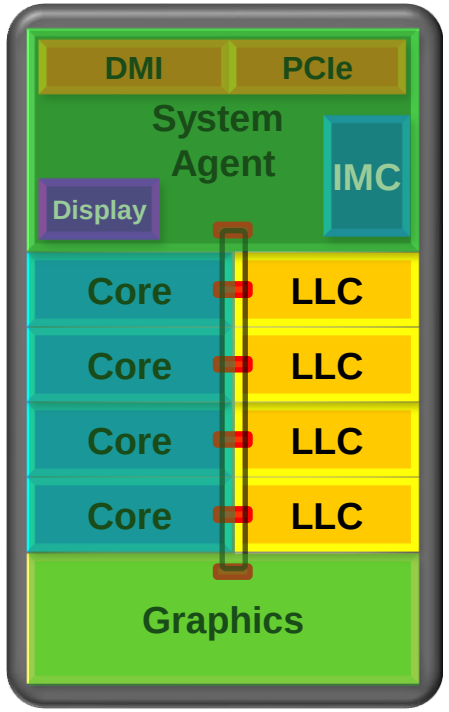
Last Level Cache – LLC

- **The LLC consists of multiple cache slices**
  - The number of slices is equal to the number of cores
  - Each slice can supply 32 bytes/cycle – a 64byte cache line fill takes two packets
- **Physical addresses are uniformly distributed among cache slices using a hash function**
  - The LLC acts as one shared cache
    - With multiple ports and BW that scales with the number of cores
    - Ring/LLC not likely to be a BW limiter
  - Prevents hot spots
  - A core can tell on which slice a requested is found according to the hash function
- **LLC is shared among all Cores, Graphics and Media**
  - GFX/media may in compete with cores on LLC



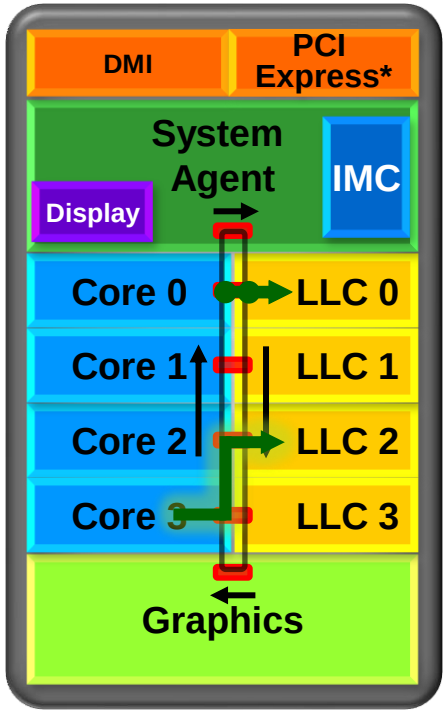
Last Level Cache – LLC (cont.)

- **LLC hit latency is ~30 cycles**
  - Depends on Core location relative to the LLC Slice (how far the request and the data need to travel on the ring)
- **Fetching data from LLC when another core has the data (in E/M states)**
  - Clean hit – data is not modified in the other core – 43 cycles
  - Dirty hit – data is modified in the other core – 60 cycles
- **LLC is fully inclusive of all core internal caches**
  - Eliminates unnecessary snoops to cores
  - Per core “Core Valid bit” indicates if the internal core caches need to be snooped for a given cache line
- **Traffic that cannot be satisfied by the LLC**
  - LLC misses, dirty line write-back, non-cacheable operations, MMIO/IO operations
  - Travels through the ring to the IMC



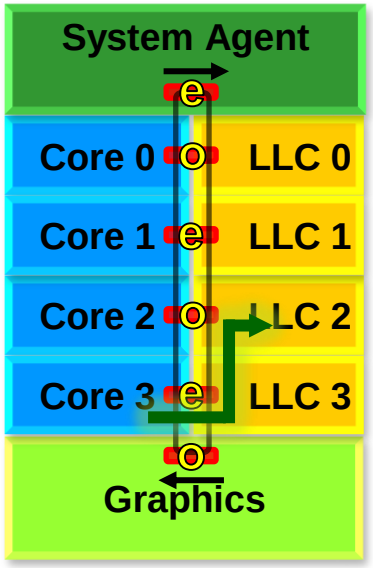
Scalable Ring Interconnect

- **High bandwidth scalable ring bus**
  - Interconnects Cores, Graphics, LLC and SA
  - 4 sub-rings: 32 Byte Data ring, Request ring, Acknowledge ring, and Snoop ring
  - Fully pipelined at core frequency/voltage: bandwidth, latency and power scale with cores
  - Ring wires run over LLC without area impact
  - Distributed arbitration, coherency and ordering
- **The ring is bidirectional**
  - Each Core/LLC can send/get data from either the “up” direction or from the “down” direction
  - Ring access always picks the shortest path
  - E.g., Core3 to LLC2 data uses the “up” stream in 1 hop
    - Rather than from the "down" stream in 7 hops
  - Ave. trip from Core to LLC is (0+1+2+3)/4 = 1.5 hops
  - Data moves across the ring at one stop per clock



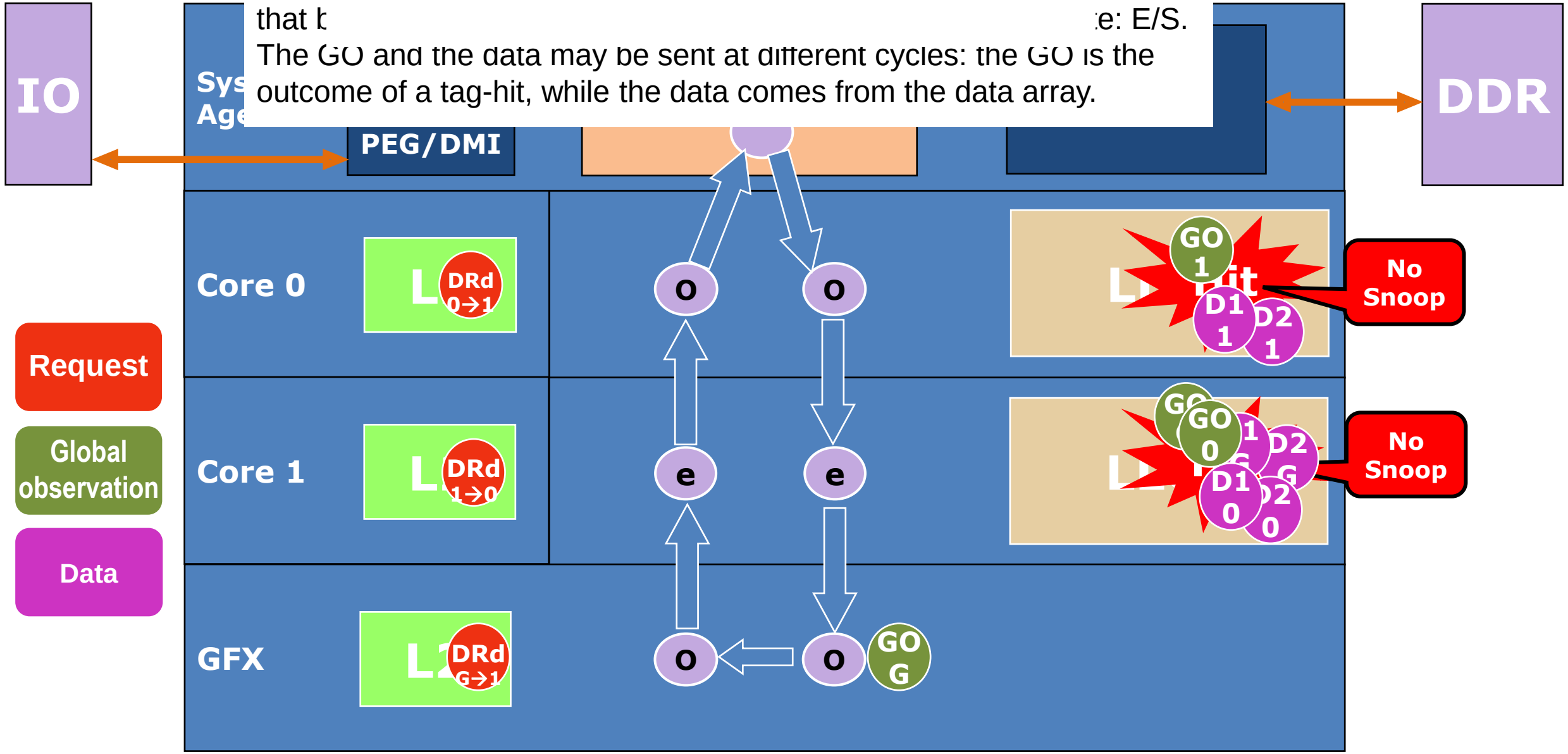
Ring Traffic Control

- **Each stop can pull one request off the ring per clock**
  - Either from the "up" or from the "down“ direction: need to avoid a case of data arriving to a given stop from both "up" and "down" directions at the same cycle
- **Solved by adding even/odd polarity to each stop**
  - The two ring directions (“up” and “down”) flip polarity every clock
  - A ring stop can pull data only from the direction that matches its polarity in the current cycle
  - The sender knows the receiver and its polarity, and the hop count from it to the receiver
  - By delaying sending the data by at most one cycle, a sender guarantees the receiver can read it
- **Example: Assume Core3 sends a request to LLC2**
  - Shortest path is on the “up” direction – 1 hop
    - Receiver polarity is odd, hop count is odd
      $\Rightarrow$ send request when ring has even polarity
  - Core3 sends the request when the “up” direction has *even polarity*
  - A cycle later the request gets to LLC2, and “up” has *odd polarity* as needed



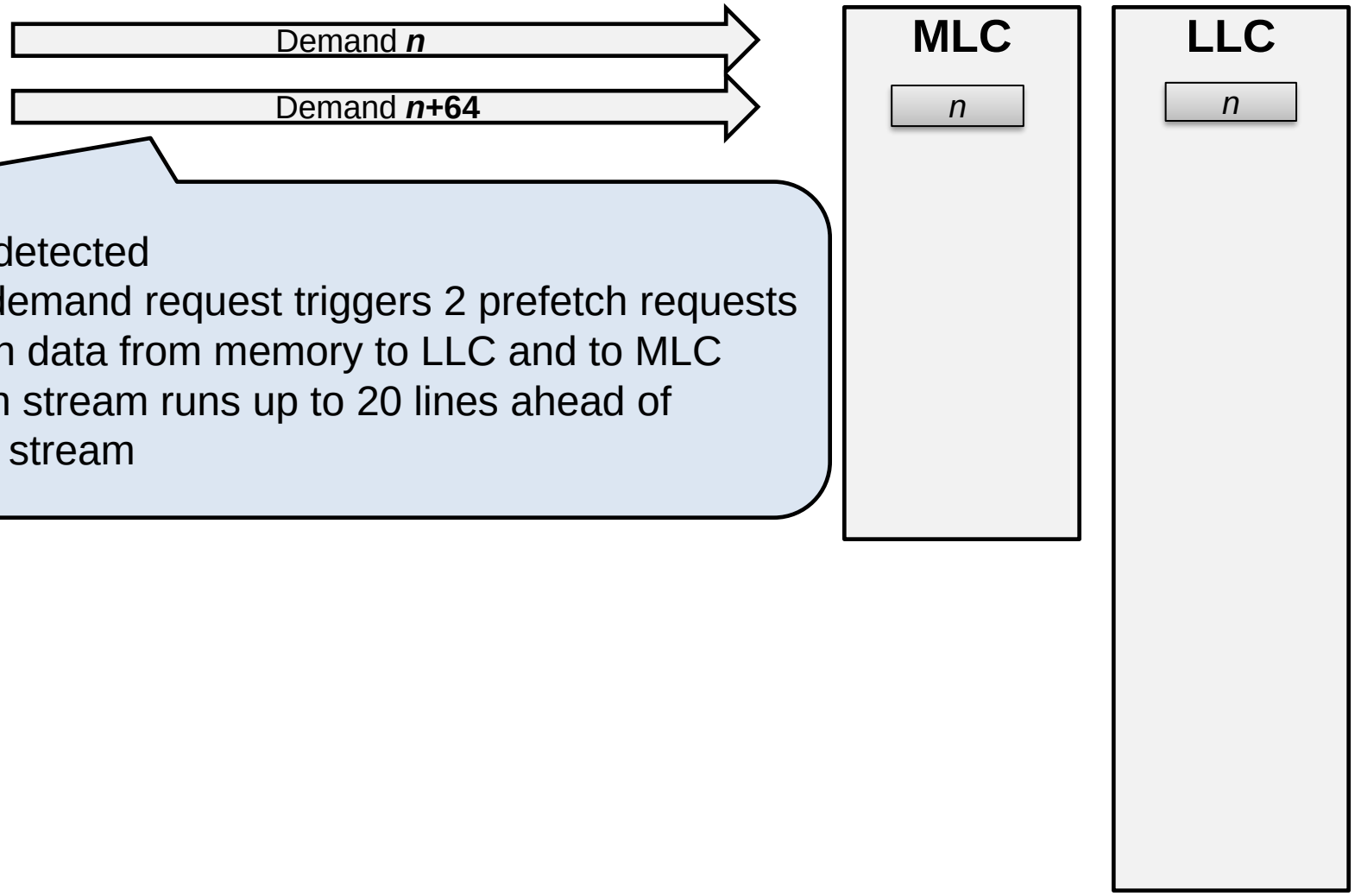
Receiver Polarity	Hop count	Send msg when ring polarity is
Odd	Odd	Even
Odd	Even	Odd
Even	Odd	Odd
Even	Even	Even

LLC s LLC sends the second chunk to each one of the cores dge
that b e: E/S.
The GO and the data may be sent at different cycles: the GO is the outcome of a tag-hit, while the data comes from the data array.



