Cache Injection Policies for End Host Networking

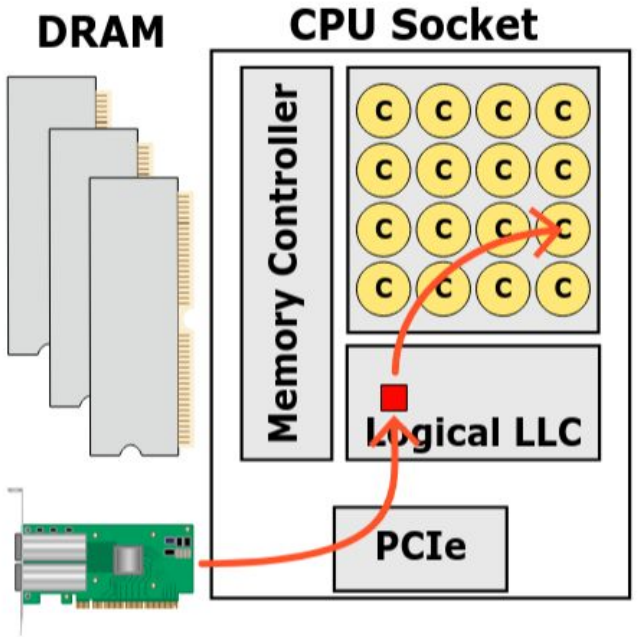
COSENERS ‘26

Marco Molè, Aurojit Panda, Gianni Antichi

Cache injection?

- DMA operations are performed directly to L3 and not to RAM

- Large adoption of DDIO (Intel's implementation) has proved the need for cache injection the host.

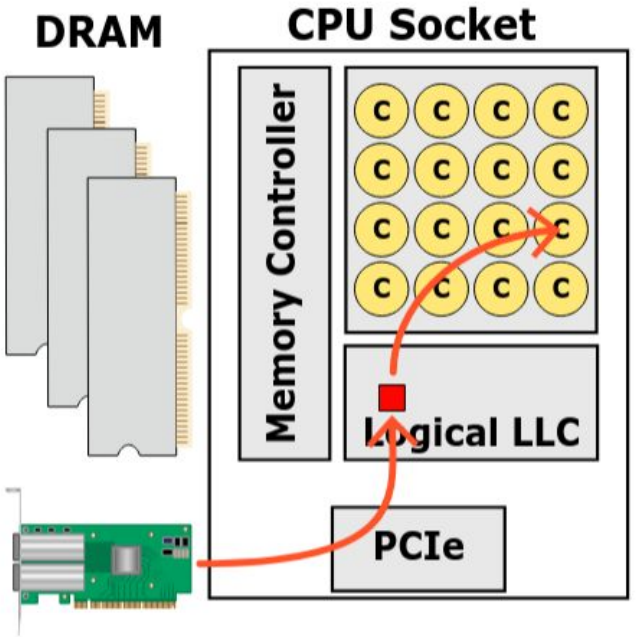


From "Reexamining Direct Cache Access to Optimize I/O Intensive Applications for Multi-hundred-gigabit Networks" ATC20

Problem Statement

Large adoption of DDIO has proved the need for cache injection the host.

At *x00 Gbps* the naiveness of DDIO becomes a double edged sword causing premature cache evictions



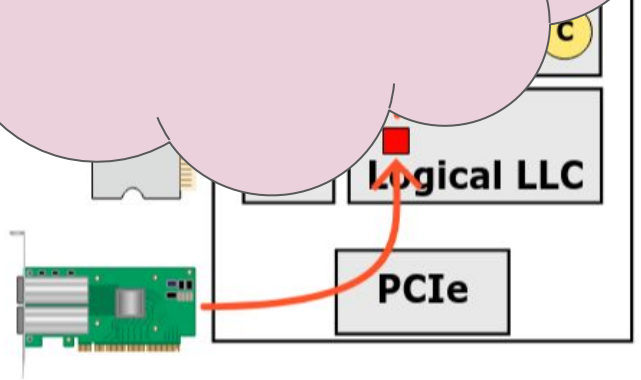
From “Reexamining Direct Cache Access to Optimize I/O Intensive Applications for Multi-hundred-gigabit Networks” ATC20

Problem Statement

Large adoption of DDIO has proved the need for cache injection the host.

At *x00 Gbps* the naiveness of DDIO becomes a double edged sword causing premature cache evictions

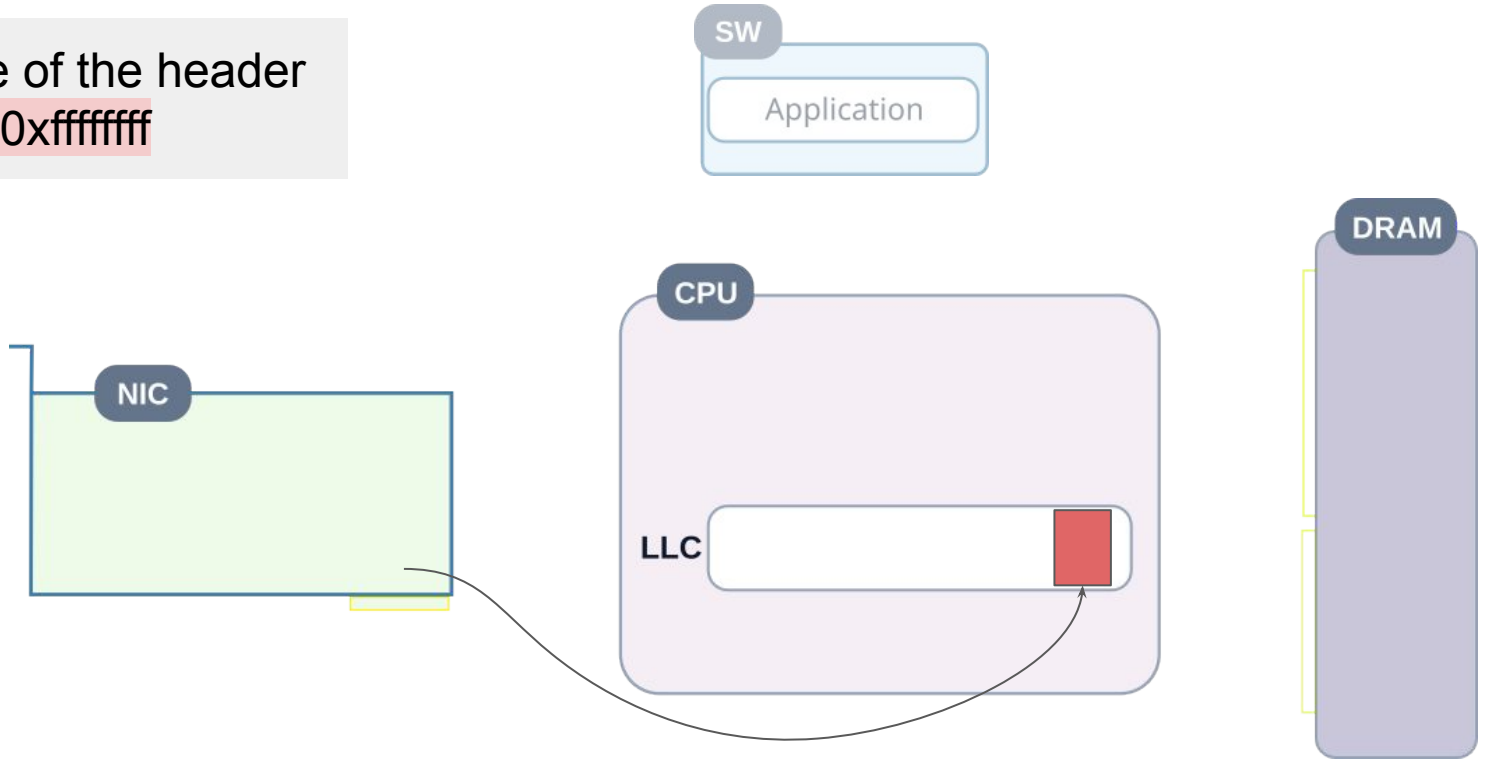
Let's govern how injection is performed via policies!



