# Ensō

## A Streaming Interface for NIC-Application Communication

**Hugo Sadok**, Nirav Atre, Zhipeng Zhao, Daniel S. Berger, James C. Hoe, Aurojit Panda, Justine Sherry, Ren Wang

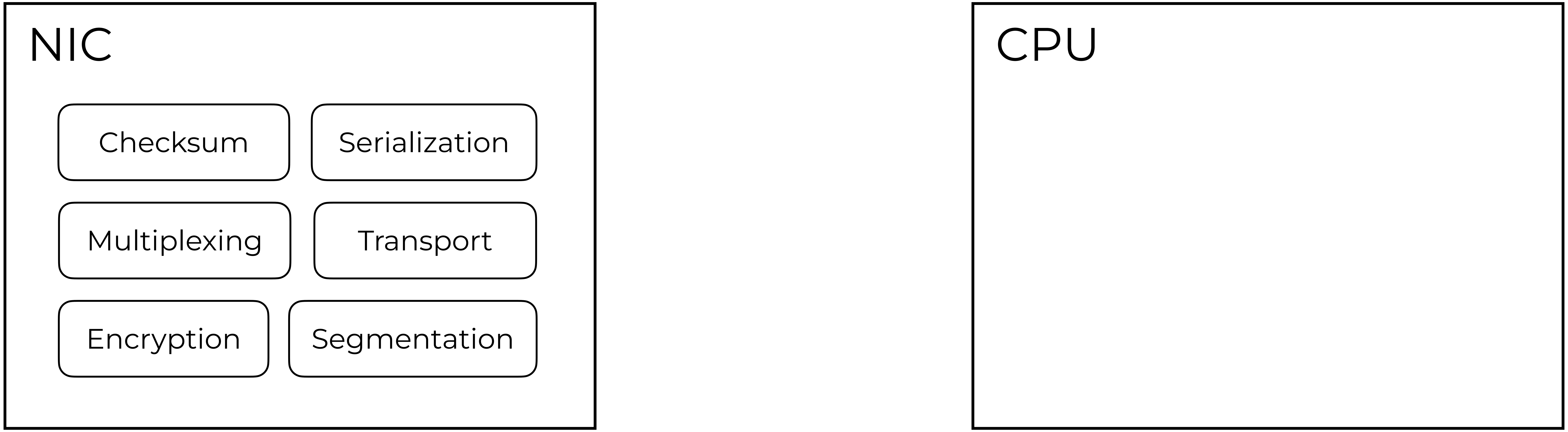
Carnegie Mellon University
NYU
Microsoft
UNIVERSITY of WASHINGTON
intel

# Two trends in high-performance networking

NIC

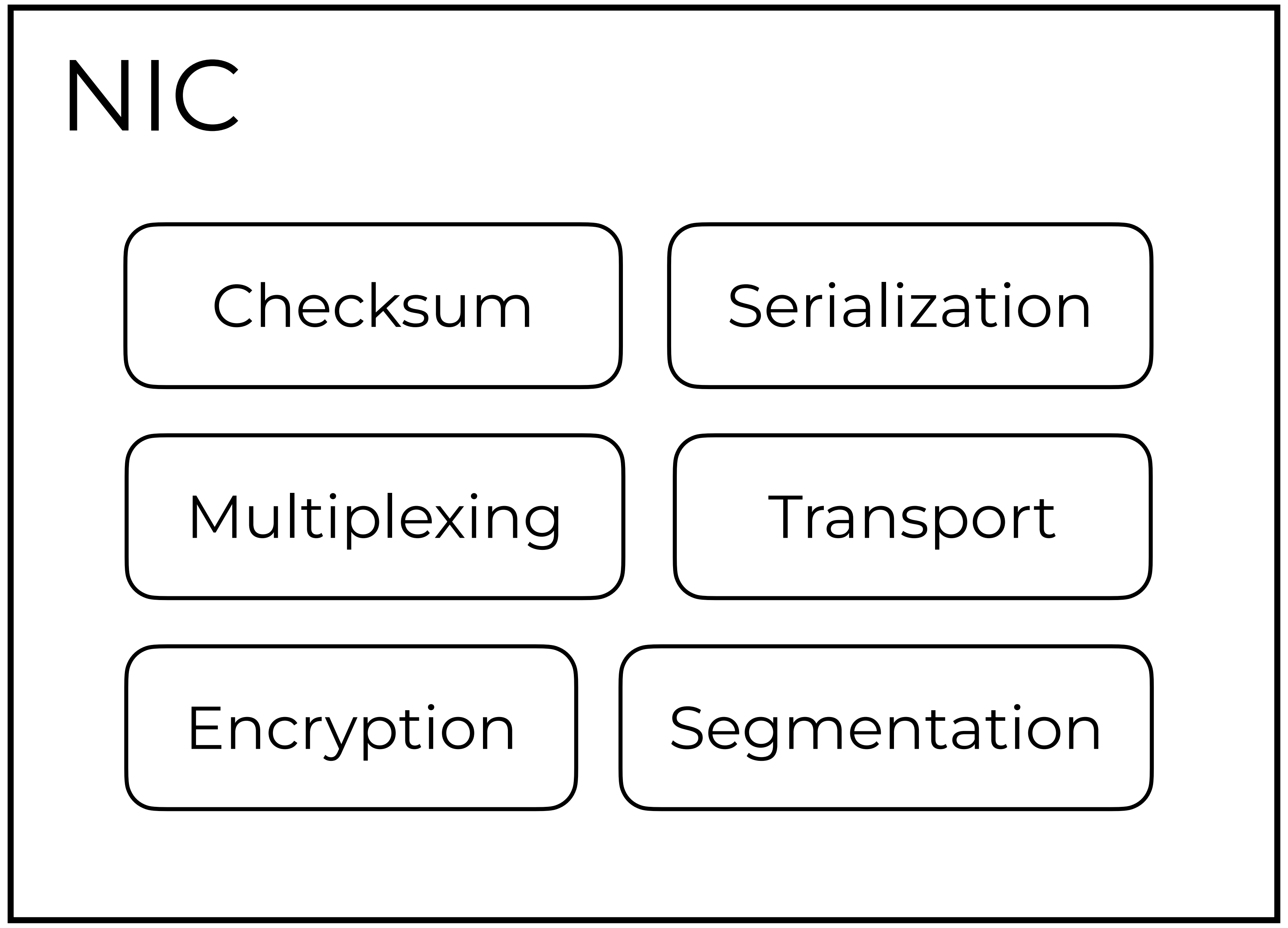
CPU

# Two trends in high-performance networking



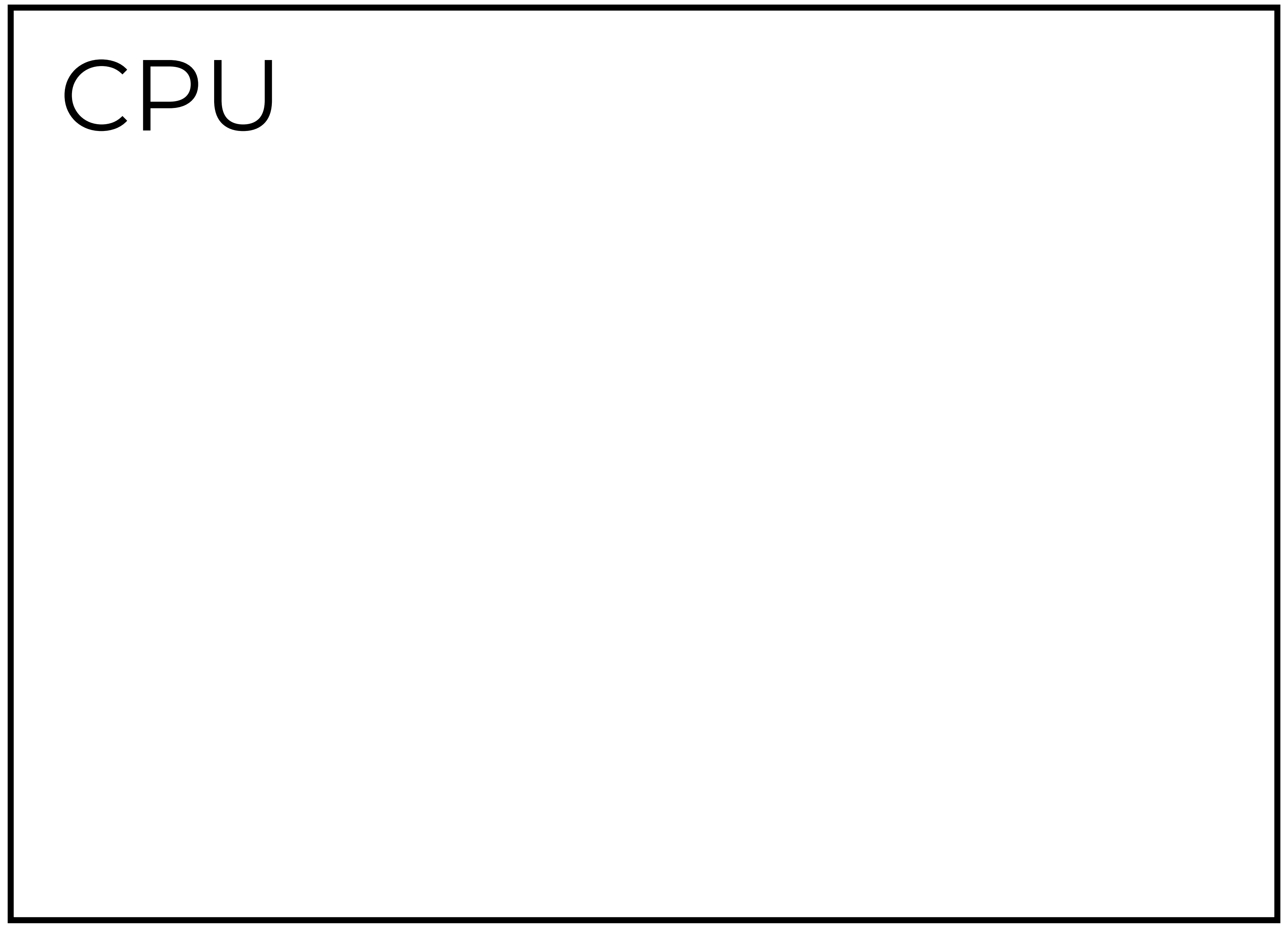

NIC offloads

# Two trends in high-performance networking




NIC offloads

Offloads operate at
higher network layers



# Two trends in high-performance networking

**NIC**

- Checksum
- Serialization
- Multiplexing
- Transport
- Encryption
- Segmentation

NIC offloads

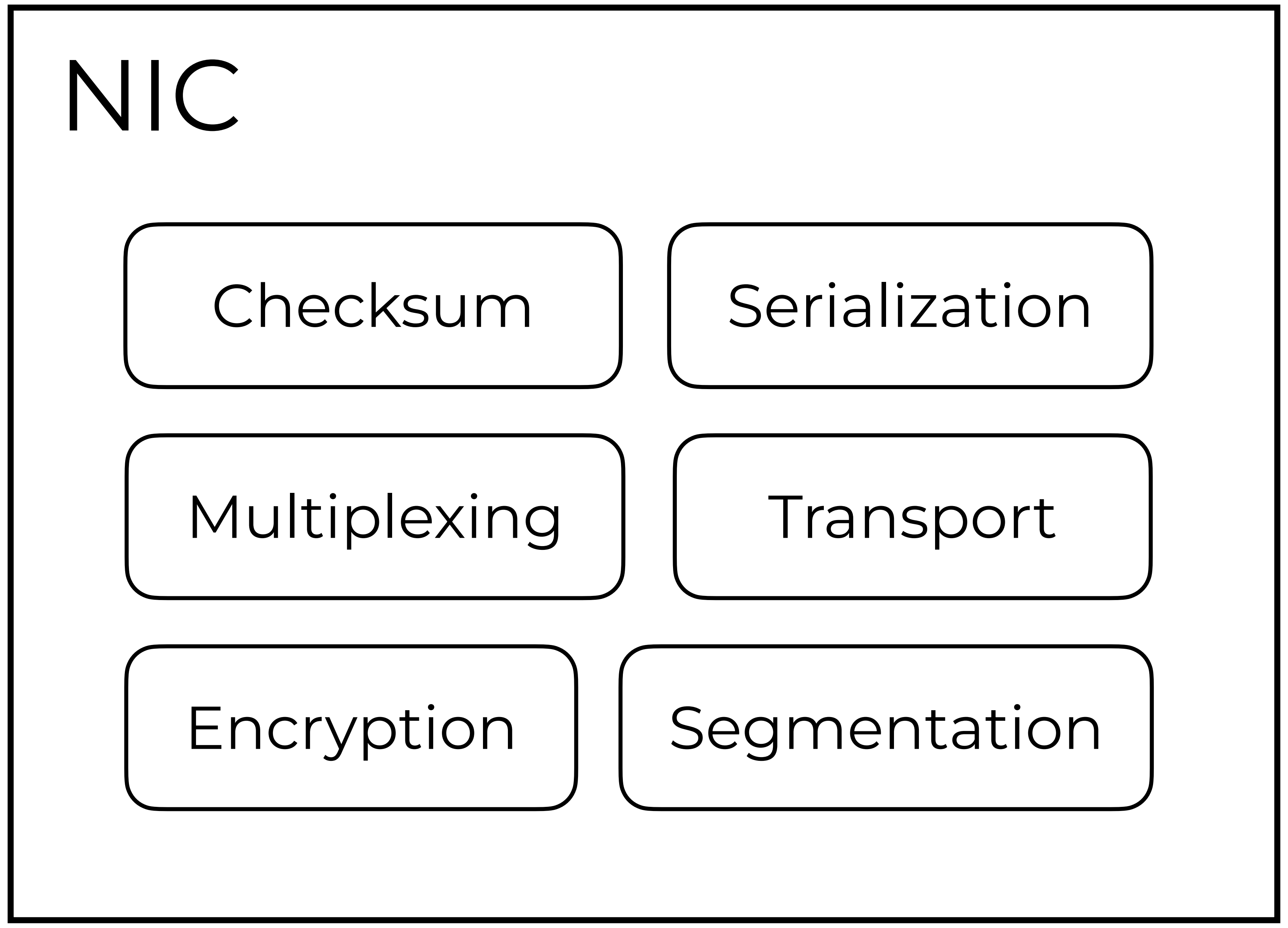
**CPU**

- DPDK
- netmap
- io_uring
- Arrakis
- mTCP
- TAS
- IX

Efficient network stacks

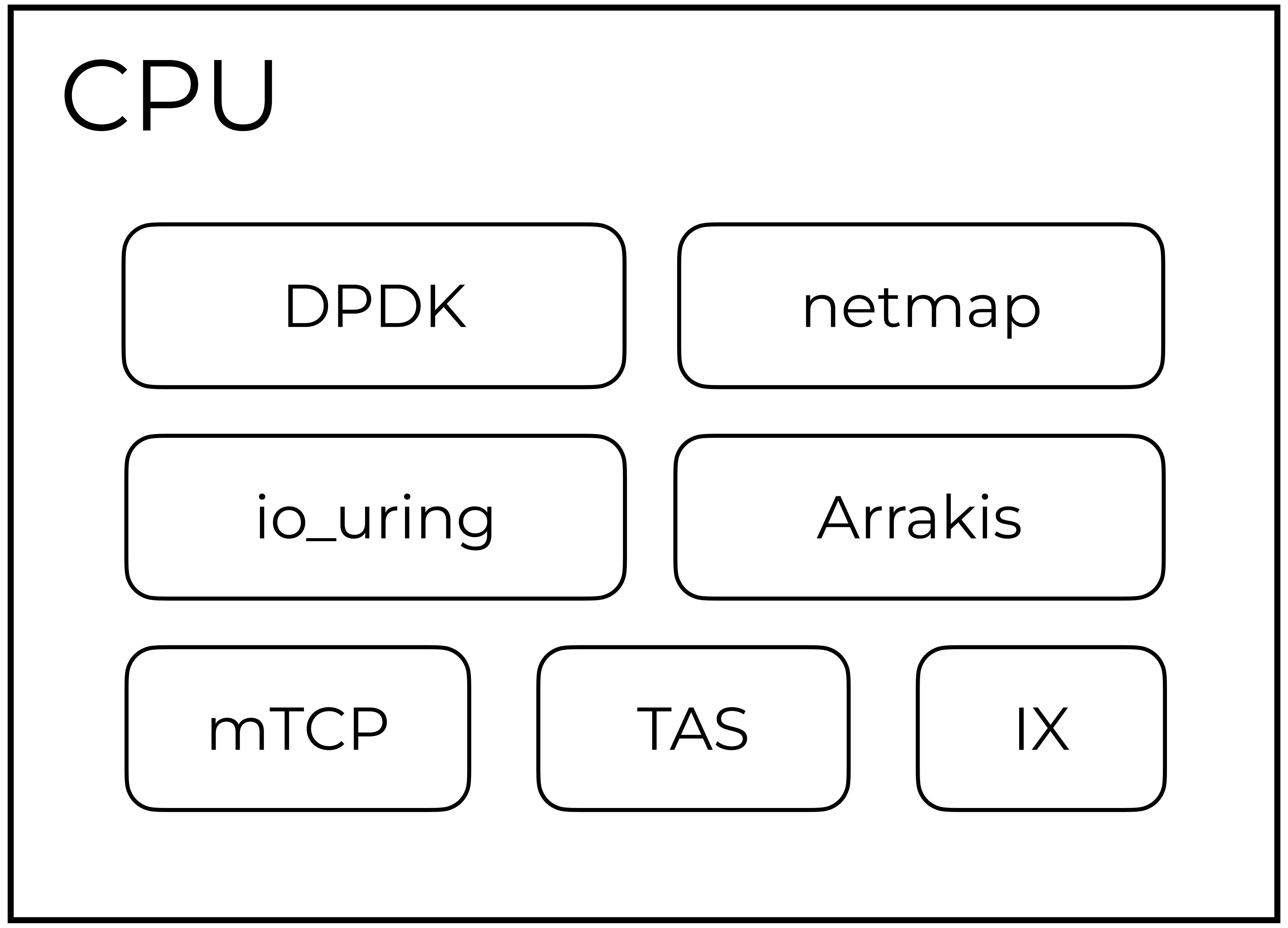
Offloads operate at
higher network layers

# Two trends in high-performance networking




NIC offloads

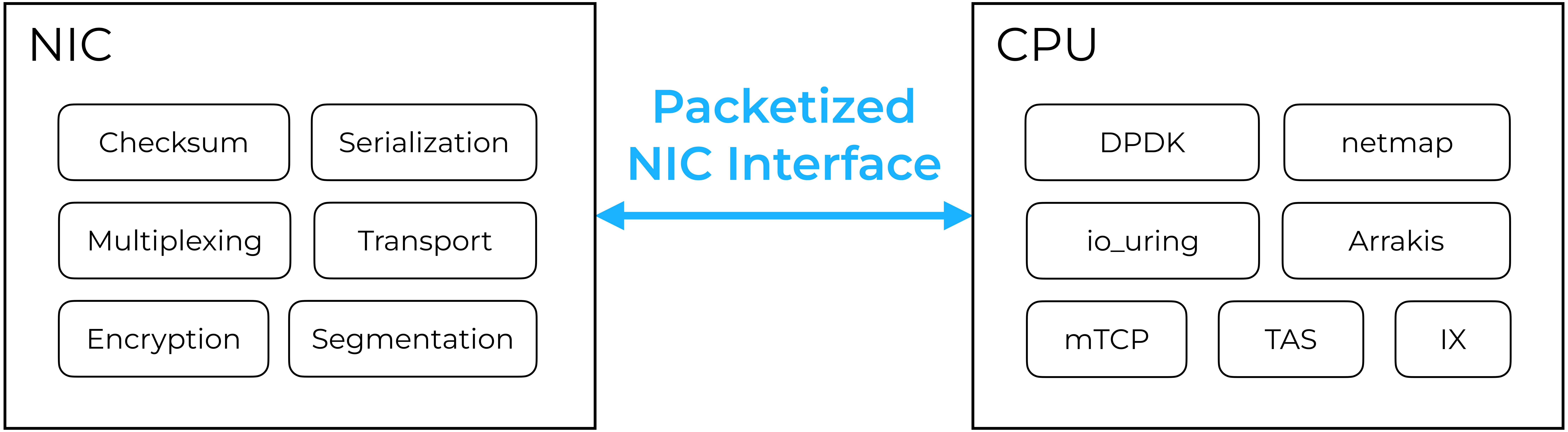
Offloads operate at
higher network layers




Efficient network stacks

Often bypass the kernel
and rely on batching

# Two trends in high-performance networking




NIC offloads

Efficient network stacks

Offloads operate at higher network layers

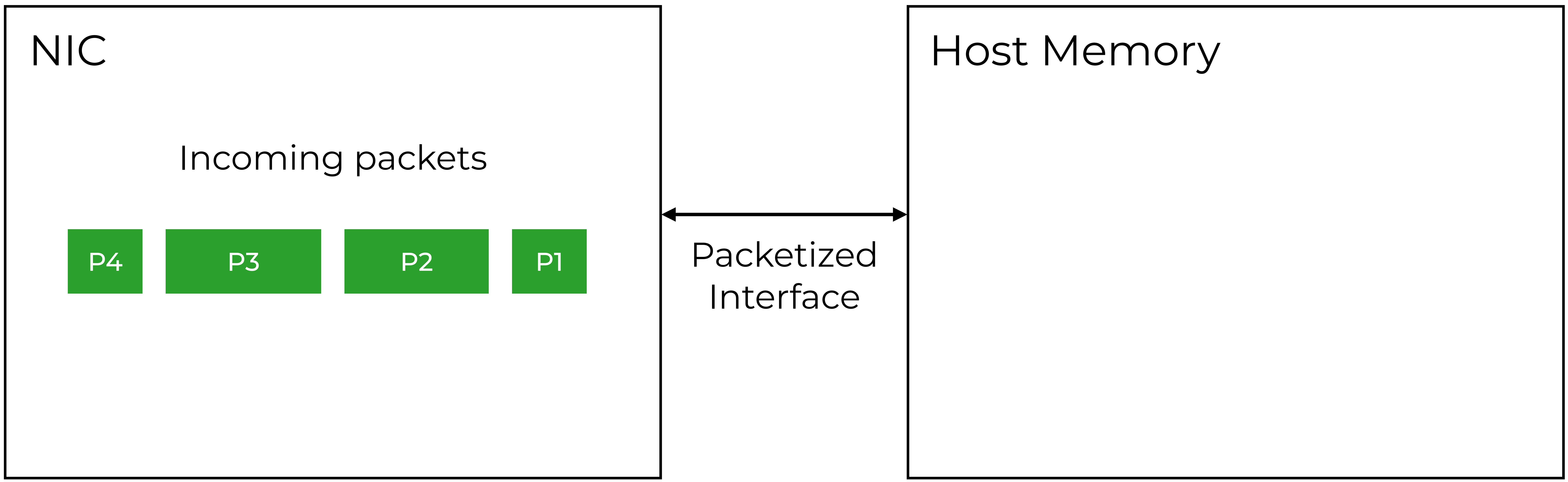
Often bypass the kernel and rely on batching

# This Talk:

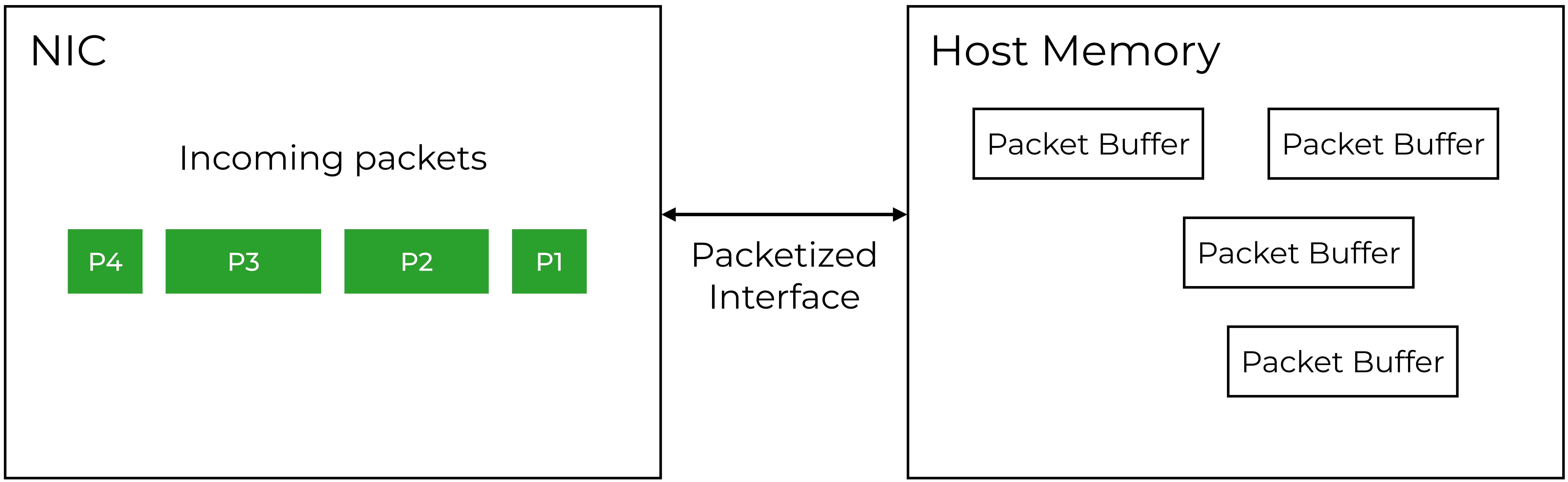
① **Mismatch** between how NICs are **used** and the **interface** that they provide

② Fixing this mismatch can significantly **improve performance** while paving the way for **higher-level offloads**

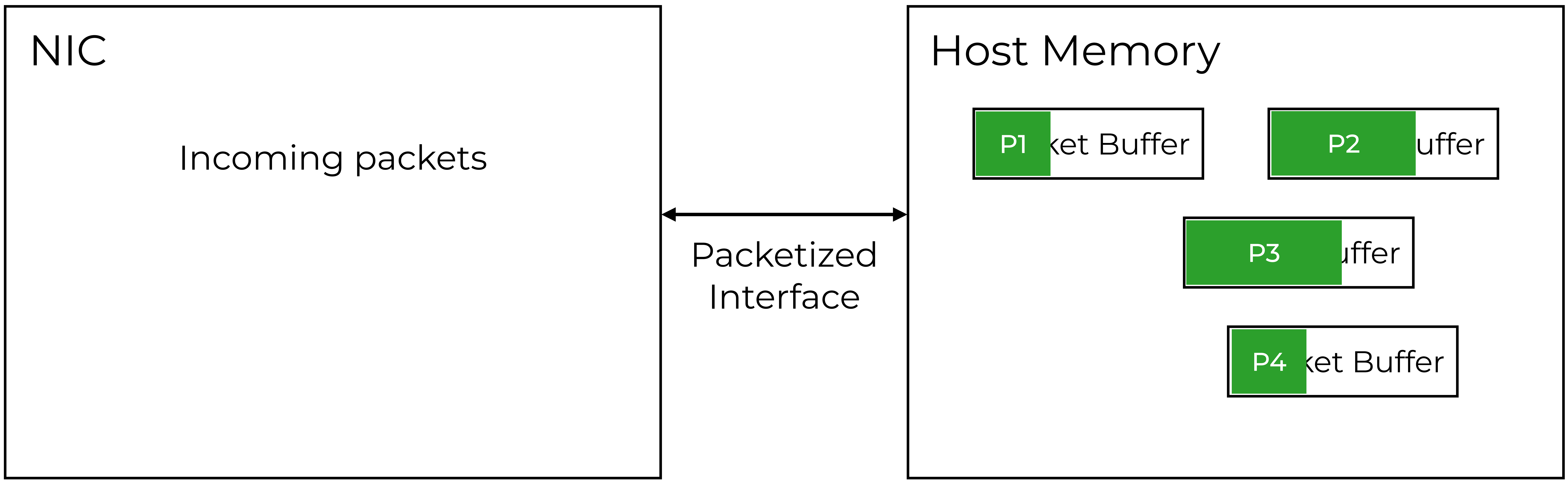
# Existing NICs provide a packetized interface