Cycle 15: up ring E, down ring O

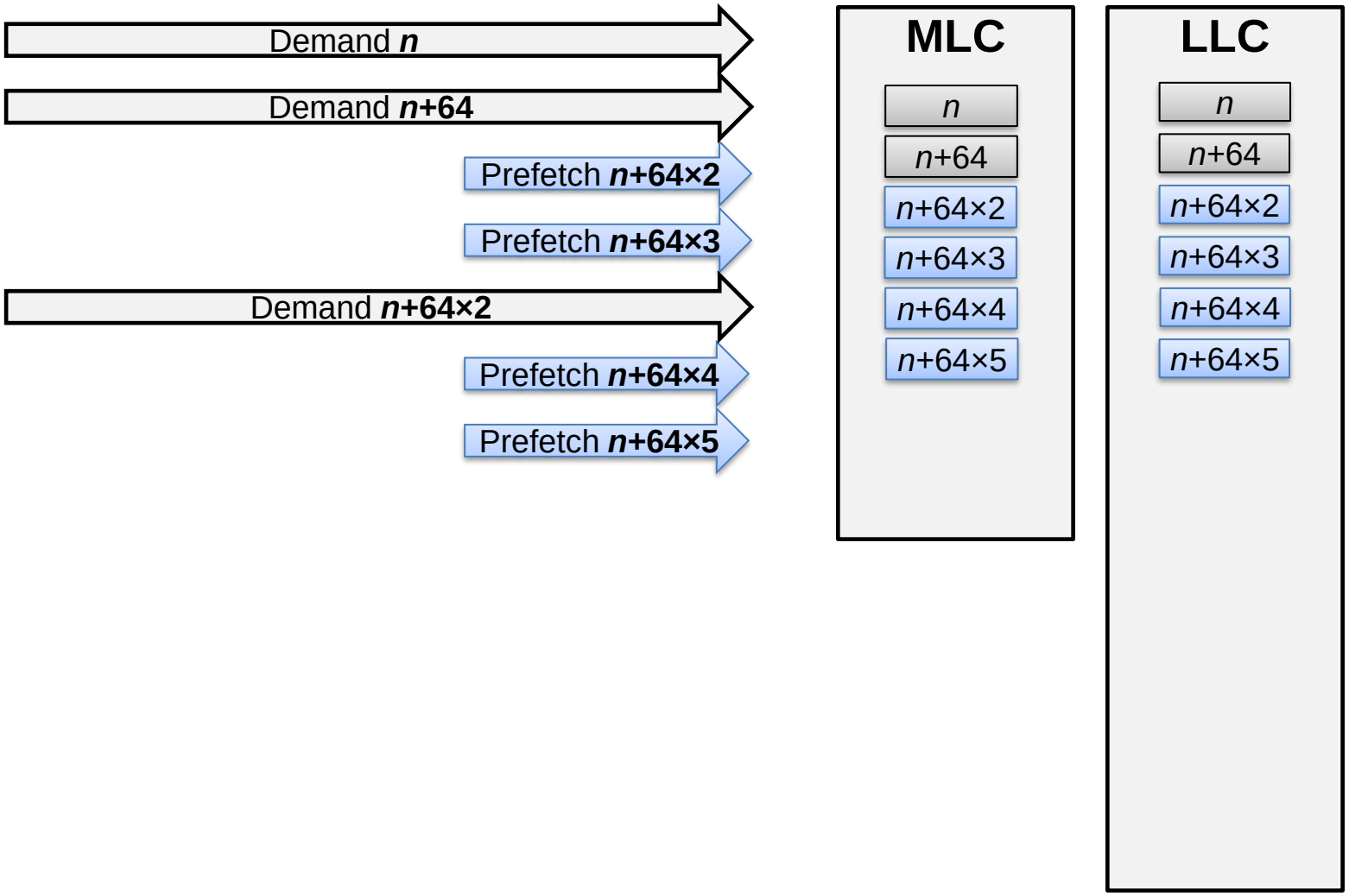
Data Prefetch to MLC and LLC



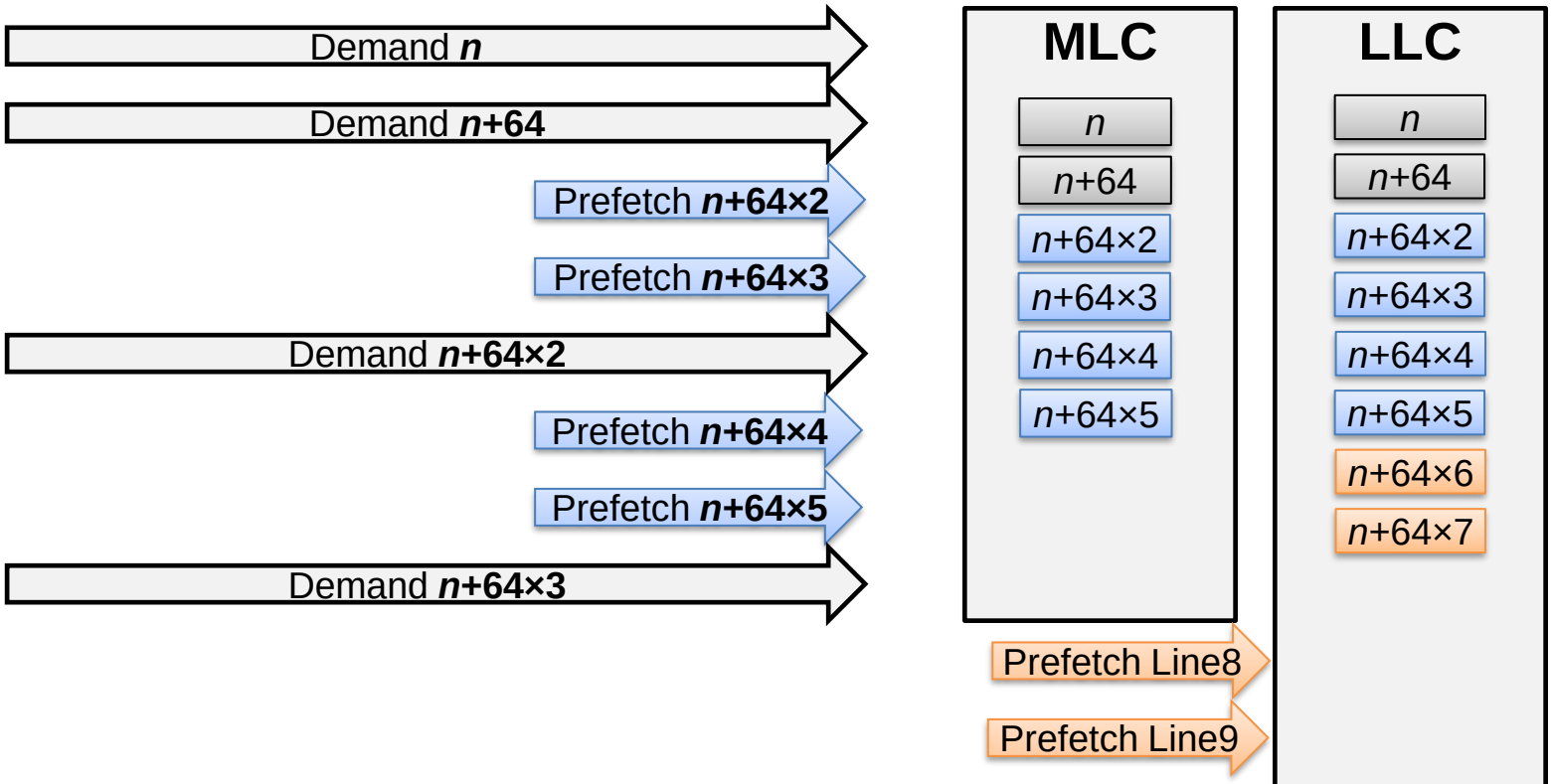
Once stride is detected

- Each MLC demand request triggers 2 prefetch requests
  - Prefetch data from memory to LLC and to MLC
- The prefetch stream runs up to 20 lines ahead of the demand stream

Data Prefetch to MLC and LLC



Data Prefetch to MLC and LLC

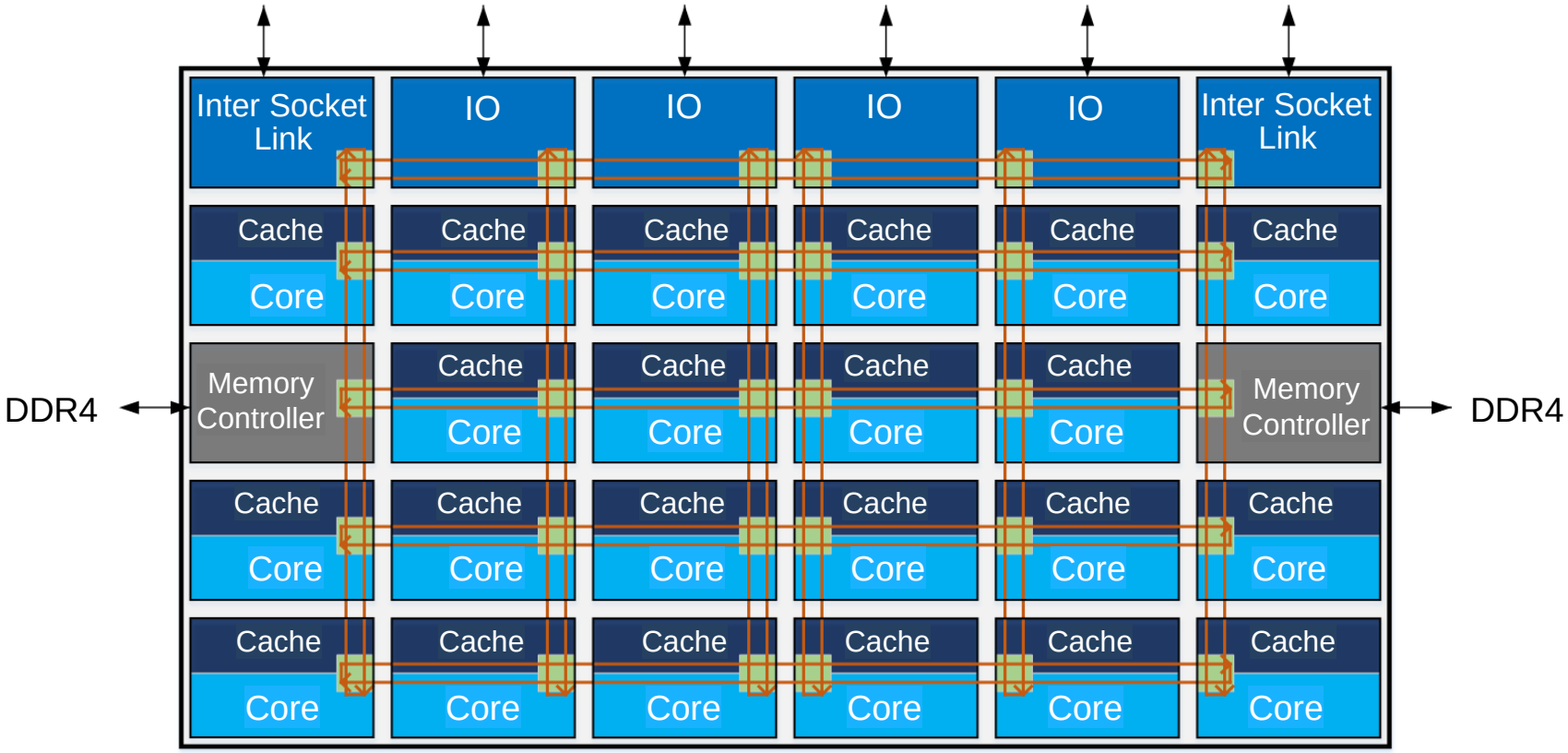


Adjusts dynamically to the number of outstanding requests per core

- Not many outstanding requests ⇒ prefetch further ahead (20 lines)
- Moderate number of outstanding requests ⇒ prefetch up to 10 lines ahead
- Many outstanding requests ⇒ prefetch to LLC only, and less far ahead
- Even more outstanding requests ⇒ stop prefetch also to LLC