From "Reexamining Direct Cache Access to Optimize I/O Intensive Applications for Multi-hundred-gigabit Networks" ATC20

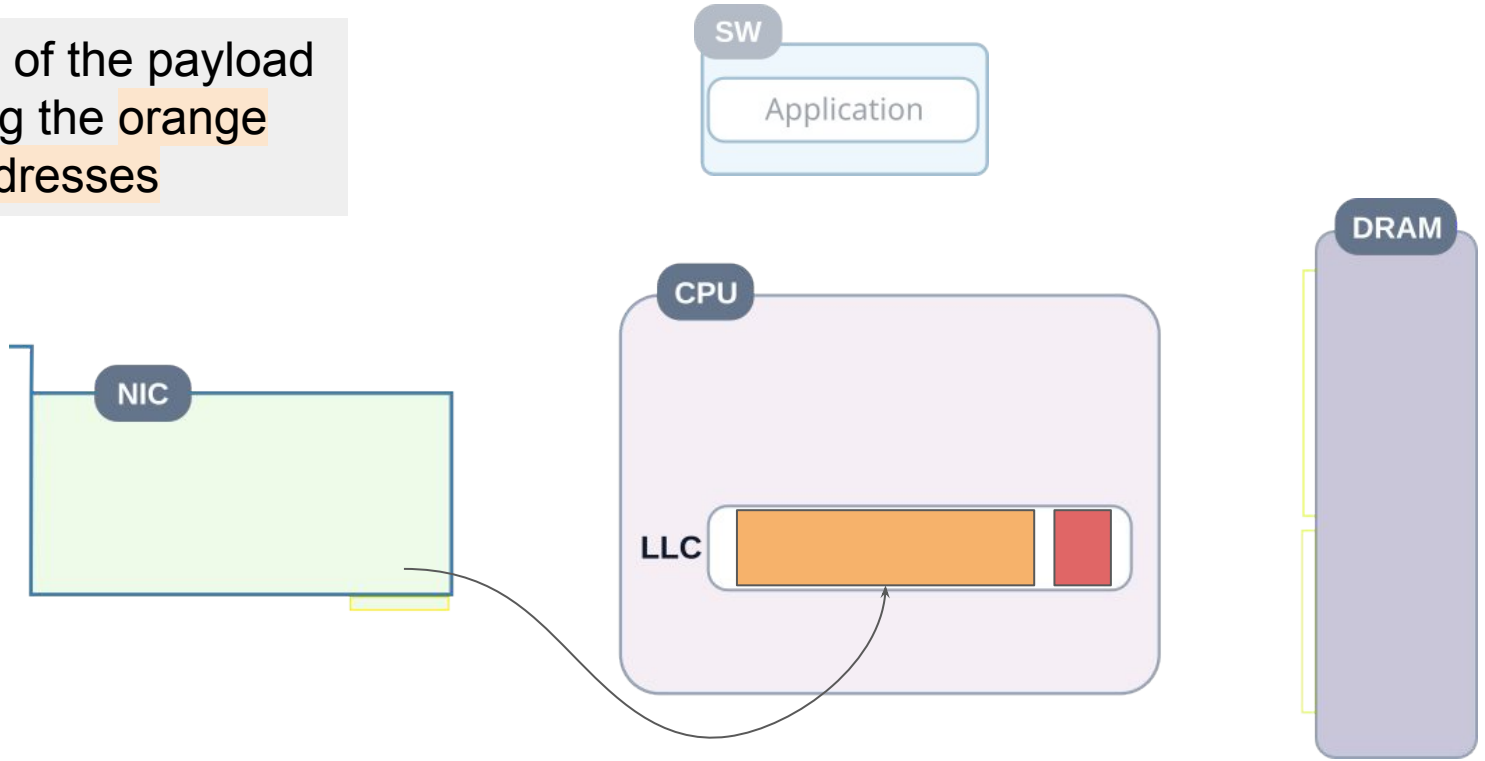
Let’s step back and do a walkthrough

DMA write of the header to 0xffffffff



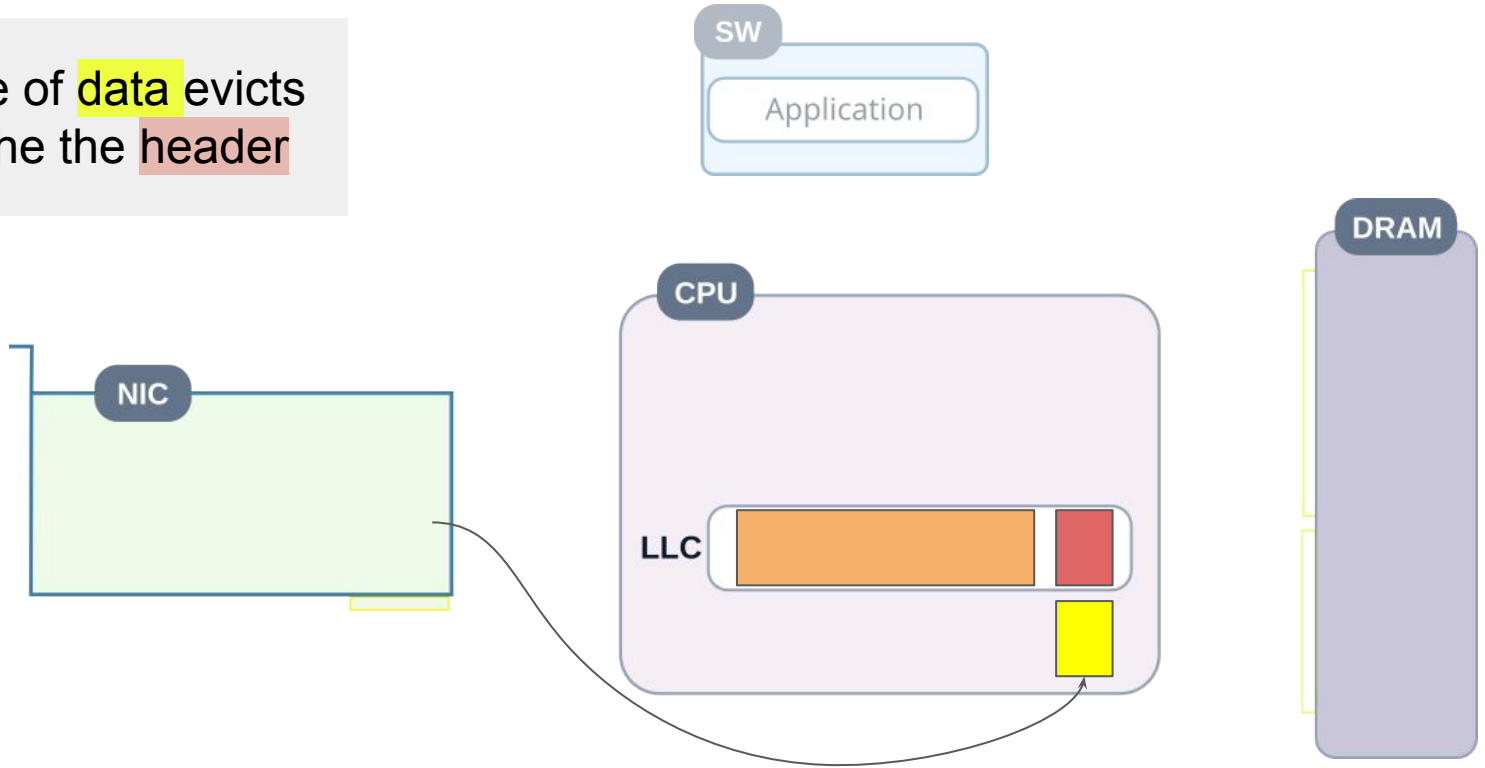
Let’s see a simple walkthrough

DMA write of the payload spanning the orange addresses



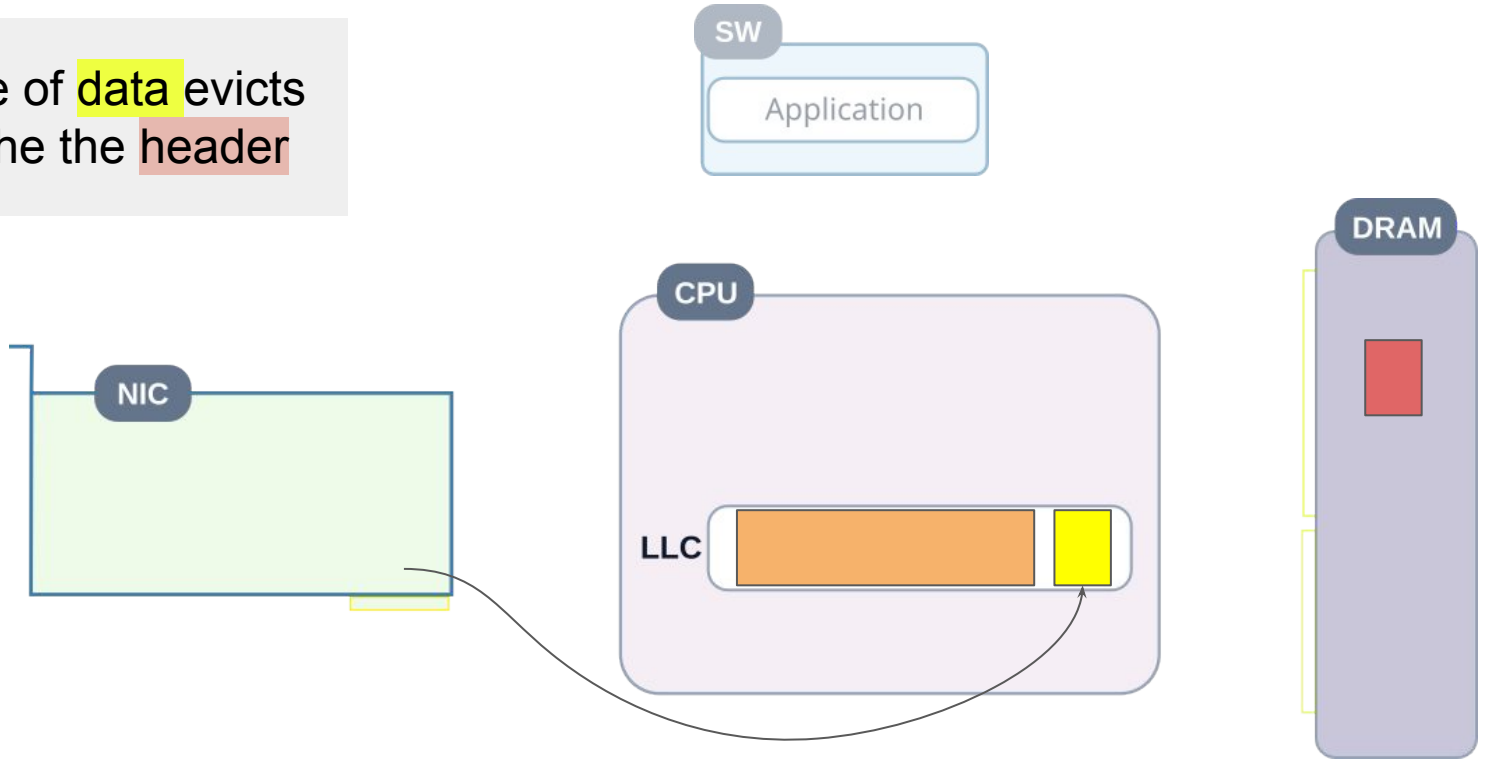
Let’s see a simple walkthrough

DMA write of data evicts from cache the header



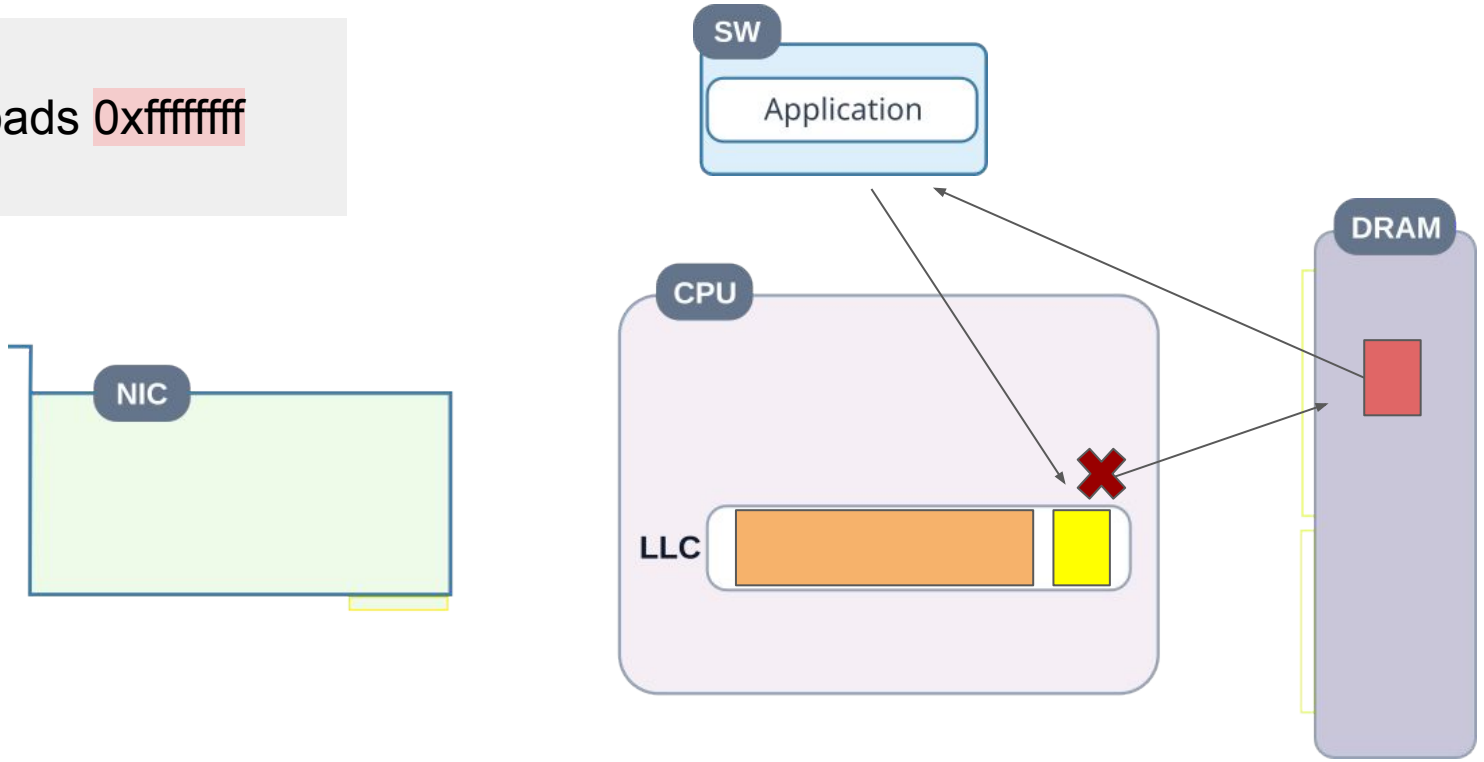
Let’s see a simple walkthrough

DMA write of data evicts from cache the header



Let’s see a simple walkthrough

CPU loads 0xffffffff



Let’s see a simple walkthrough

The header we wanted to process has been “writebacked” to RAM :(

Let’s see a simple walkthrough

The header we wanted to process has been “writebacked” to RAM :(

~~~

Applications work in FIFO

Caches work in LIFO


~~~

Standing on the shoulders of giants we see…

Reduce cache interference by exploiting the limited knobs of DDIO

A4: Microarchitecture-Aware LLC Management for Datacenter Servers with Emerging I/O Devices