# Existing NICs provide a packetized interface

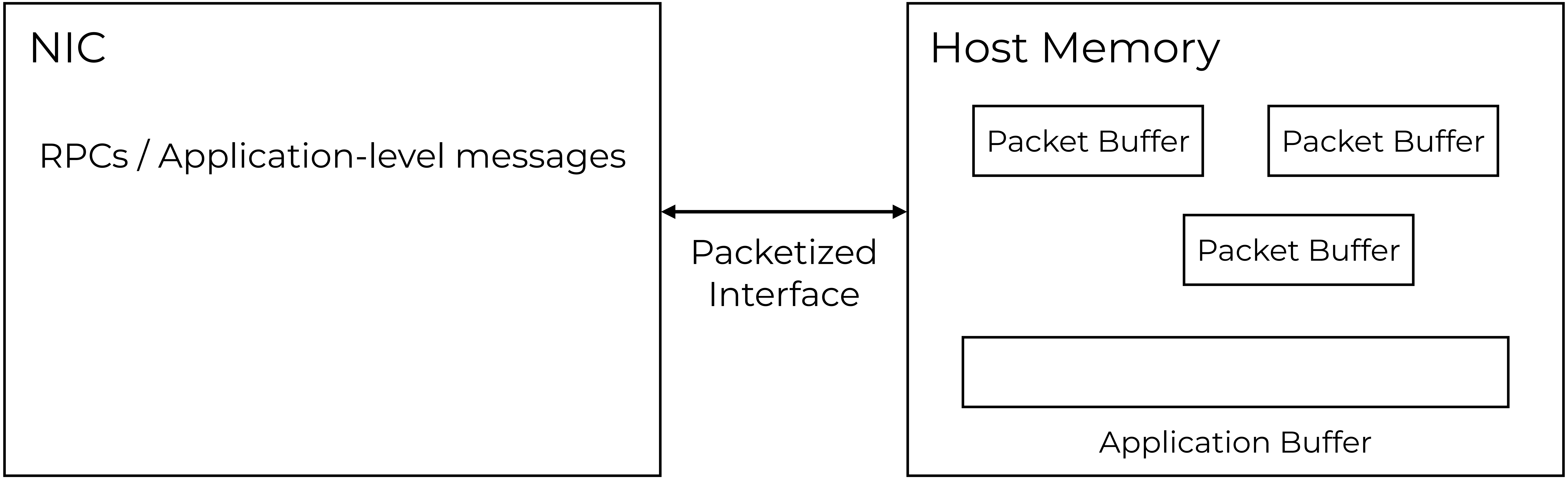


# Existing NICs provide a **packetized** interface



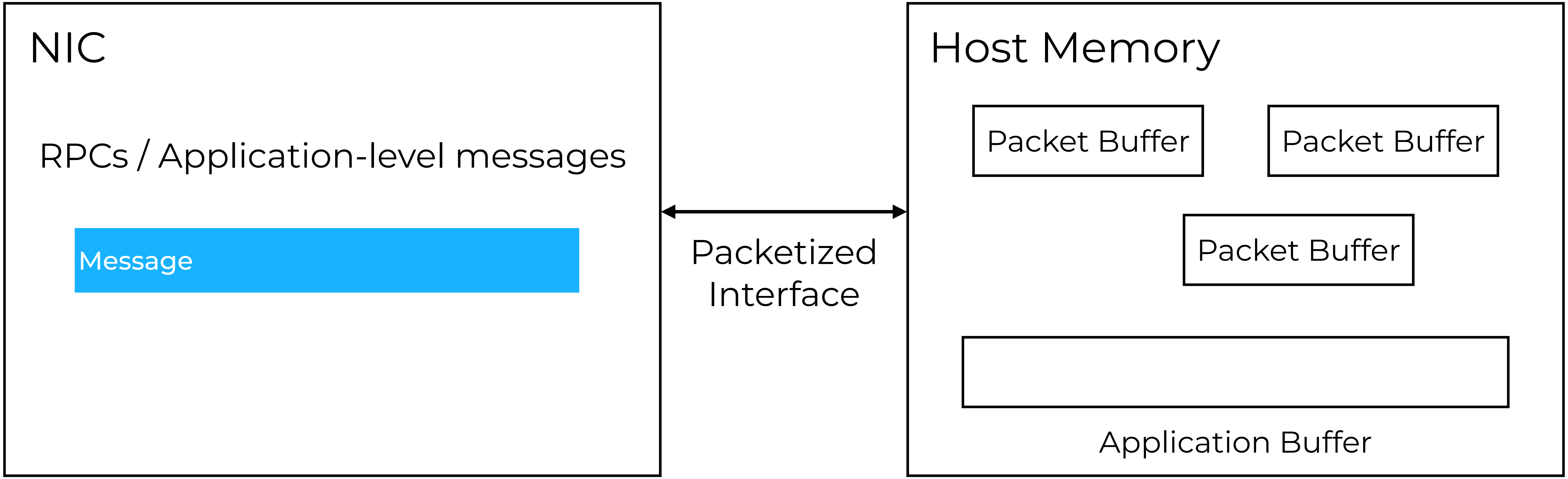
# Problem #1 Packetized Abstraction

Packetized abstraction is unsuitable for higher-level offloads



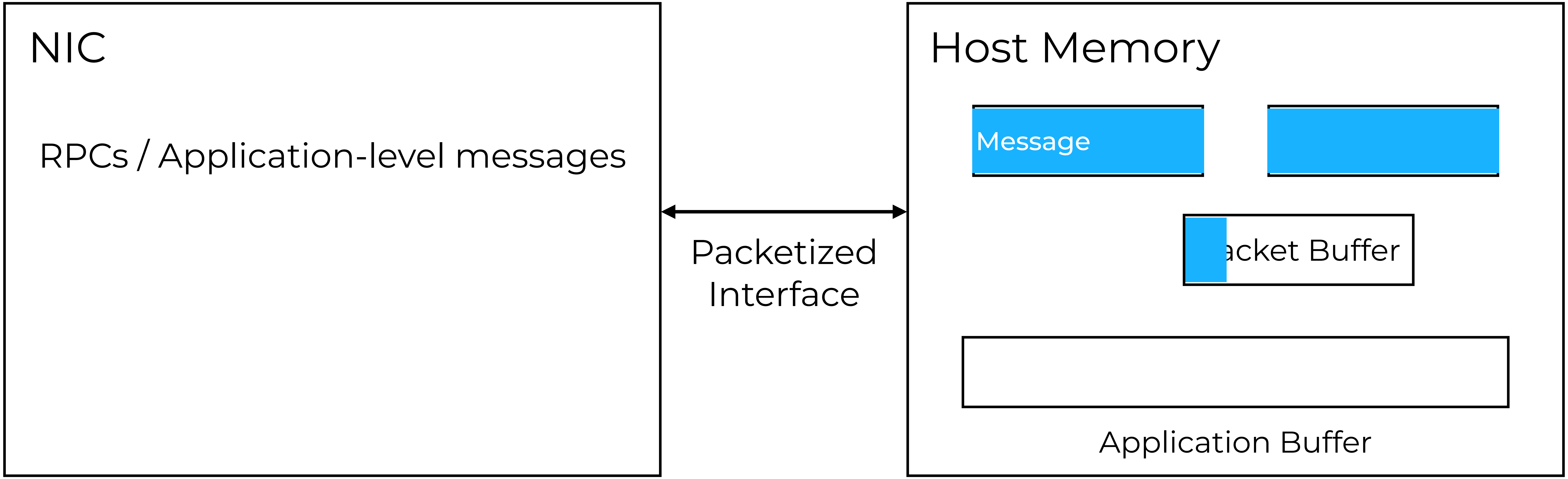
# Problem #1 Packetized Abstraction

Packetized abstraction is unsuitable for higher-level offloads



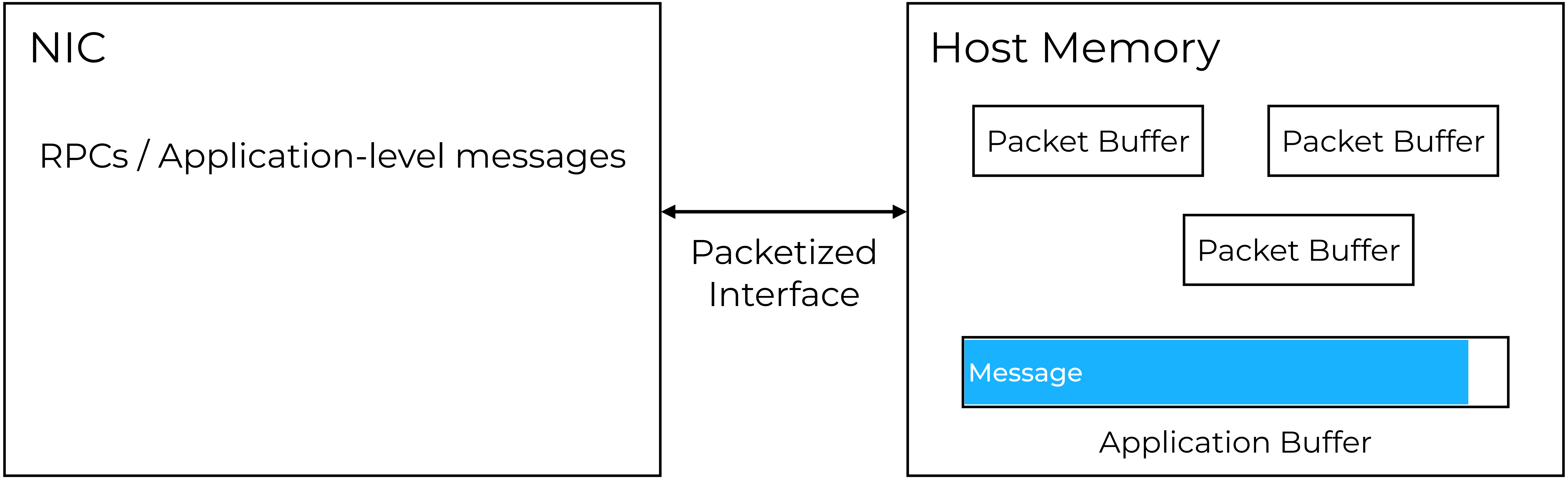
# Problem #1 Packetized Abstraction

Packetized abstraction is unsuitable for higher-level offloads



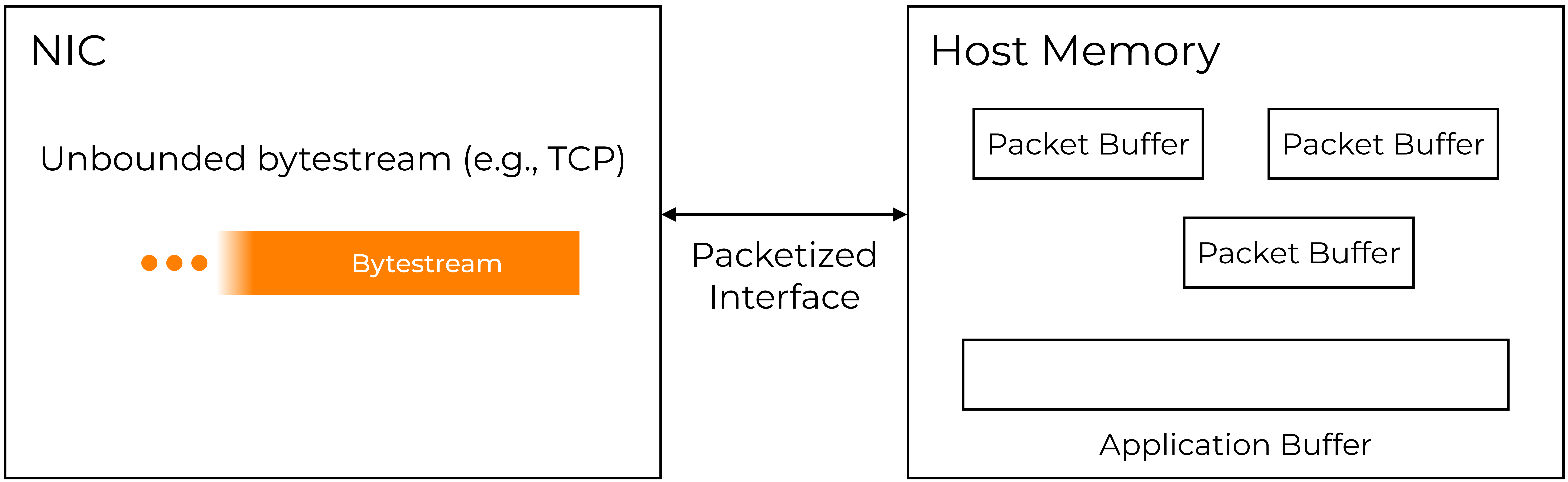
# Problem #1 Packetized Abstraction

Packetized abstraction is unsuitable for higher-level offloads



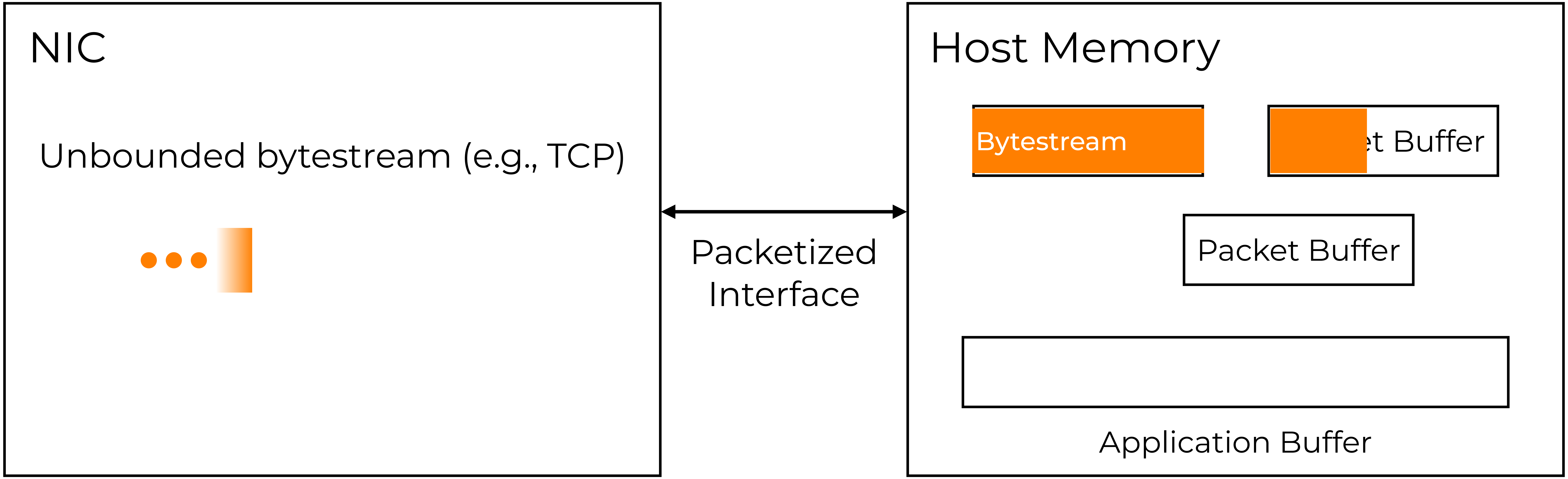
# Problem #1 Packetized Abstraction

Packetized abstraction is unsuitable for higher-level offloads



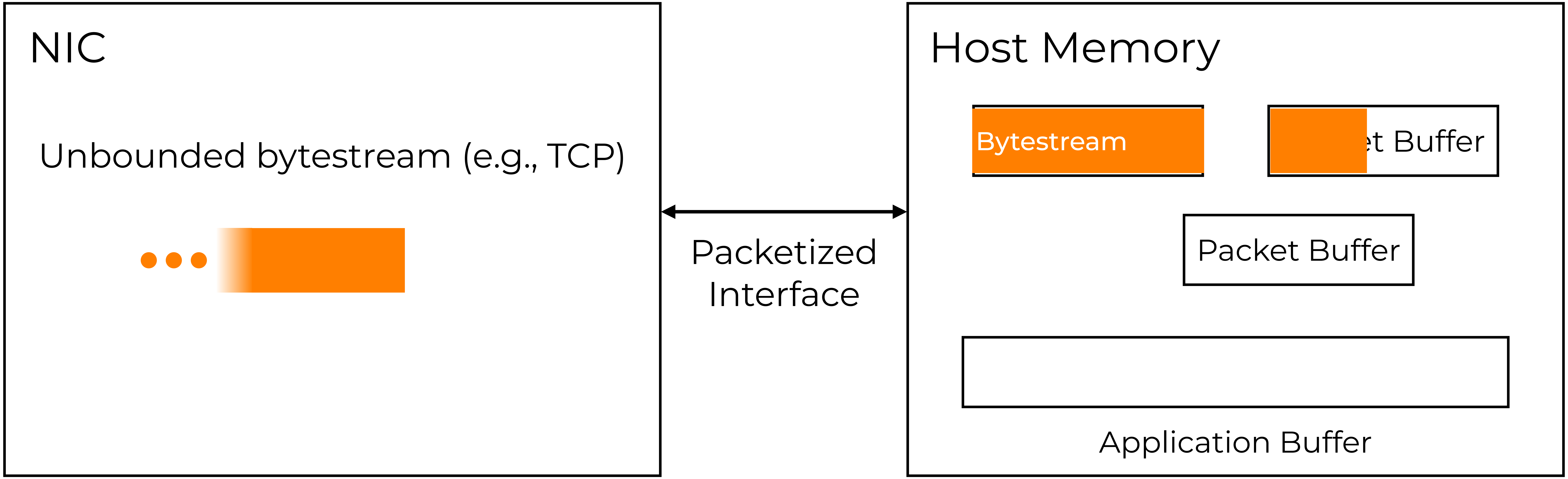
# Problem #1 Packetized Abstraction

Packetized abstraction is unsuitable for higher-level offloads



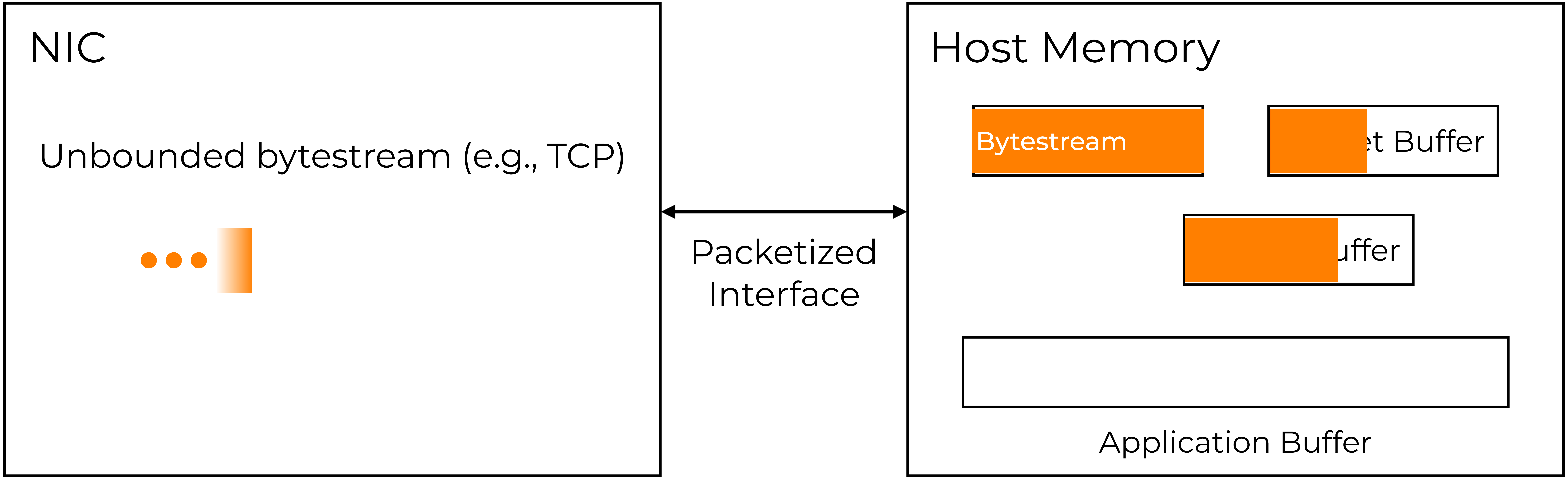
# Problem #1 Packetized Abstraction

Packetized abstraction is unsuitable for higher-level offloads



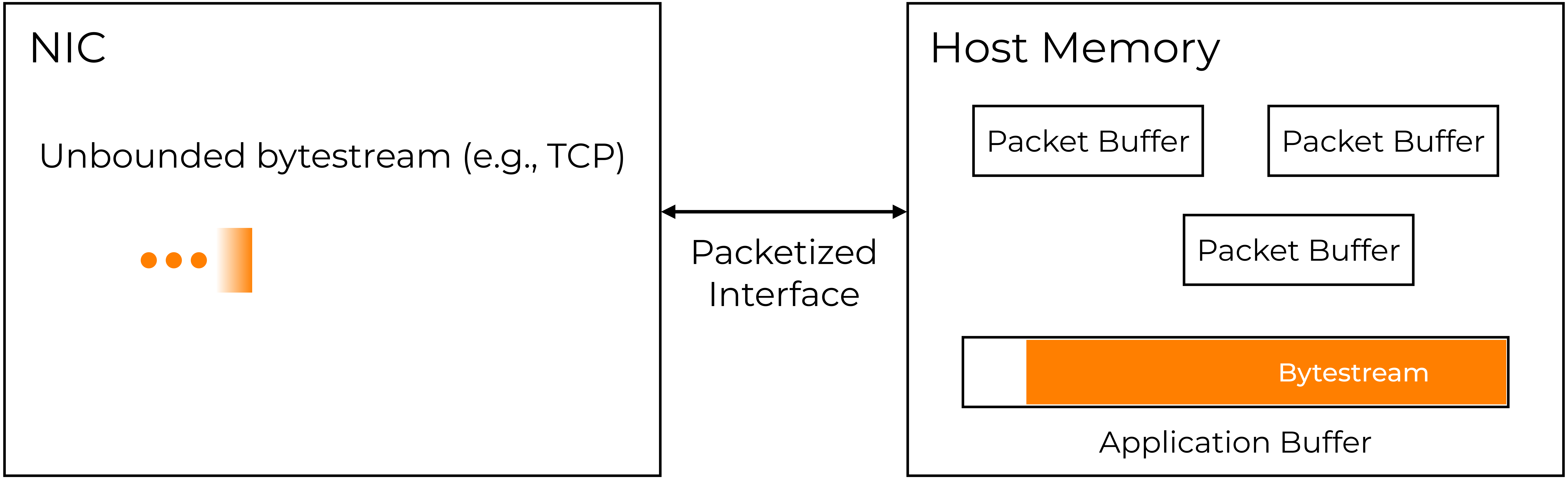
# Problem #1 Packetized Abstraction

Packetized abstraction is unsuitable for higher-level offloads



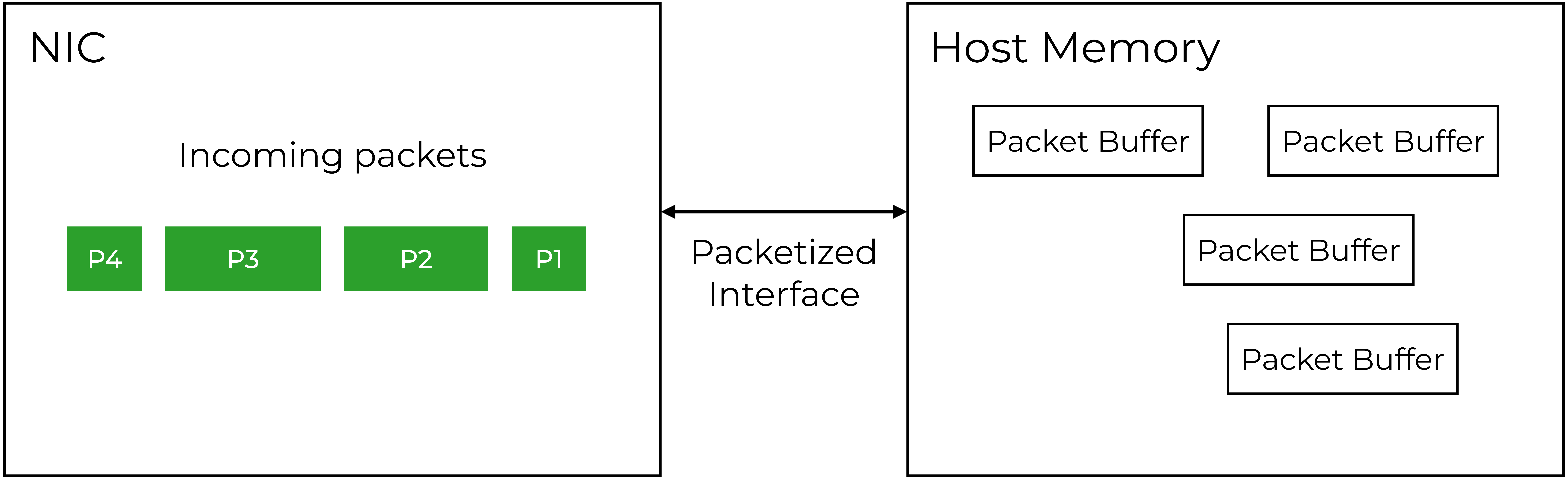
# Problem #1 Packetized Abstraction

Packetized abstraction is unsuitable for higher-level offloads



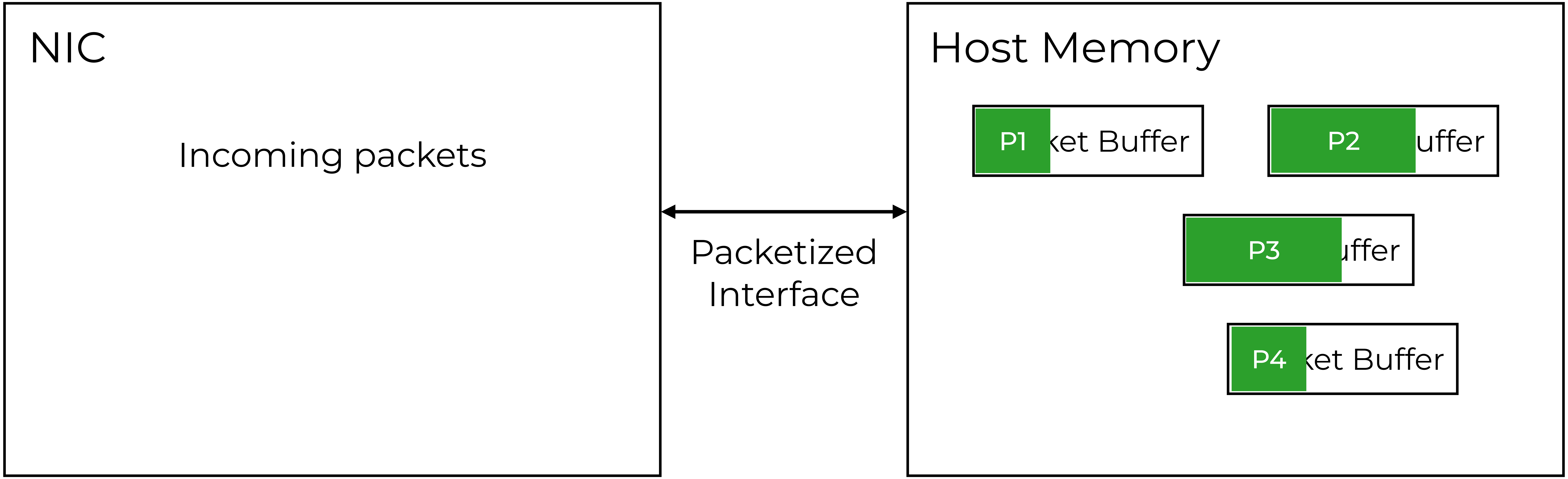
# Problem #2 Poor Cache Interaction

Poor cache interaction due to chaotic memory access



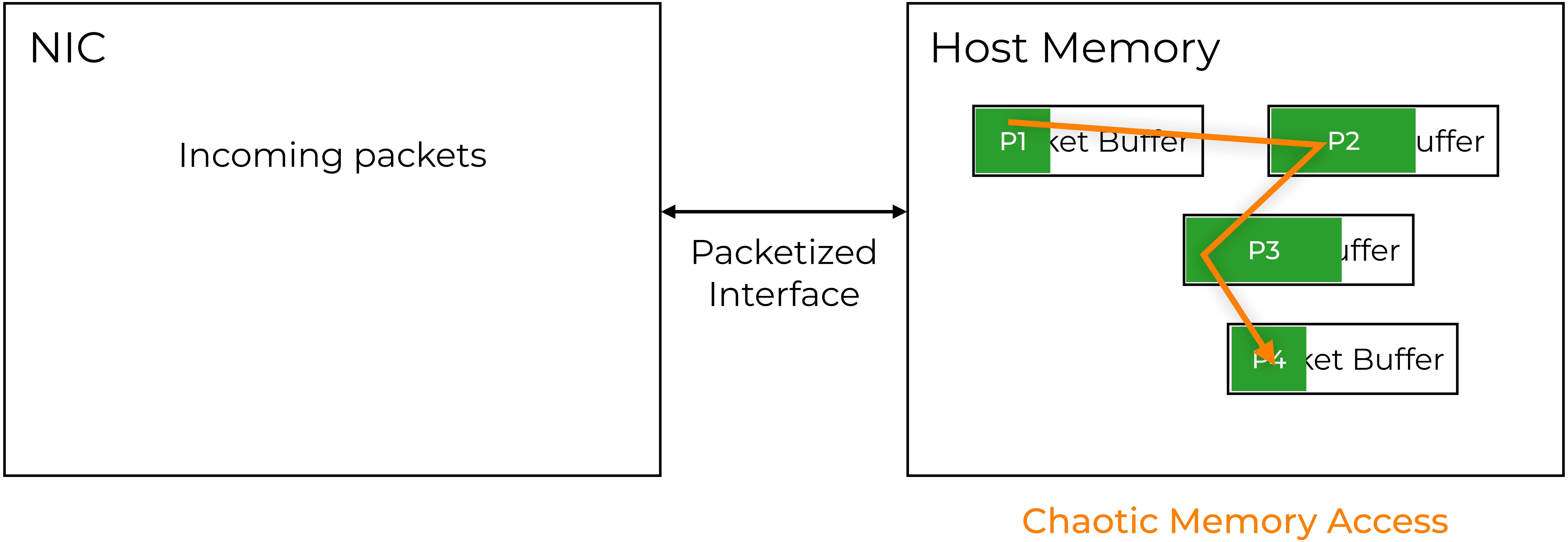
# Problem #2 Poor Cache Interaction

Poor cache interaction due to chaotic memory access



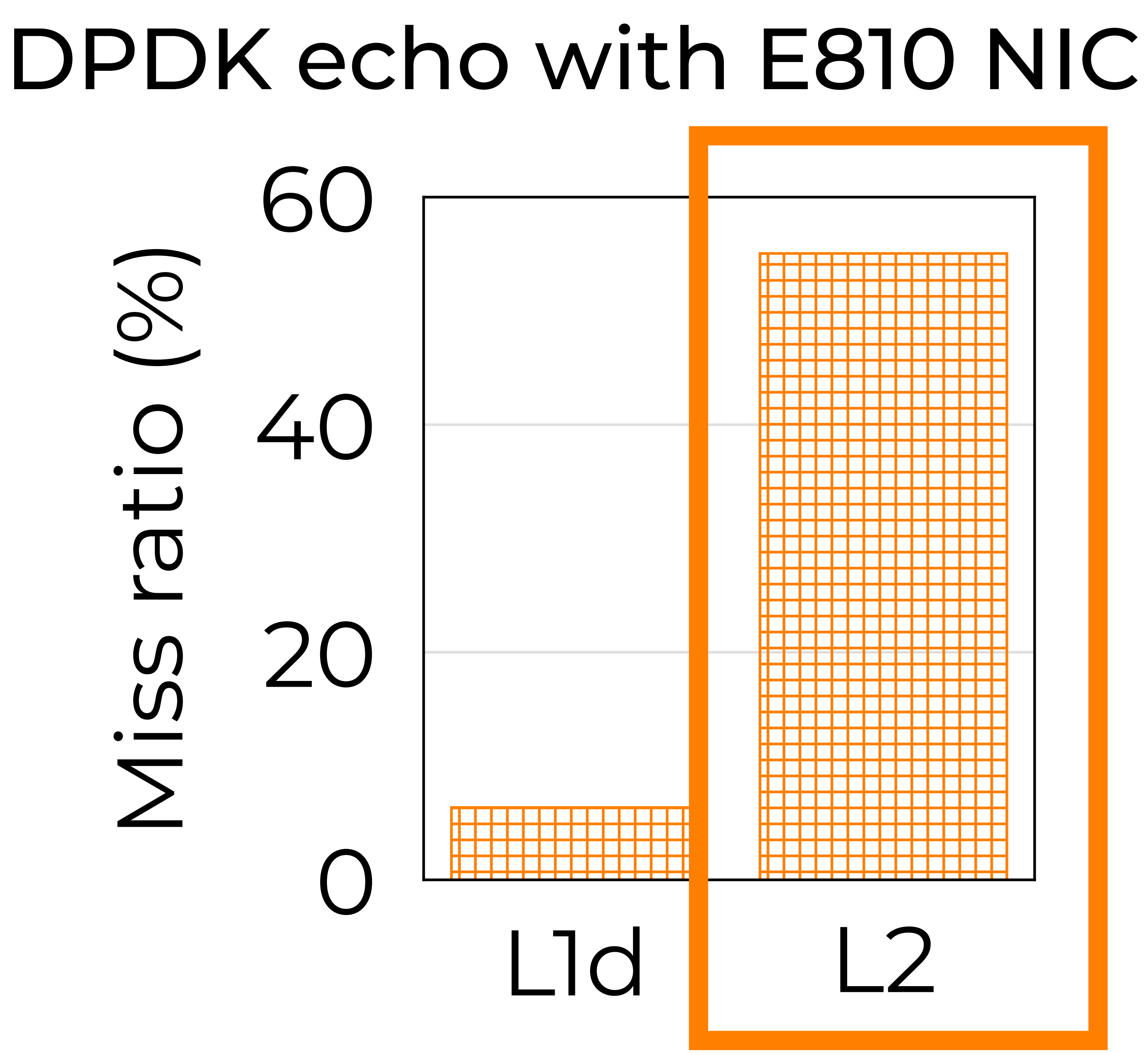
# Problem #2 Poor Cache Interaction

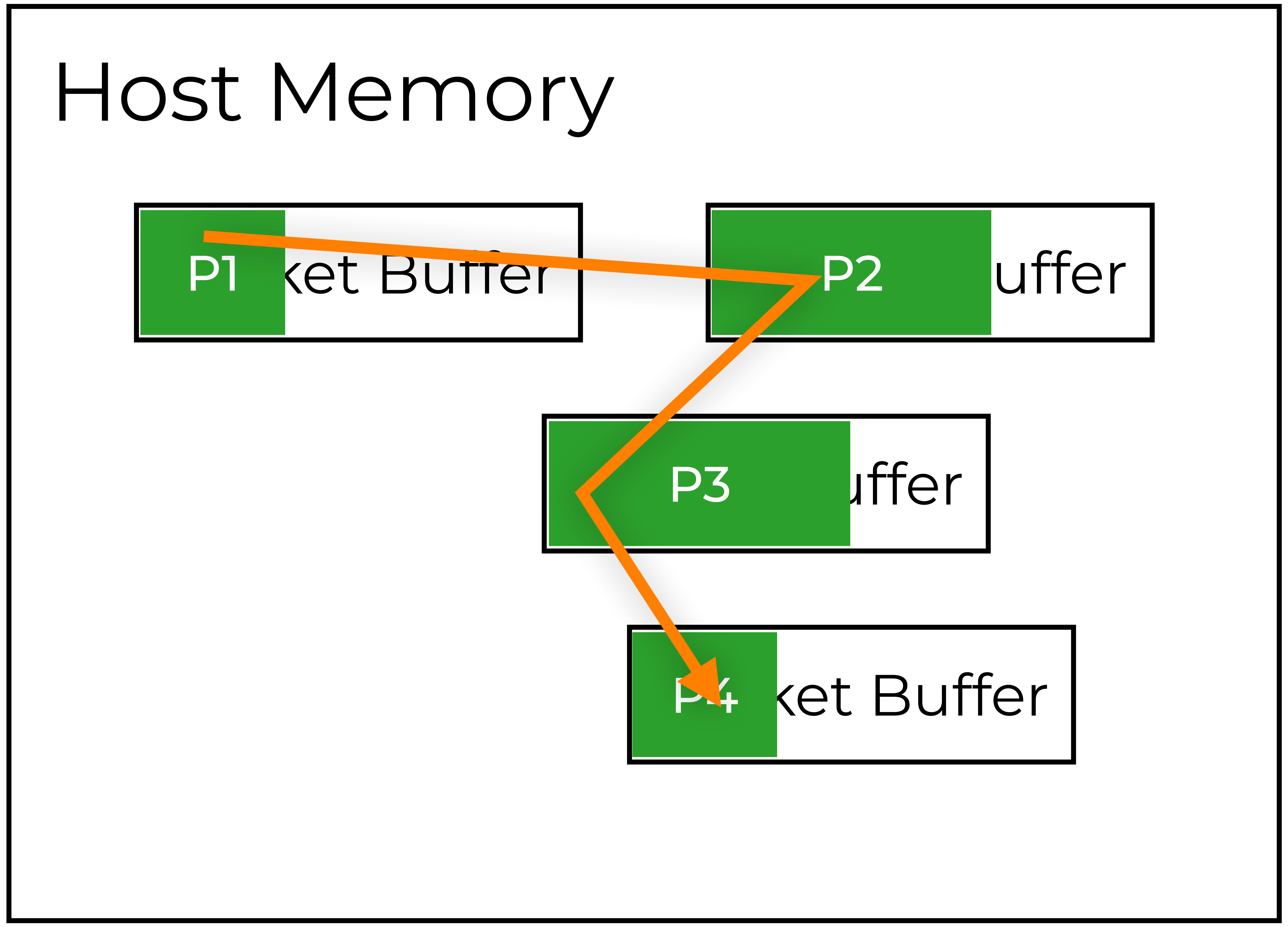
Poor cache interaction due to chaotic memory access