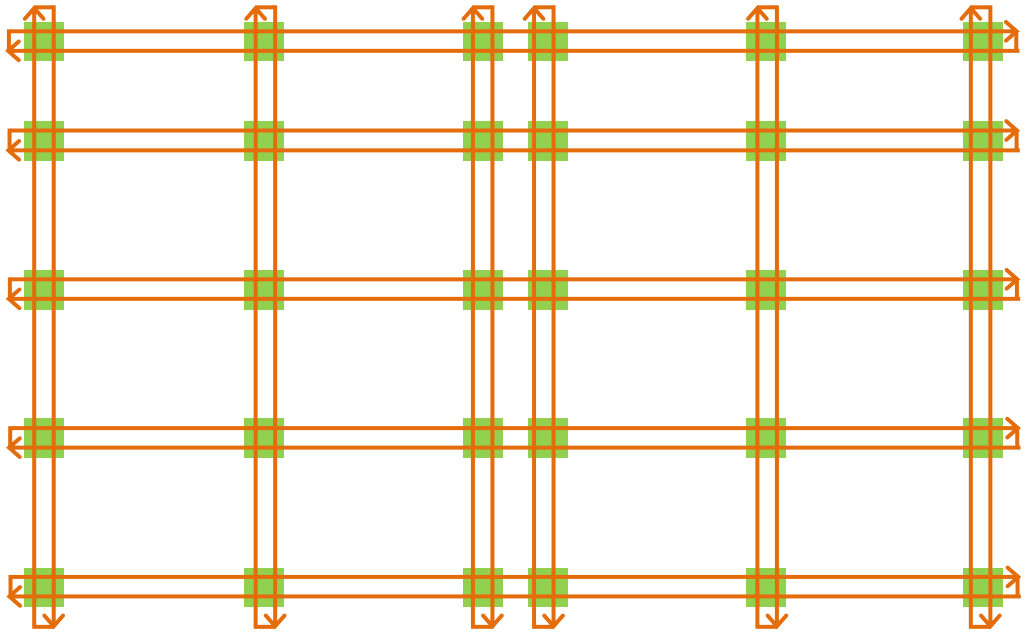
Server SOC

- Large LLC with snooping capabilities that support multiple processors
- Interfaces for supporting multi-socket platforms
- Support for high BW traffic from memory and I/O
- Mesh interconnect between Cores, LLC, I/O, MC, Inter Socket Link
- No Graphics Unit



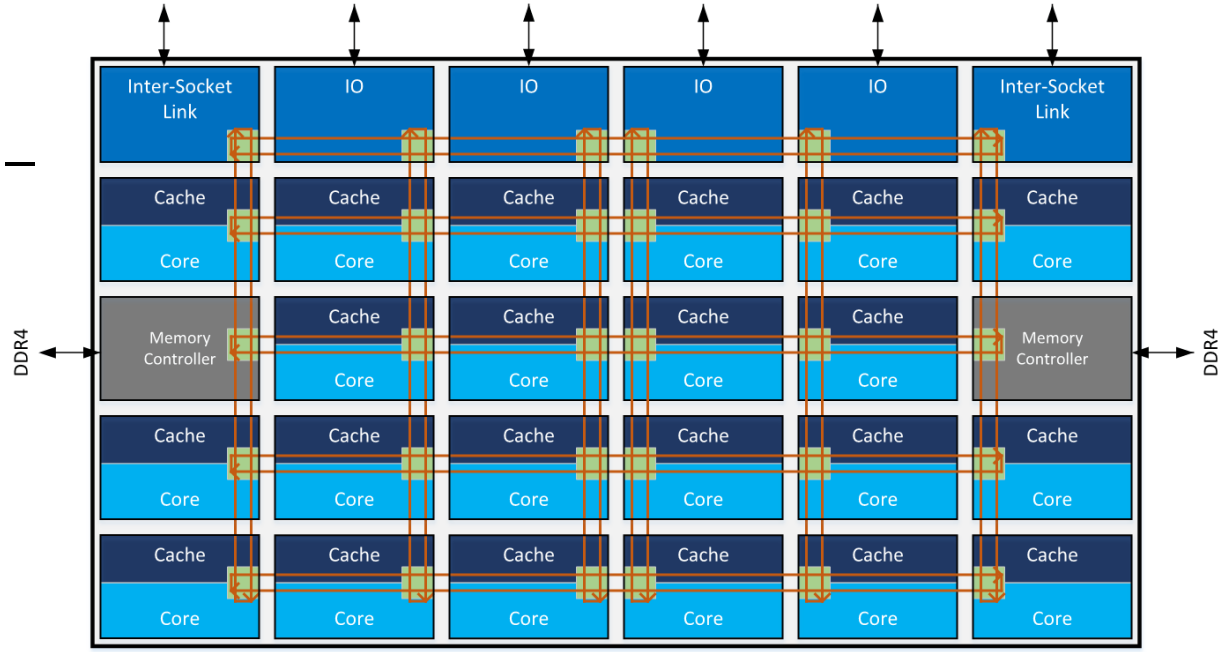
Mesh Interconnect

- **A synchronous, high-bandwidth, scalable 2-dimensional array of half rings**
  - Ring is limited by the number of cores it can serve in terms of latency and bandwidth, thus not suitable for server



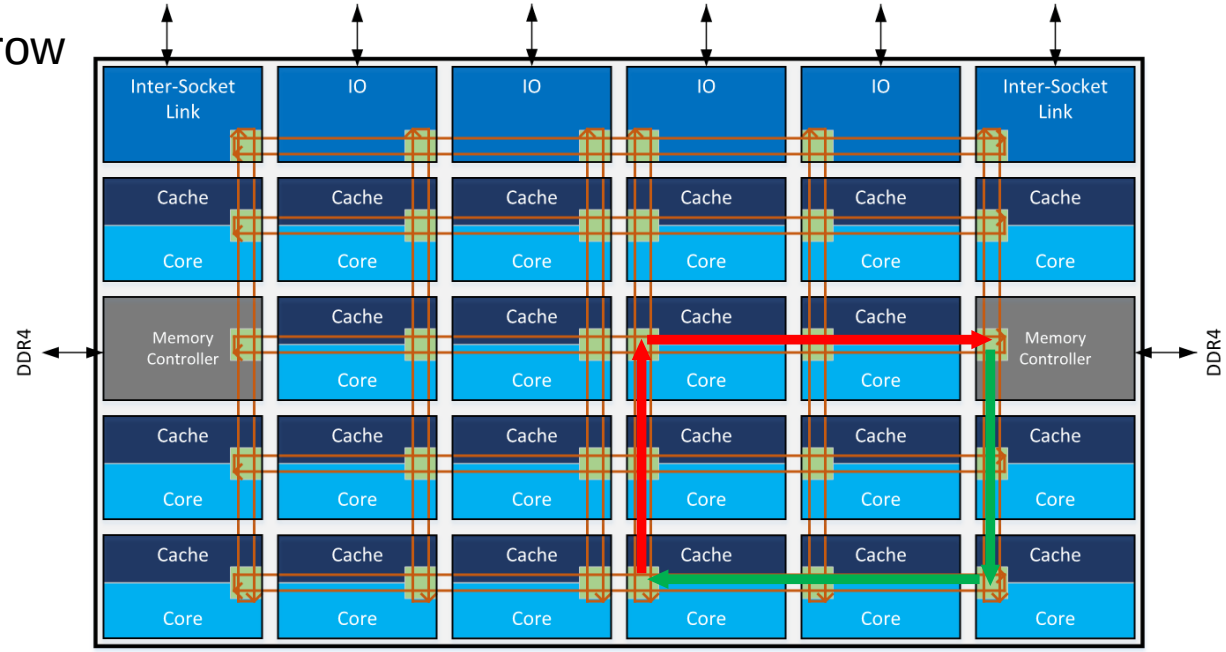
Mesh Components

- Mesh – the fabric, a 2-dimensional array of half rings forming a grid
  - Every vertical column forms a bi-directional half ring
  - Every horizontal row forms a bi-directional half ring
- Tile – a modular IP block that can be replicated across the grid
  - Core Tile, IMC Tile (integrated memory controller)
- Caching/Home Agent (CHA) - a unit inside the core tiles that maintains the cache coherency between tiles
  - The CHA also interfaces with the CMS
- Converged/Common Mesh Stop (CMS) – a mesh stop station
  - Facilitating the interface between a tile and the fabric