Haneul Park
University of Illinois Urbana-Champaign
Urbana, USA
hnpark2@illinois.edu

Jiaqi Lou
University of Illinois Urbana-Champaign
Urbana, USA
jiaqil6@illinois.edu

Sangjin Lee
Chung-Ang University
Seoul, Republic of Korea
tkdwls0727@cau.ac.kr

Yifan Yuan
Meta
Menlo Park, USA

KyoungSoo Park
Seoul National University
Seoul, Republic of Korea

Yongseok Son
Chung-Ang University
Seoul, Republic of Korea

Don't Forget the I/O When Allocating Your LLC

Yifan Yuan[1], Mohammad Alian[2], Yipeng Wang[3], Ren Wang[3], Ilia Kurakin[3], Charlie Tai[3], Nam Sung Kim[1]
[1]*UIUC*, [2]*University of Kansas*, [3]*Intel*
{yifany3, nskim}@illinois.edu alian@ku.edu {yipeng1.wang, ren.wang, ilia.kurakin, charlie.tai}@intel.com

Standing on the shoulders of giants we see…

Reduce cache interference by exploiting the limited knobs of DDIO

HW/SW NIC interfaces that reduce the I/O working set

Disentangling the Dual Role of NIC Receive Rings

Boris Pismenny* — *EPFL and NVIDIA*
Adam Morrison — *Tel Aviv University*
Dan Tsafrir — *Technion – Israel Institute of Technology*