# Problem #2 Poor Cache Interaction

Poor cache interaction due to chaotic memory access



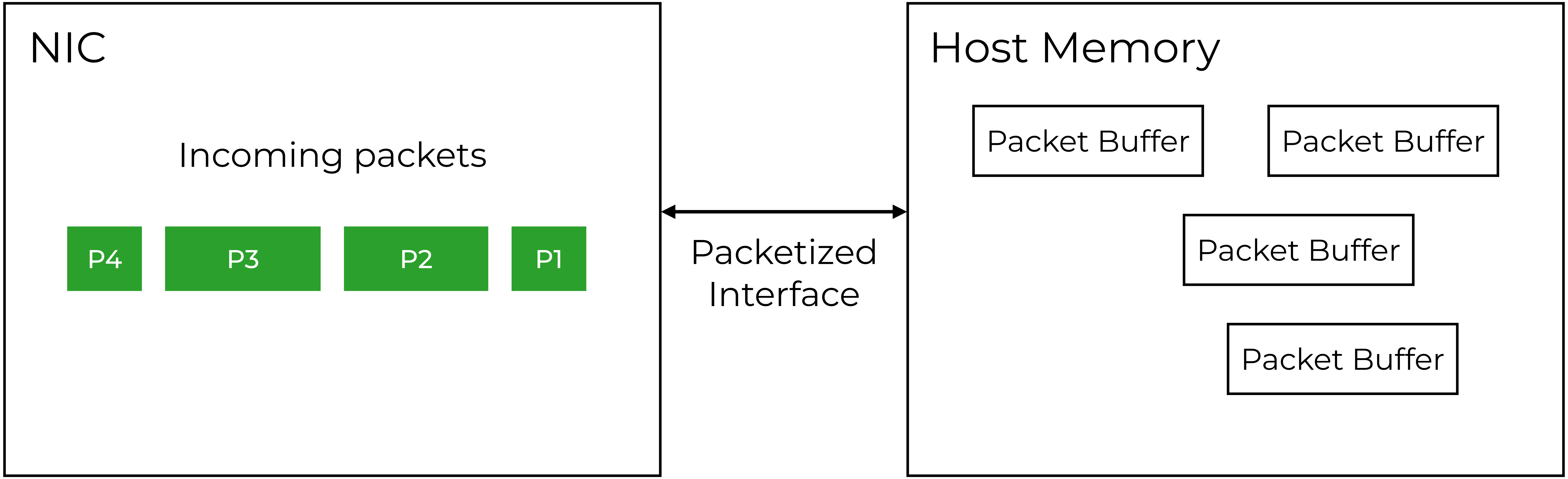




55% Miss Ratio for the L2 Cache

Chaotic Memory Access

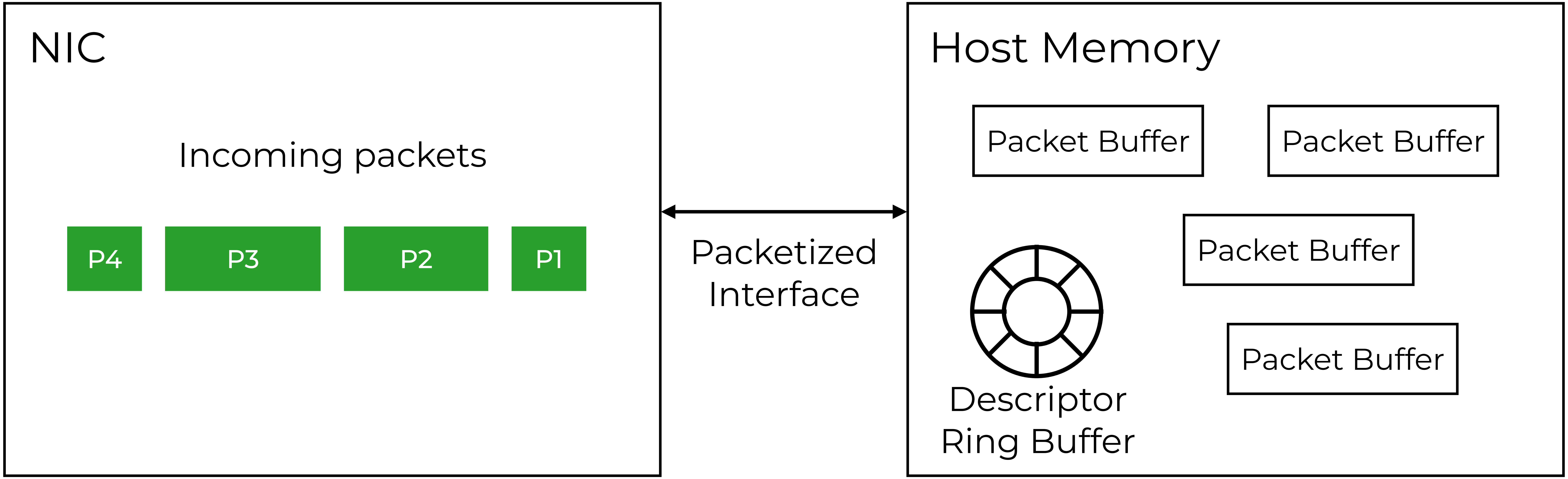
# Problem #3 Metadata Overhead

Overhead (PCIe bandwidth and CPU cycles) due to per-packet metadata



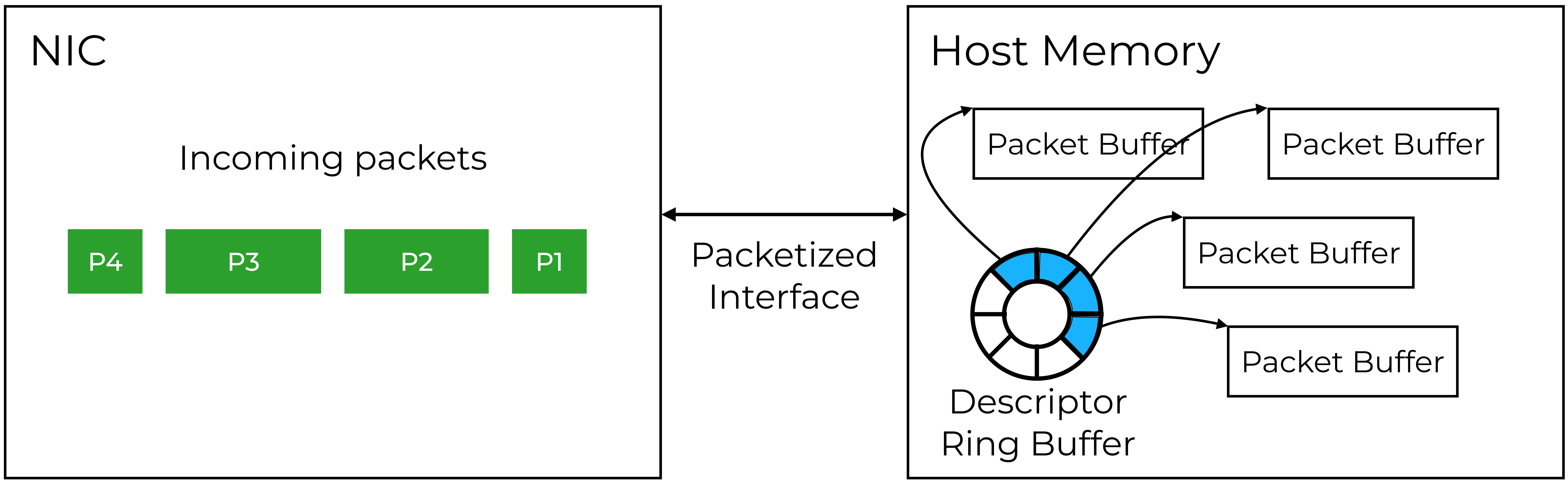
# Problem #3 Metadata Overhead

Overhead (PCIe bandwidth and CPU cycles) due to per-packet metadata



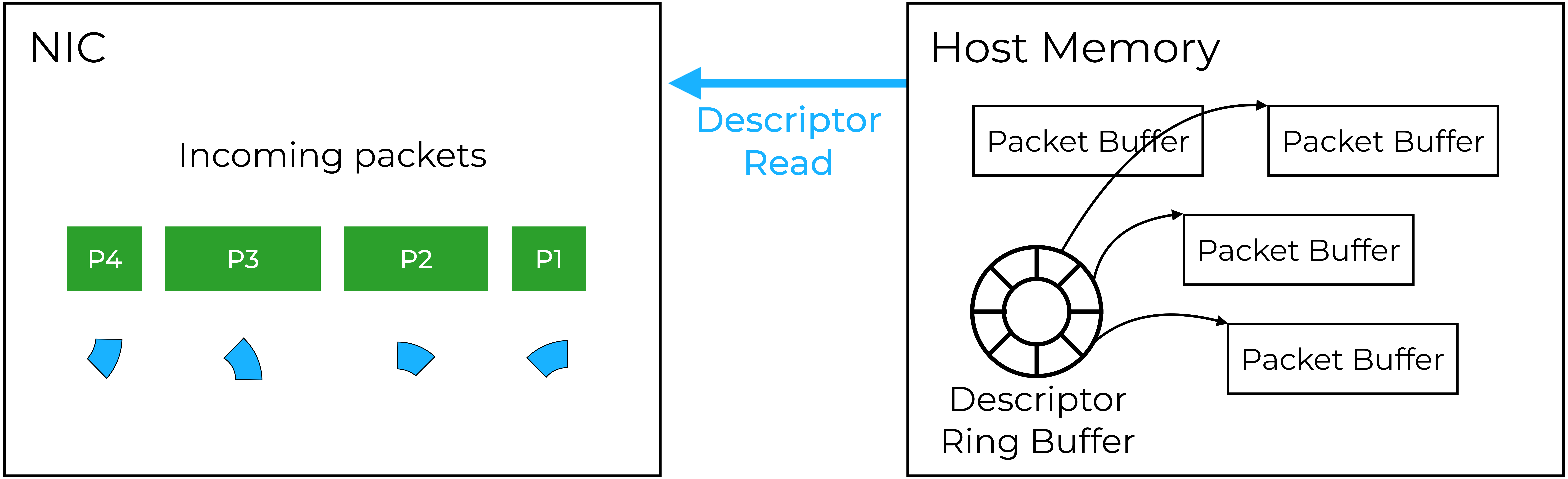
# Problem #3 Metadata Overhead

Overhead (PCIe bandwidth and CPU cycles) due to per-packet metadata



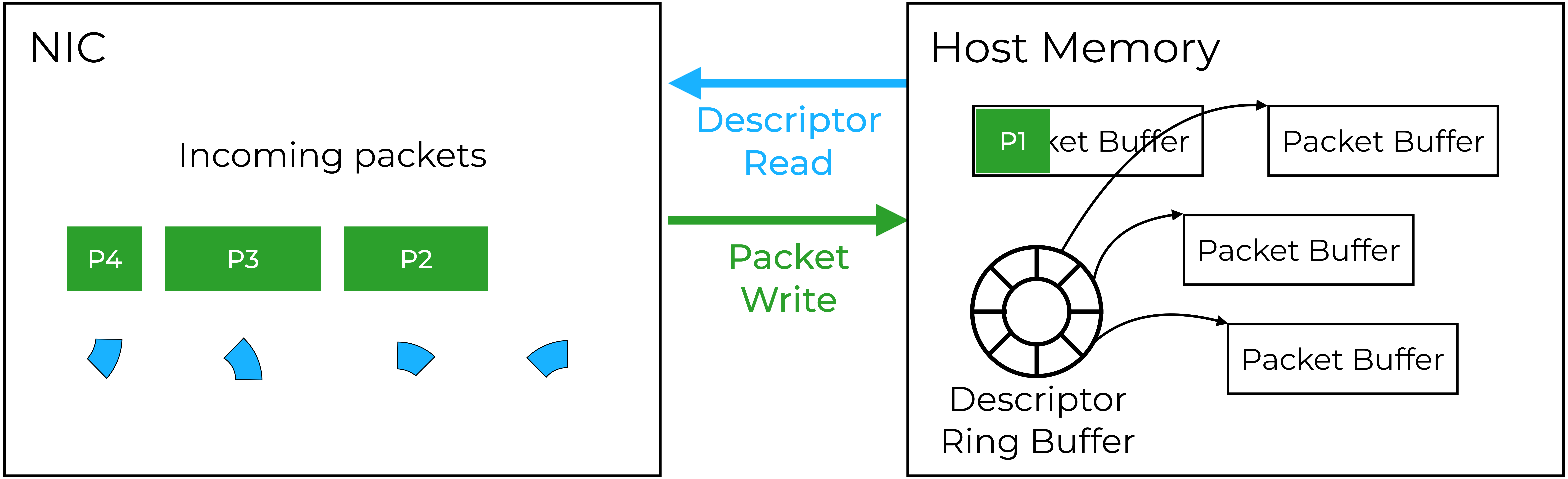
# Problem #3 Metadata Overhead

Overhead (PCIe bandwidth and CPU cycles) due to per-packet metadata



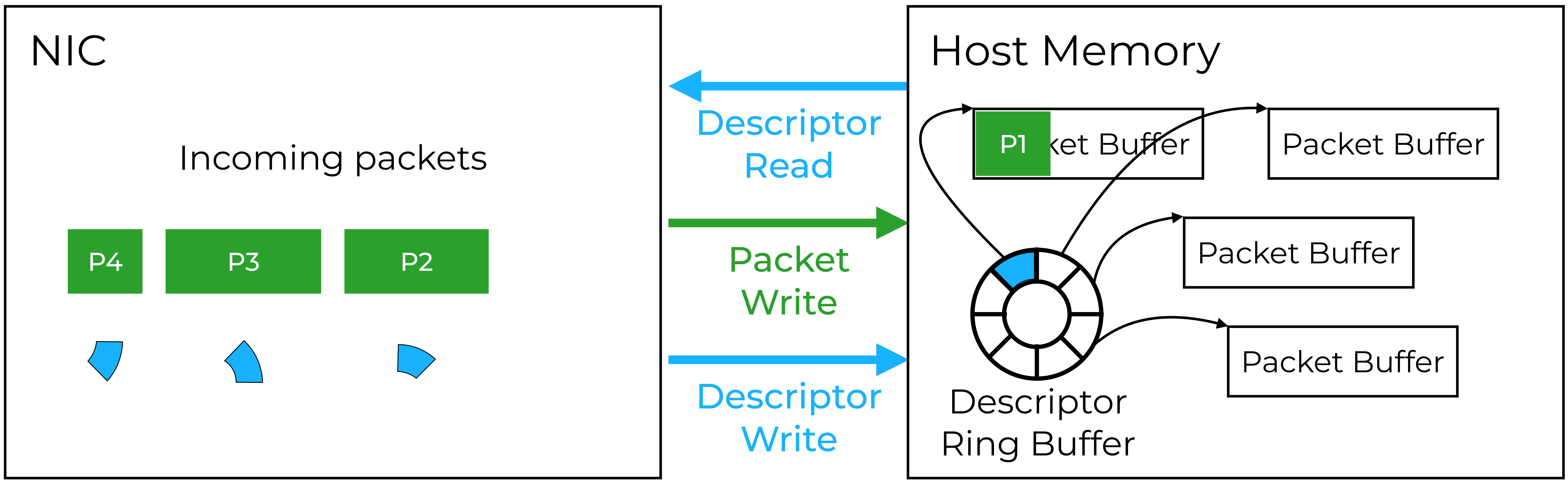
# Problem #3 Metadata Overhead

Overhead (PCIe bandwidth and CPU cycles) due to per-packet metadata



# Problem #3 Metadata Overhead

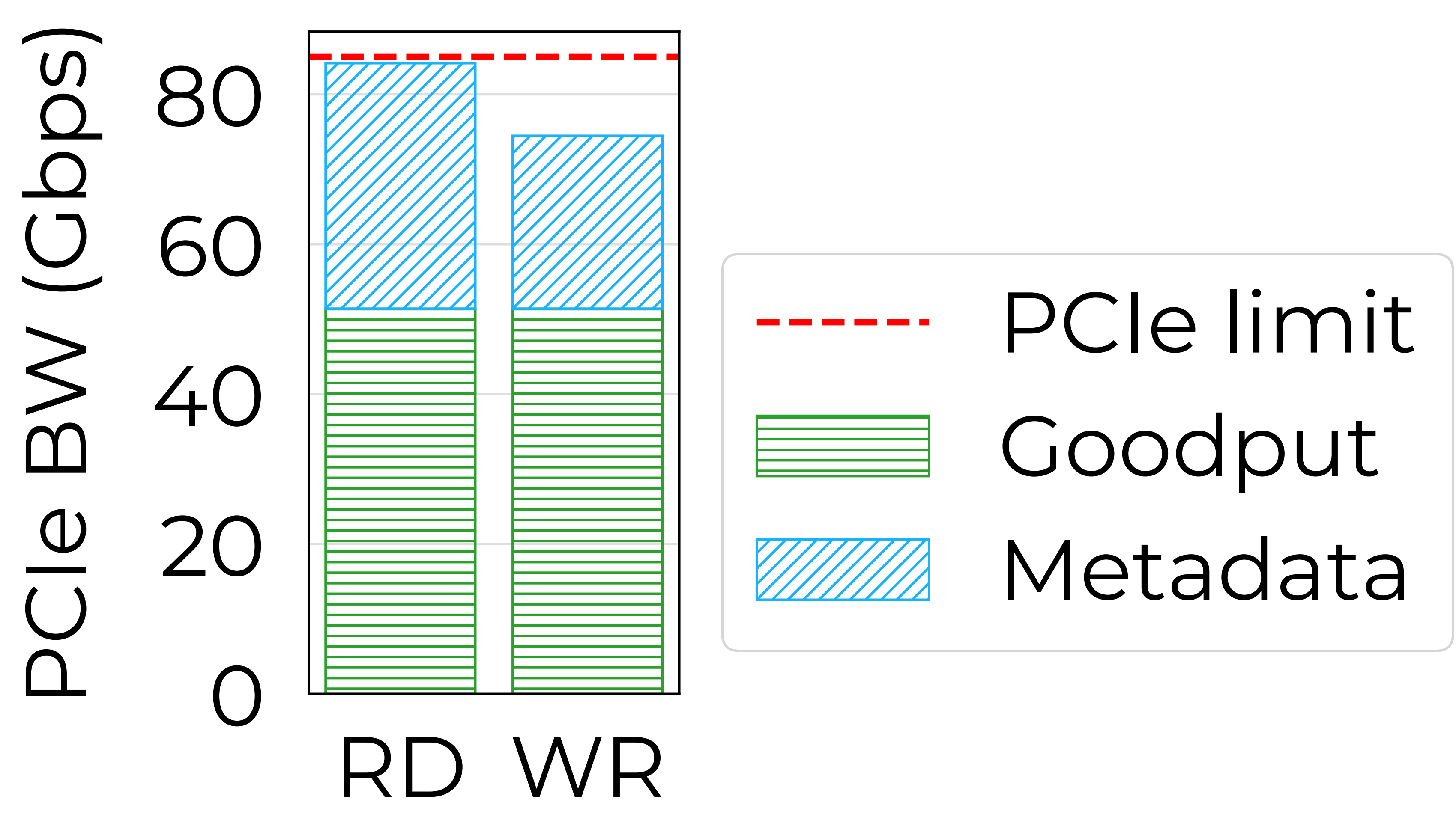
Overhead (PCIe bandwidth and CPU cycles) due to per-packet metadata



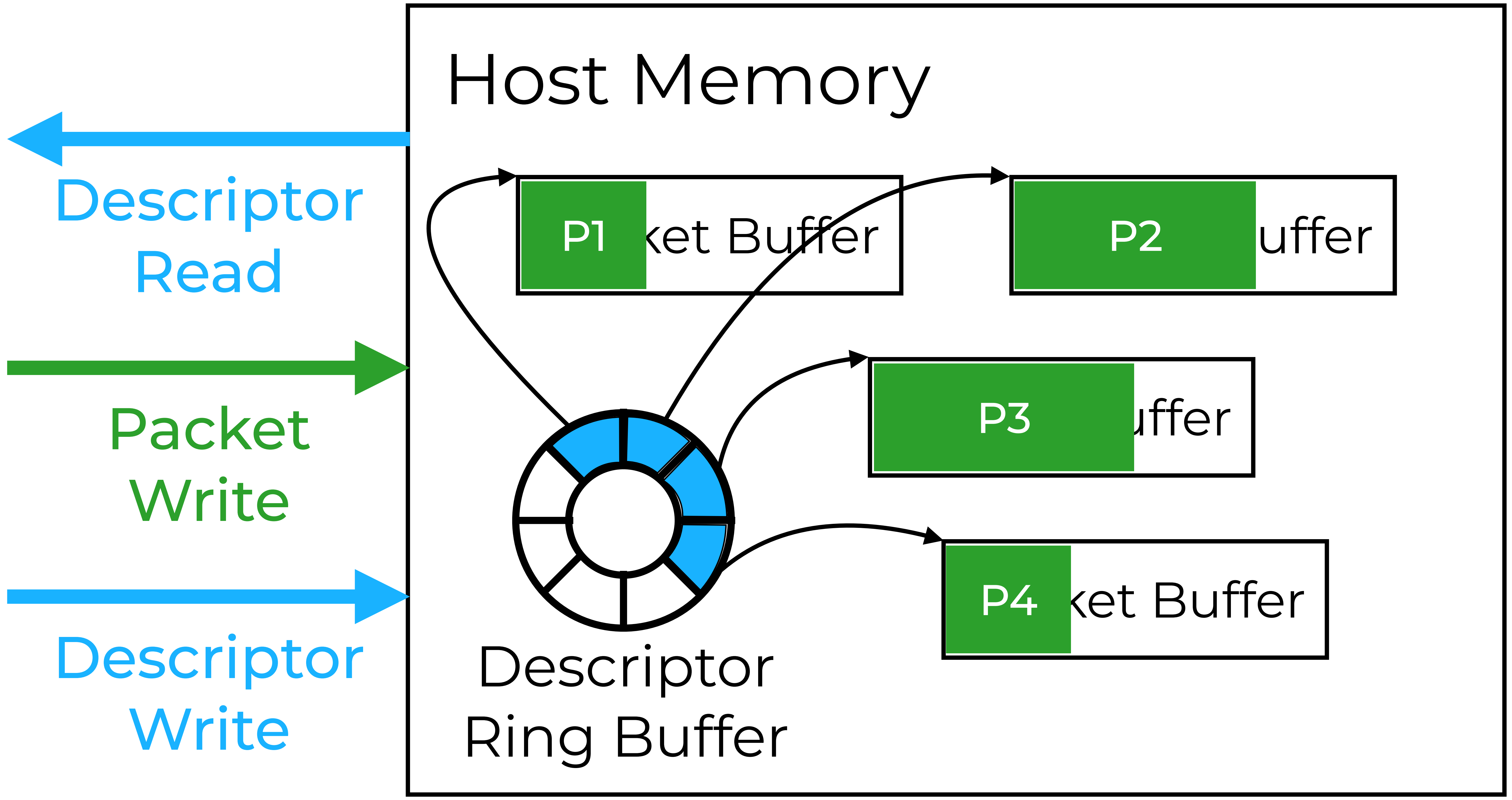
# Problem #3 Metadata Overhead

Overhead (PCIe bandwidth and CPU cycles) due to per-packet metadata

DPDK echo with E810 NIC



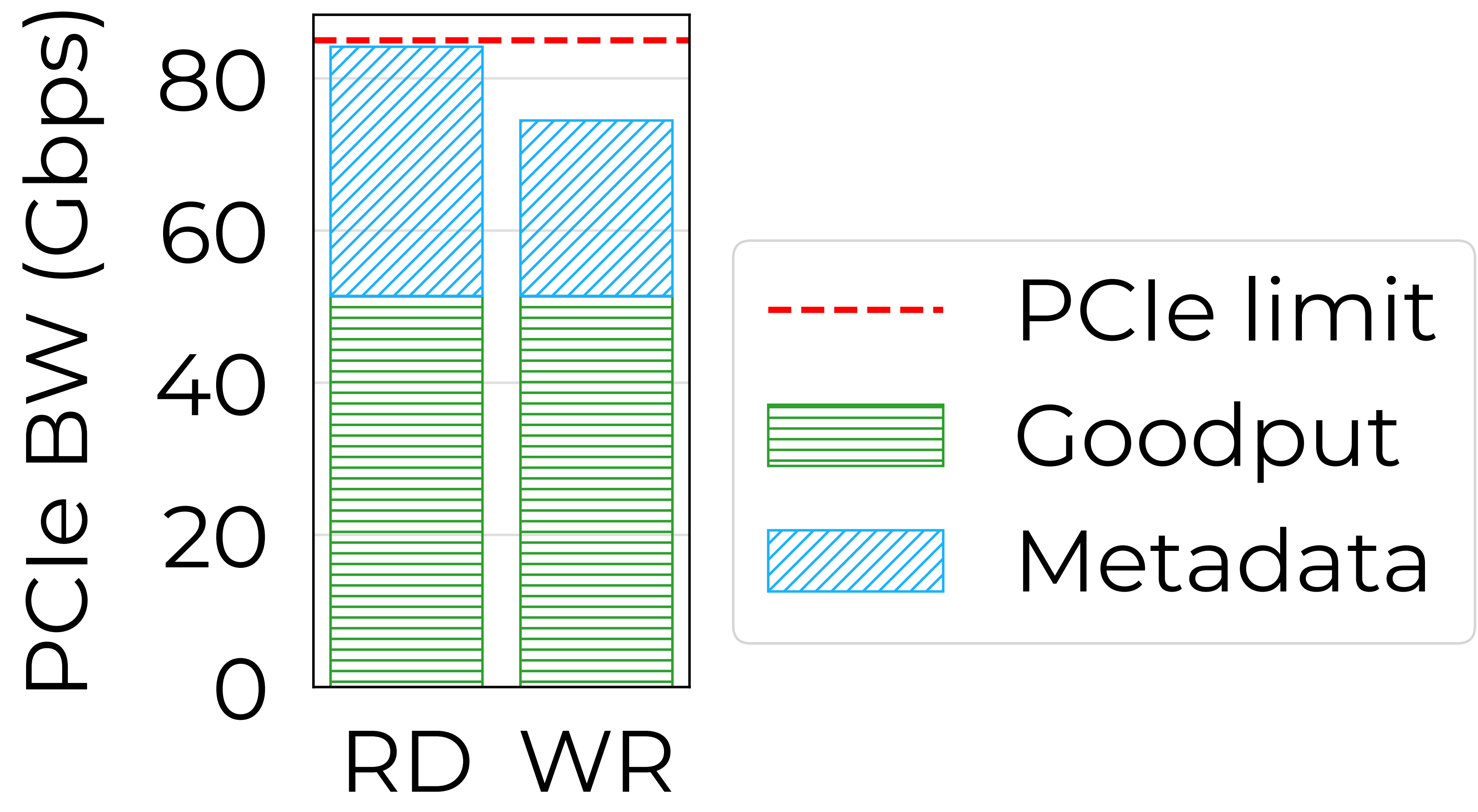

Up to 39% of PCIe bandwidth consumed with metadata



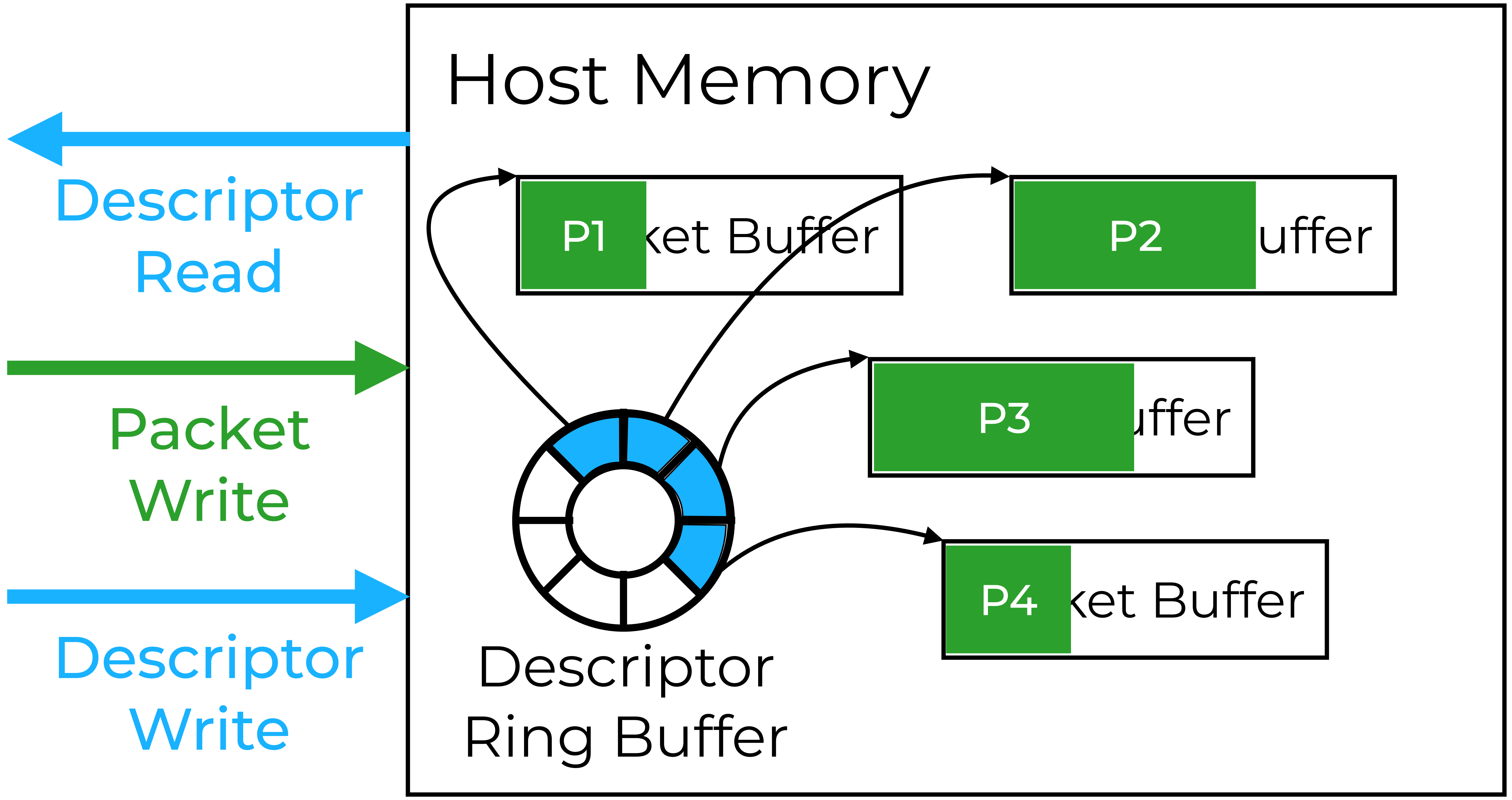
# Problem #3 Metadata Overhead

Overhead (PCIe bandwidth and CPU cycles) due to per-packet metadata

DPDK echo with E810 NIC

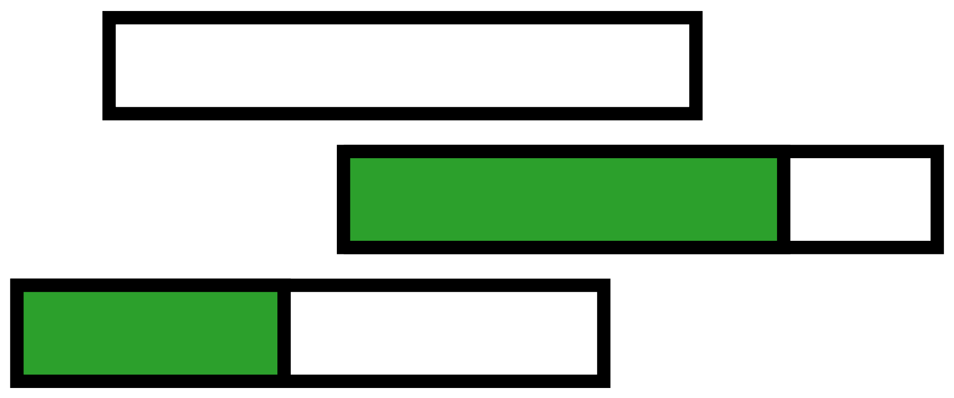



Up to 39% of PCIe bandwidth consumed with metadata

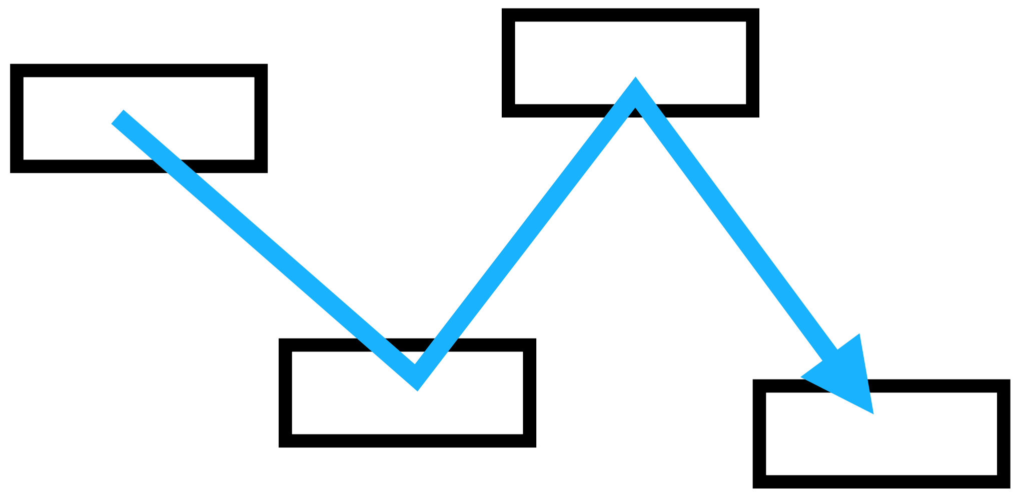



*Similar process to transmit packets*

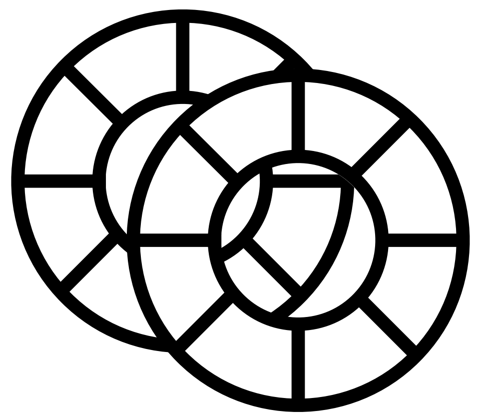
# Mismatch between how NICs are used and their interface