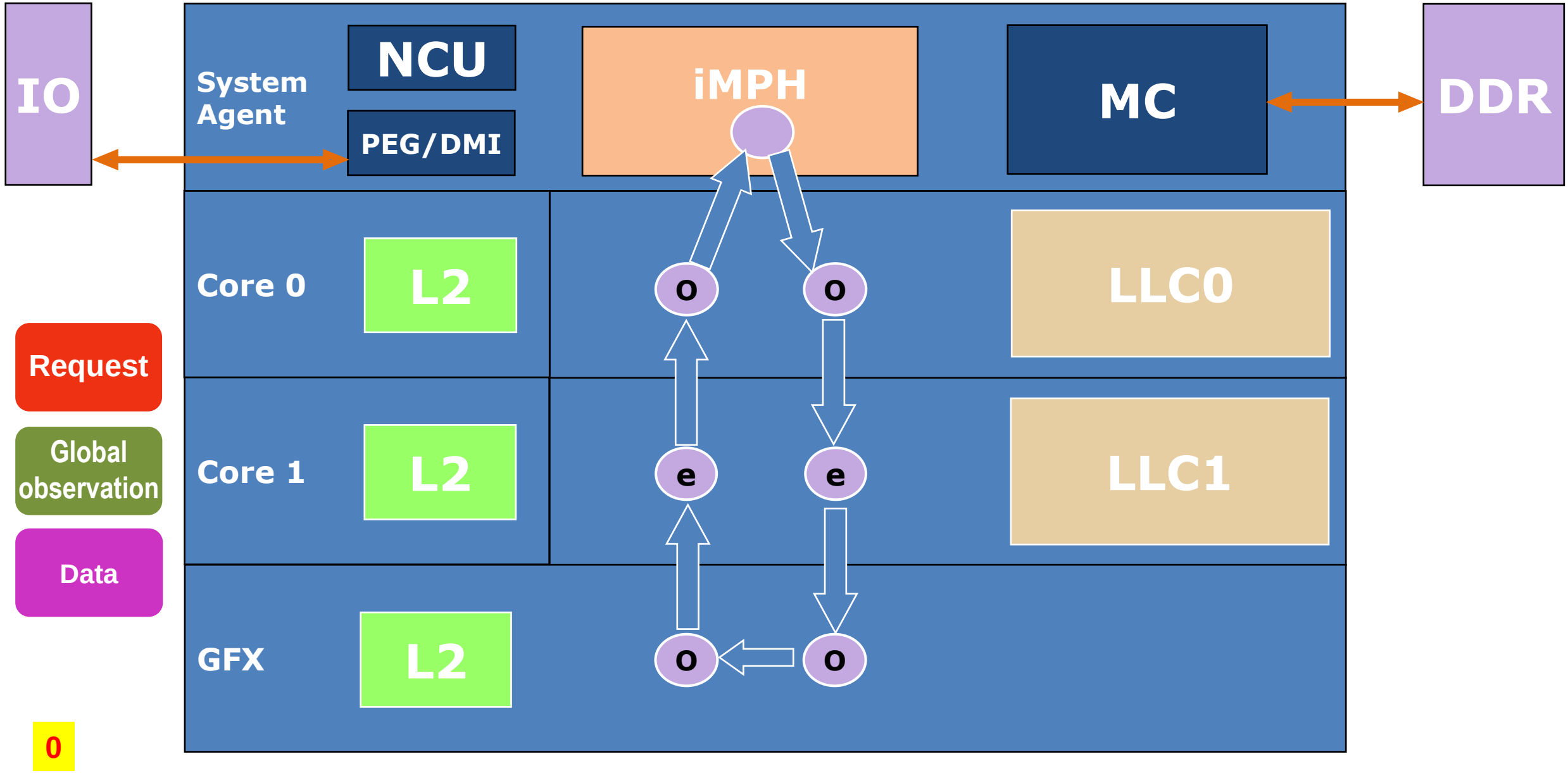


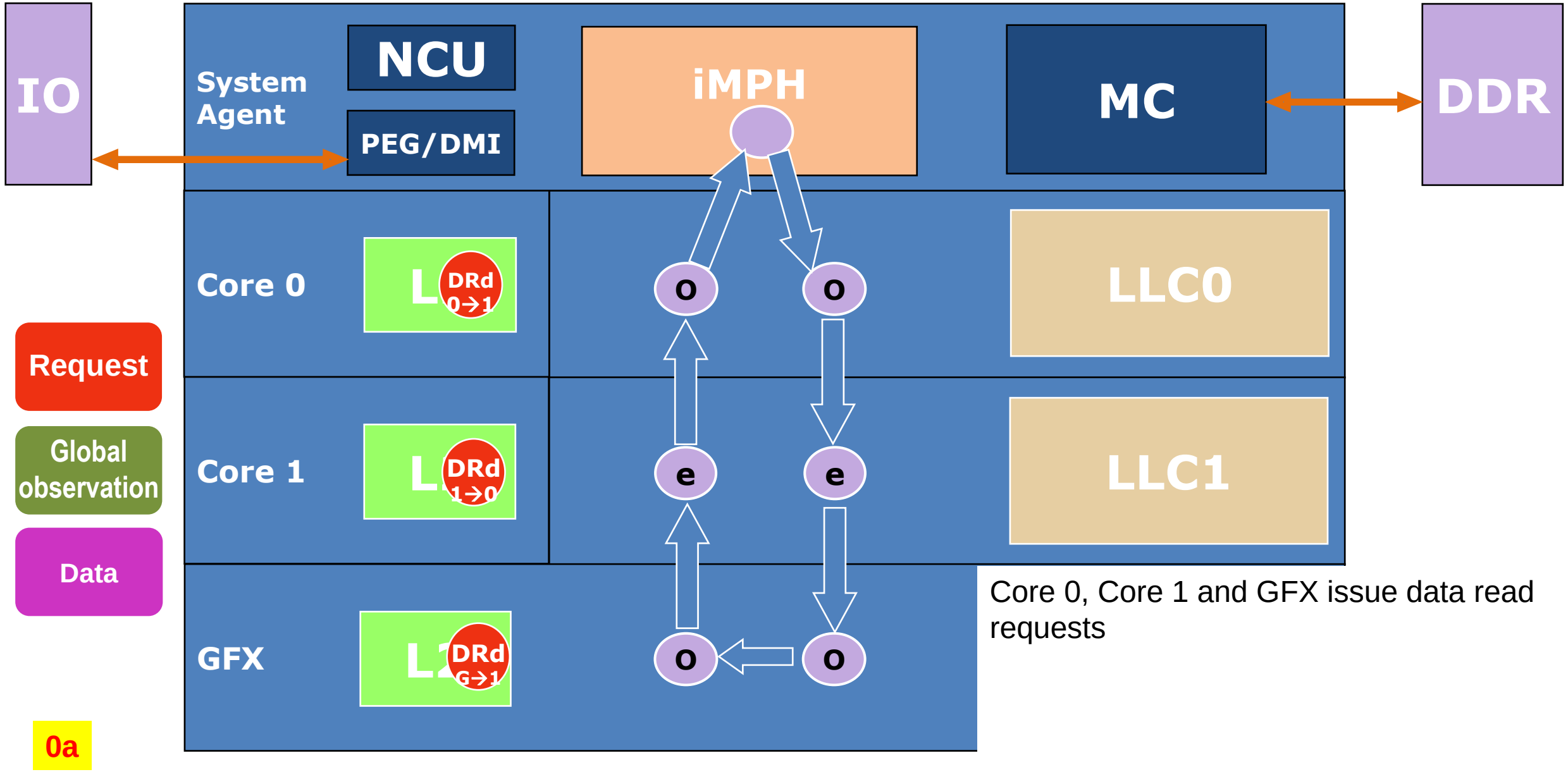
Mesh Routing

- **A packet follows a simple routing algorithm**
  - Packets are first routed vertically
  - Packets are then routed horizontally
- **A packet originates at a tile (e.g. from the CHA)**
  - It enters the fabric at its local Mesh Stop (CMS)
  - The packet is routed along the vertical half ring, either north or south
    - Always taking the shortest path
  - Once the packet reaches its destination row
    - It is taken off the vertical half ring and placed on the horizontal half ring where it continues to the destination tile
  - Once the packet reaches the destination tile
    - It interfaces back with the tile via Mesh Stop
  - The return path is different

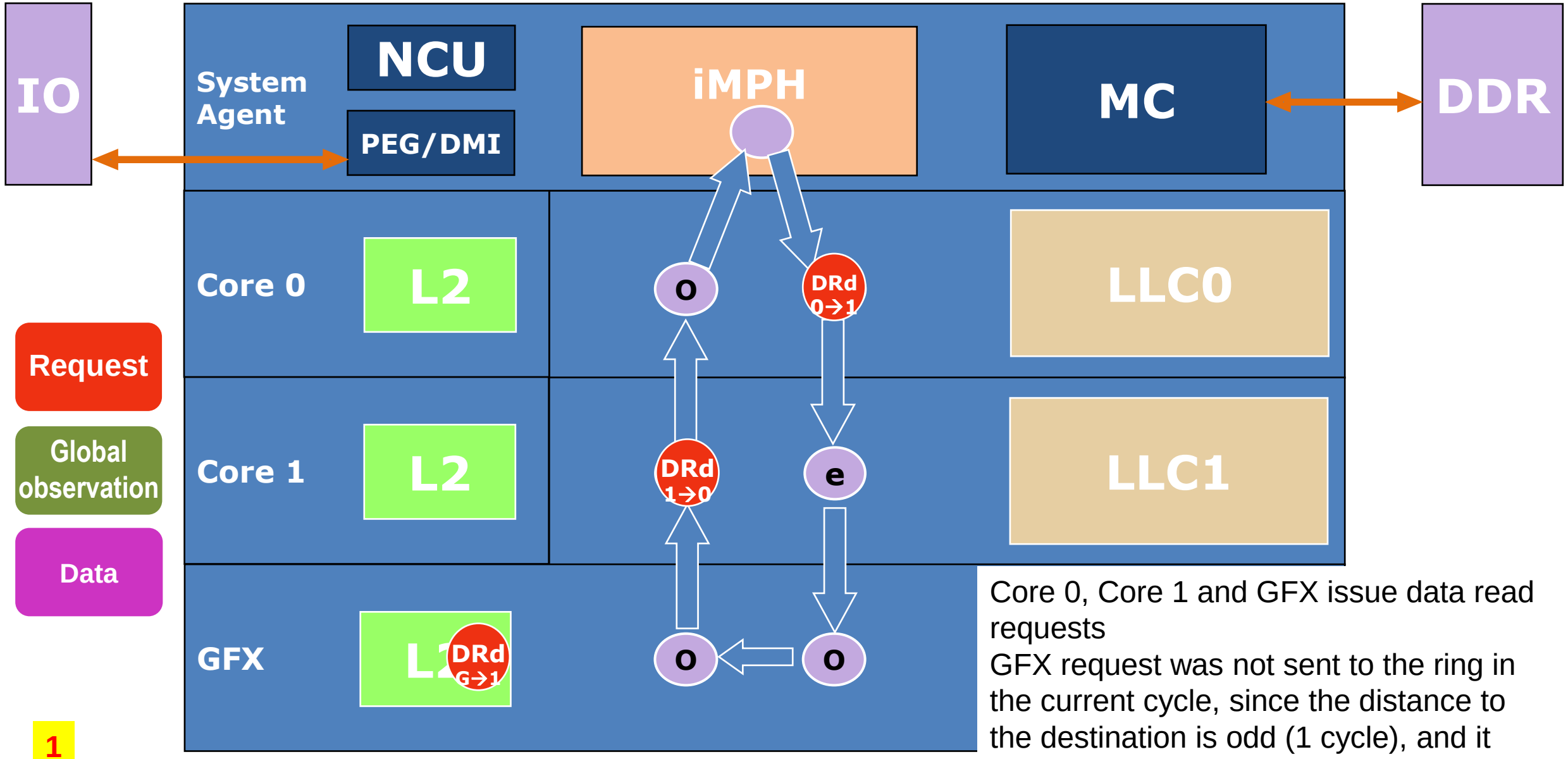


Backup


IO
System Agent
NCU
PEG/DMI
iMPH
MC
DDR
Core 0
L2
Core 1
L2
GFX
L2
LLC0
LLC1
o
o
e
e
o
o
Request
Global observation
Data
0