CEIO: A Cache-Efficient Network I/O Architecture for NIC-CPU Data Paths

Bowen Liu* Xinyang Huang* Qijing Li

ShRing: Networking with Shared Receive Rings

Boris Pismenny†◇ Adam Morrison‡ Dan Tsafrir†±

Standing on the shoulders of giants we see…

Reduce cache interference by exploiting the limited knobs of DDIO

HW/SW NIC interfaces that reduce the I/O working set

New HW design that allows for more injection targets

IDIO: Network-Driven, Inbound Network Data Orchestration on Server Processors

Mohammad Alian[1], Siddharth Agarwal[2], Jongmin Shin[3], Neel Patel[1], Yifan Yuan[2], Daehoon Kim[3], Ren Wang[4]
Nam Sung Kim[2]
[1]*University of Kansas,* [2]*University of Illinois, Urbana-Champaign,* [3]*DGIST,* [4]*Intel Labs*

AMD

Smart Data Cache Injection (SDCI) White Paper

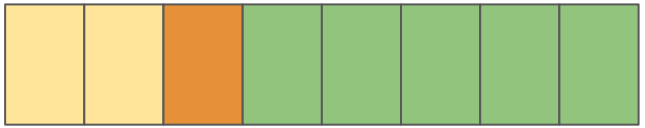
Some bytes are more equal than others

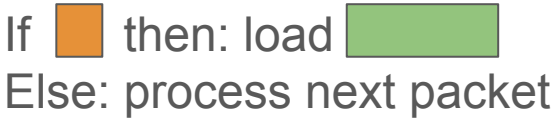
- Applications touch at different time different packet regions: headers, prefixes, flags, payloads.
- Example: NFV needs headers; RDMA payload does not need CPU cache.

DDIO assumes the whole packet is accessed at the same time

What are the applications even doing?

- Access patterns depend on application semantics and packet contents.

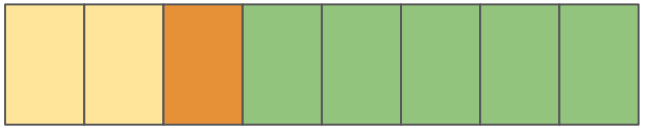




DDIO assumes the applications will process the entire packet

What are the applications even doing?

- Access patterns depend on application semantics and packet contents.
- Prefetcher-like hardware cannot reliably know this.
  - Even if it could we the area/power in a CPU is a scarce resource



If  then: load 
Else: process next packet

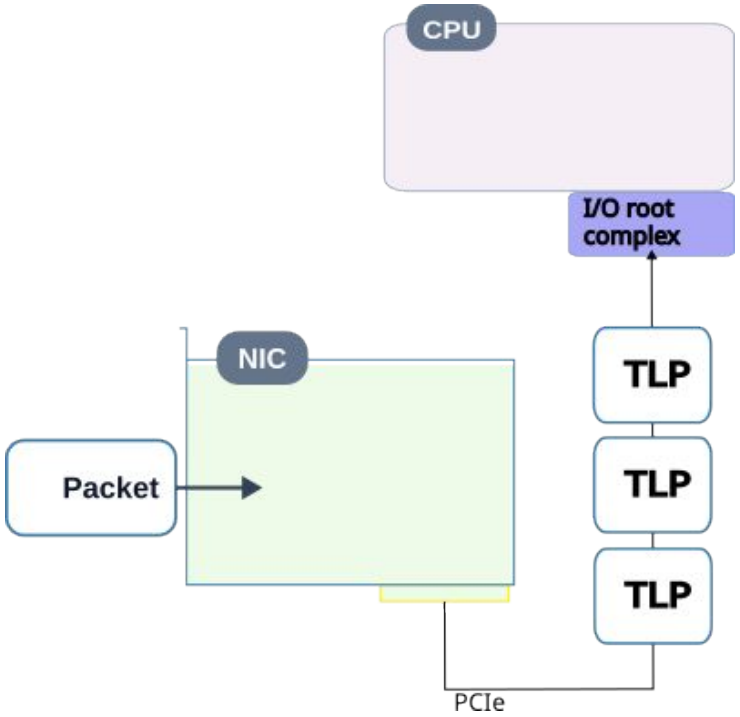
What are the applications even doing? Let's ask them!

Let's have *application defined* injection policies to accurately predict the working set!

But how?

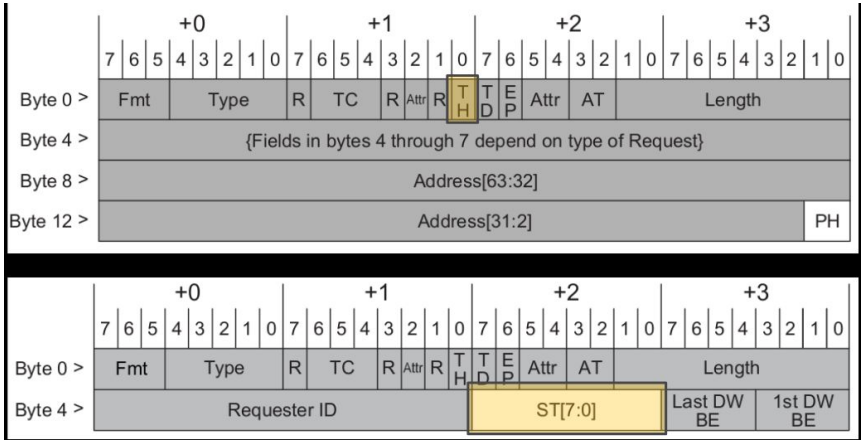
What are the applications even doing? Let's ask them!

- NIC-CPU communication is done via PCIe packets: **T**ransaction **L**ayer **P**ackets



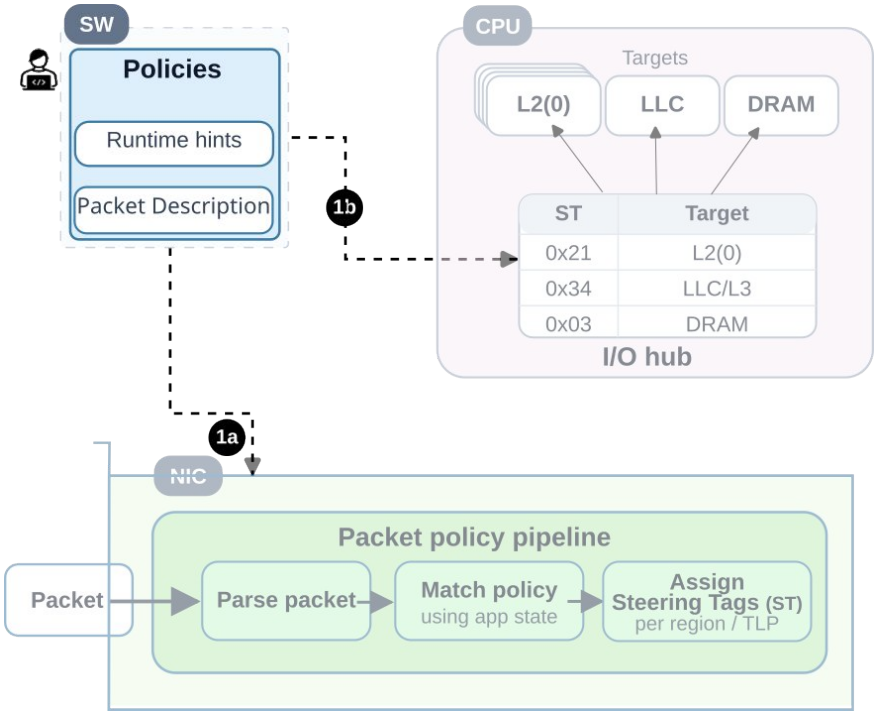
What are the applications even doing? Let’s ask them!