**#1** Packetized Abstraction



**#2** Poor Cache Interaction



**#3** Metadata Overhead

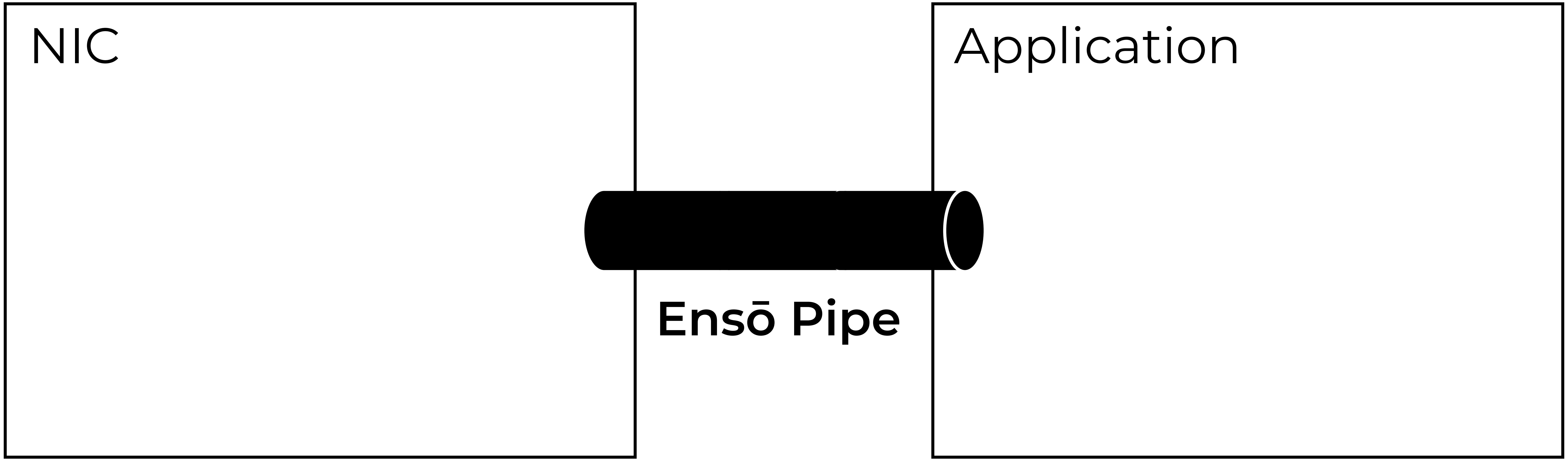
# Ensō

## New interface for NIC-Application Communication

# Ensō

## New interface for NIC-Application Communication

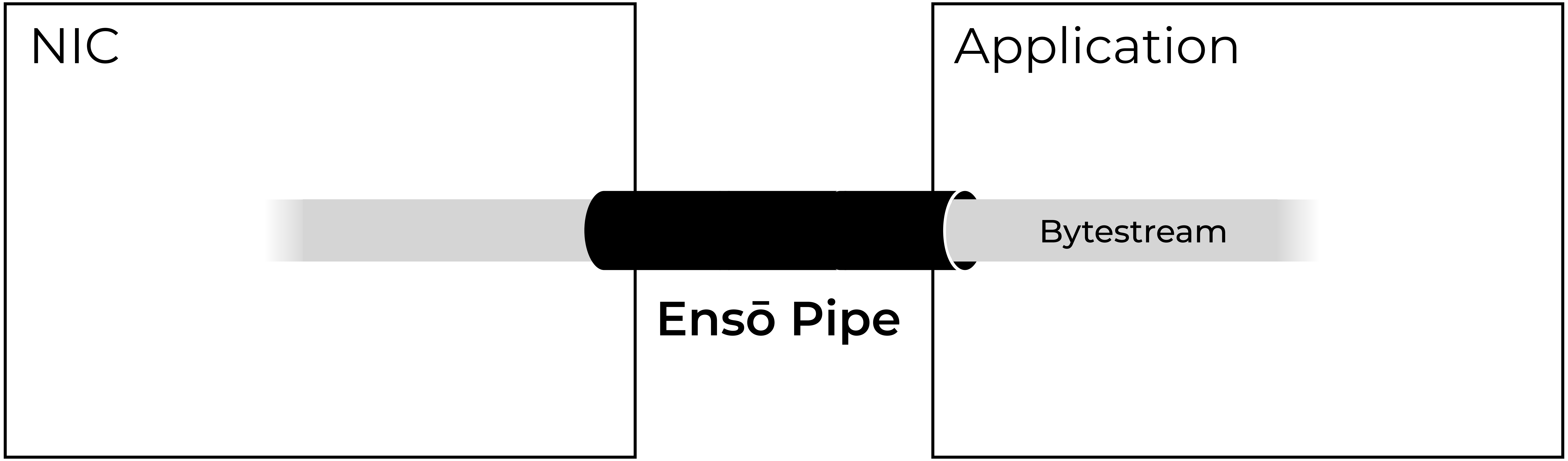
**Key Idea:** Streaming abstraction



# Ensō

## New interface for NIC-Application Communication

**Key Idea:** Streaming abstraction



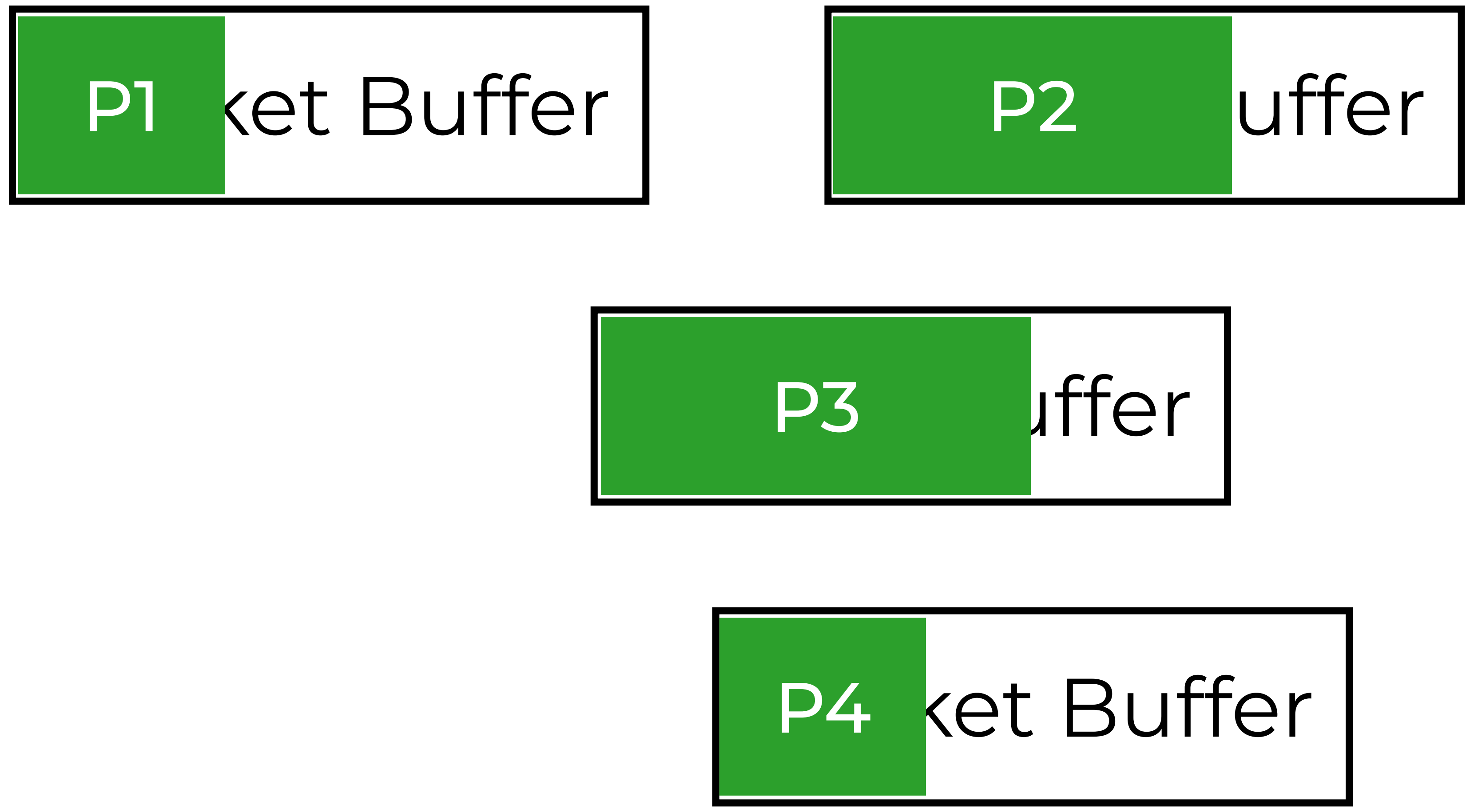
# What is a Streaming Abstraction?

# What is a Streaming Abstraction?

Provide the illusion of an unbounded buffer

# What is a Streaming Abstraction?

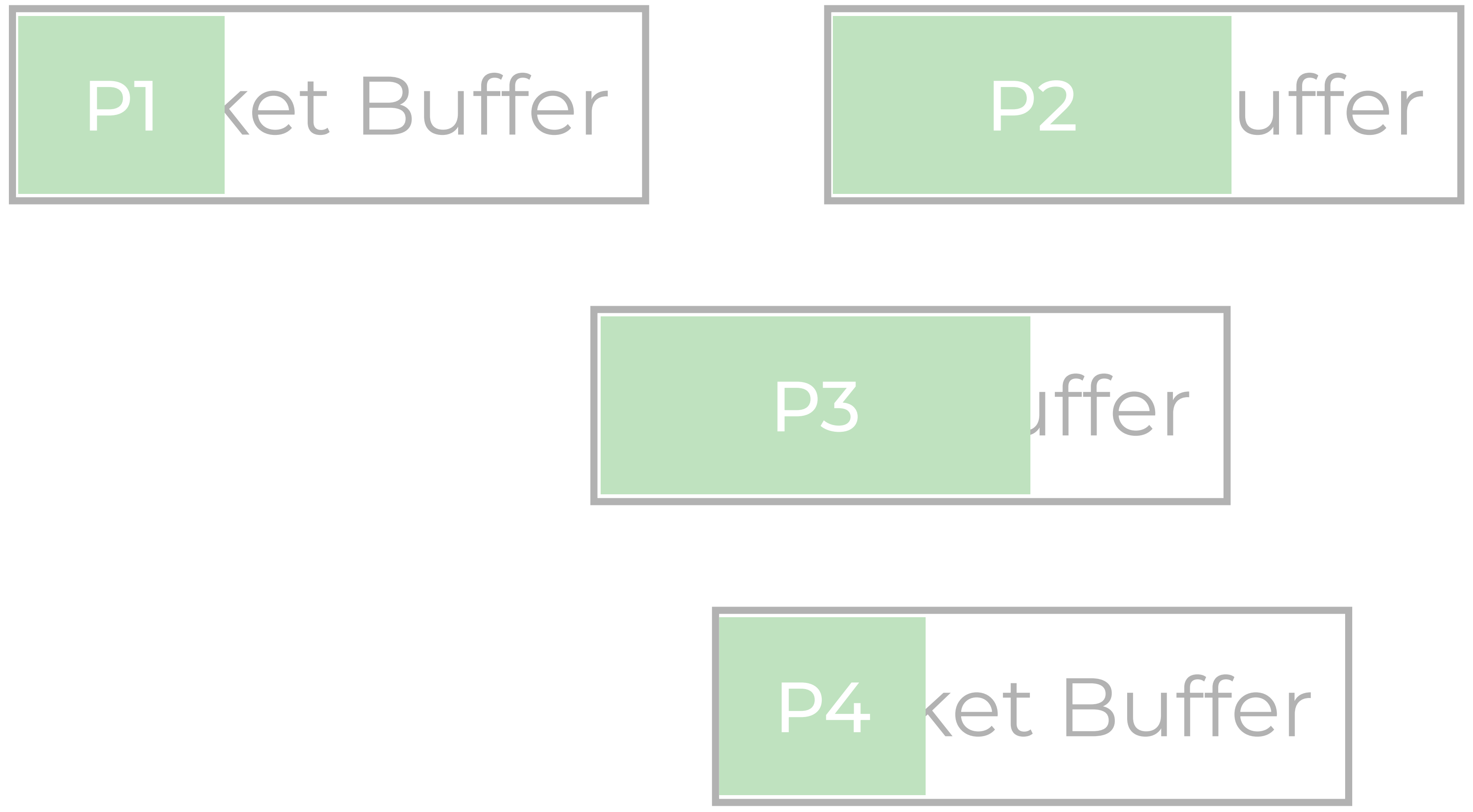
## Provide the illusion of an unbounded buffer




Packetized Abstraction

# What is a Streaming Abstraction?

## Provide the illusion of an unbounded buffer



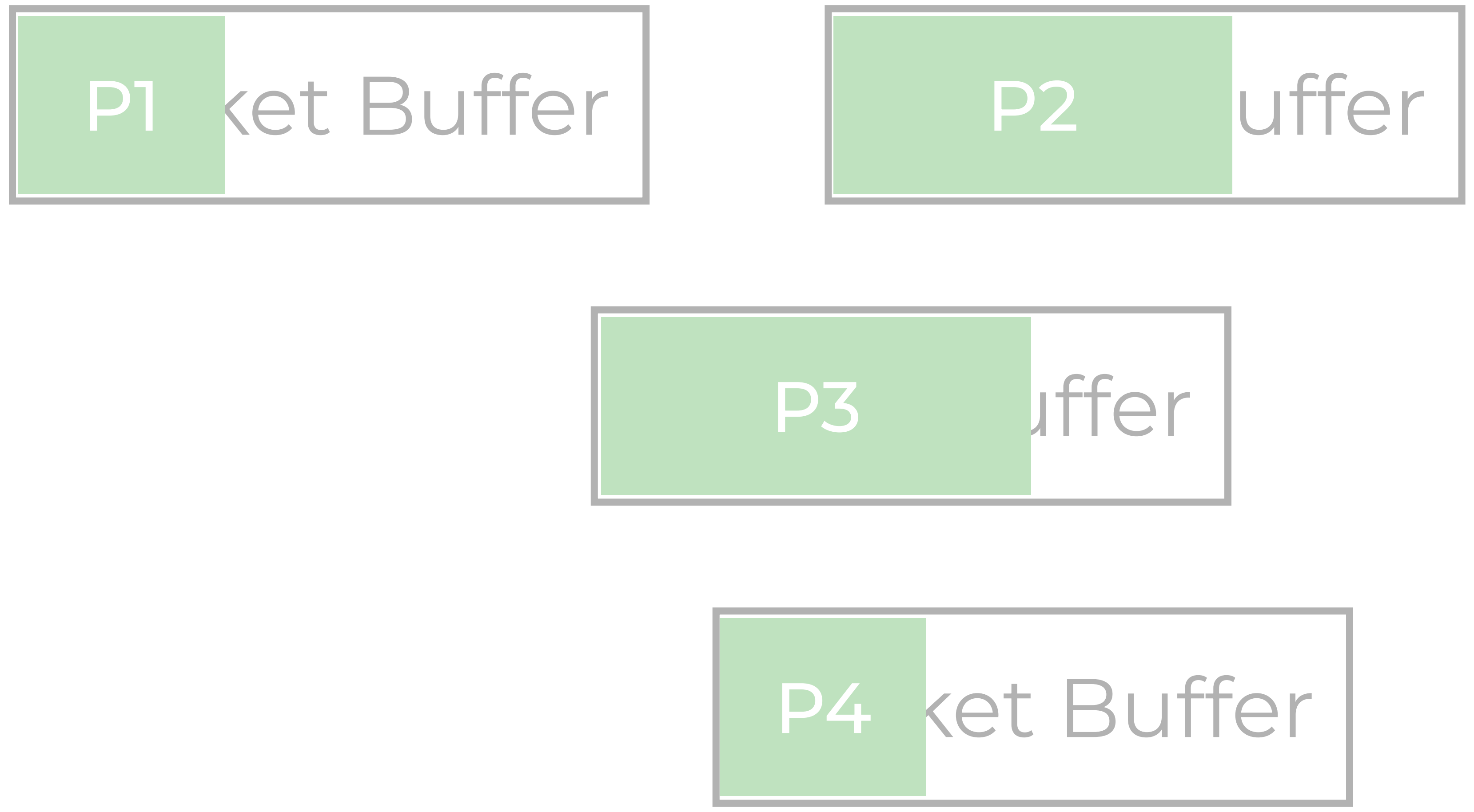

Unbounded Buffer

Packetized Abstraction

Streaming Abstraction

# What is a Streaming Abstraction?

## Provide the illusion of an unbounded buffer



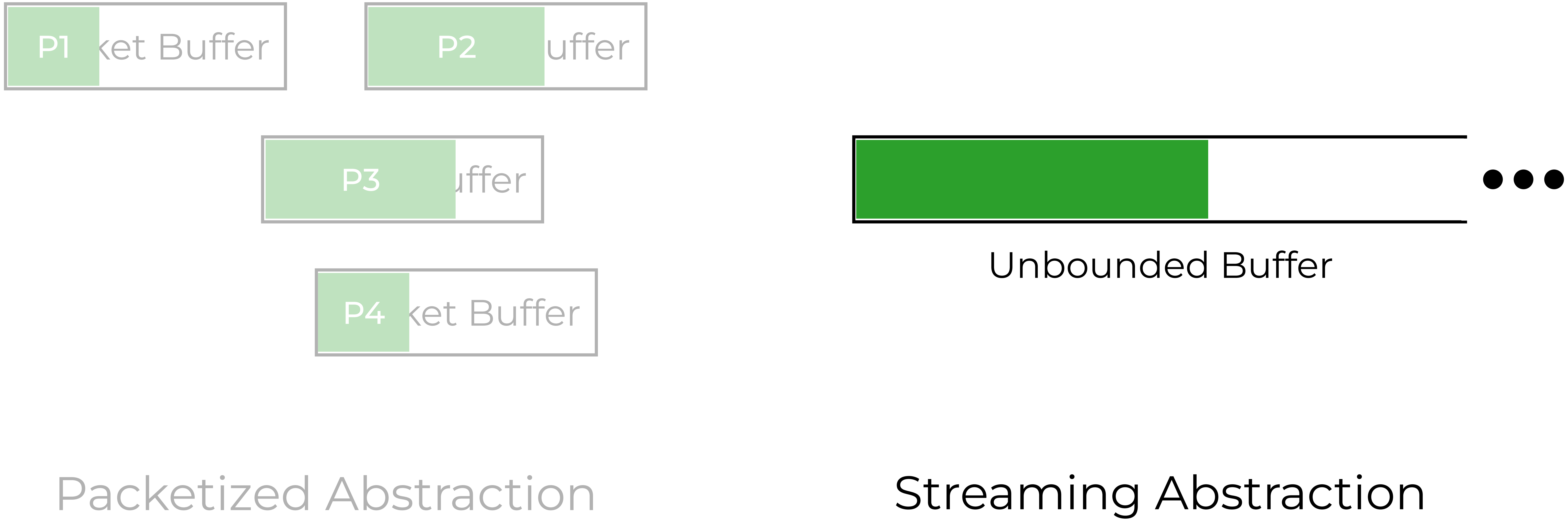


Packetized Abstraction

Streaming Abstraction

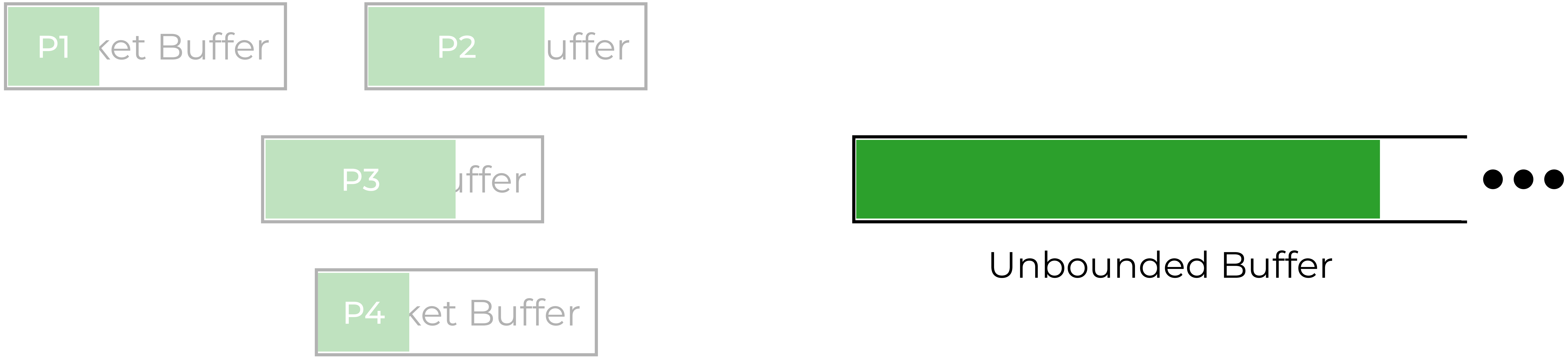
# What is a Streaming Abstraction?

Provide the illusion of an unbounded buffer



# What is a Streaming Abstraction?

## Provide the illusion of an unbounded buffer

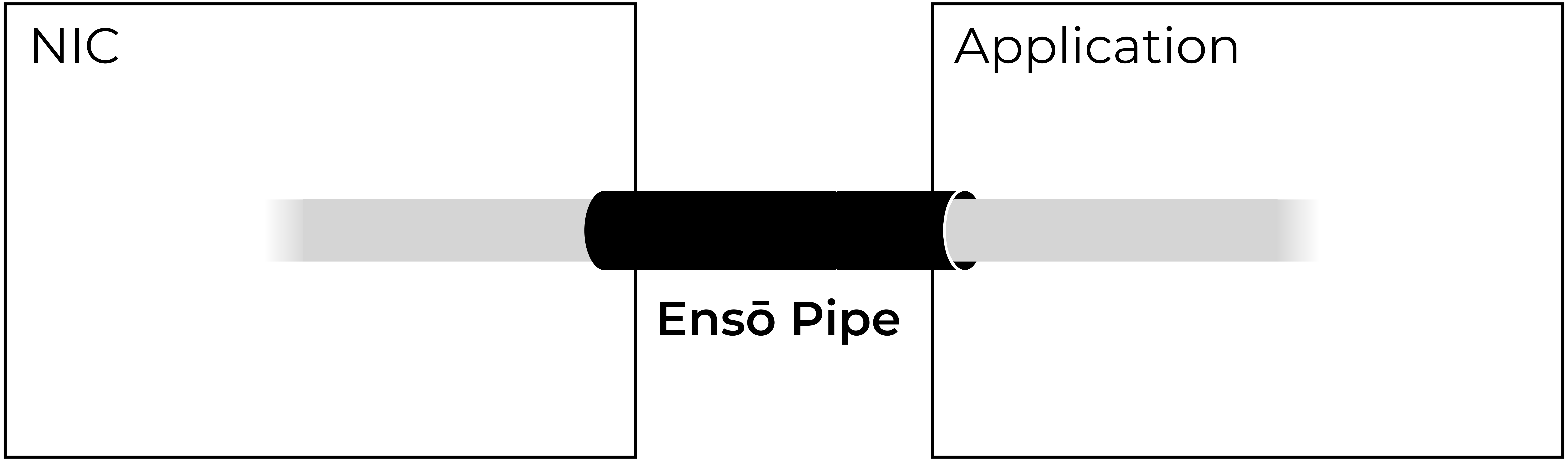



Packetized Abstraction

Streaming Abstraction

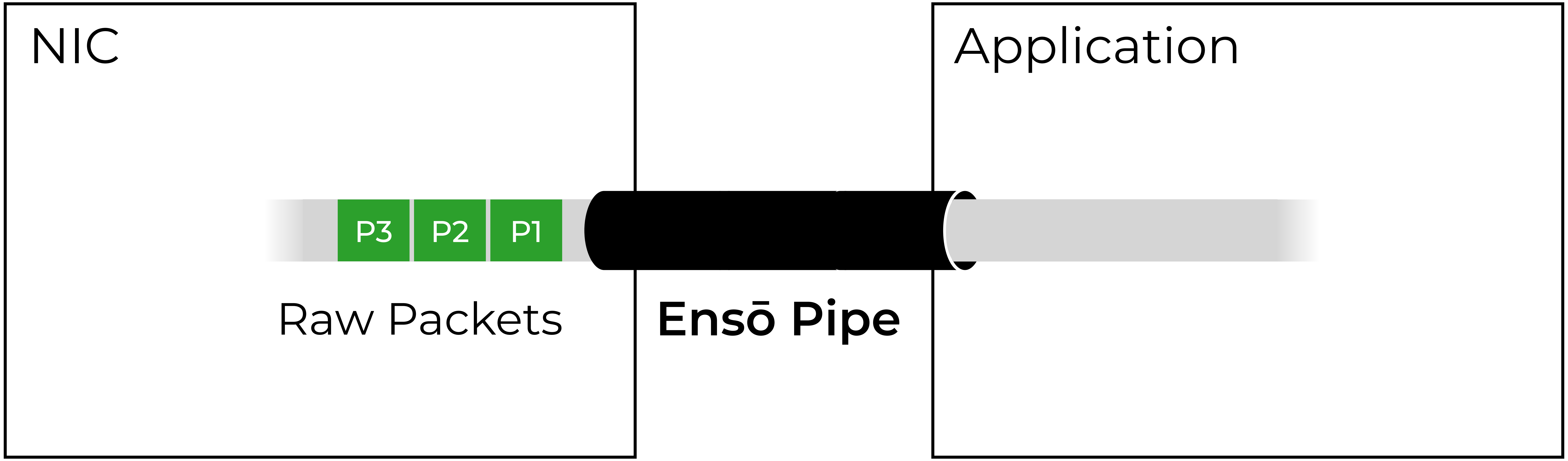
# Flexibility of a Streaming Abstraction

**Example 1:** NIC with no offloads



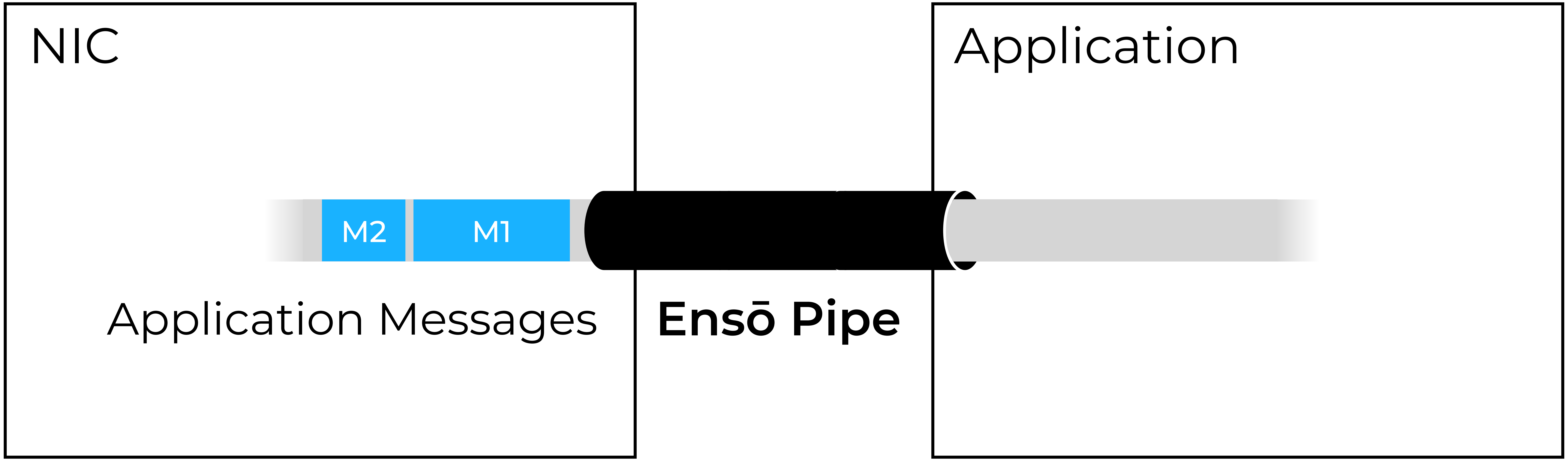
# Flexibility of a Streaming Abstraction

**Example 1:** NIC with no offloads



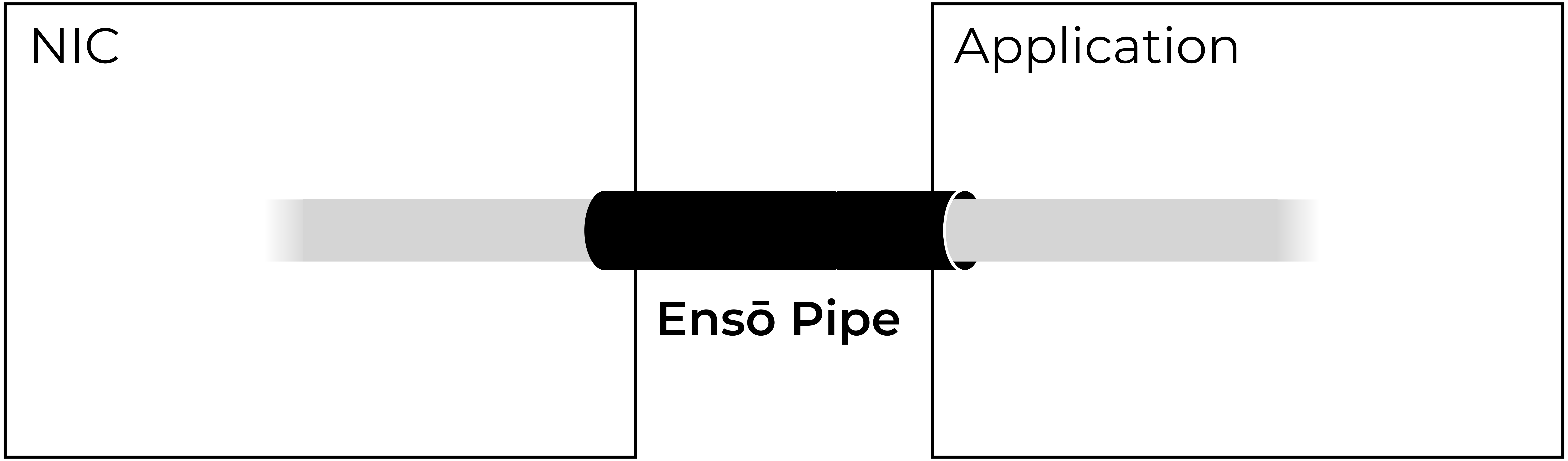
# Flexibility of a Streaming Abstraction

**Example 2:** NIC that is aware of application-level messages



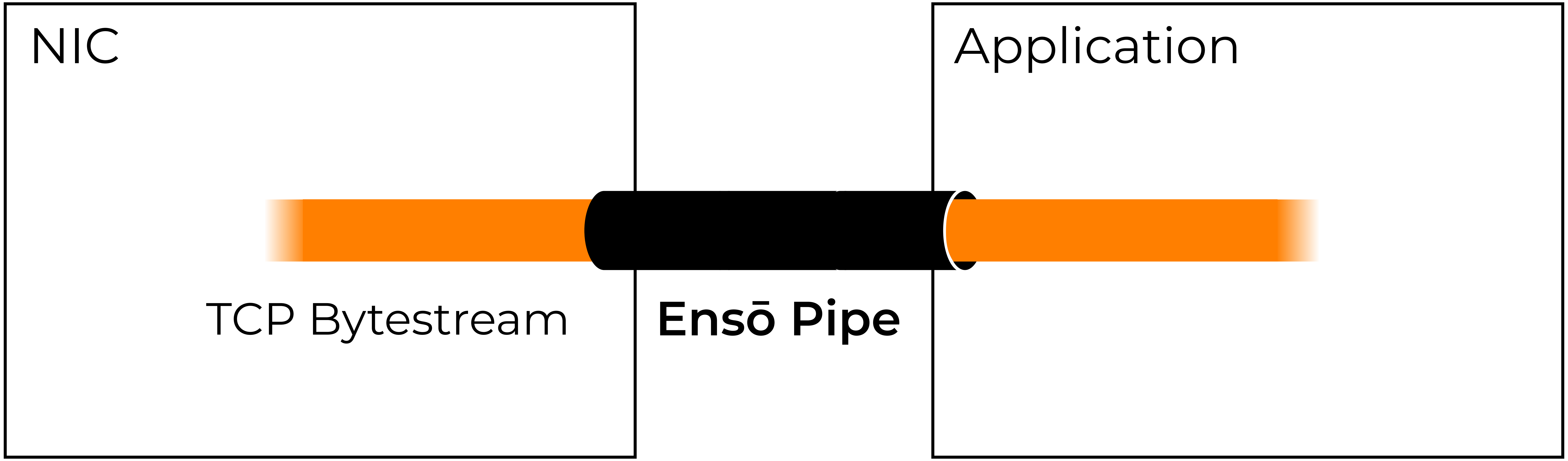
# Flexibility of a Streaming Abstraction

**Example 3:** NIC that implements a transport protocol



# Flexibility of a Streaming Abstraction

**Example 3:** NIC that implements a transport protocol



# ① How to implement a streaming abstraction?

① How to implement a streaming abstraction?