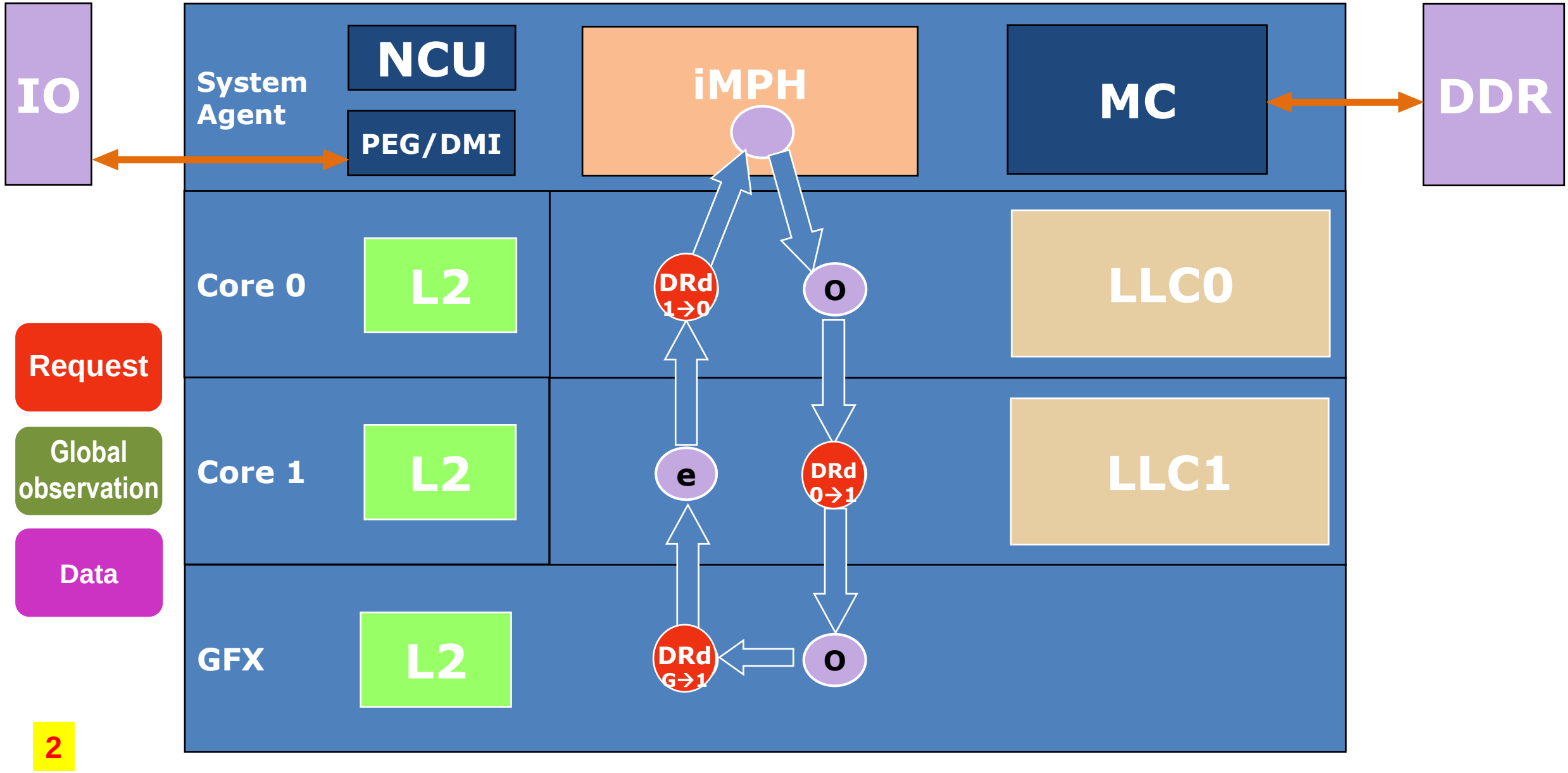


Core 0, Core 1 and GFX issue data read requests

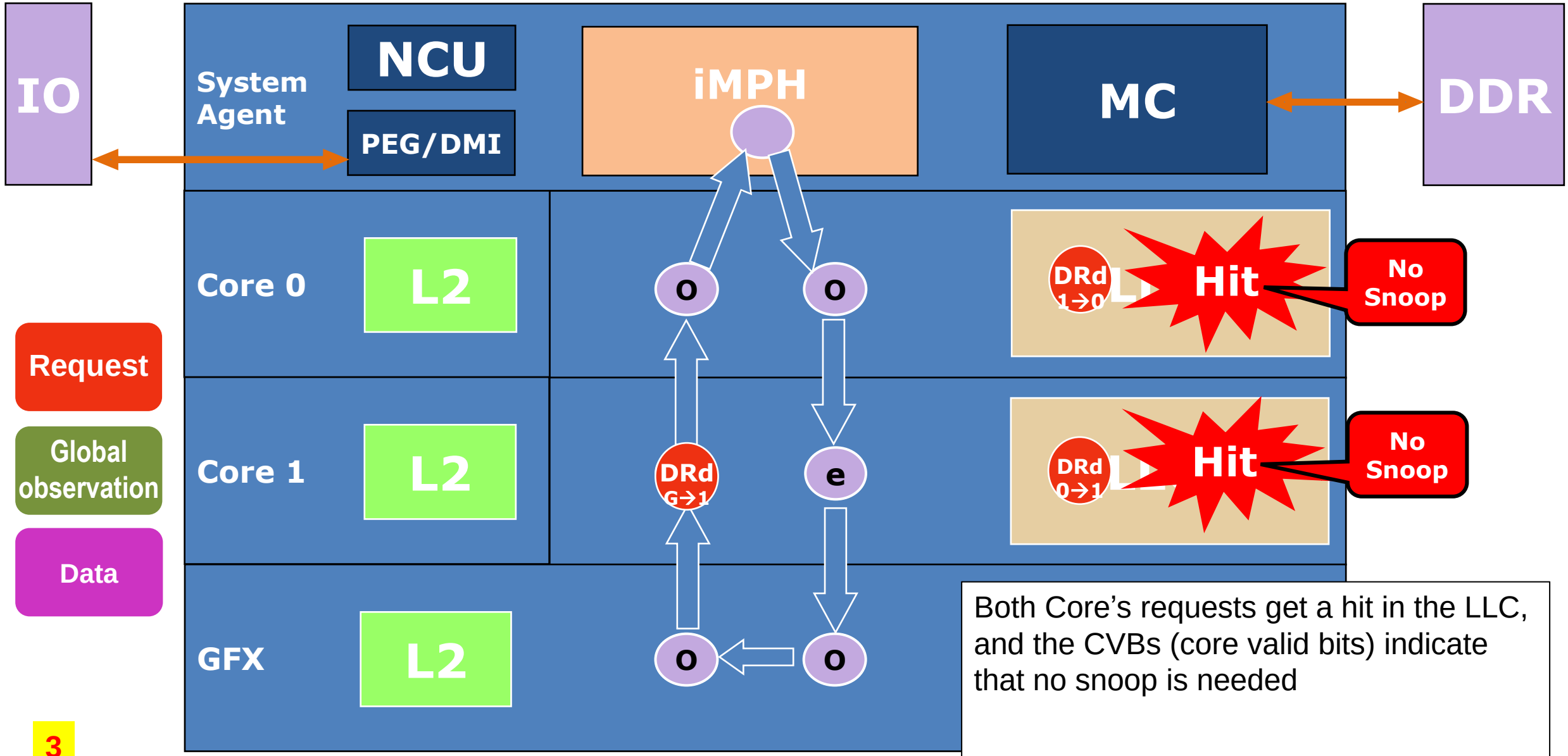


Core 0, Core 1 and GFX issue data read requests
GFX request was not sent to the ring in the current cycle, since the distance to the destination is odd (1 cycle), and it must arrive there on an Even cycle

Cycle 1: up ring E, down ring O

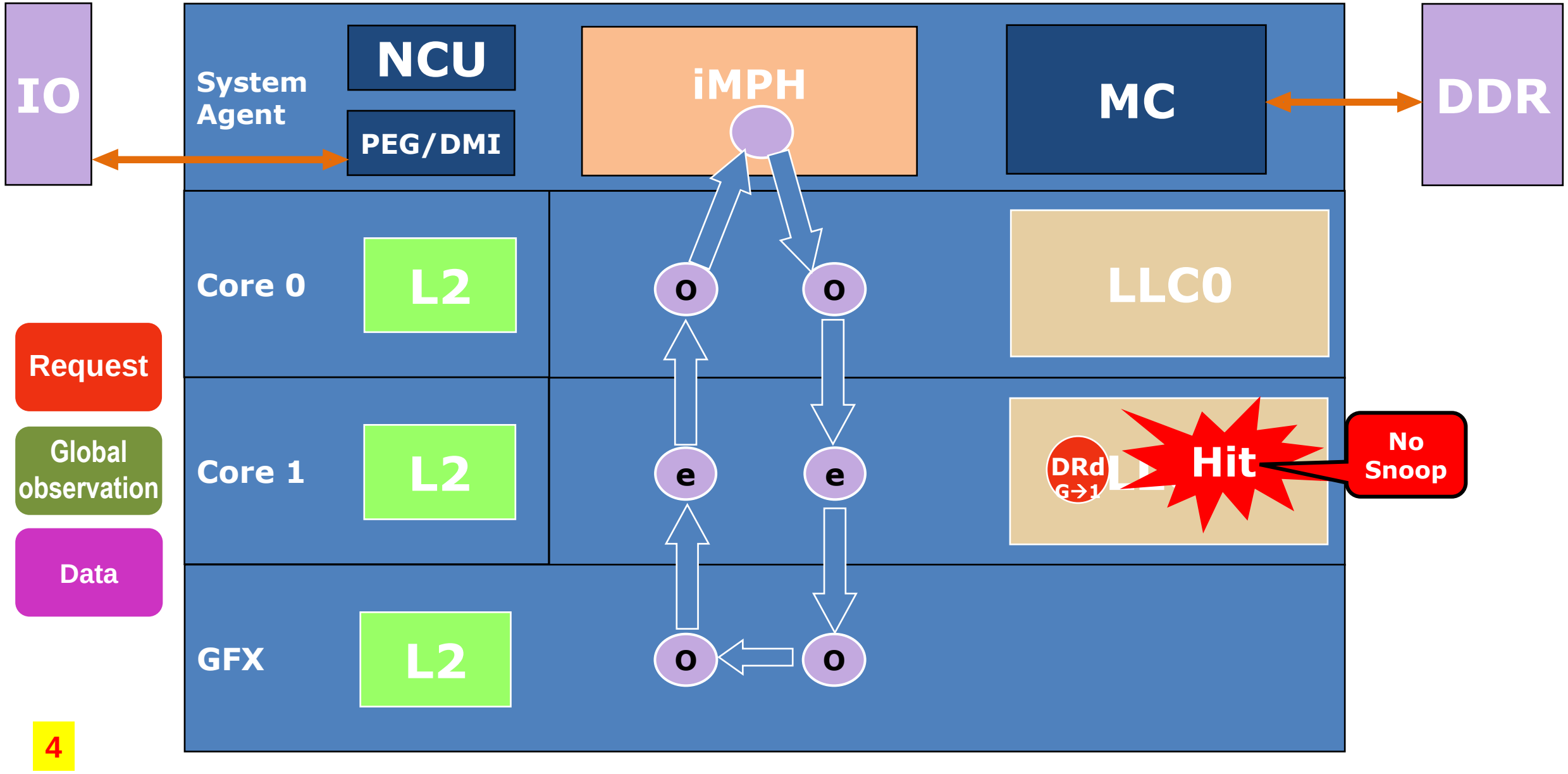


Cycle 2: up ring O, down ring E

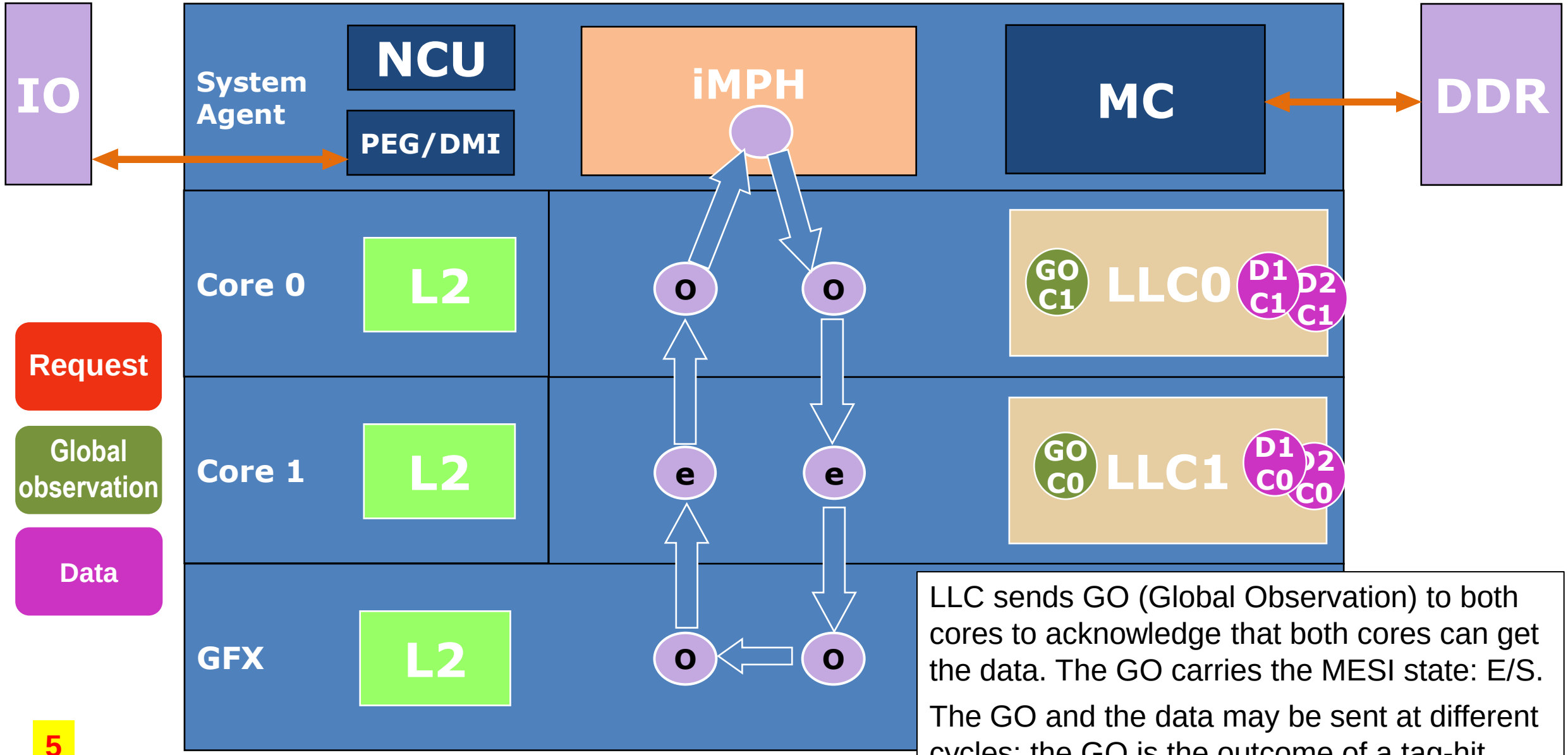


Both Core's requests get a hit in the LLC, and the CVBs (core valid bits) indicate that no snoop is needed