- NIC-CPU communication is done via PCie packets: **T**ransaction **L**ayer **P**ackets
- TLP Processing Hints as the coordination system between CPU-NIC-IO Complex



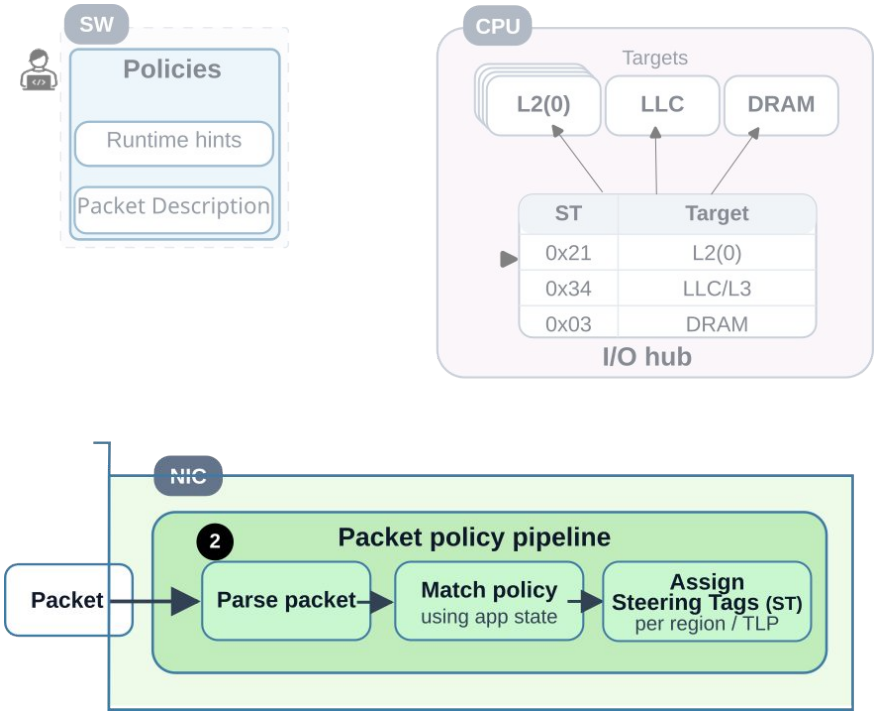
Stateless in the CPU, Tasteful in the NIC

1. Applications dictate the policies
   a. The application defines packet descriptions and placement policies
   b. Programs an indirect steering-tag table



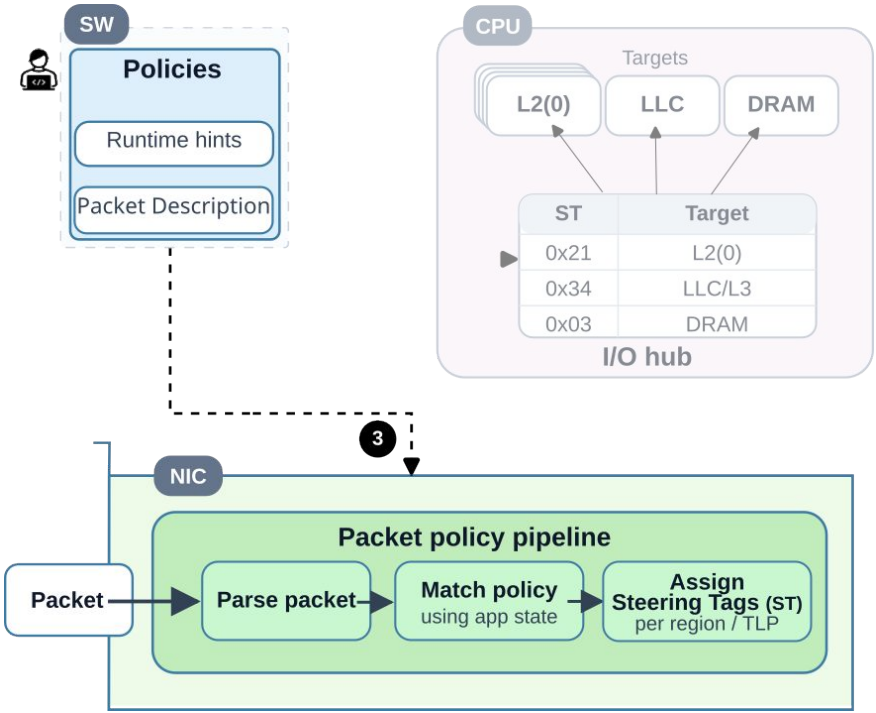
Stateless in the CPU, Tasteful in the NIC

2. Parse/match/action pipeline able to inspect the packet and attach steering tags to outgoing PCIe transactions



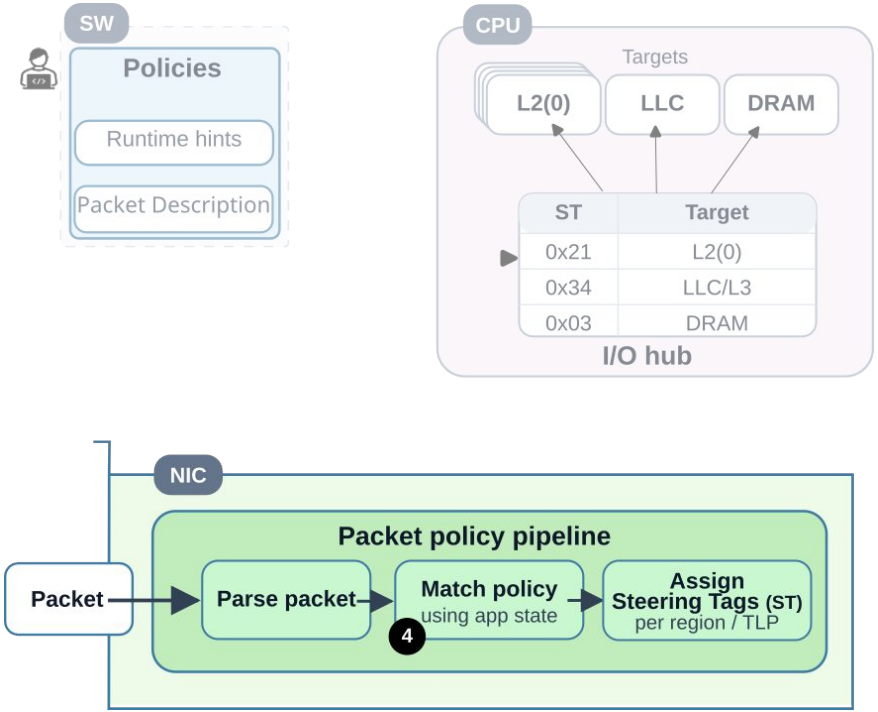
Stateless in the CPU, Tasteful in the NIC

3. At runtime, the application can export additional hints to the NIC, such as the set of hot entries currently handled by a fast path



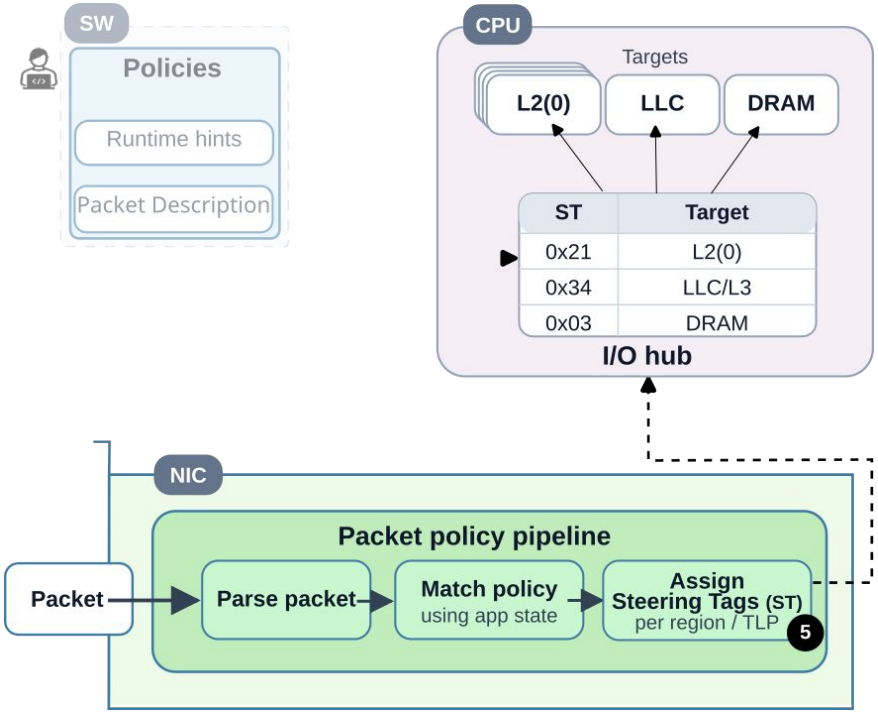
Stateless in the CPU, Tasteful in the NIC