② How can a streaming abstraction improve performance?

# ① How to implement a streaming abstraction?

# ① How to implement a streaming abstraction?

Provide the illusion of an unbounded buffer

# ① How to implement a streaming abstraction?

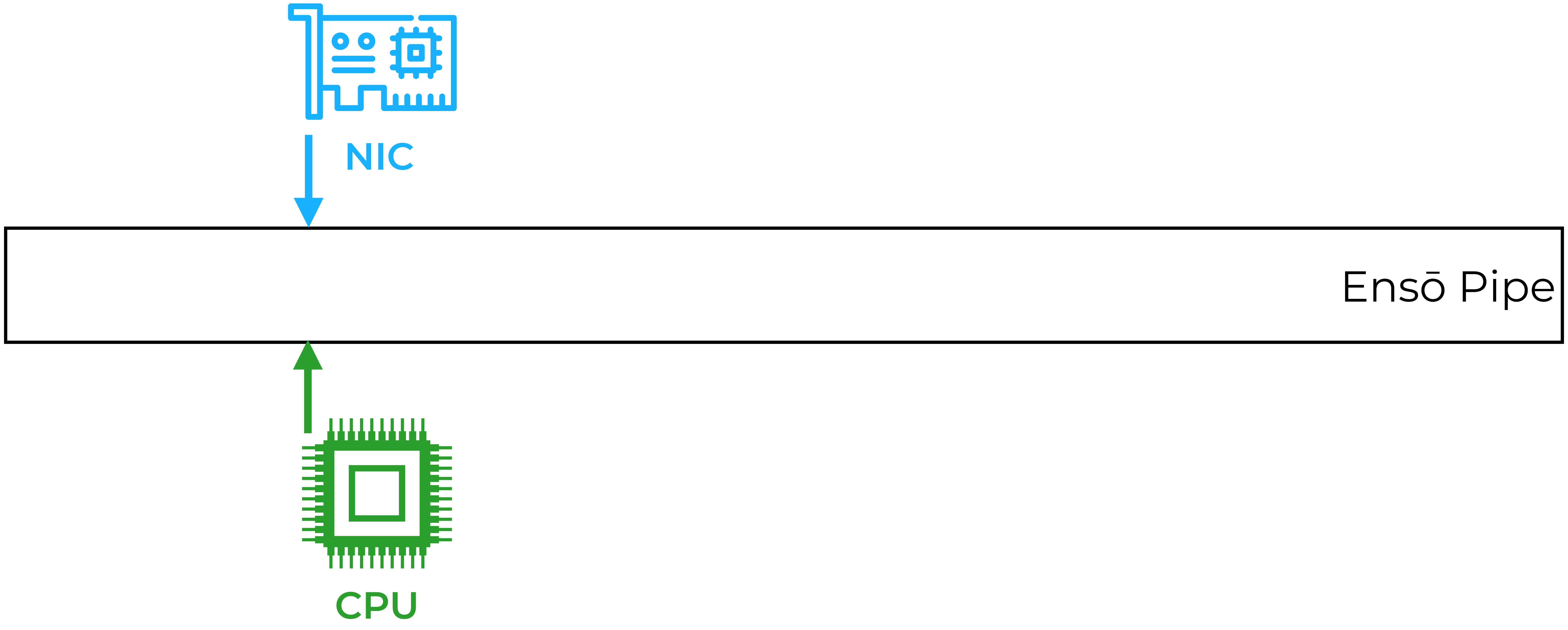
Provide the illusion of an unbounded buffer

| Ensō Pipe |
|---|

Each pipe consists of a single contiguous buffer

# ① How to implement a streaming abstraction?

Provide the illusion of an unbounded buffer

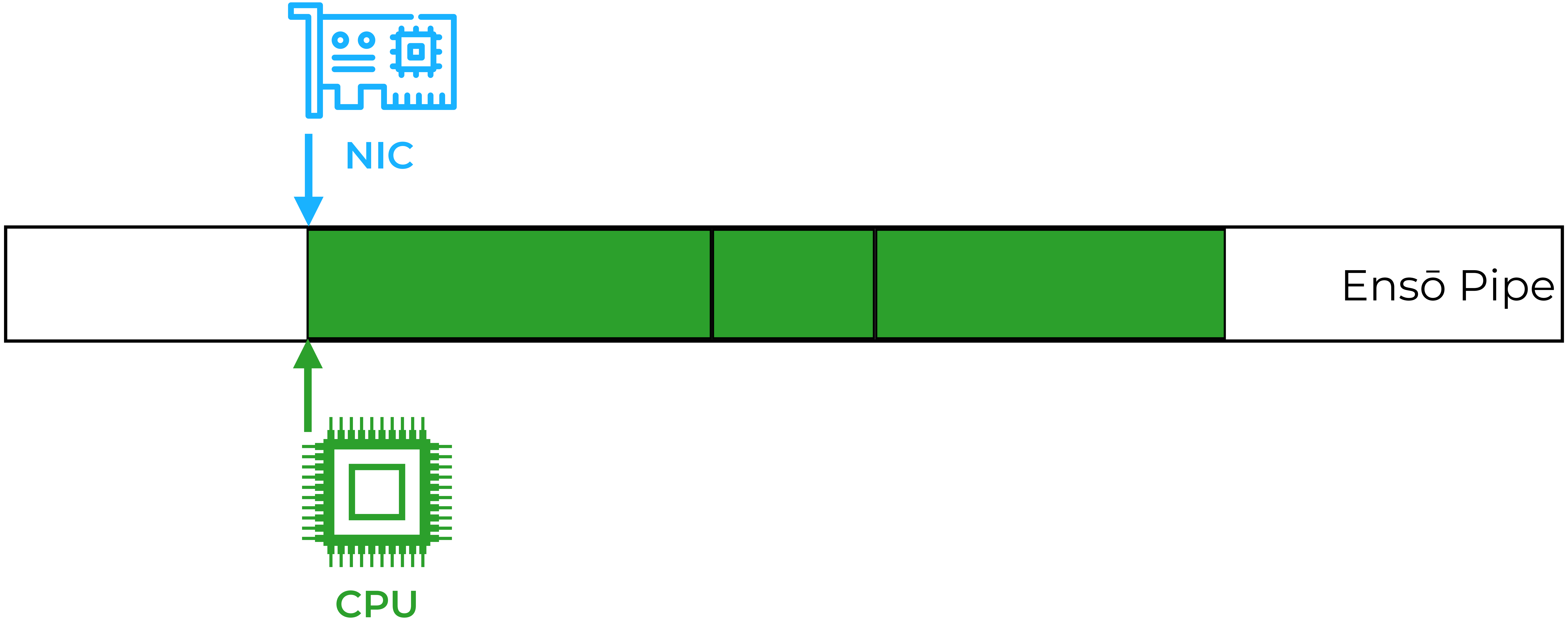



Each pipe consists of a single contiguous buffer

We treat this buffer as a ring buffer for data

# ① How to implement a streaming abstraction?

Provide the illusion of an unbounded buffer

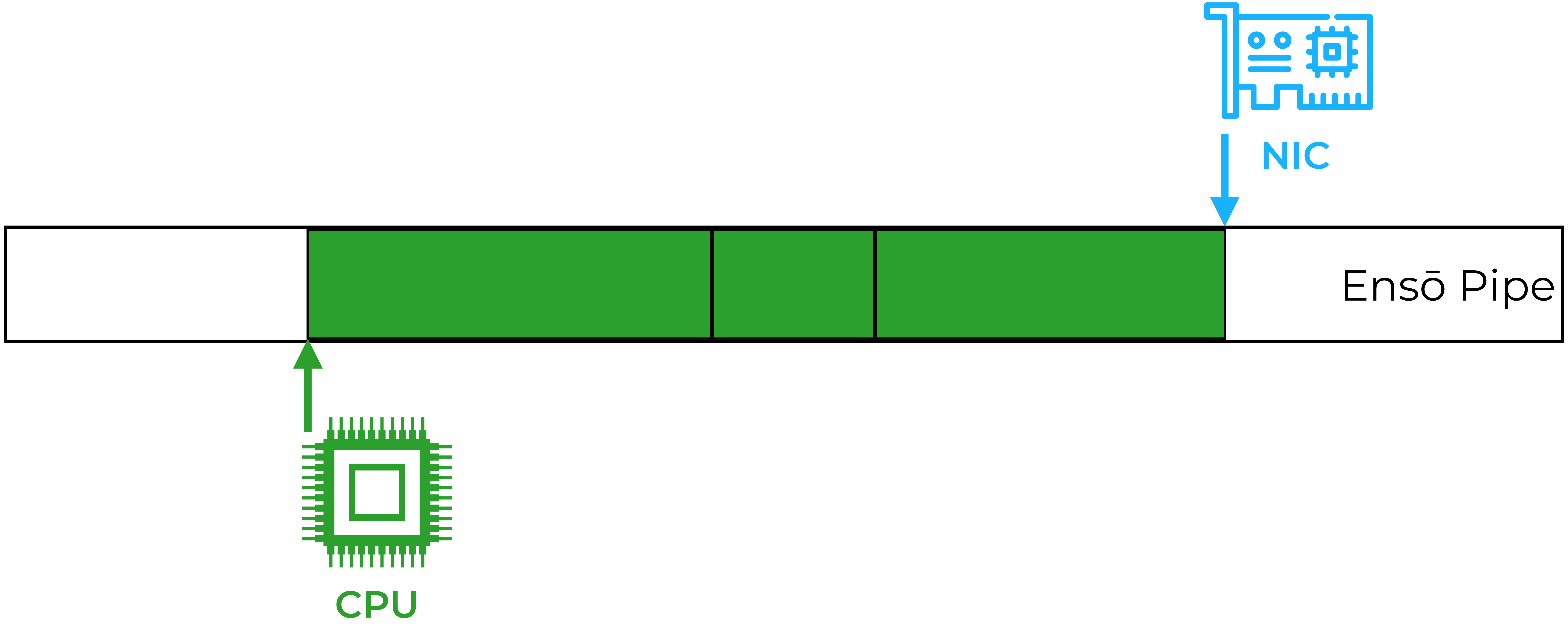



Each pipe consists of a single contiguous buffer

We treat this buffer as a ring buffer for data

# ① How to implement a streaming abstraction?

Provide the illusion of an unbounded buffer



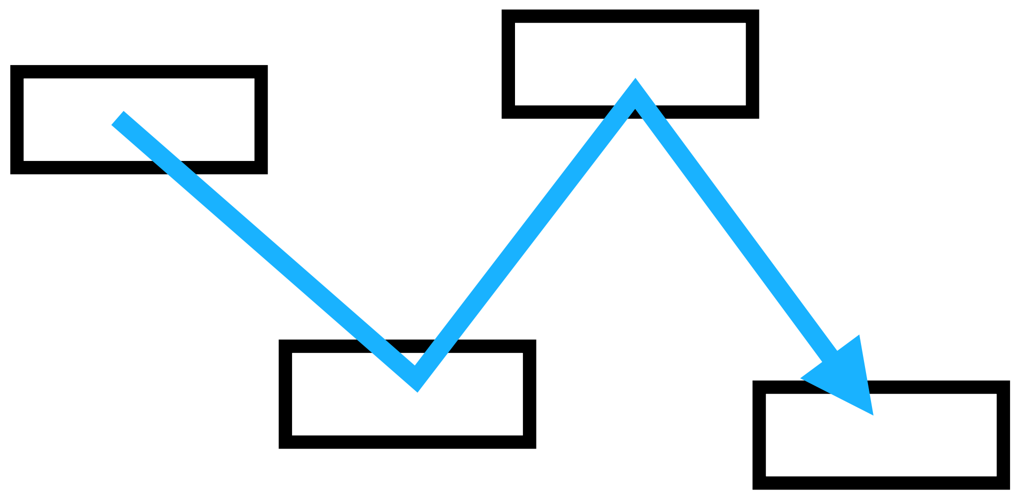

Each pipe consists of a single contiguous buffer

We treat this buffer as a ring buffer for data

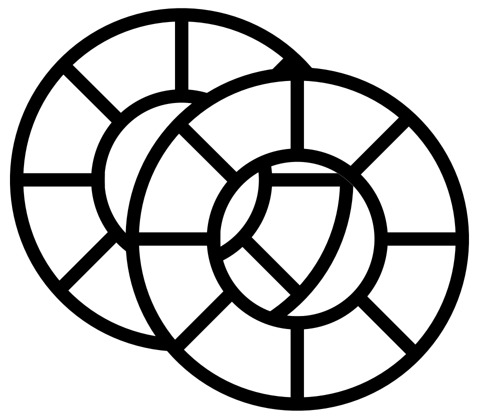
# ② How can a streaming abstraction improve performance?

# ② How can a streaming abstraction improve performance?

Packetized
Interface



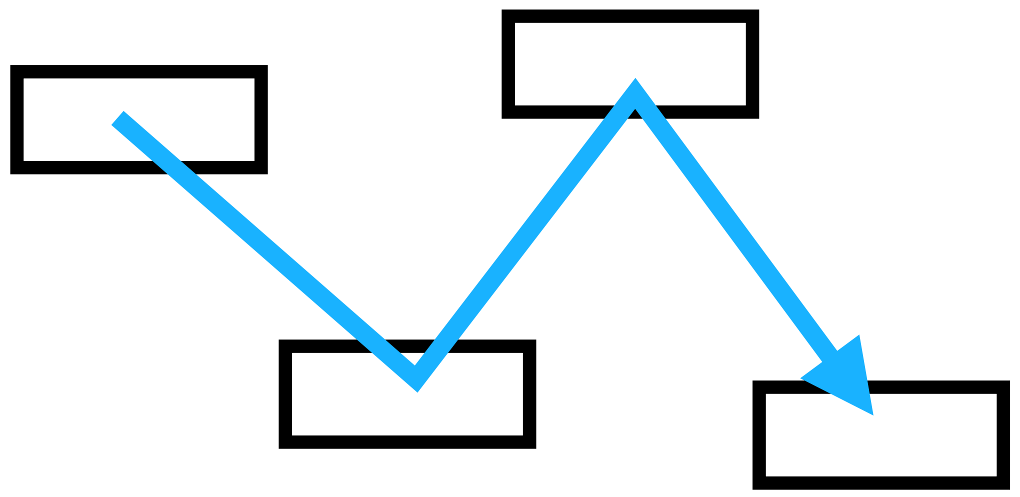
Poor Cache Interaction



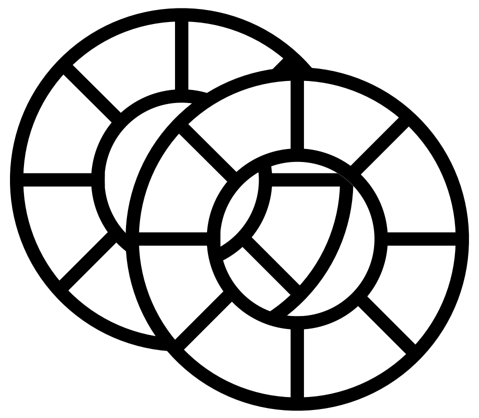
Metadata Overhead

# ② How can a streaming abstraction improve performance?

Packetized
Interface



Poor Cache Interaction

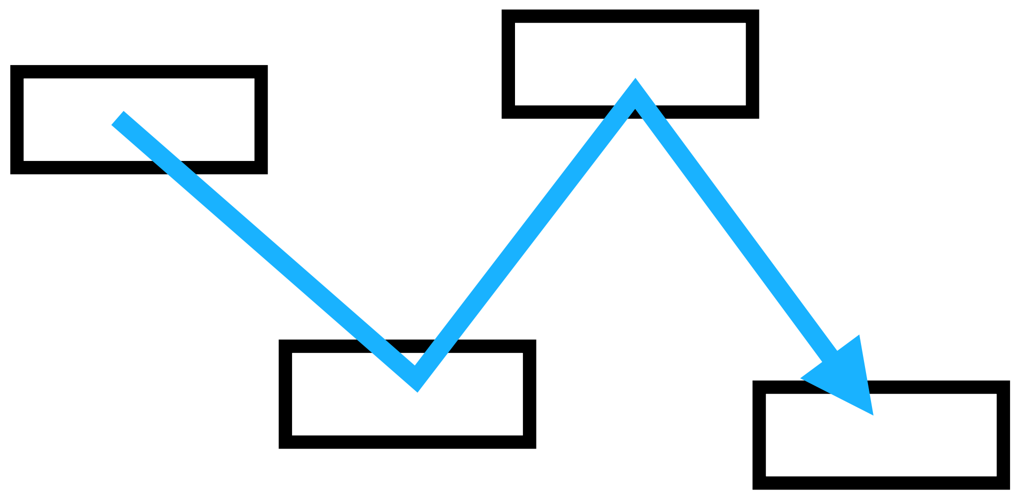


Metadata Overhead

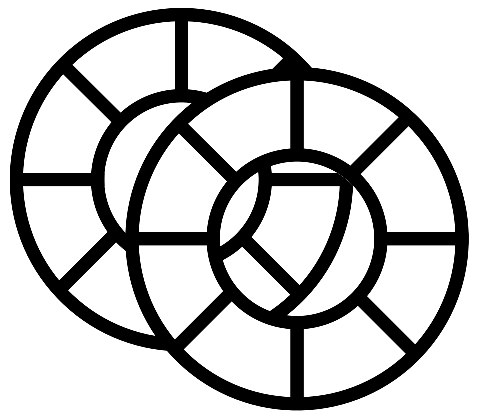
Ensō

# ② How can a streaming abstraction improve performance?

Packetized
Interface



Poor Cache Interaction



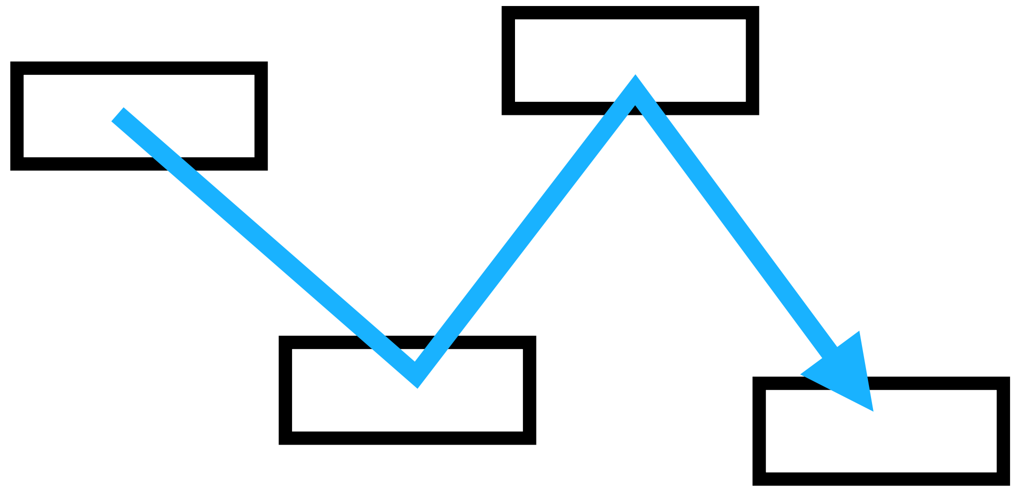
Metadata Overhead

Ensō

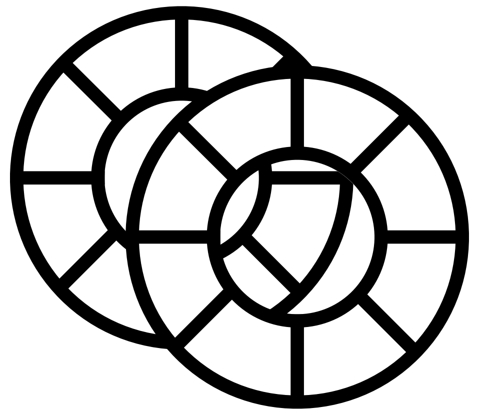
Sequential Memory Access

# ② How can a streaming abstraction improve performance?

Packetized Interface



Poor Cache Interaction



Metadata Overhead

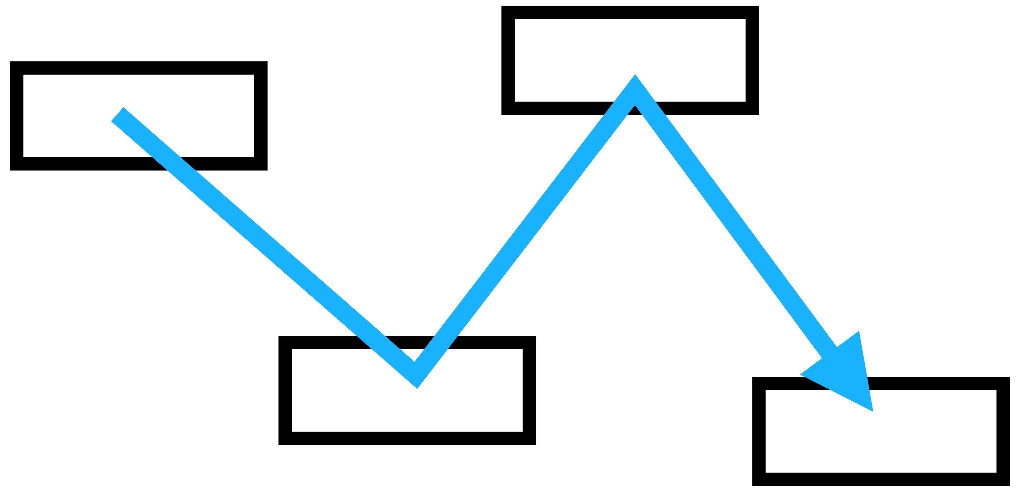
Ensō

Sequential Memory Access

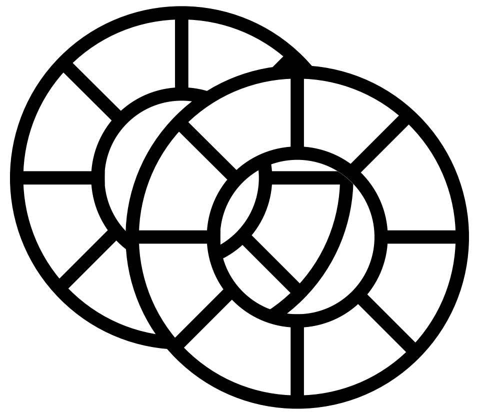
Reduces L1 misses by 95.9%
and L2 misses by 99.5%

# ② How can a streaming abstraction improve performance?

Packetized Interface



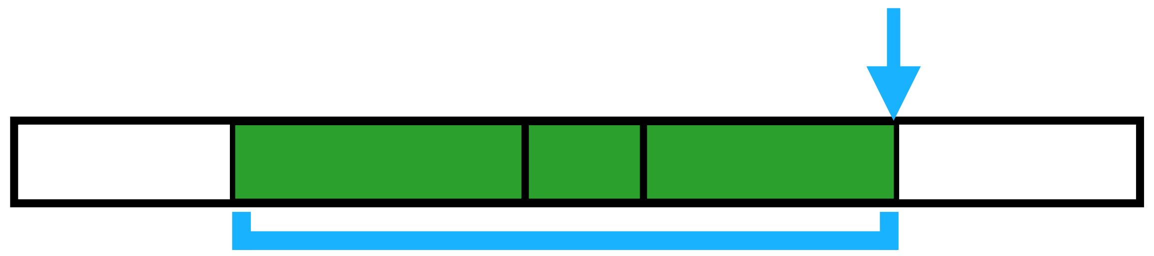
Poor Cache Interaction



Metadata Overhead

Ensō

Sequential Memory Access

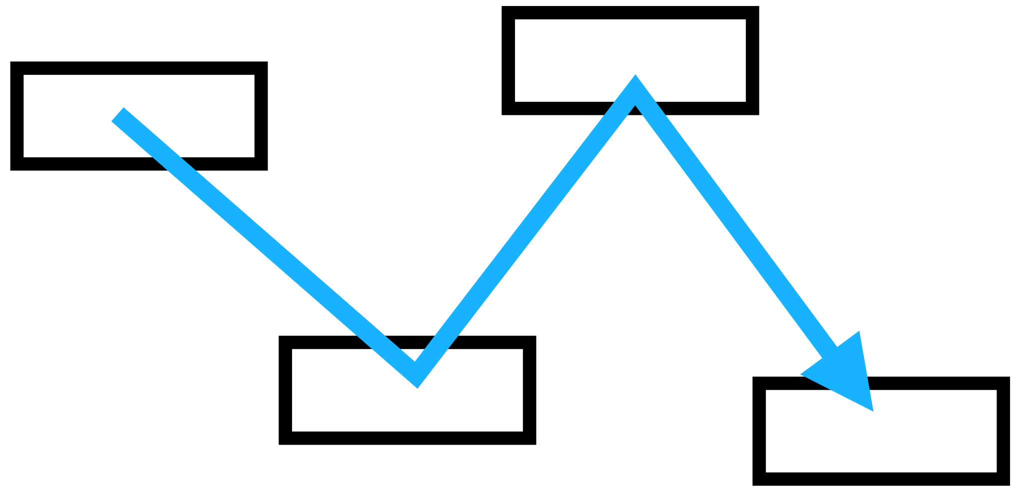


Notifying Batches

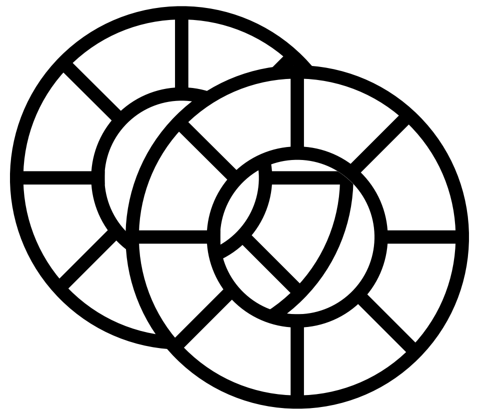
Reduces L1 misses by 95.9%
and L2 misses by 99.5%

# ② How can a streaming abstraction improve performance?

Packetized Interface



Poor Cache Interaction

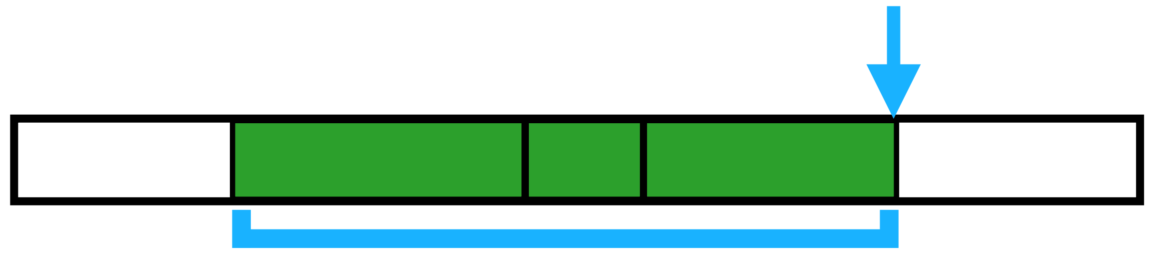


Metadata Overhead

Ensō

Sequential Memory Access

Reduces L1 misses by 95.9% and L2 misses by 99.5%



Notifying Batches

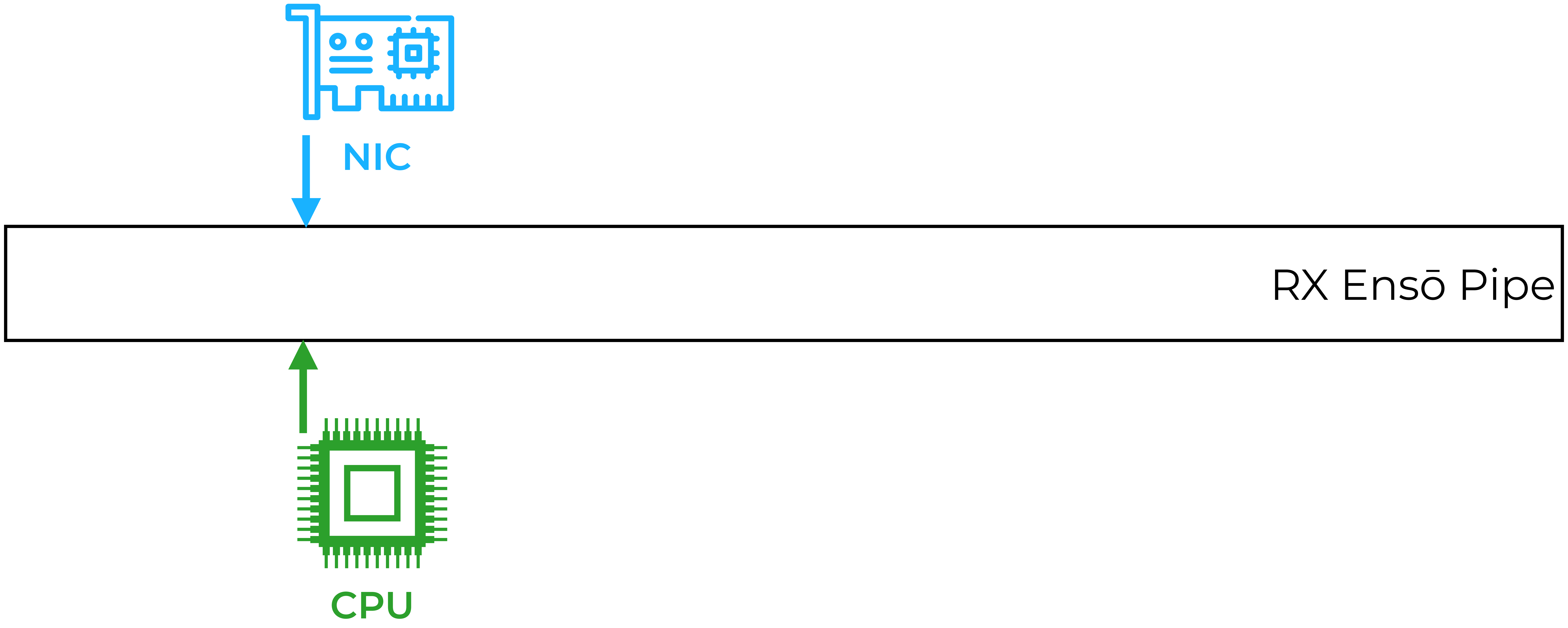
Reduces PCIe metadata traffic by 96.9%

# How often should the NIC notify a batch?

RX Ensō Pipe

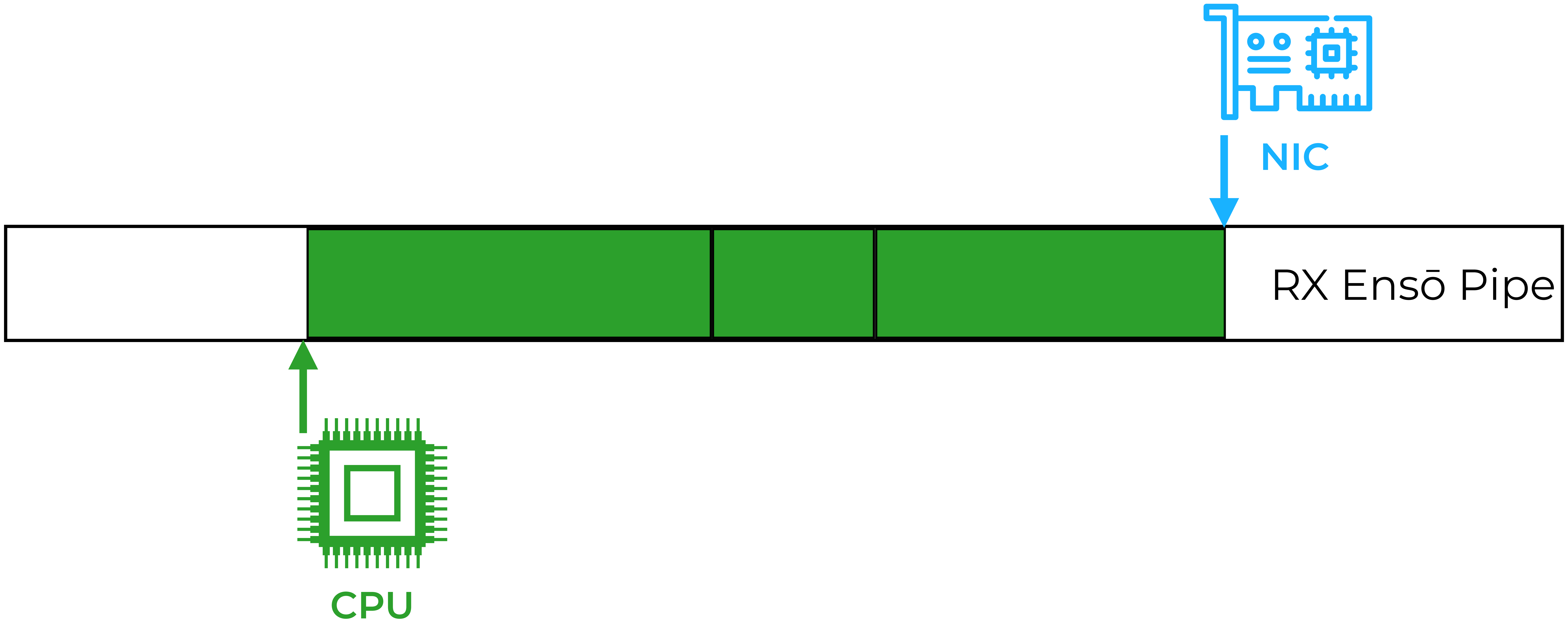
# How often should the NIC notify a batch?