Cycle 3: up ring E, down ring O



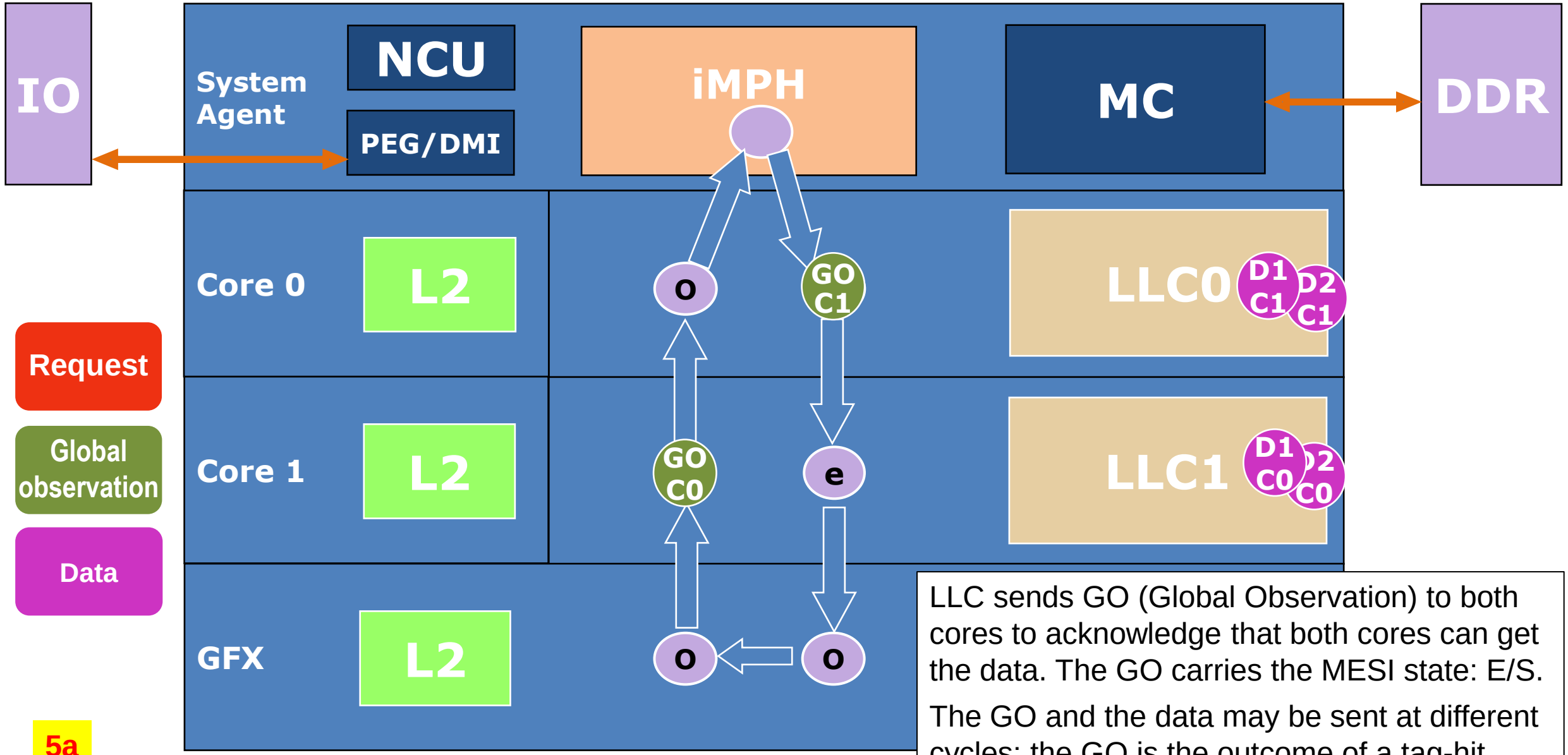
Cycle 4: up ring O, down ring E



Cycle 5: up ring E, down ring O

LLC sends GO (Global Observation) to both cores to acknowledge that both cores can get the data. The GO carries the MESI state: E/S.

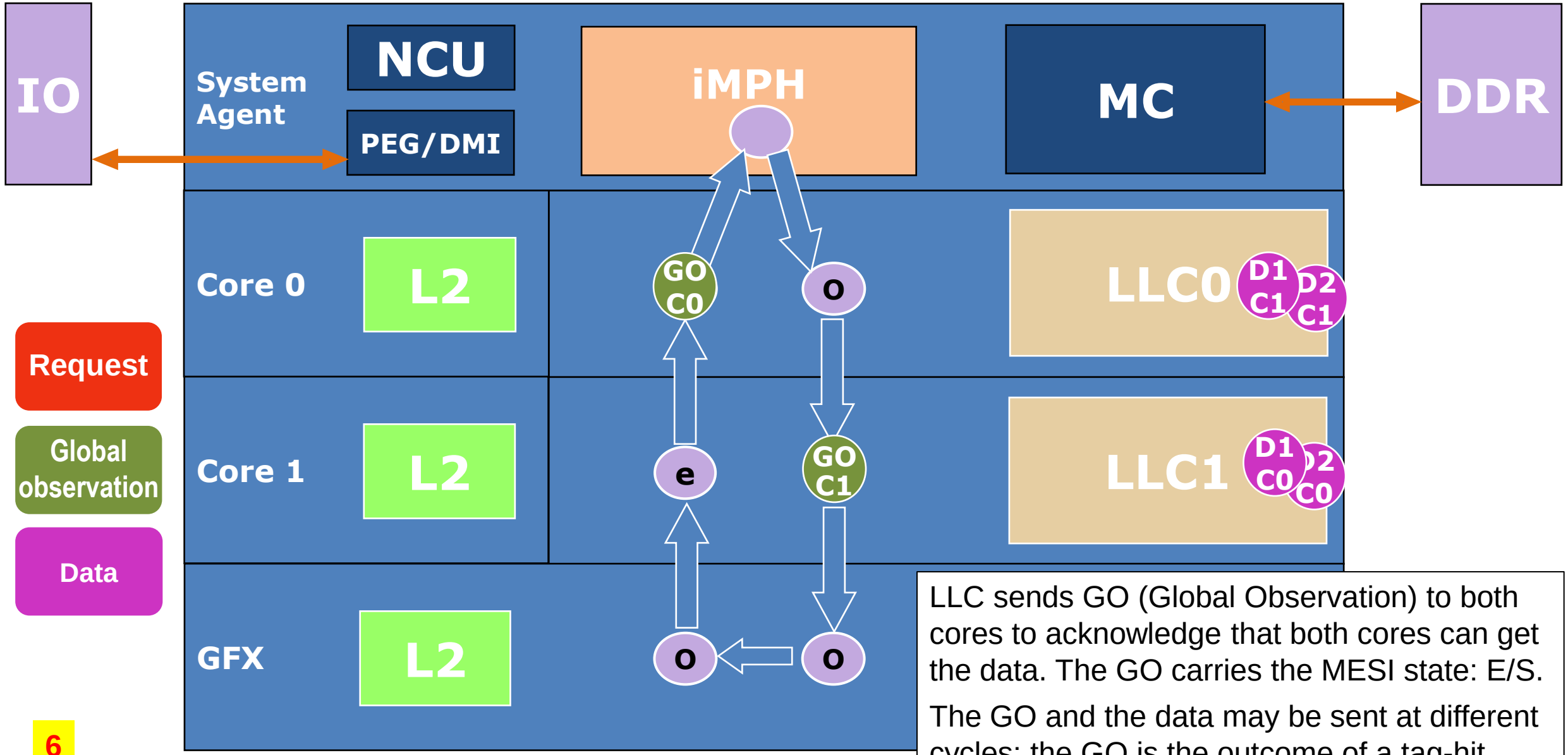
The GO and the data may be sent at different cycles: the GO is the outcome of a tag-hit, while the data comes from the data array.



Cycle 5: up ring E, down ring O

LLC sends GO (Global Observation) to both cores to acknowledge that both cores can get the data. The GO carries the MESI state: E/S.

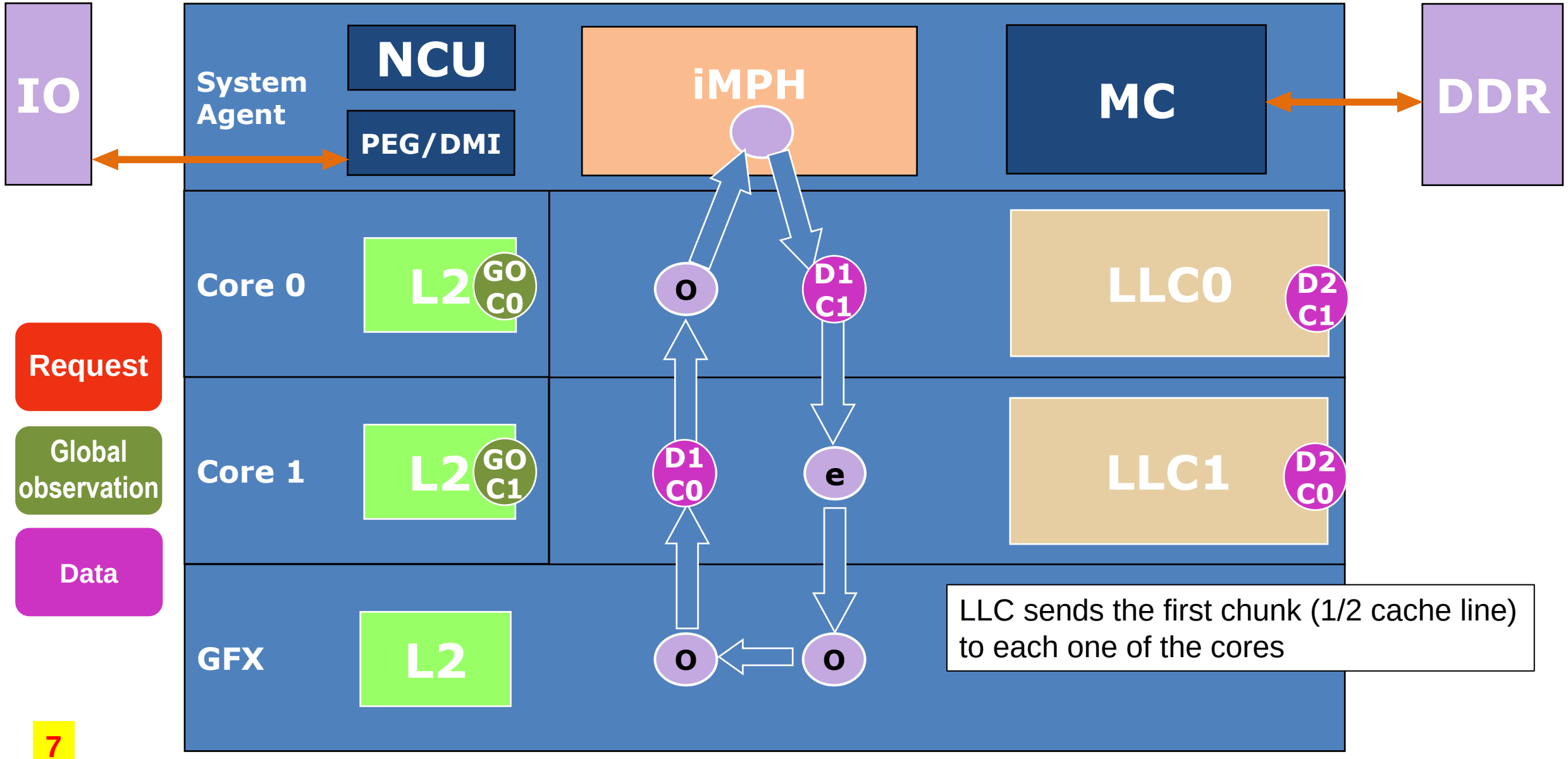
The GO and the data may be sent at different cycles: the GO is the outcome of a tag-hit, while the data comes from the data array.



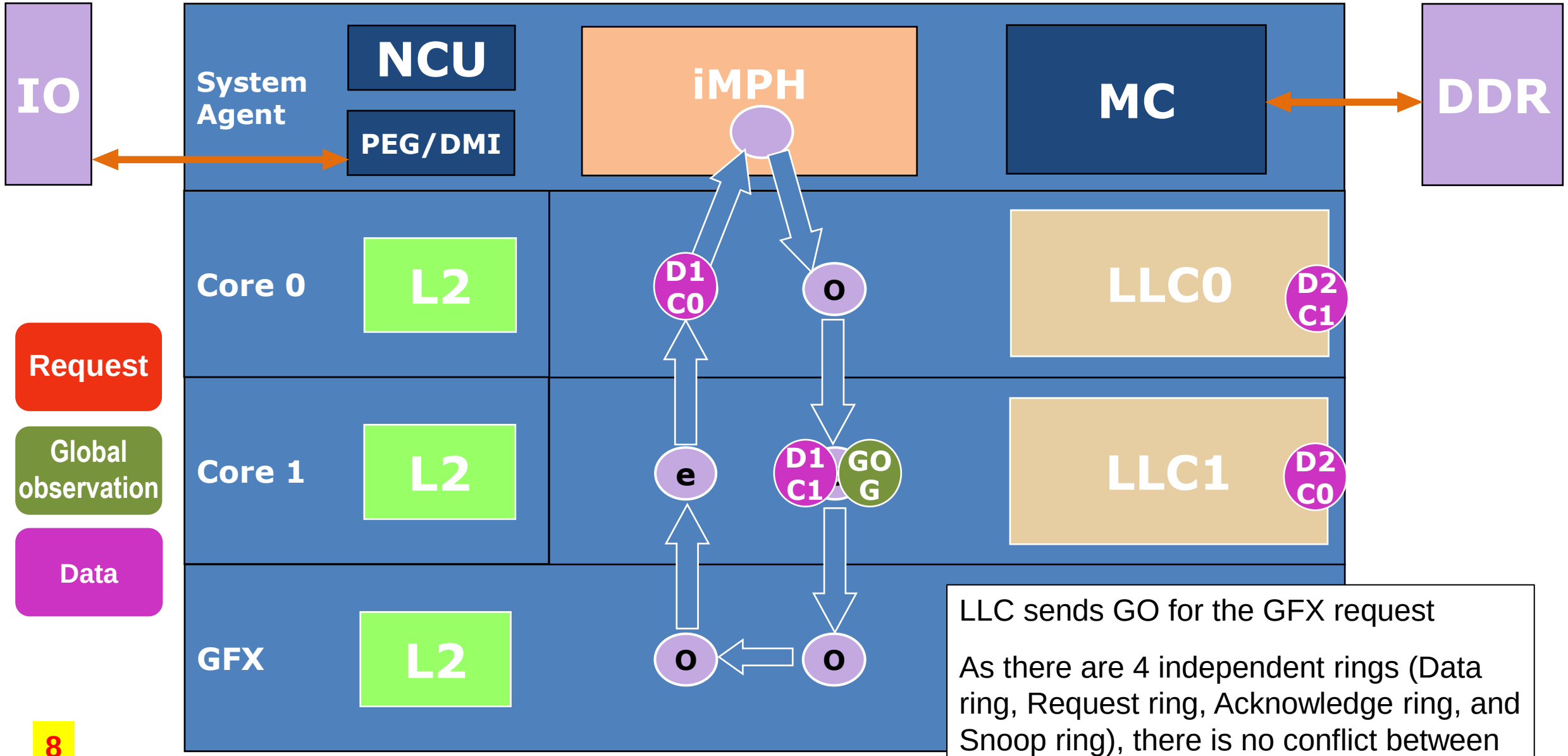
Cycle 6: up ring O, down ring E

LLC sends GO (Global Observation) to both cores to acknowledge that both cores can get the data. The GO carries the MESI state: E/S.