4. Determine the ST to attach to each region of the packet according to ❶ ❷ ❸



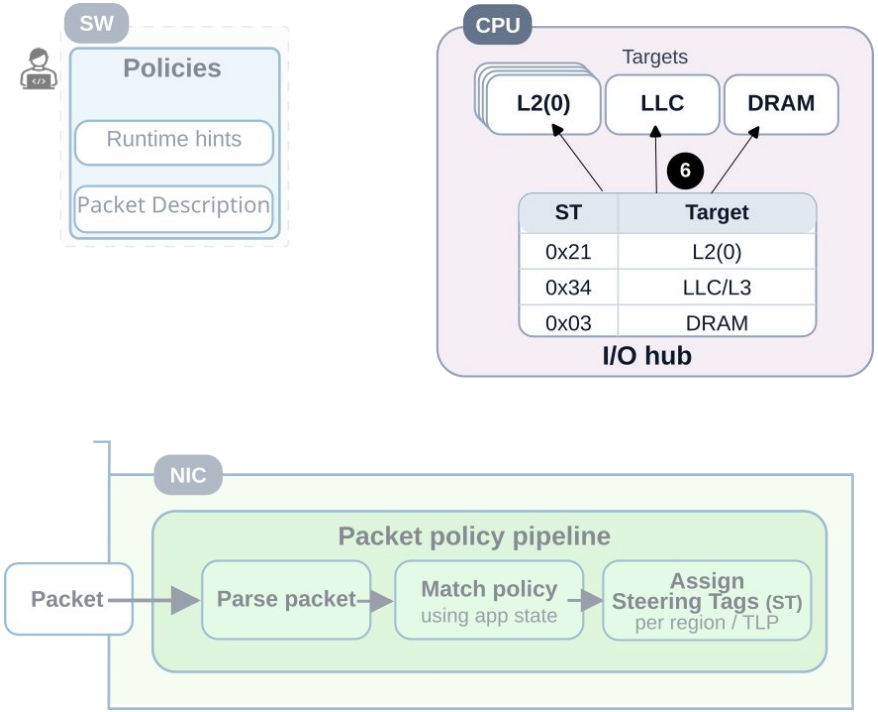
Stateless in the CPU, Tasteful in the NIC

5. The NIC then emits PCIe TLPs annotated with the selected steering tags



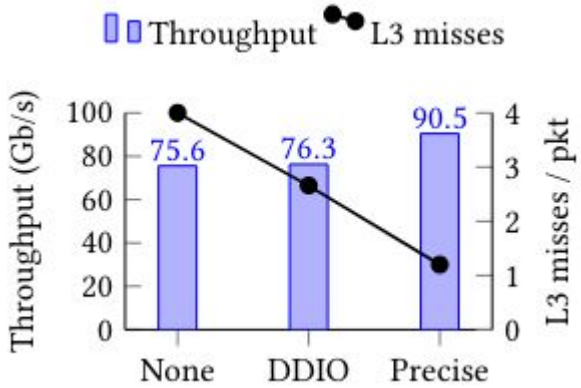
Stateless in the CPU, Tasteful in the NIC

6. The I/O root complex resolves each tag through the configured mapping and injects the corresponding data into the selected destination



Preliminary results and challenges

- Having better accuracy actually helps
- On gem5 simulations of synthetic NFV we have a 20-30% improvement depending on the workload



Preliminary results and challenges

1. The I/O root complex we describe does not quite exists in reality -> *gem5 simulation + validation on real but slightly different HW*

Preliminary results and challenges

1. The I/O root complex we describe does not quite exists in reality -> *gem5 simulation + validation on real but slightly different HW*

2. Writing bad policies will worsen performance -> *Having tight HW/SW control loops may be necessary (CXL may come in handy here)*

Preliminary results and challenges

1. The I/O root complex we describe does not quite exists in reality -> *gem5 simulation + validation on real but slightly different HW*

2. Writing bad policies will worsen performance -> *Having tight HW/SW control loops may be necessary (CXL may come in handy here)*

3. Generalization of the idea on CXL hardware -> *If you have a CXL-capable FPGA please have a chat :)*

~~Beer?~~

Questions?

Standing on the shoulders of giants we see…

Reduce cache interference by exploiting the limited knobs of DDIO

A4: Microarchitecture-Aware LLC Management for Datacenter Servers with Emerging I/O Devices

Haneul Park
University of Illinois Urbana-Champaign
Urbana, USA
hnpark2@illinois.edu

Jiaqi Lou
University of Illinois Urbana-Champaign
Urbana, USA
jiaqil6@illinois.edu

Sangjin Lee
Chung-Ang University
Seoul, Republic of Korea
tkdwls0727@cau.ac.kr

Don't Forget the I/O When Allocating Your LLC

Yifan Yuan[1], Mohammad Alian[2], Yipeng Wang[3], Ren Wang[3], Ilia Kurakin[3], Charlie Tai[3], Nam Sung Kim[1]
[1]*UIUC*, [2]*University of Kansas*, [3]*Intel*
{yifany3, nskim}@illinois.edu alian@ku.edu {yipeng1.wang, ren.wang, ilia.kurakin, charlie.tai}@intel.com

Standing on the shoulders of giants we see…

HW/SW NIC interfaces that reduce the I/O working set

Disentangling the Dual Role of NIC Receive Rings

Boris Pismenny* Adam Morrison Dan Tsafrir
EPFL and NVIDIA *Tel Aviv University* *Technion – Israel Institute of Technology*

CEIO: A Cache-Efficient Network I/O Architecture for NIC-CPU Data Paths

Bowen Liu* Xinyang Huang* Qijing Li

ShRing: Networking with Shared Receive Rings

Boris Pismenny†◇ Adam Morrison‡ Dan Tsafrir†±

Standing on the shoulders of giants we see…

New HW design that allows for more injection targets

IDIO: Network-Driven, Inbound Network Data Orchestration on Server Processors

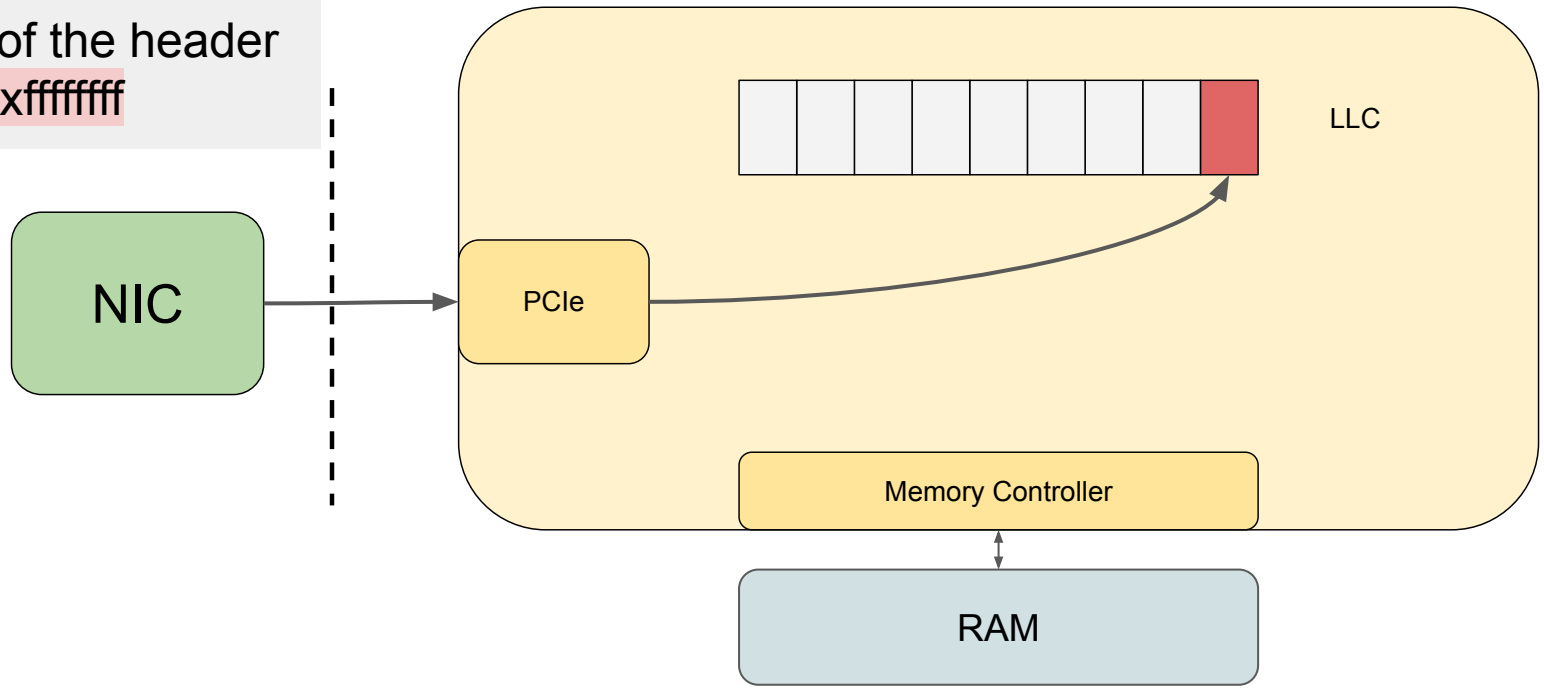
Mohammad Alian[1], Siddharth Agarwal[2], Jongmin Shin[3], Neel Patel[1], Yifan Yuan[2], Daehoon Kim[3], Ren Wang[4]
Nam Sung Kim[2]
[1]*University of Kansas,* [2]*University of Illinois, Urbana-Champaign,* [3]*DGIST,* [4]*Intel Labs*