**Naïve strategy:** send an update for *every* piece of data



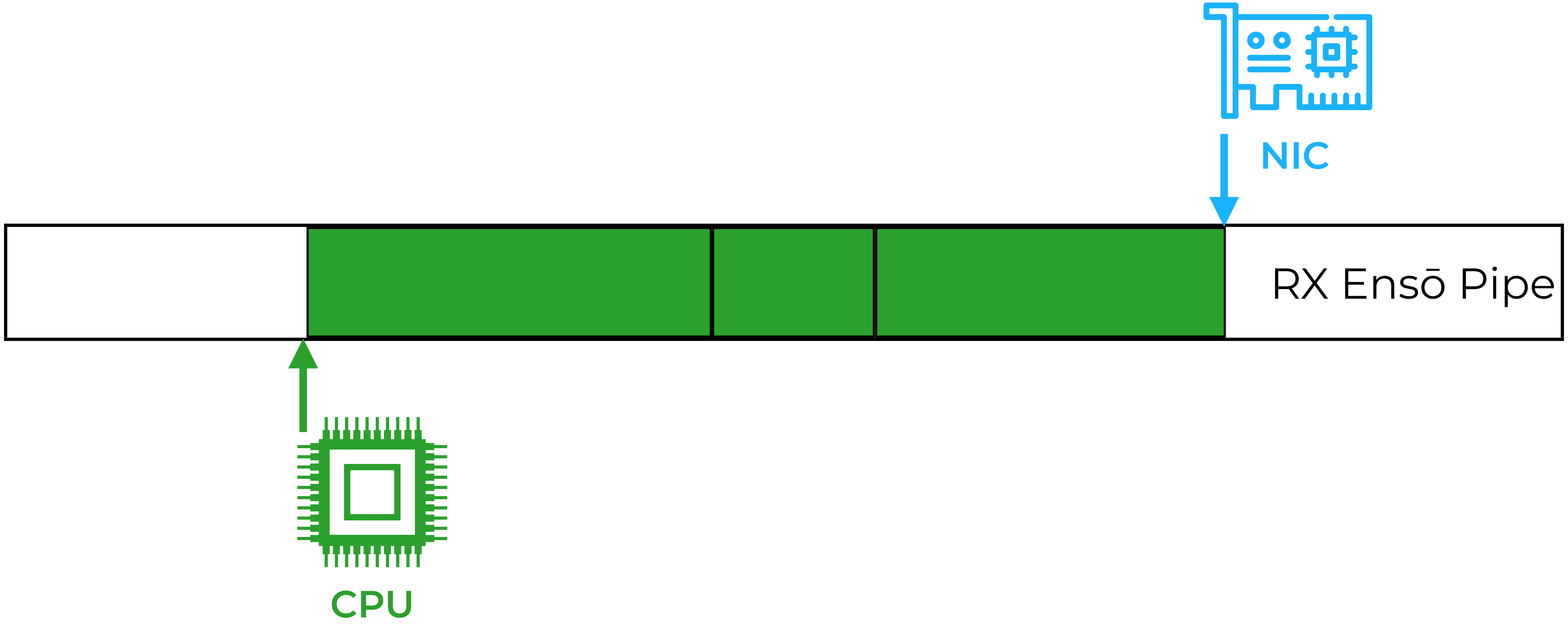
# How often should the NIC notify a batch?

**Naïve strategy:** send an update for *every* piece of data



# How often should the NIC notify a batch?

**Naïve strategy:** send an update for *every* piece of data




**Problem:** Per-packet overhead

# Notification Pacing in Ensō

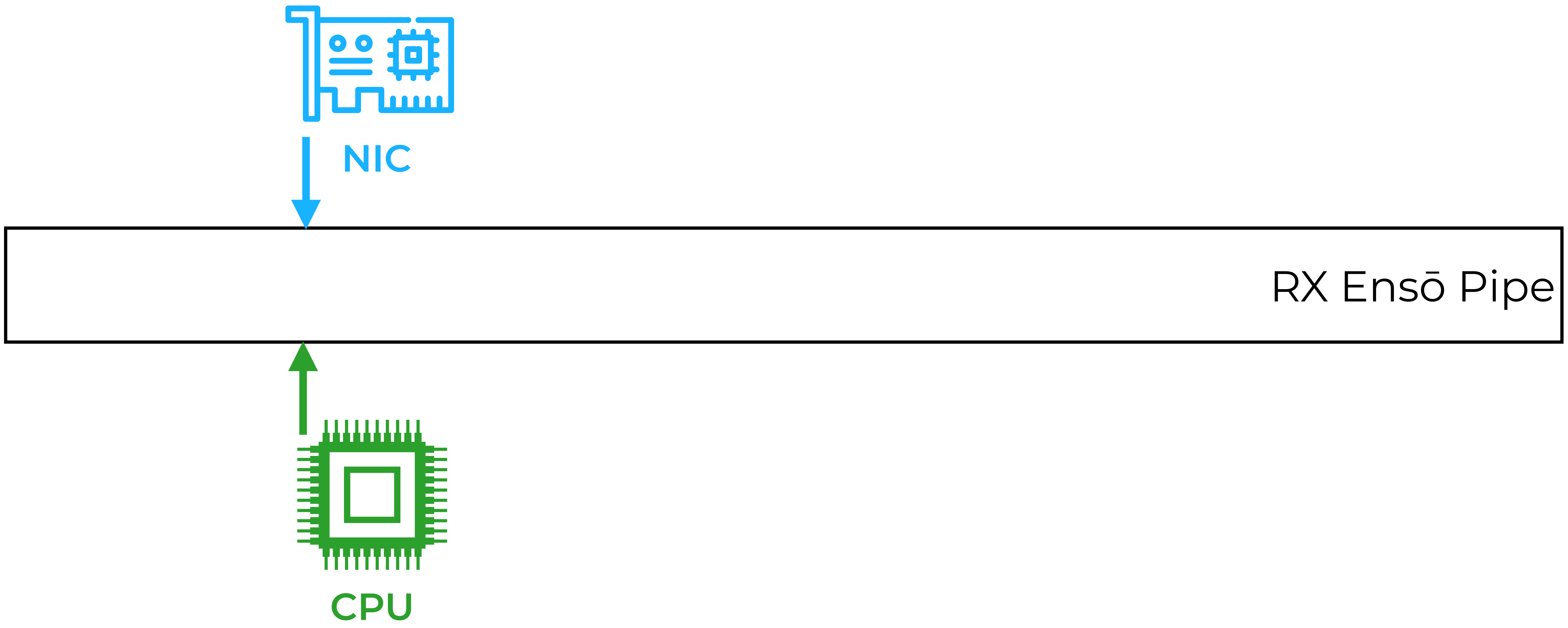
Ensō combines two techniques

① Reactive Notifications

② Notification Prefetching

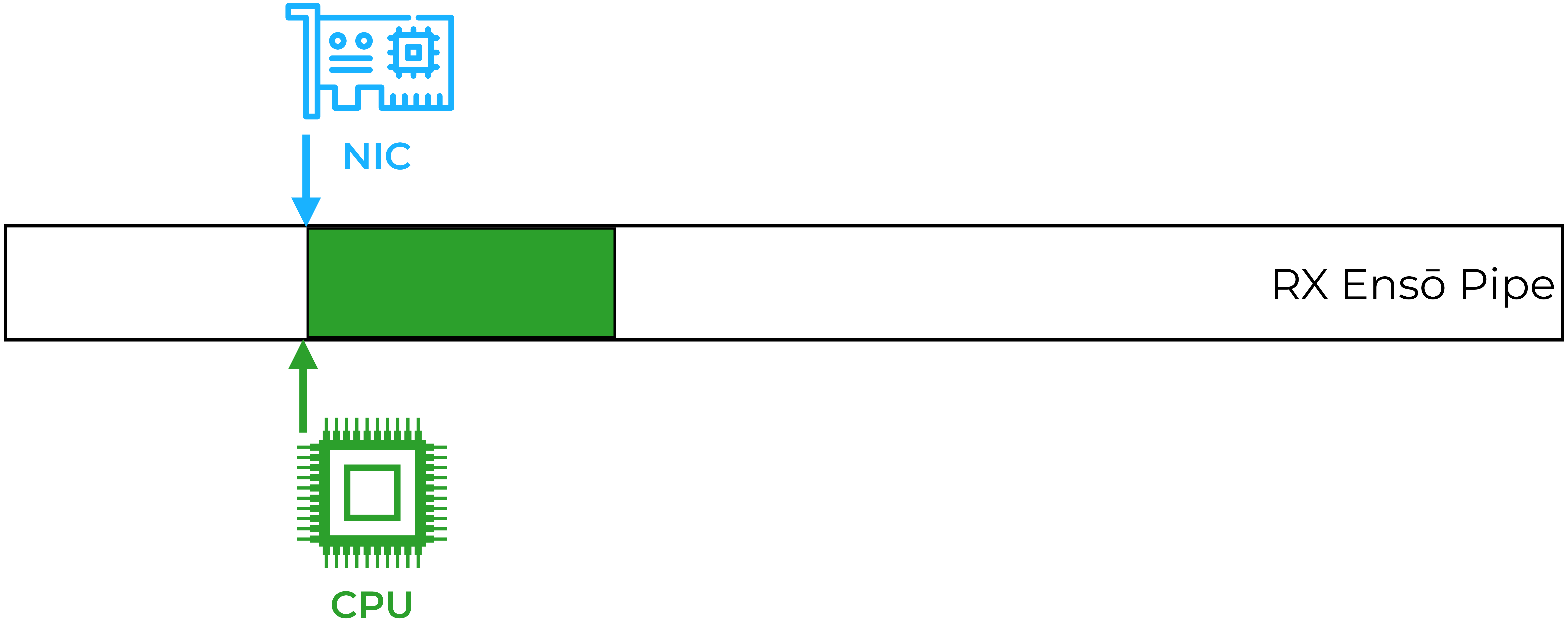
# ① Reactive Notifications

The NIC updates its pointer in *reaction* to CPU pointer updates



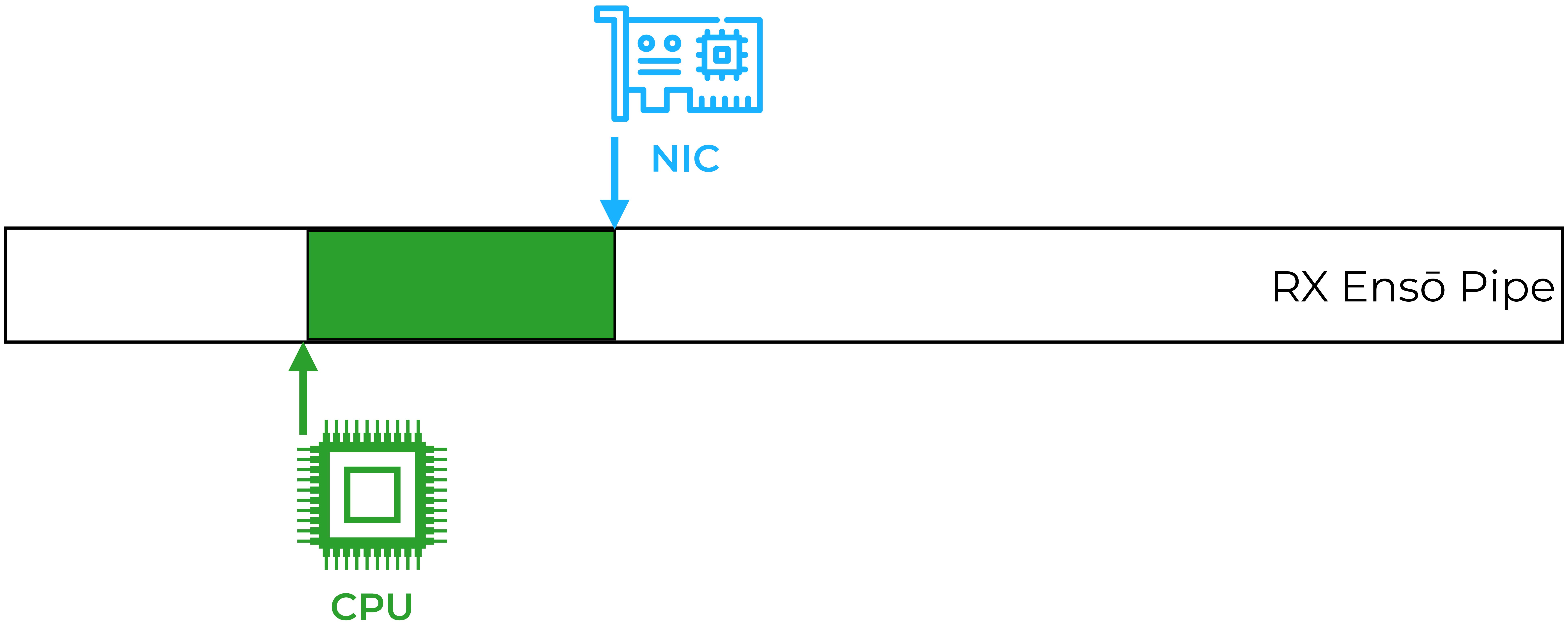
# ① Reactive Notifications

The NIC updates its pointer in *reaction* to CPU pointer updates



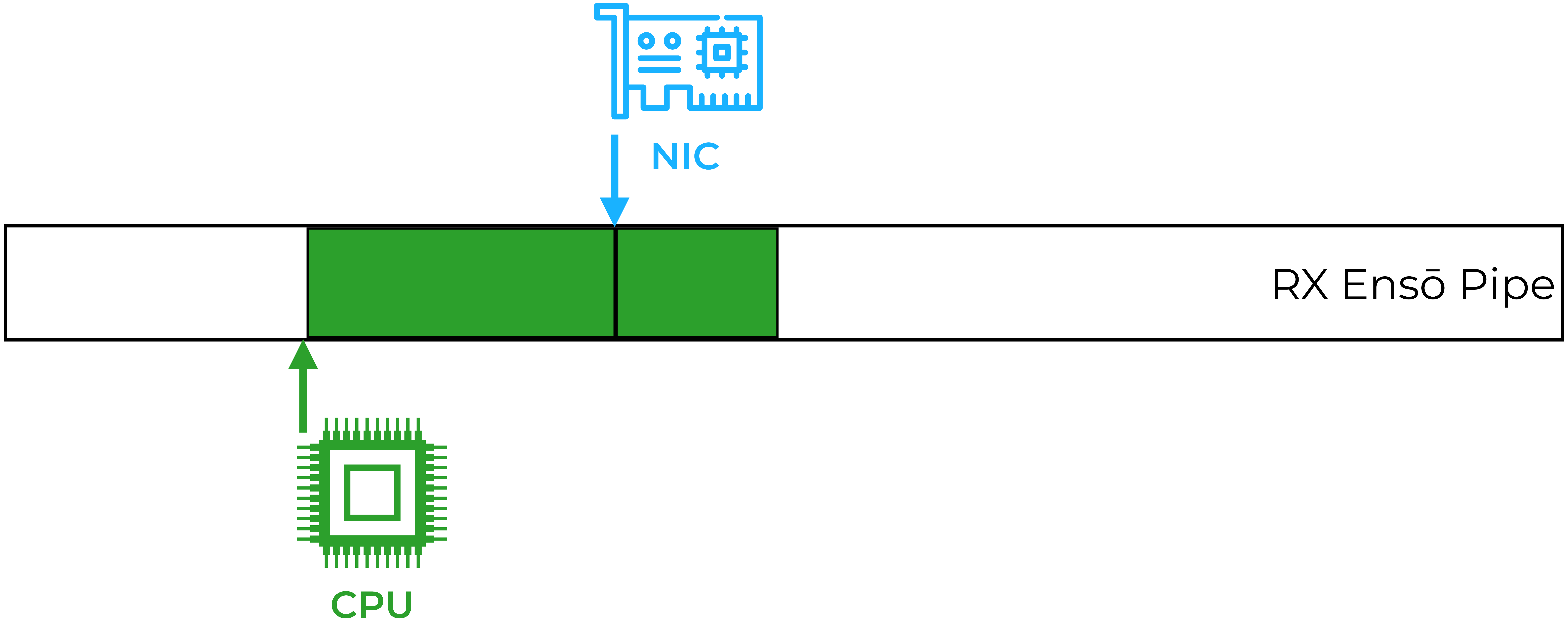
# ① Reactive Notifications

The NIC updates its pointer in *reaction* to CPU pointer updates



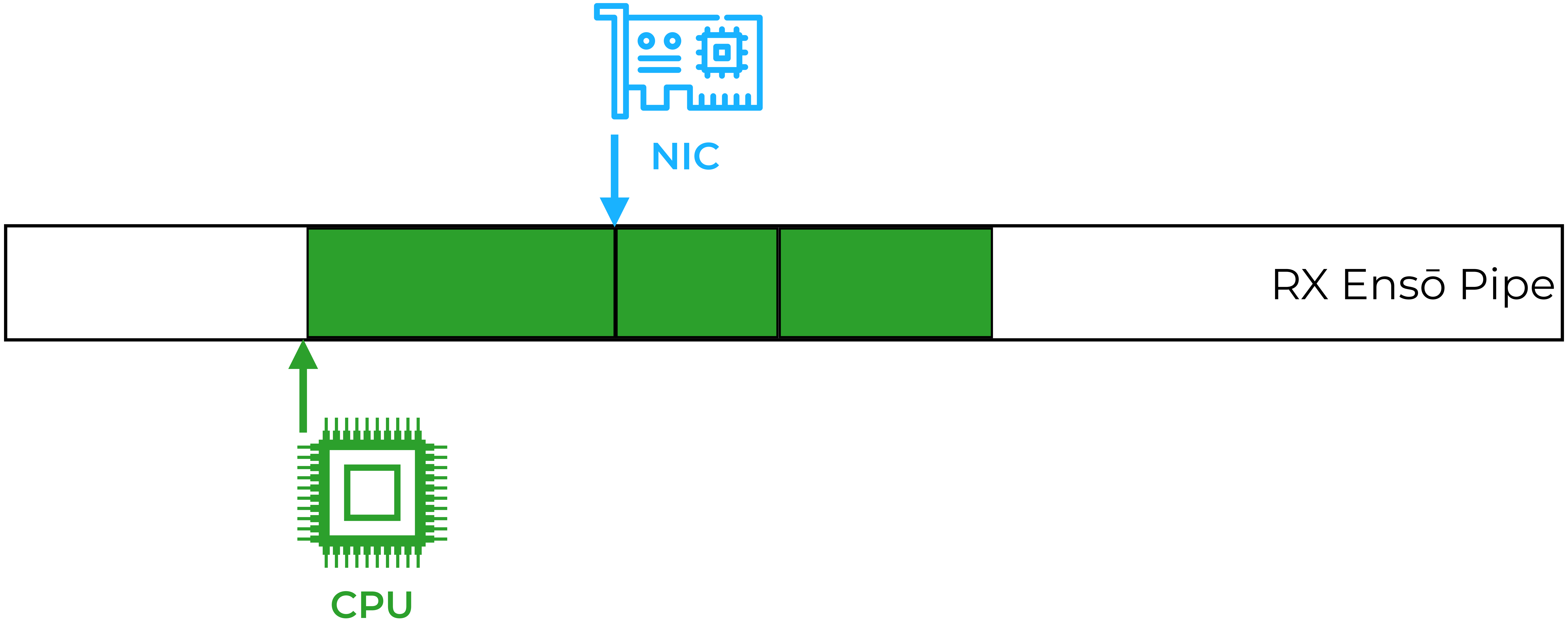
# ① Reactive Notifications

The NIC updates its pointer in *reaction* to CPU pointer updates



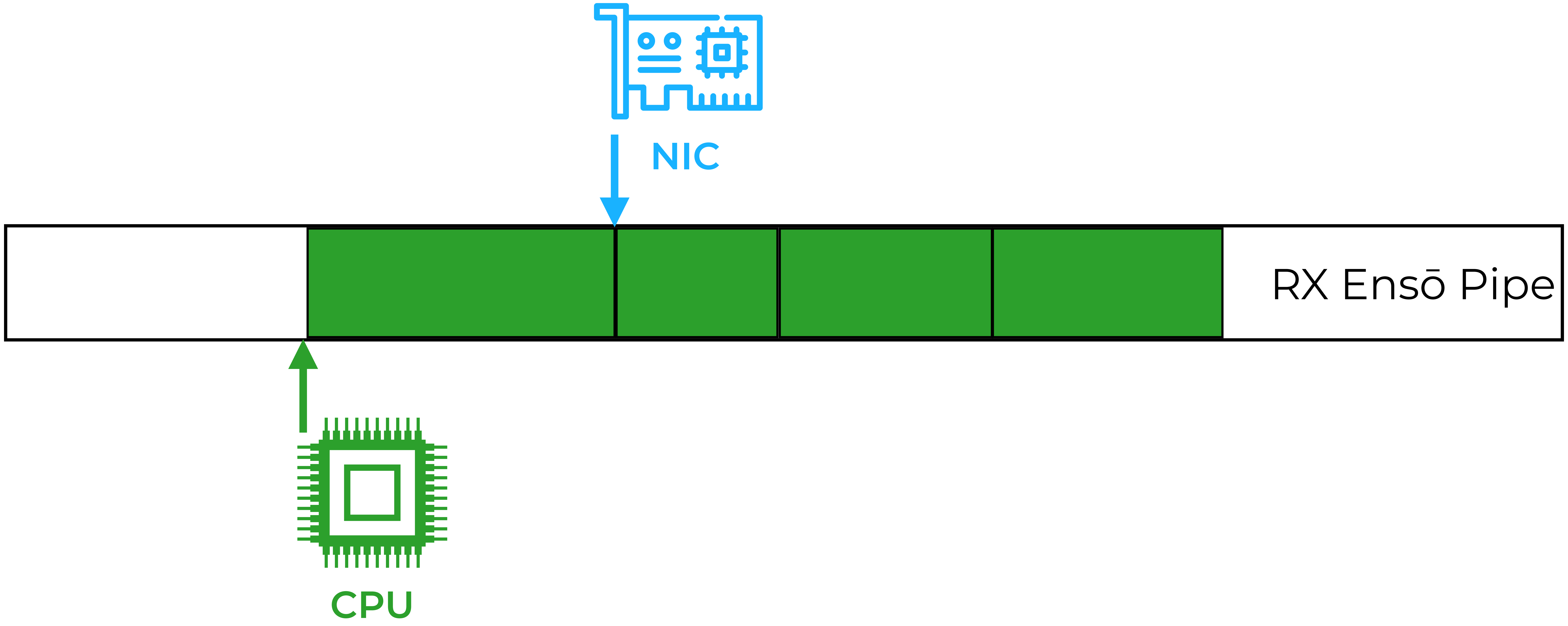
# ① Reactive Notifications

The NIC updates its pointer in *reaction* to CPU pointer updates



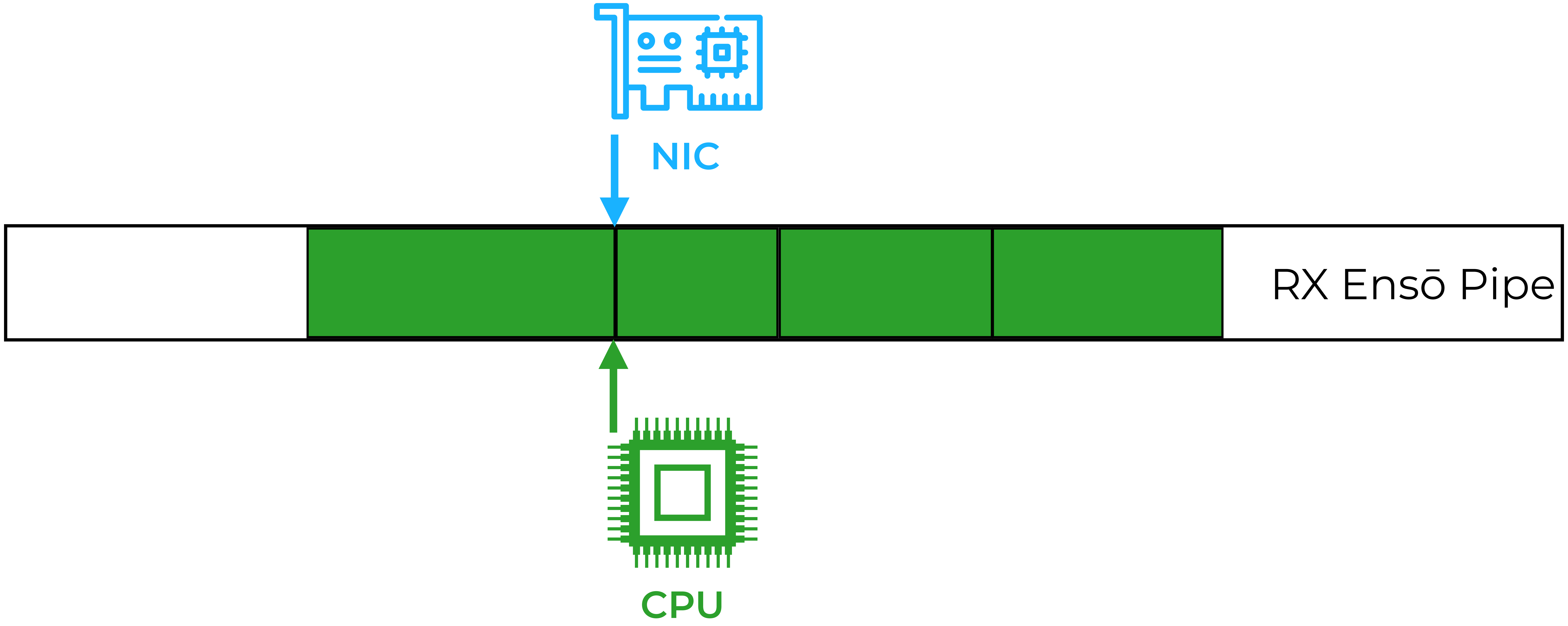
# ① Reactive Notifications

The NIC updates its pointer in *reaction* to CPU pointer updates



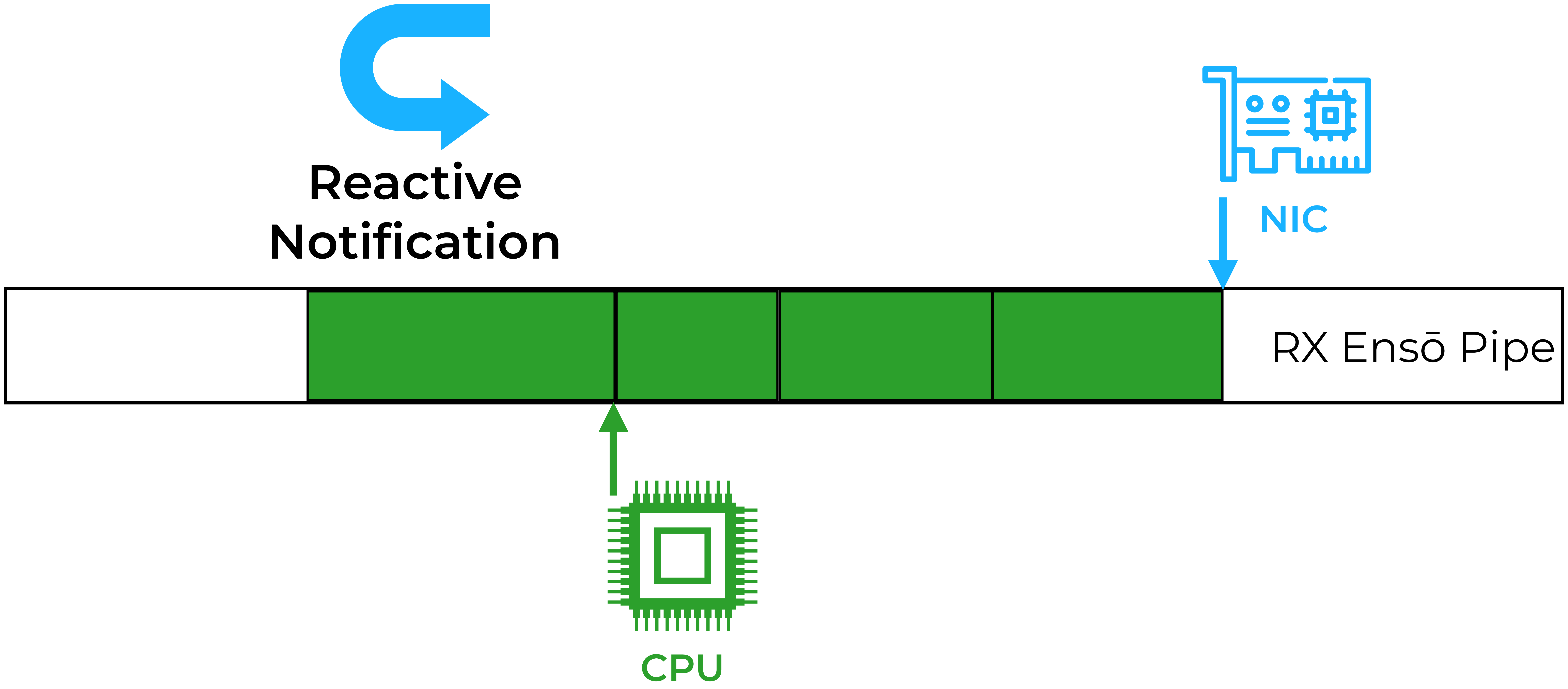
# ① Reactive Notifications

The NIC updates its pointer in *reaction* to CPU pointer updates



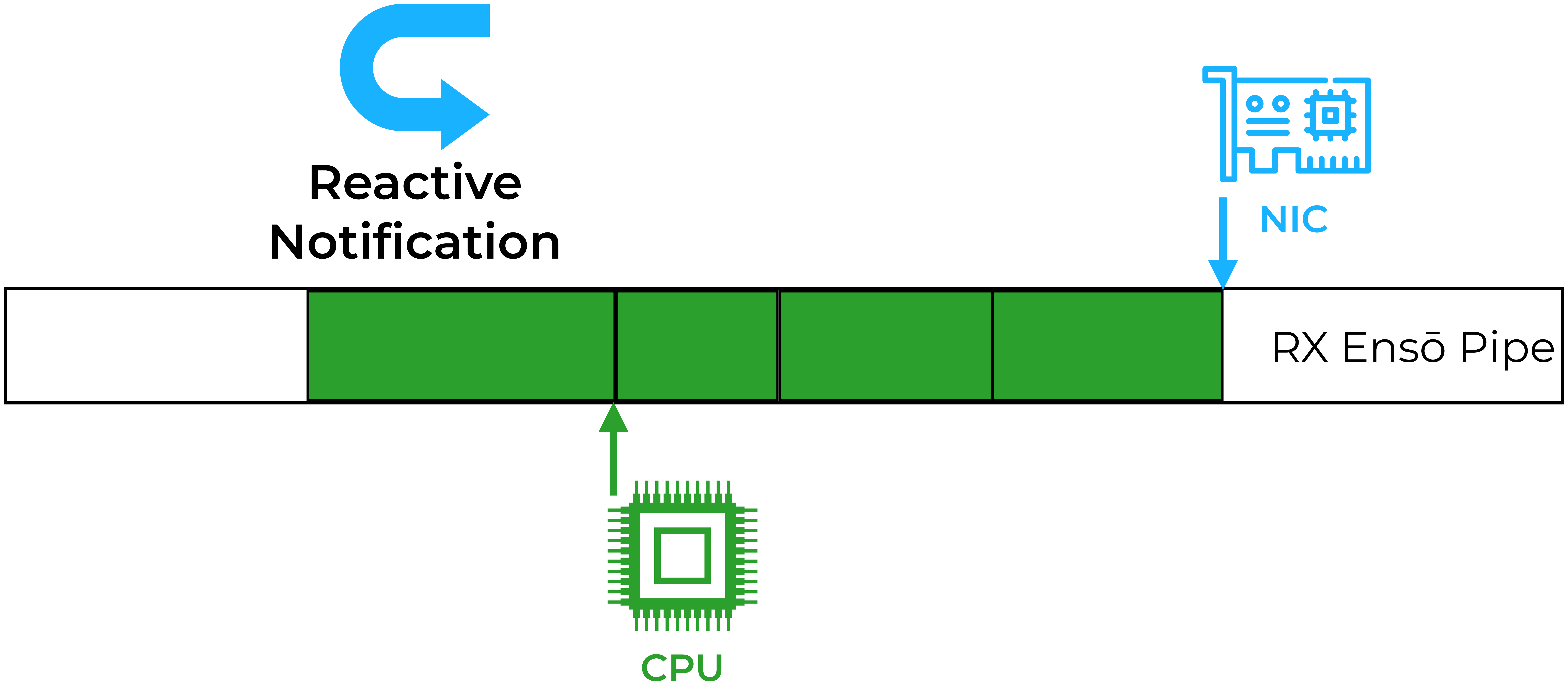
# ① Reactive Notifications

The NIC updates its pointer in *reaction* to CPU pointer updates



# ① Reactive Notifications

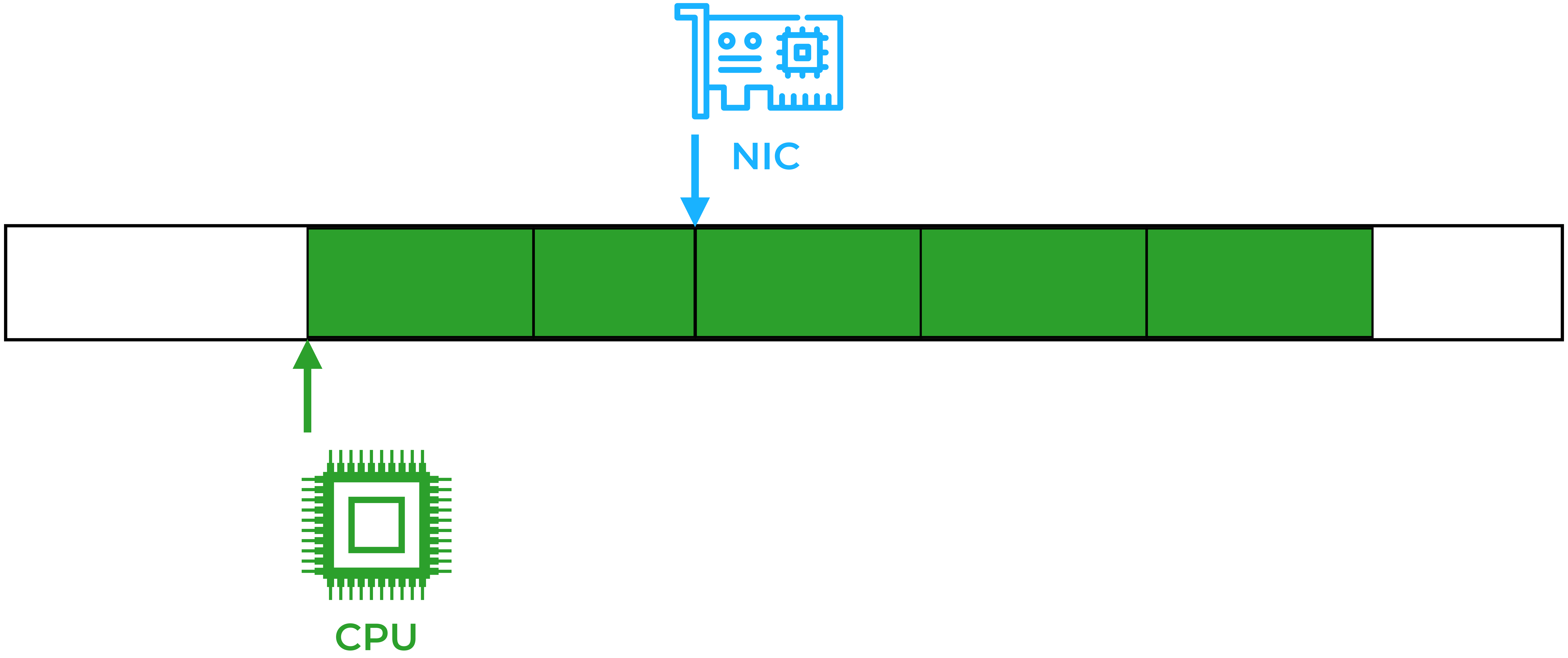
The NIC updates its pointer in *reaction* to CPU pointer updates