The GO and the data may be sent at different cycles: the GO is the outcome of a tag-hit, while the data comes from the data array.



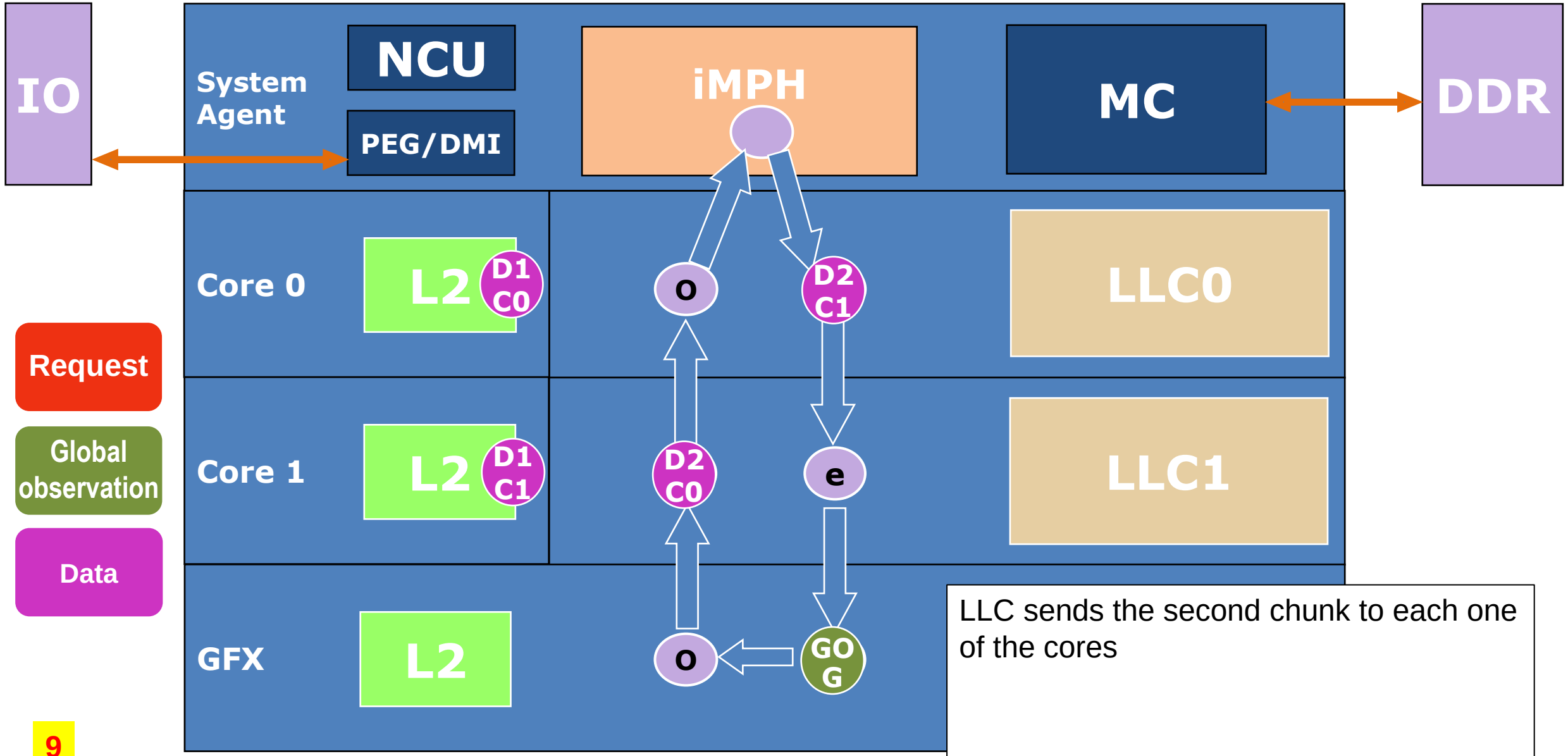
Cycle 7: up ring E, down ring O



Cycle 8: up ring O, down ring E

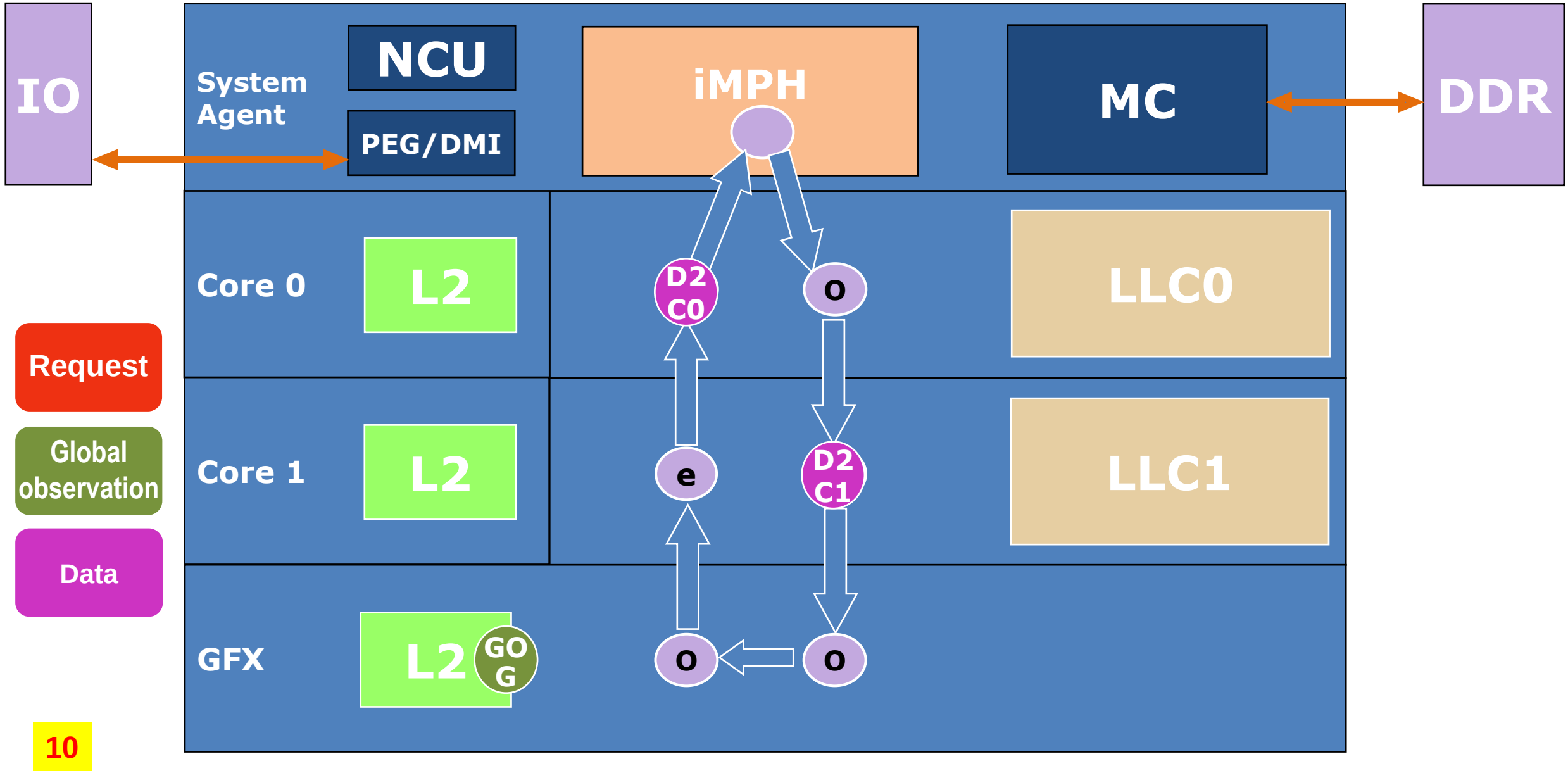
LLC sends GO for the GFX request

As there are 4 independent rings (Data ring, Request ring, Acknowledge ring, and Snoop ring), there is no conflict between Data and GO.