AMD

Smart Data Cache Injection (SDCI) White Paper

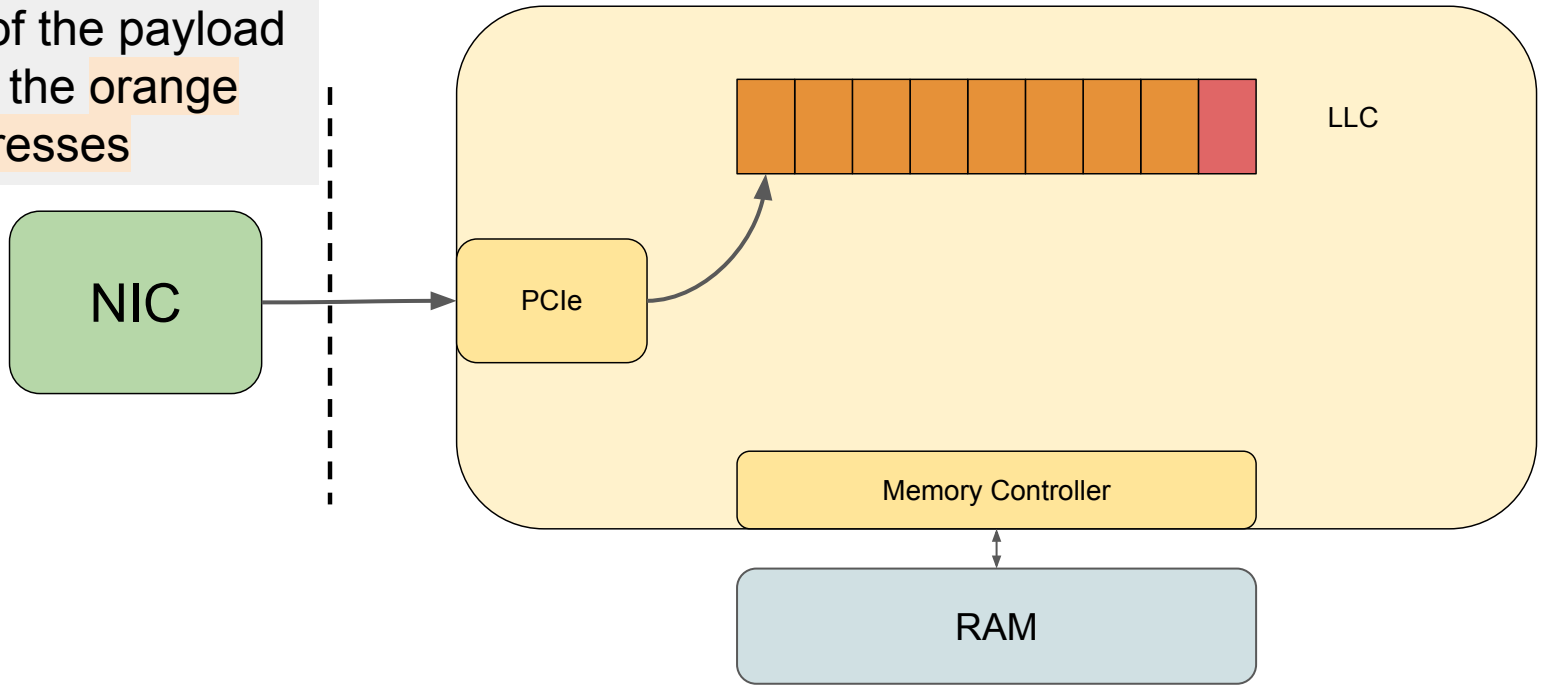
Let’s see a simple walkthrough

DMA write of the header to 0xffffffff



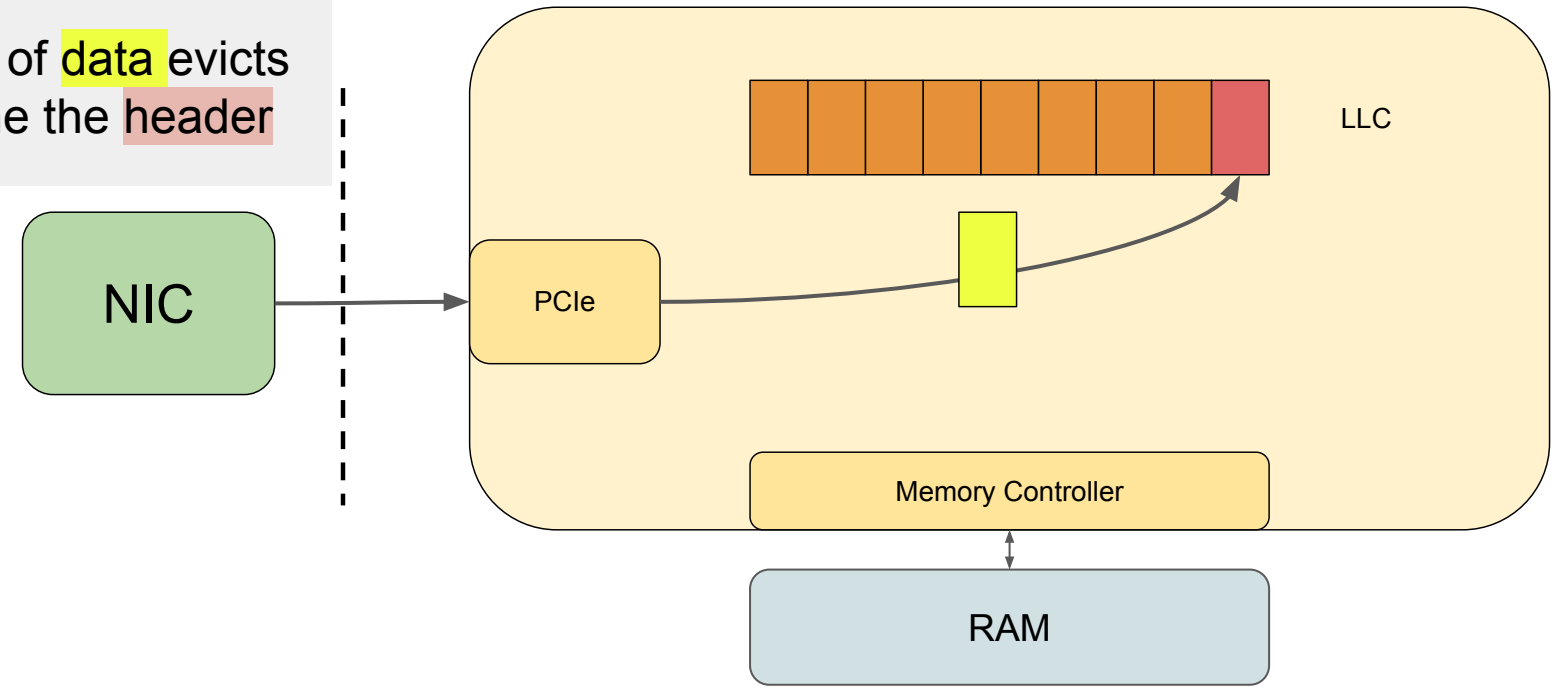
Let’s see a simple walkthrough

DMA write of the payload spanning the orange addresses



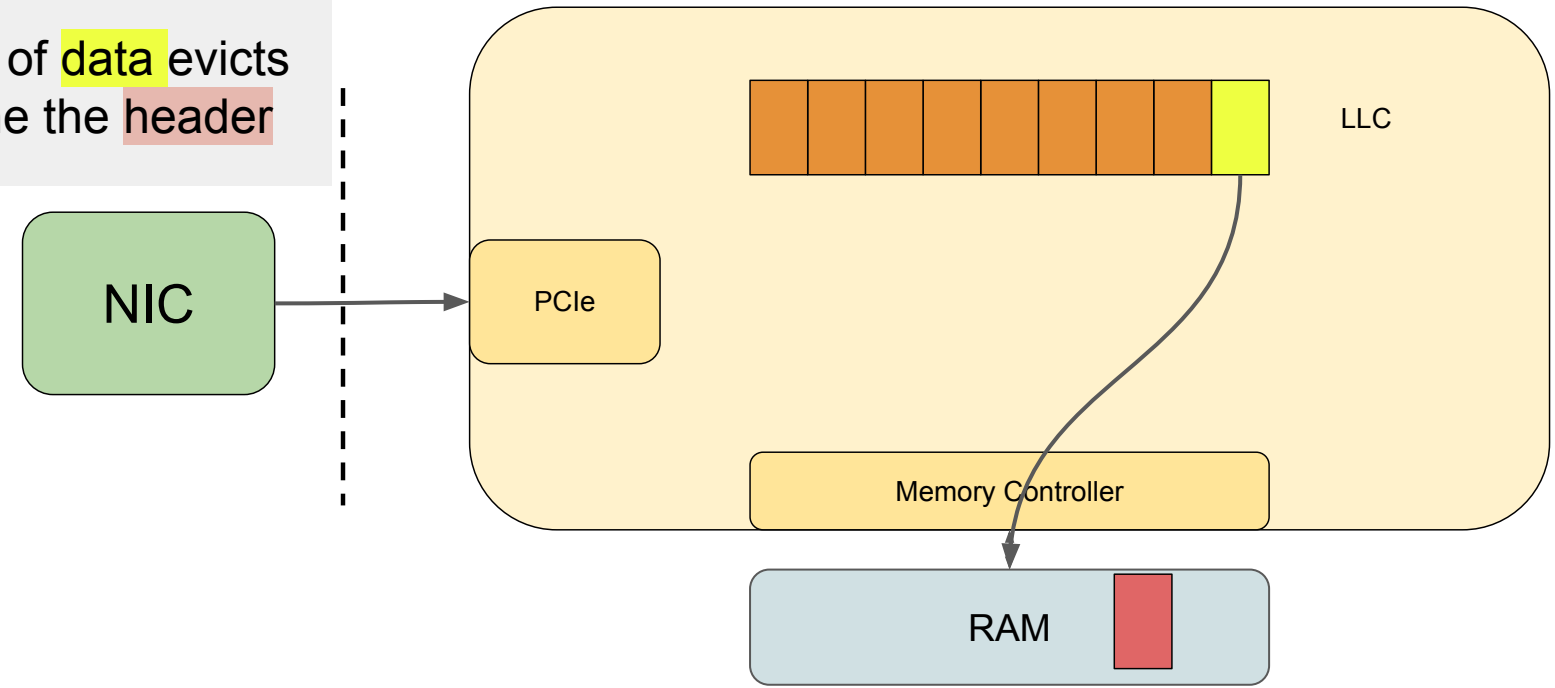
Let’s see a simple walkthrough

DMA write of data evicts from cache the header

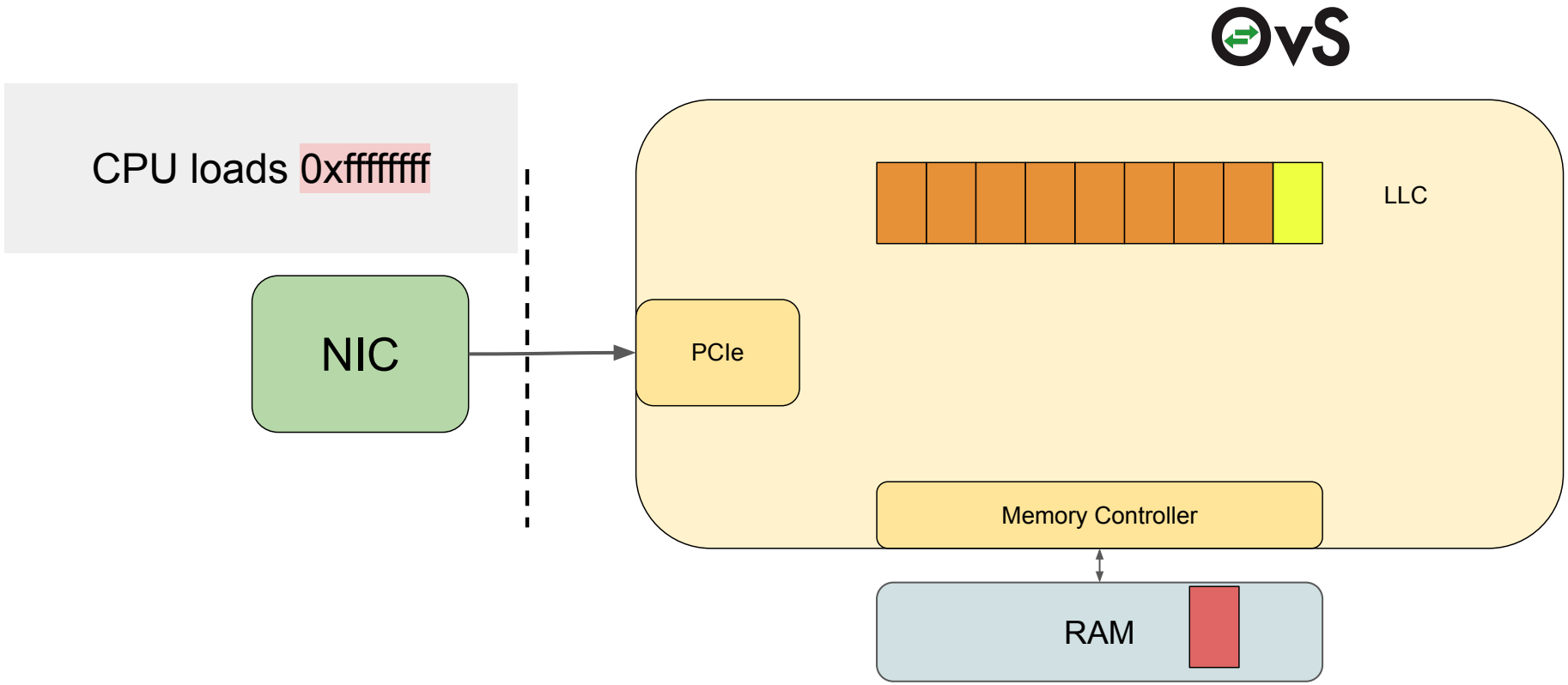