**Only sends notifications that are strictly necessary**

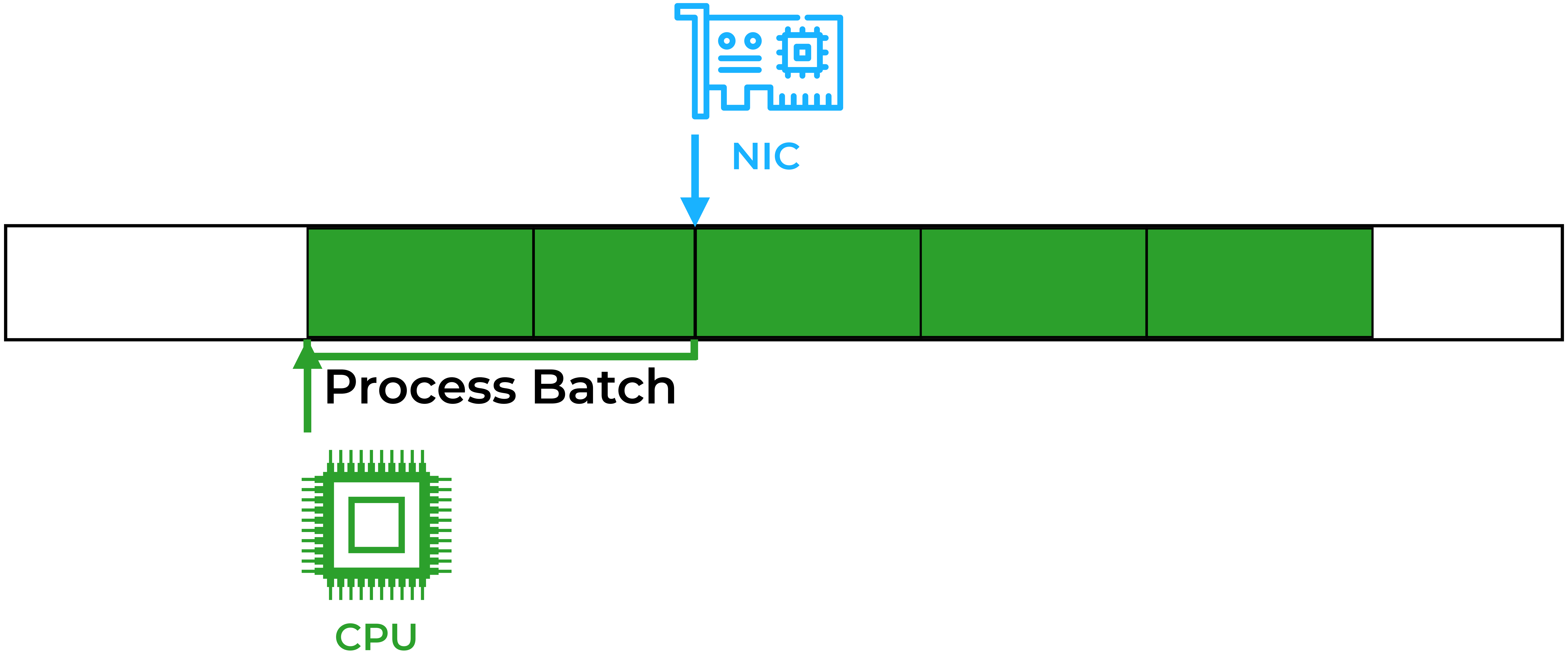
# Problem: PCIe Latency

Software may need to wait up to 1 PCIe RTT for a notification



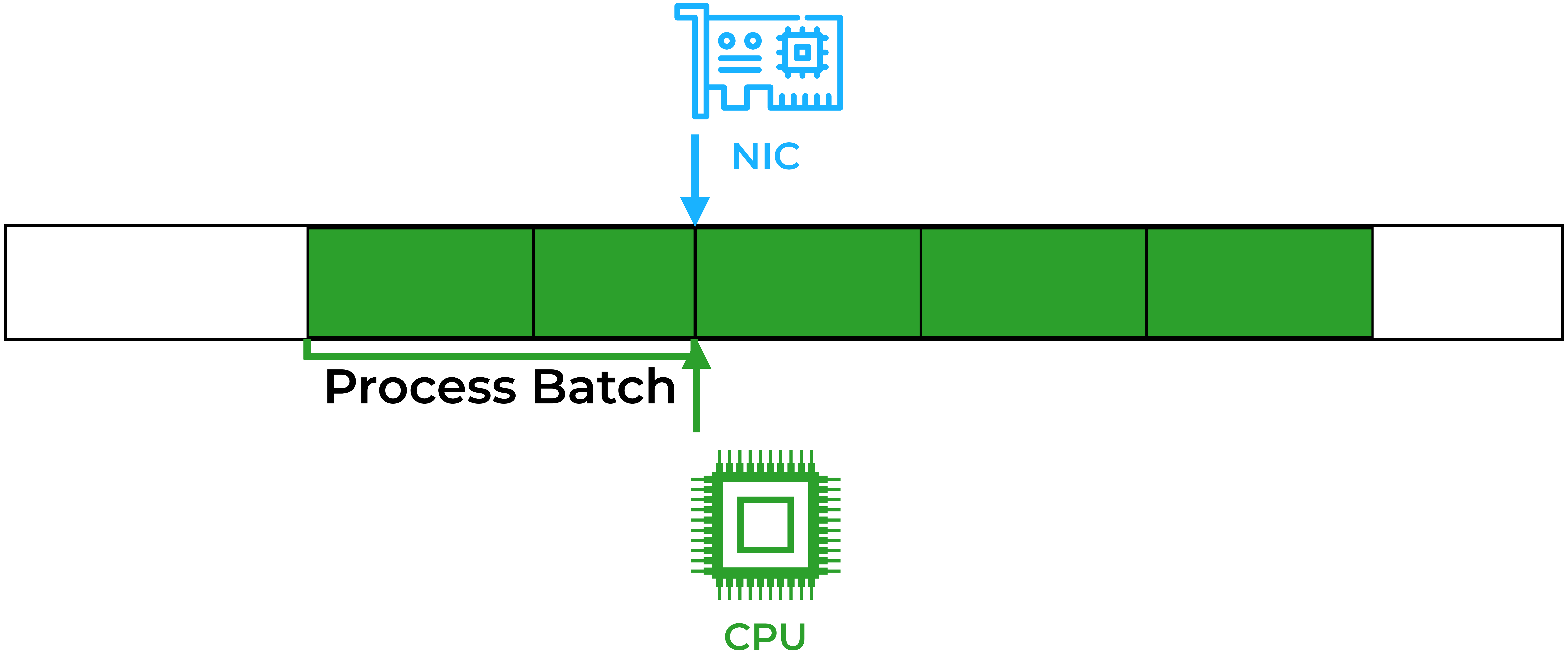
# Problem: PCIe Latency

Software may need to wait up to 1 PCIe RTT for a notification



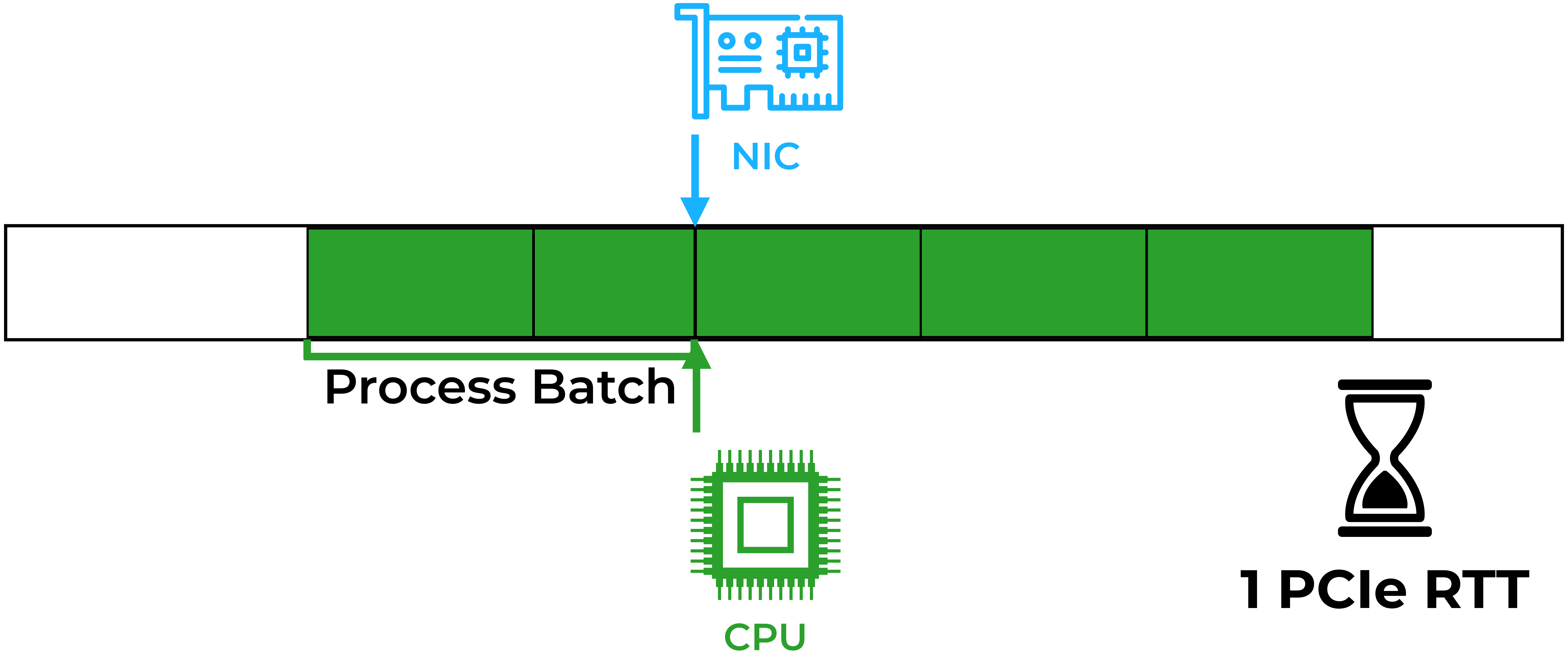
# Problem: PCIe Latency

Software may need to wait up to 1 PCIe RTT for a notification



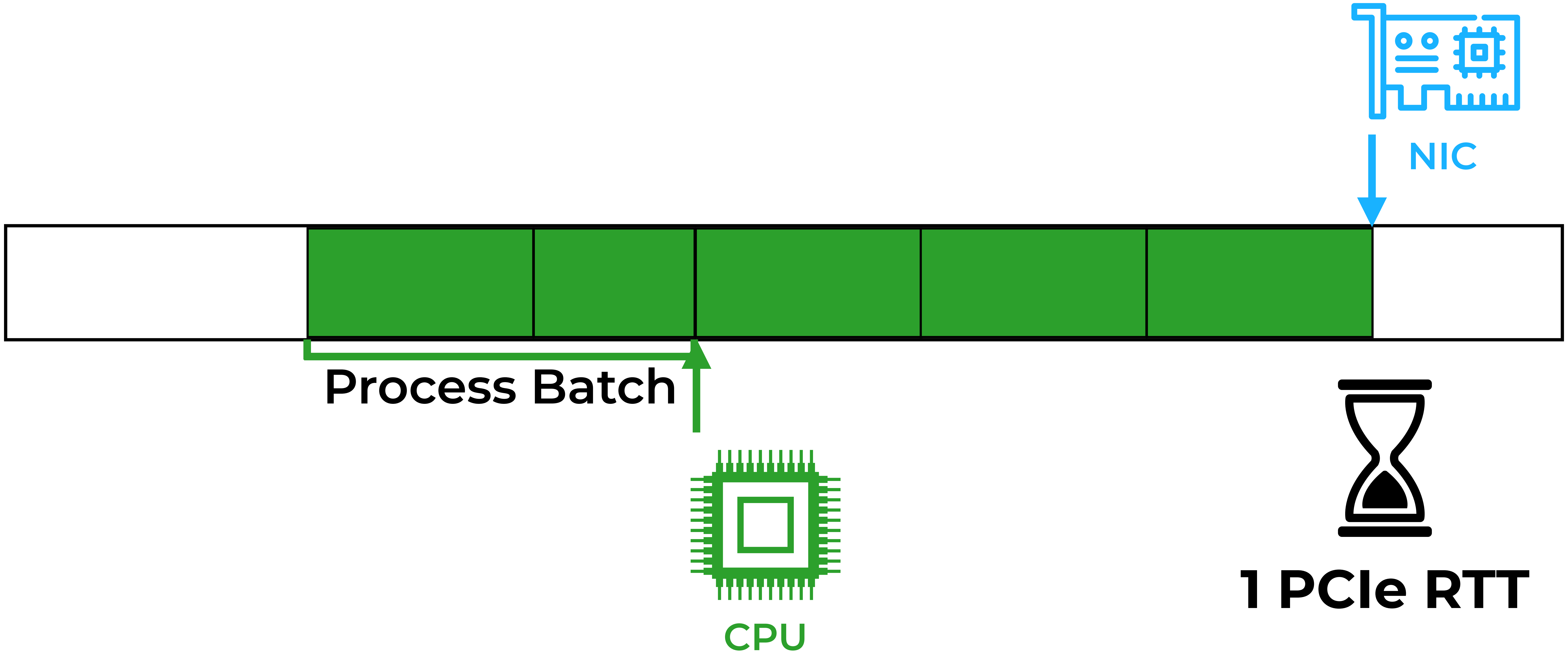
# Problem: PCIe Latency

Software may need to wait up to 1 PCIe RTT for a notification



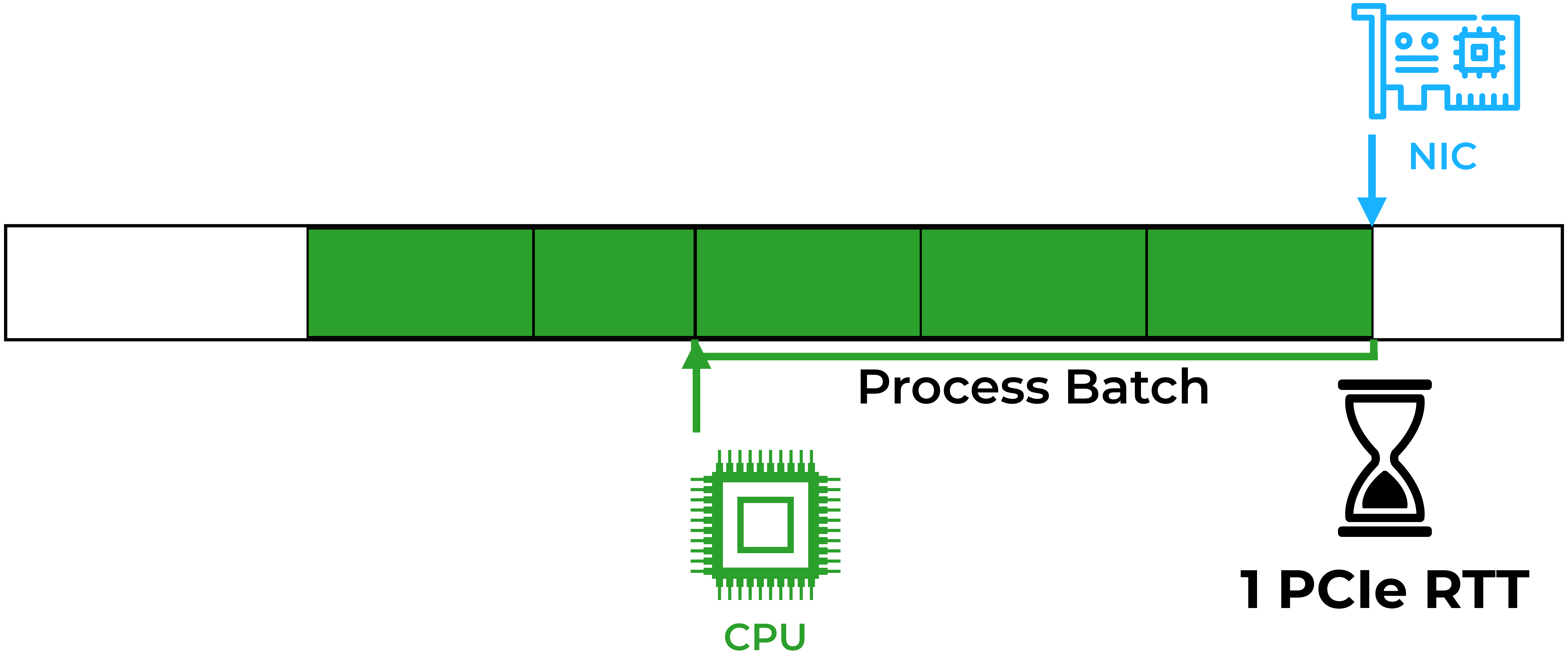
# Problem: PCIe Latency

Software may need to wait up to 1 PCIe RTT for a notification



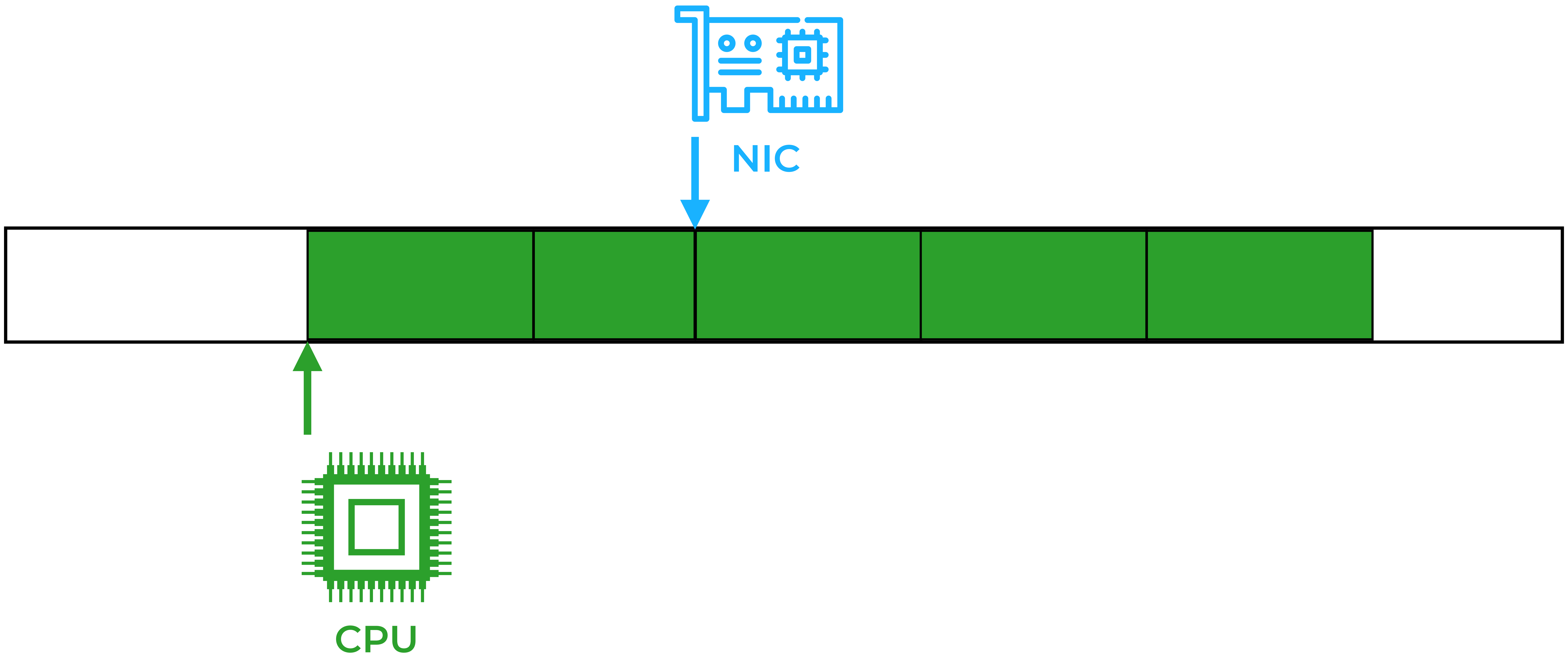
# Problem: PCIe Latency

Software may need to wait up to 1 PCIe RTT for a notification



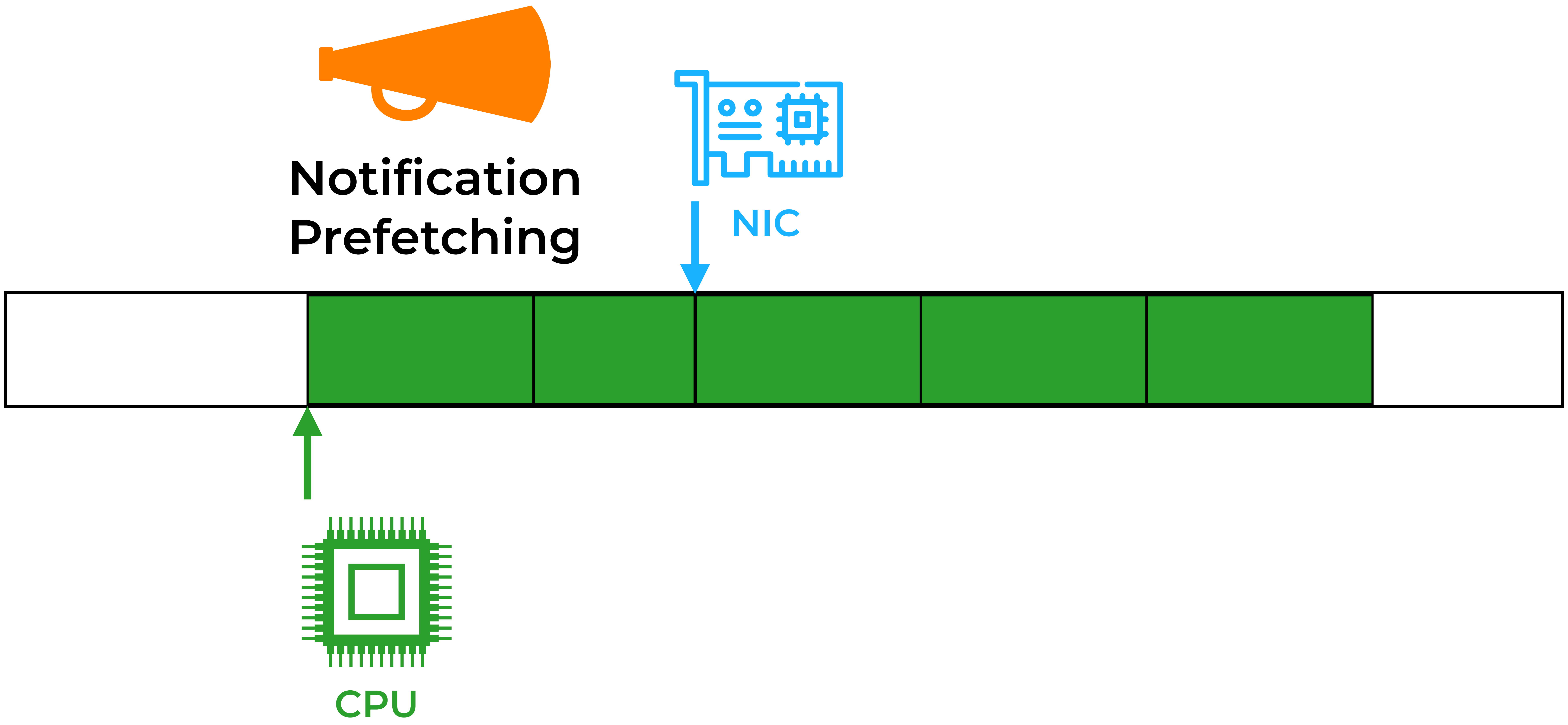
# ② Notification Prefetching

Software can *explicitly* request pointer updates from the NIC



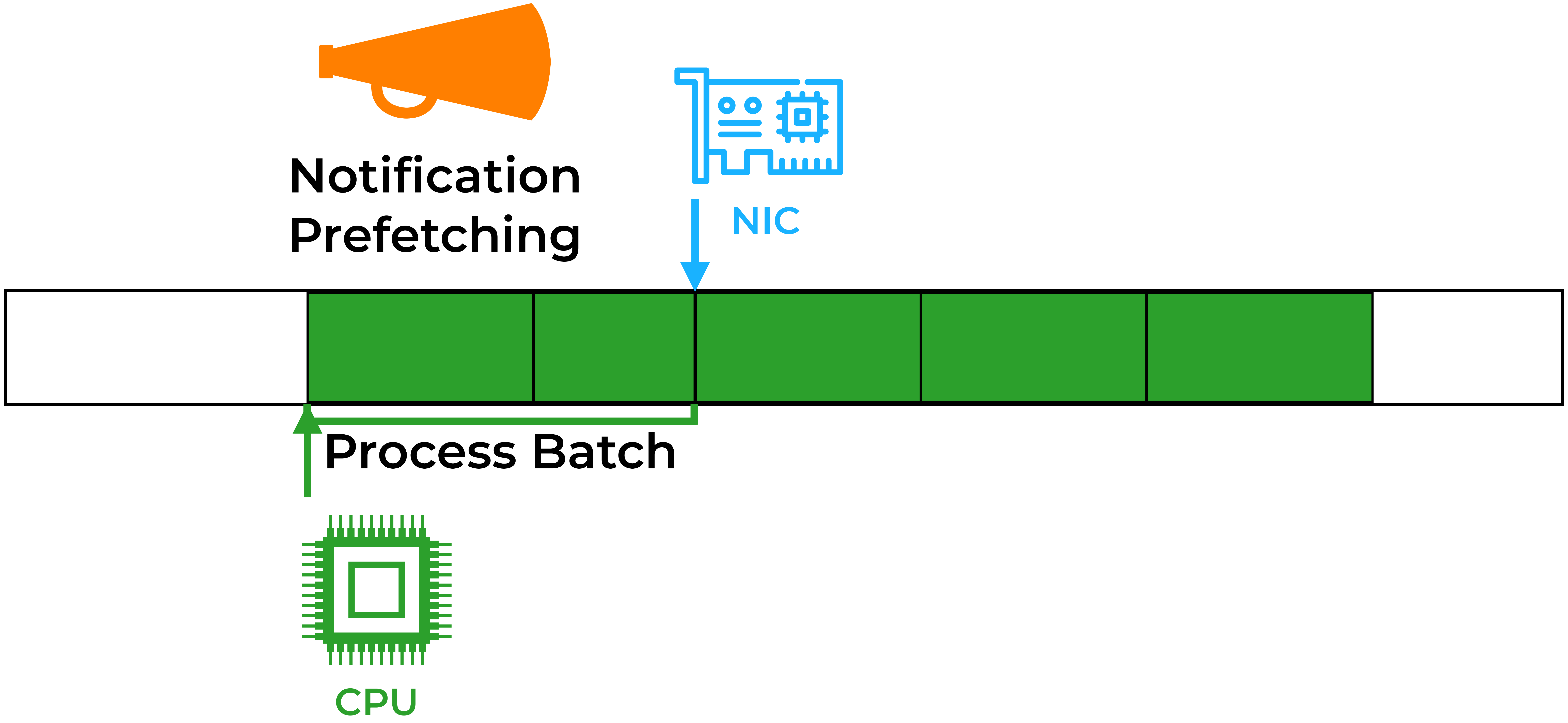
# ② Notification Prefetching

Software can *explicitly* request pointer updates from the NIC



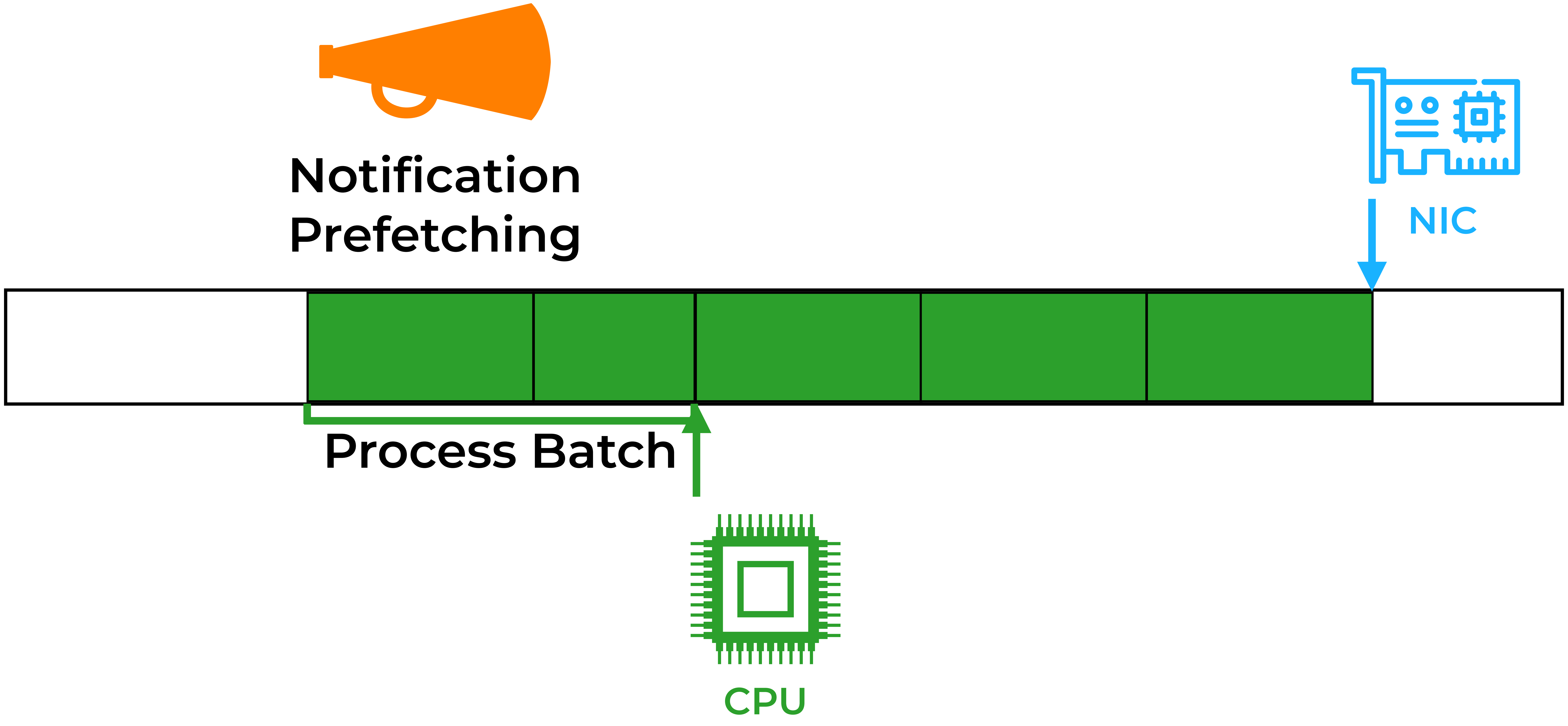
# ② Notification Prefetching

Software can *explicitly* request pointer updates from the NIC



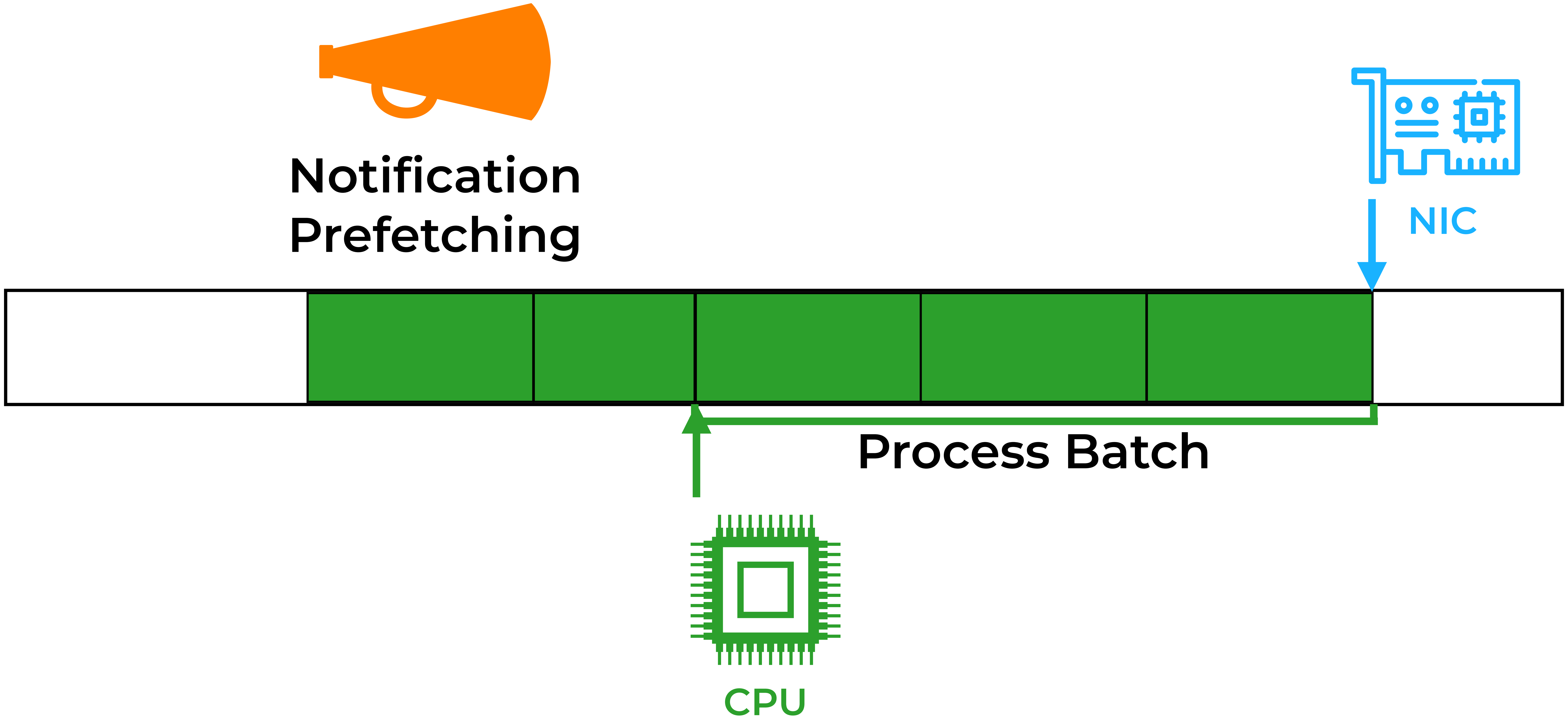
# ② Notification Prefetching

Software can *explicitly* request pointer updates from the NIC



# ② Notification Prefetching

Software can *explicitly* request pointer updates from the NIC



# Many other design challenges...

How to notify pointer updates efficiently?