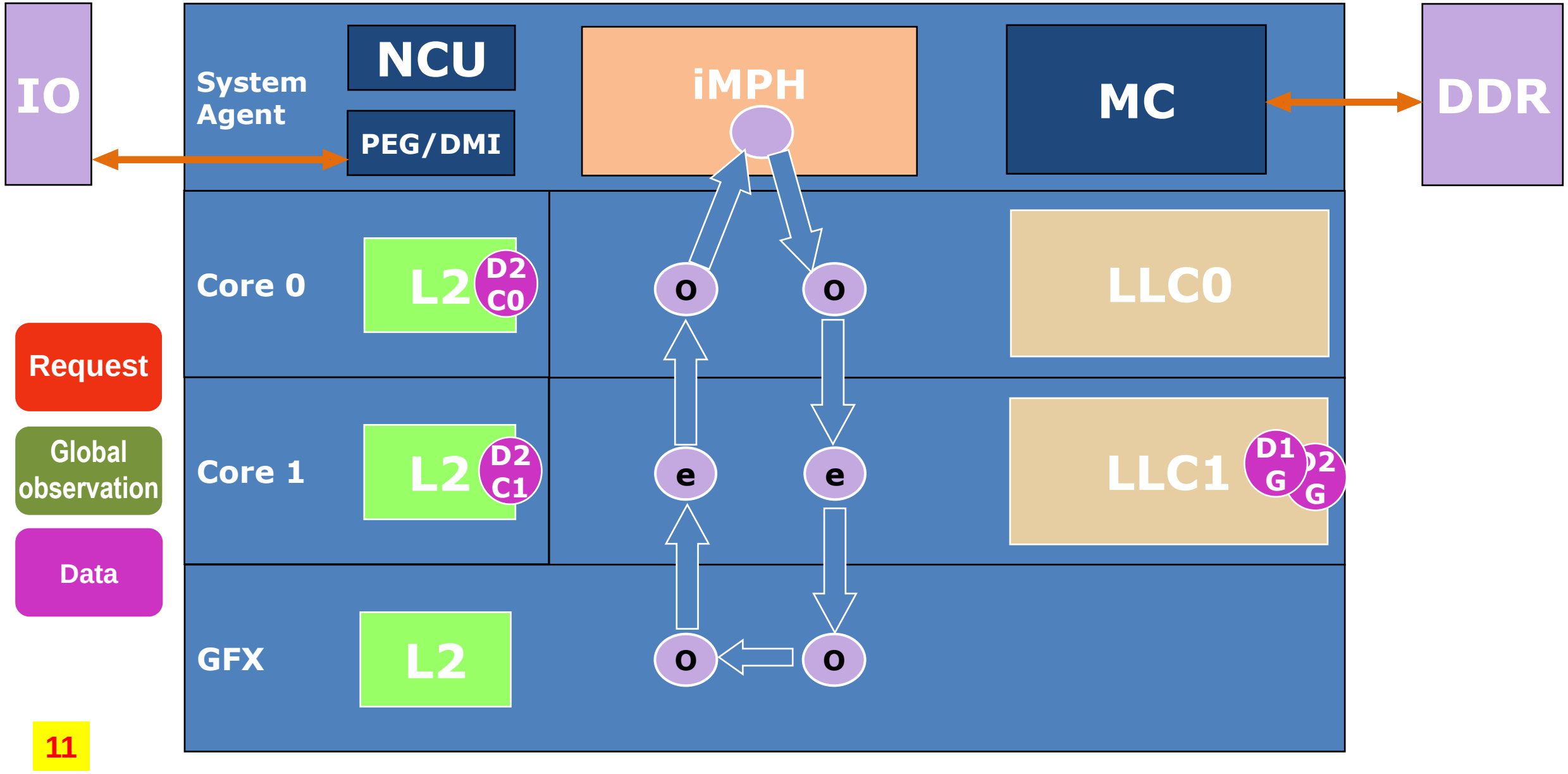


LLC sends the second chunk to each one of the cores

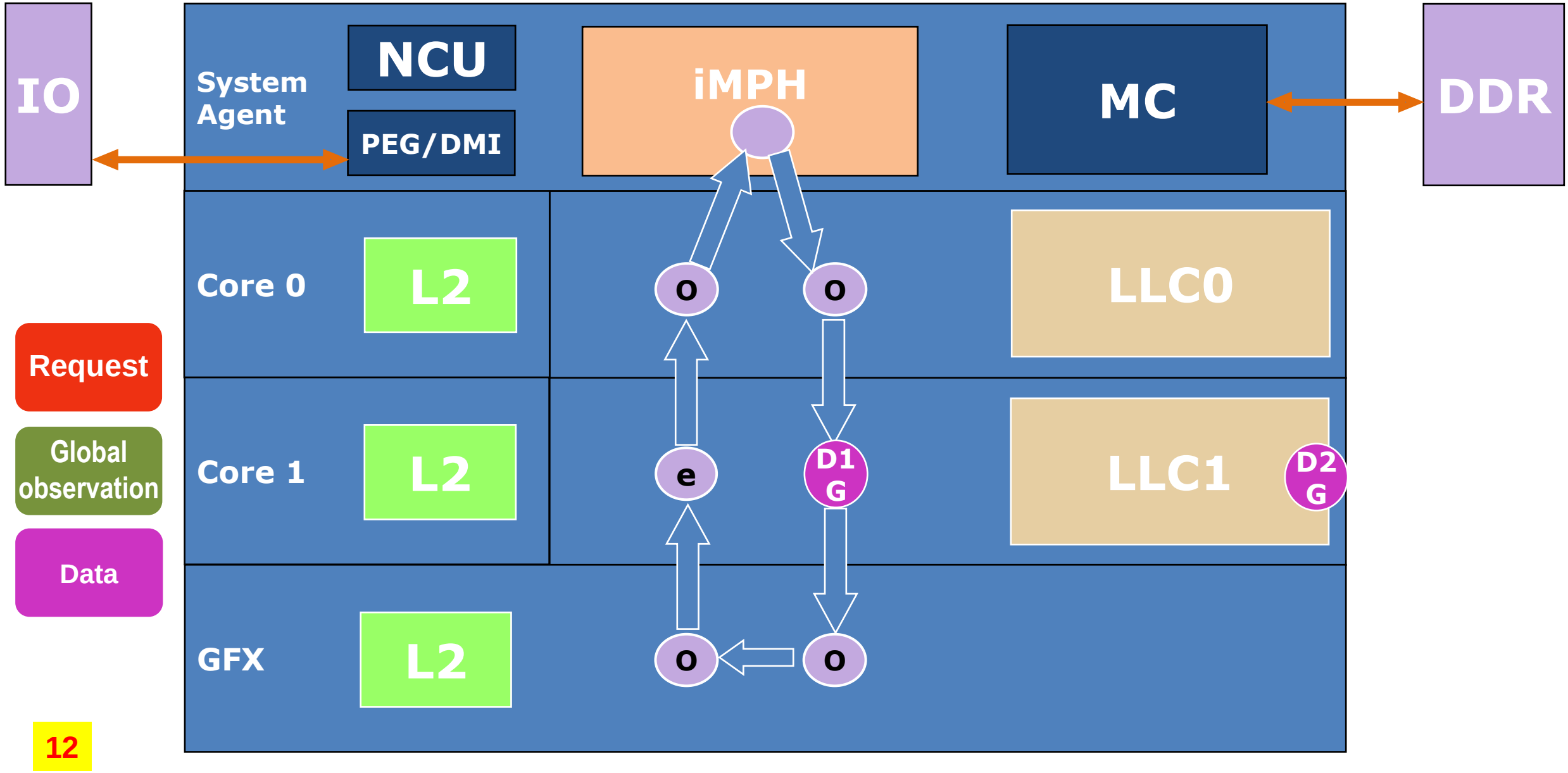
Cycle 9: up ring E, down ring O



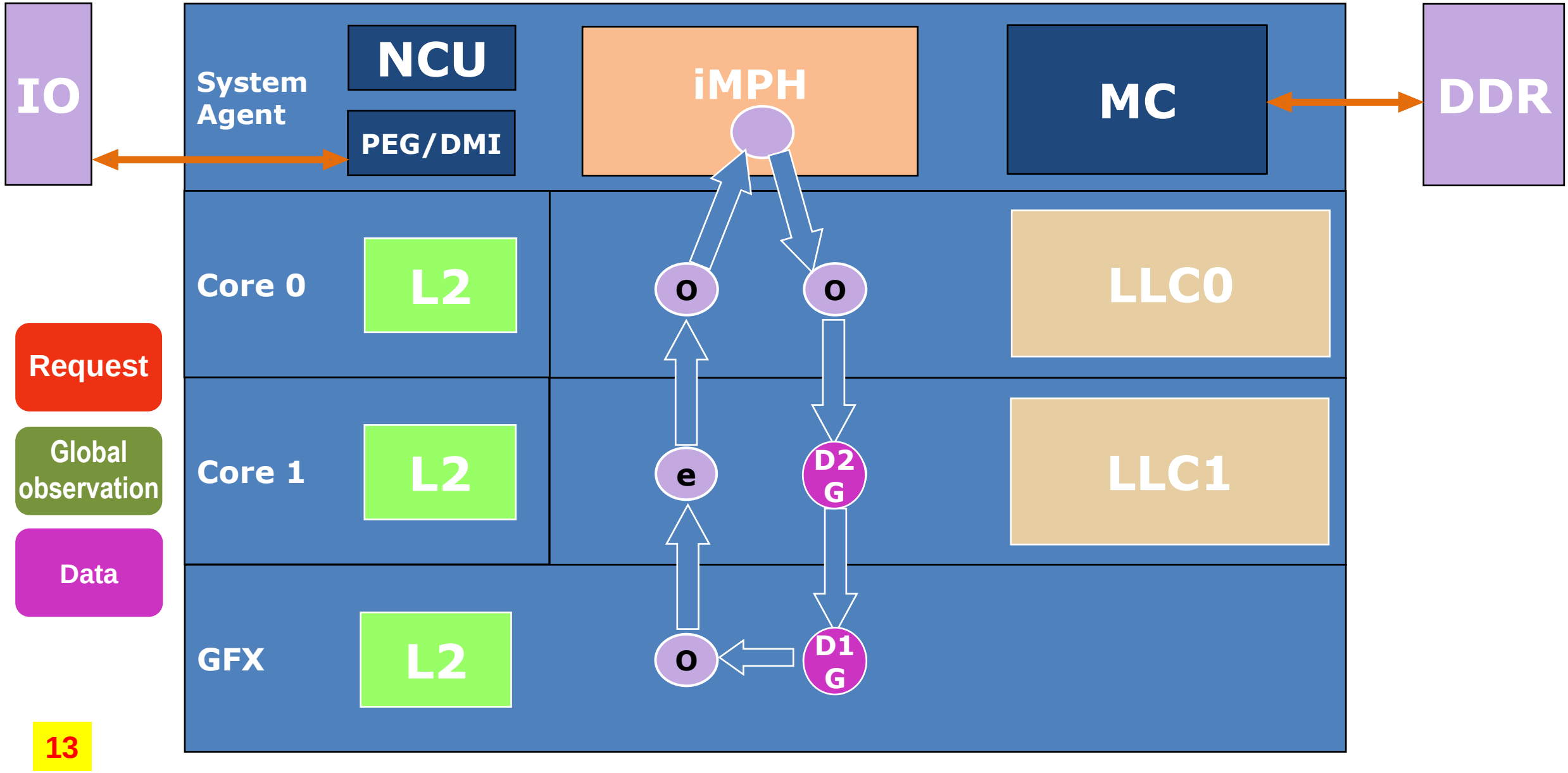
Cycle 10: up ring O, down ring E



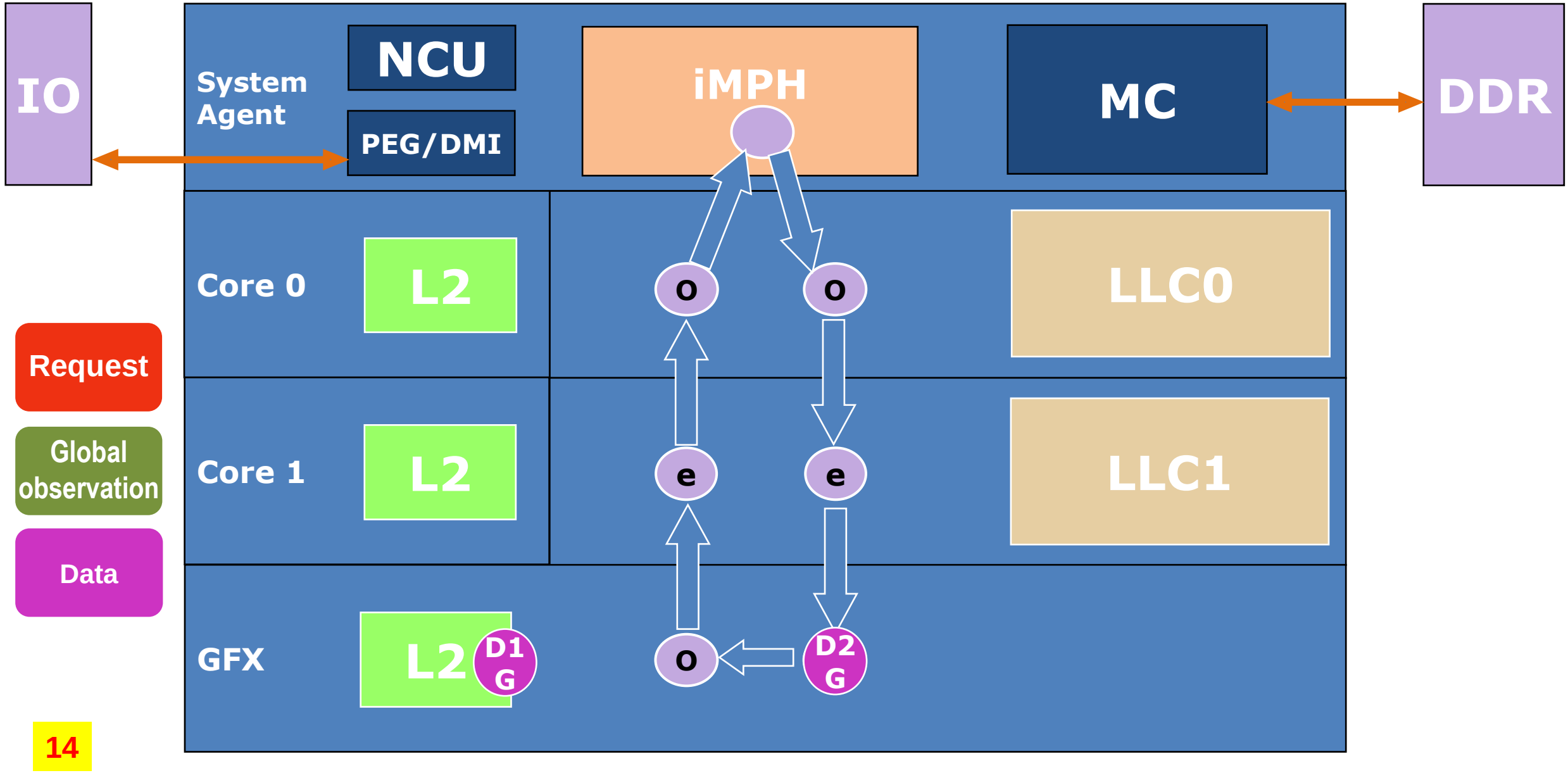
Cycle 11: up ring E, down ring O



Cycle 12: up ring O, down ring E



Cycle 13: up ring E, down ring O



Cycle 14: up ring O, down ring E

System Agent Components

- **PCIe controllers that connect to external PCIe devices**
  - Support different configurations: ×16+×4, ×8+×8+×4, ×8+×4+×4+×4
- **DMI (Direct Media Interface) controller**
  - Connects to the PCH (Platform Controller Hub)
- **Integrated display engine**
  - Handles delivering the pixels to the screen
- **Flexible Display Interface (FDI)**
  - Connects to the PCH, where the display connectors (HDMI, DVI) are attached
- **DisplayPort (used for integrated display)**
  - e.g., a laptop's LCD
- **Integrated Memory Controller (IMC)**
  - Connects to and controls the DRAM
- **An arbiter that handles accesses from Ring and from I/O (PCIe & DMI)**
  - Routes the accesses to the right place
  - Routes main memory traffic to the IMC