Let’s see a simple walkthrough

DMA write of data evicts from cache the header

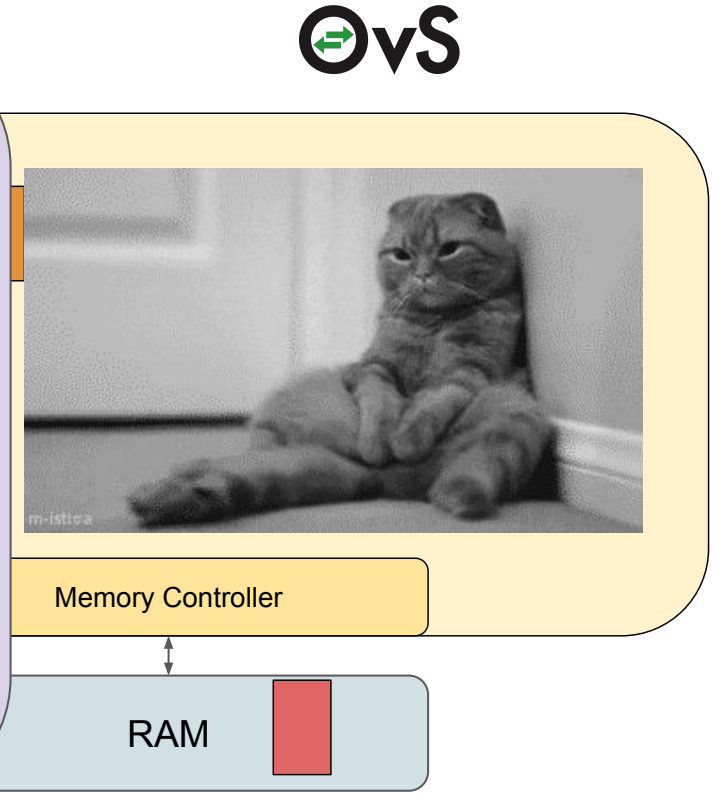


Let’s see a simple walkthrough



Let’s see a simple walkthrough

The header we wanted to process has been “writebacked” to RAM :(



Let’s see a simple walkthrough

The header we wanted to process has been “writebacked” to RAM :(

~~~

Applications work in FIFO

Caches work in LIFO

Memory Controller

RAM
~~~