How to deal with data that wrap around?

How to design a scalable hardware?

How to avoid copies in applications that send data back (e.g., Network Functions)?

# Many other design challenges...

How to notify pointer updates efficiently?

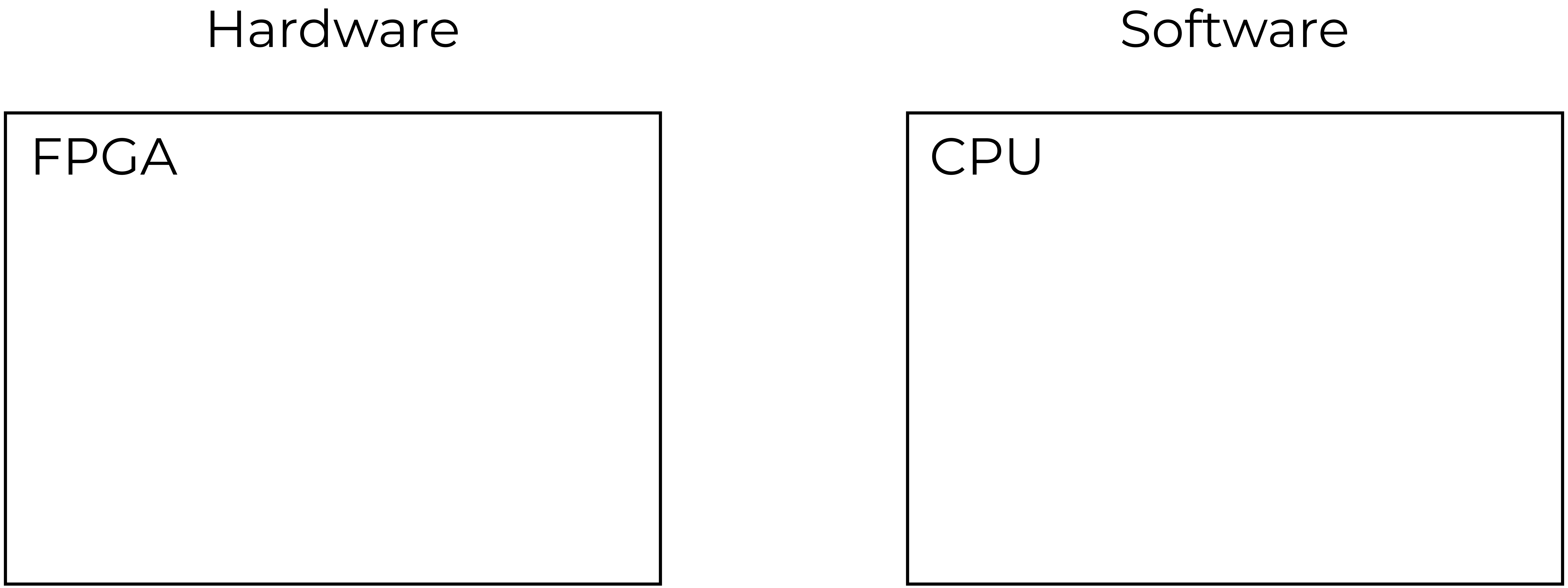
How to deal with data that wrap around?

How to design a scalable hardware?

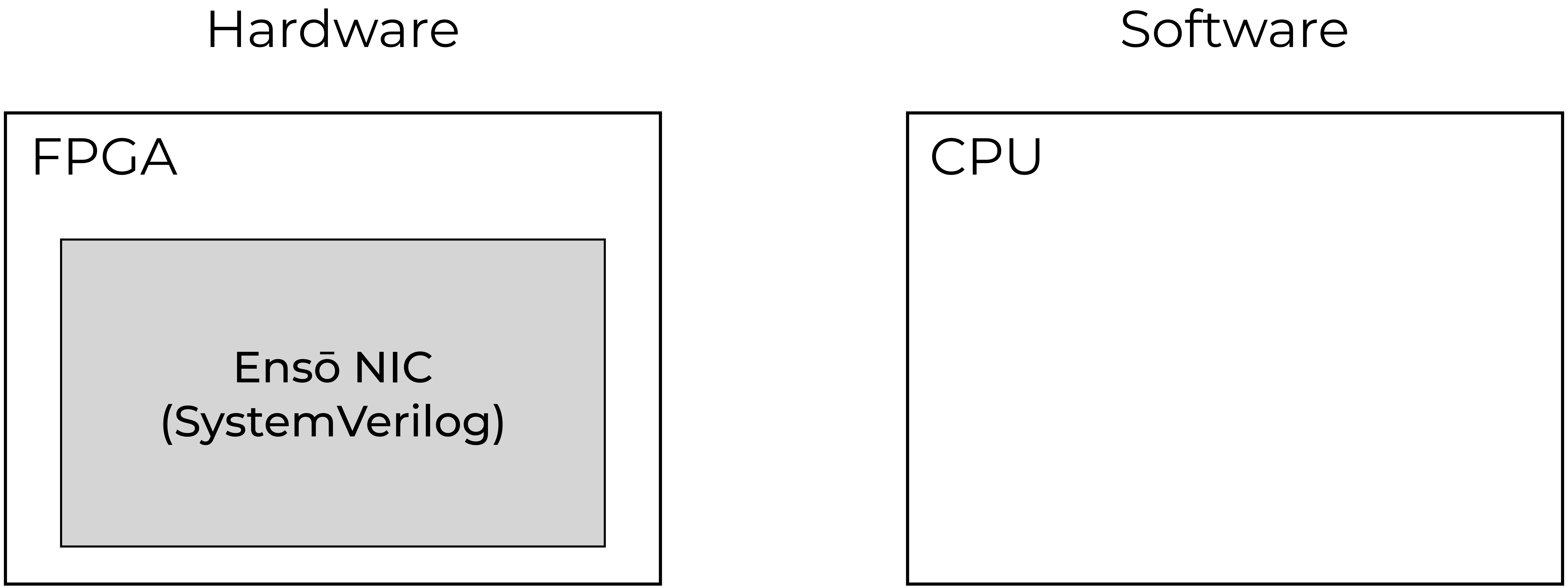
How to avoid copies in applications that send data back (e.g., Network Functions)?

**Refer to the paper for details**

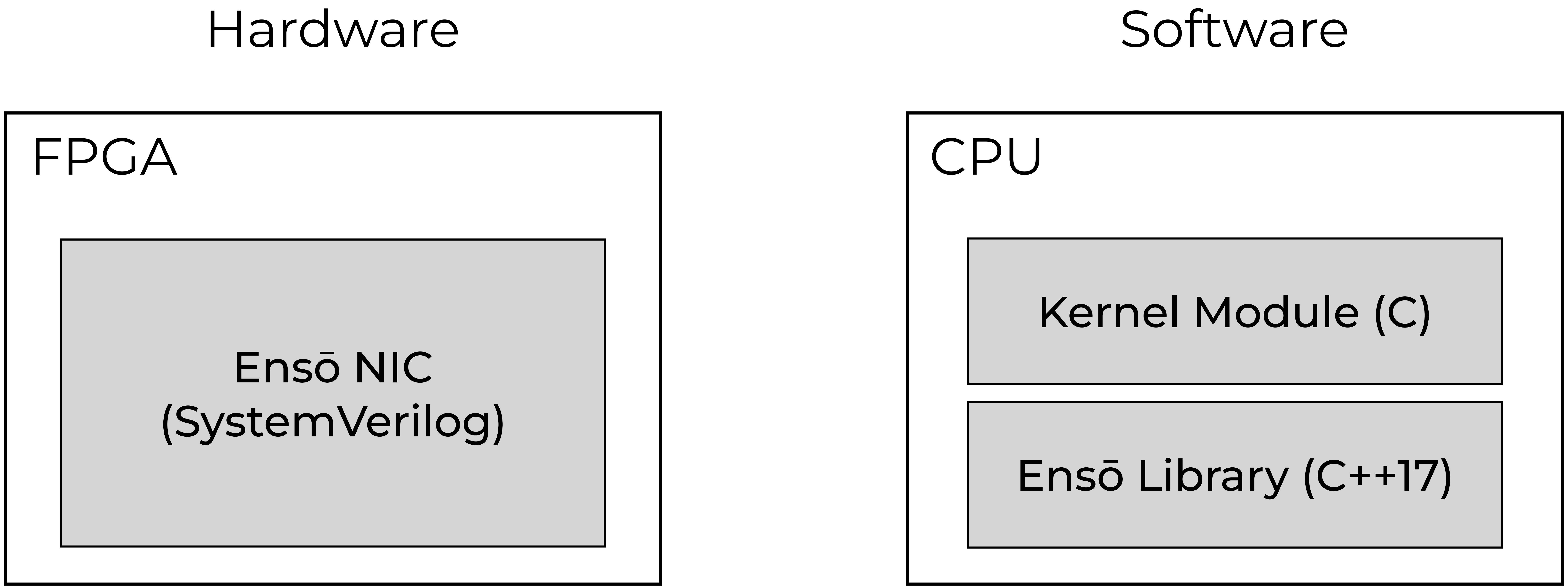
# Ensō Implementation



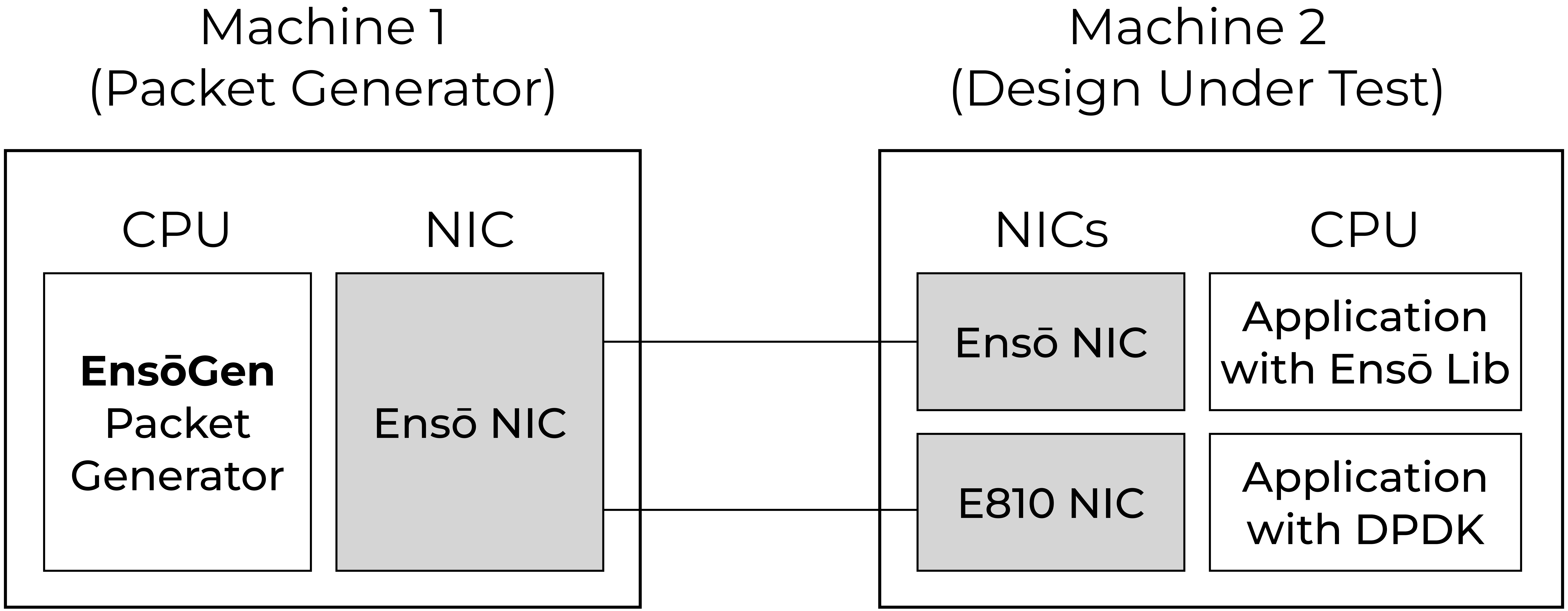
# Ensō Implementation



# Ensō Implementation

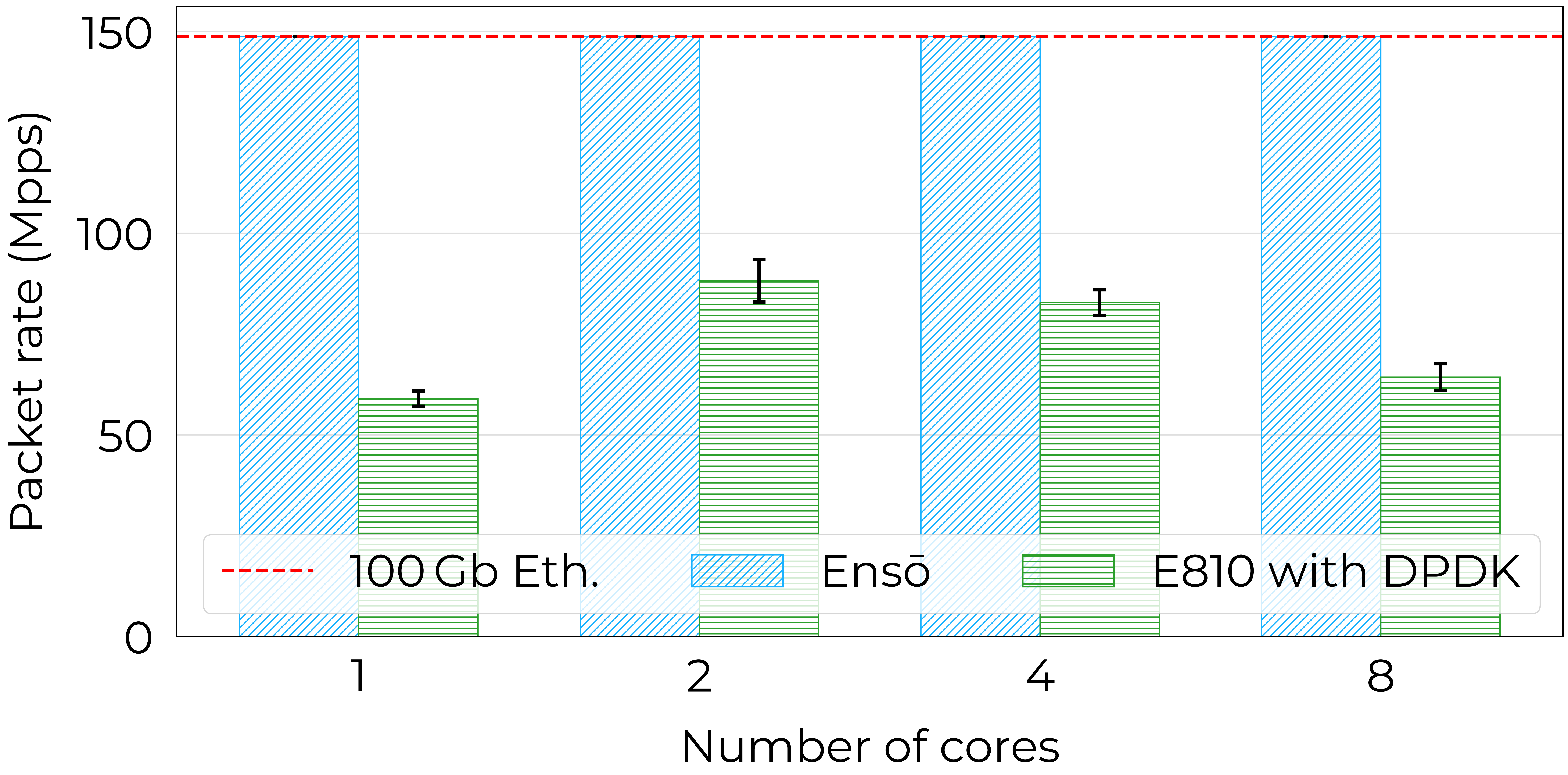


# Evaluation

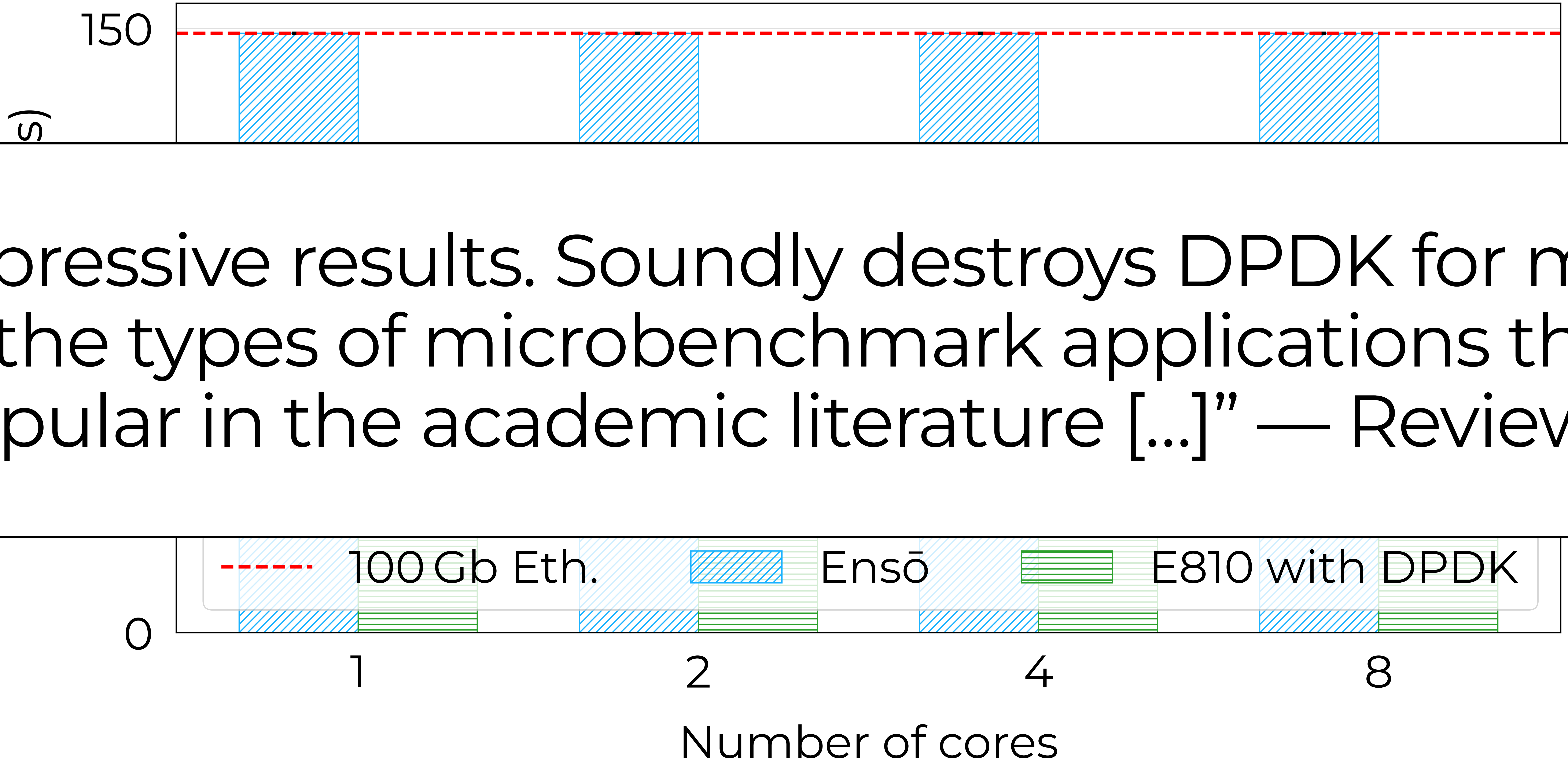


Machine 1
(Packet Generator)
CPU
NIC
**EnsōGen**
Packet
Generator
Ensō NIC
Machine 2
(Design Under Test)
NICs
CPU
Ensō NIC
Application
with Ensō Lib
E810 NIC
Application
with DPDK

# Ensō achieves 100 Gbps line rate (148.8 Mpps) using a single core



# Ensō achieves 100 Gbps line rate (148.8 Mpps) using a single core

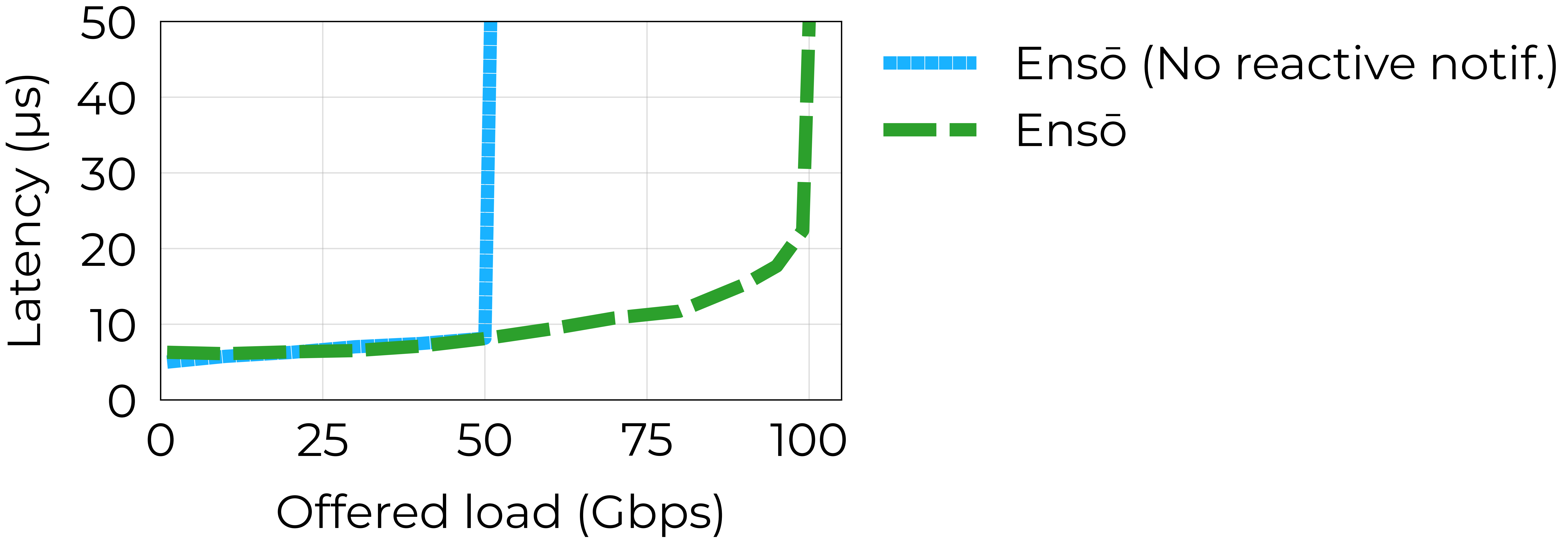



"Impressive results. Soundly destroys DPDK for many of the types of microbenchmark applications that are popular in the academic literature [...]" — Reviewer D

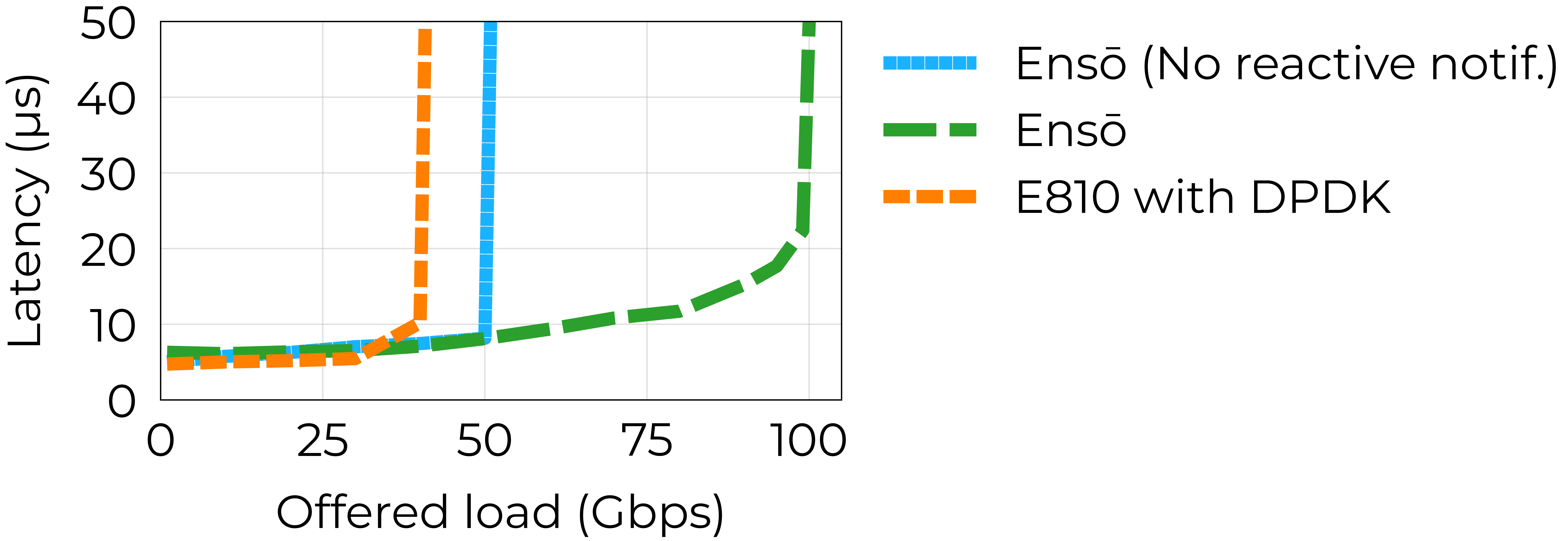
# Ensō improves application throughput by up to 6x

| Application | Throughput Improvement |
|---|---|
| Maglev Load Balancer [NSDI ’16] | Up to 6x |
| Network Telemetry with NitroSketch [SIGCOMM ’19] | Up to 3.5x |
| MICA Key-Value Store [NSDI ’14] | Up to 47% |
| Log Monitor | Up to 95% |

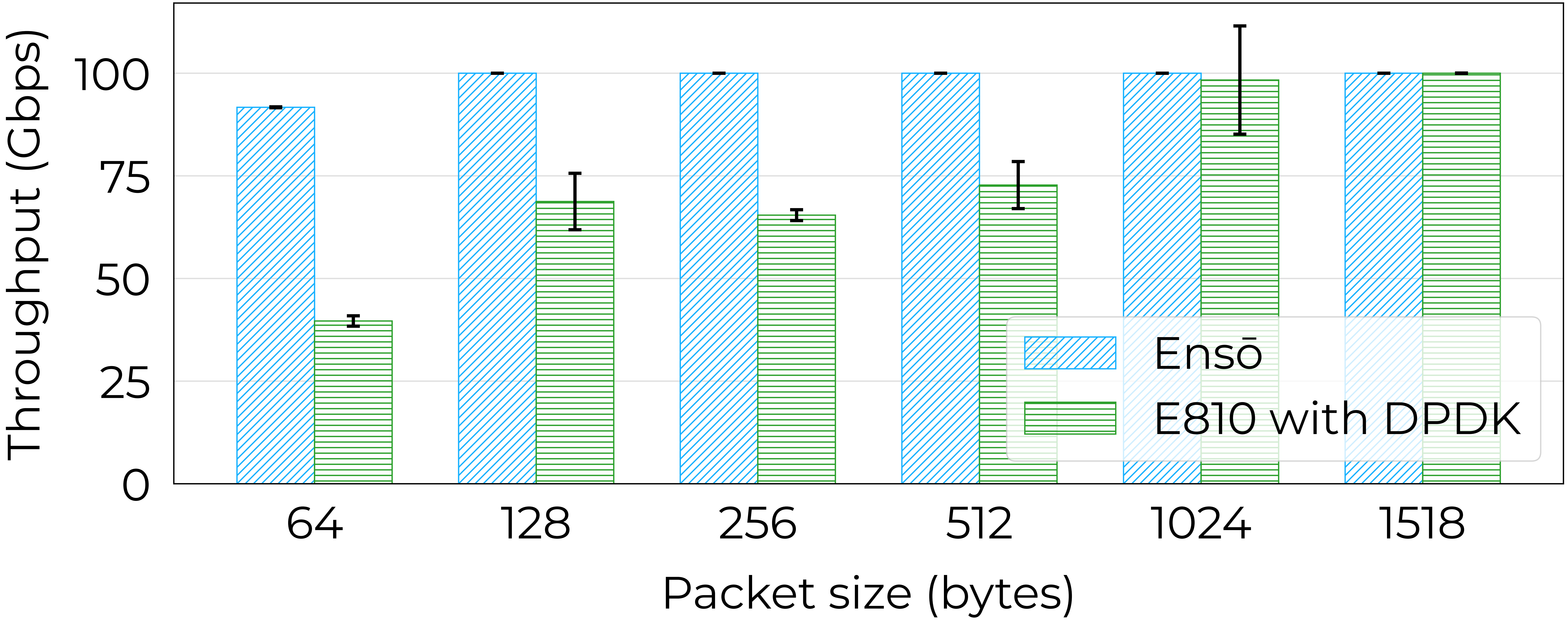
# Reactive Notifications + Notification Prefetching improve throughput without impairing latency



# Ensō achieves similar latency to the E810 NIC with DPDK, while sustaining a much greater load



# Ensō outperforms the packetized interface even when copying data



# Conclusion

**Ensō** is a **streaming interface** for NIC-Application communication

# Conclusion

**Ensō** is a **streaming interface** for NIC-Application communication

Improves application throughput by **up to 6x** even with no offloads

# Conclusion

**Ensō** is a **streaming interface** for NIC-Application communication

Improves application throughput by **up to 6x** even with no offloads

Allows easier and more efficient **high-level offload** implementations

# Conclusion

**Ensō** is a **streaming interface** for NIC-Application communication

Improves application throughput by **up to 6x** even with no offloads

Allows easier and more efficient **high-level offload** implementations

Ensō is open source: enso.cs.cmu.edu

Contact: sadok@cmu.edu