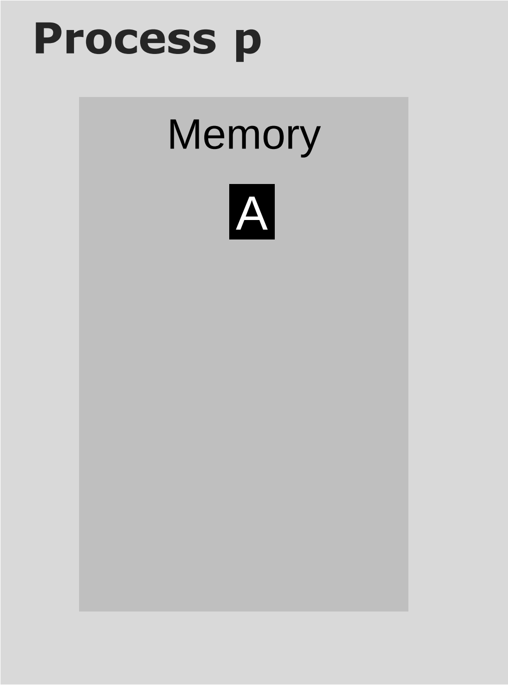
Active Access: A Mechanism for High-Performance Distributed Data-Centric Computations

Maciej Besta, Torsten Hoefler

REMOTE MEMORY ACCESS (RMA) PROGRAMMING

Remote Memory Access (RMA) Programming


Process p

Memory

A

Remote Memory Access (RMA) Programming

Process p
Memory
A
Process q
Memory
B

Remote Memory Access (RMA) Programming

Process p

Memory

A

Process q

Memory

B

Cray
BlueWaters

Remote Memory Access (RMA) Programming

Process p

Memory

A

Process q

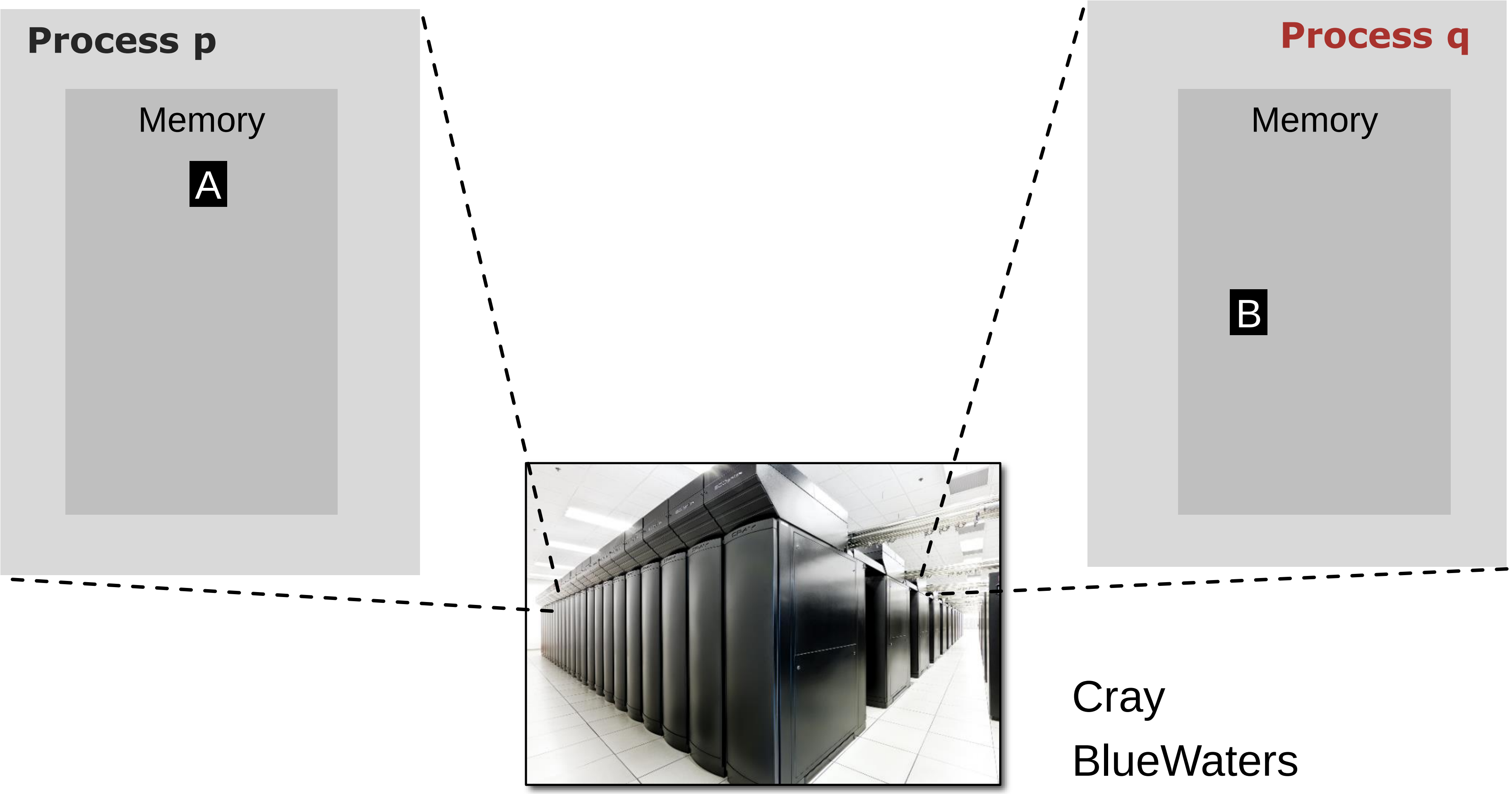
Memory

B

Cray

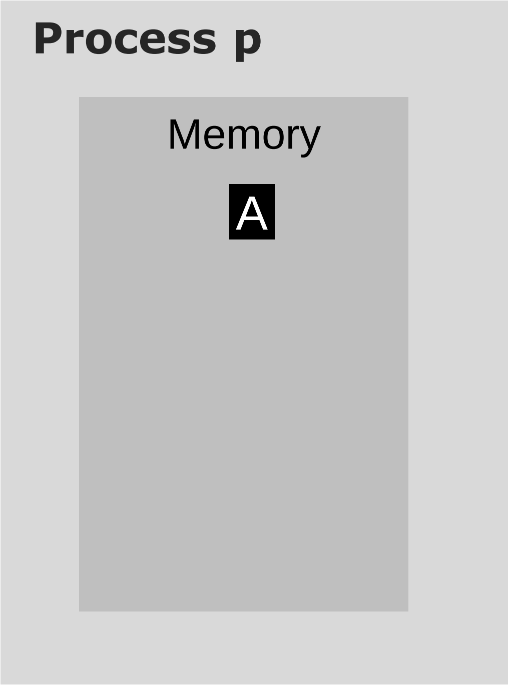
BlueWaters

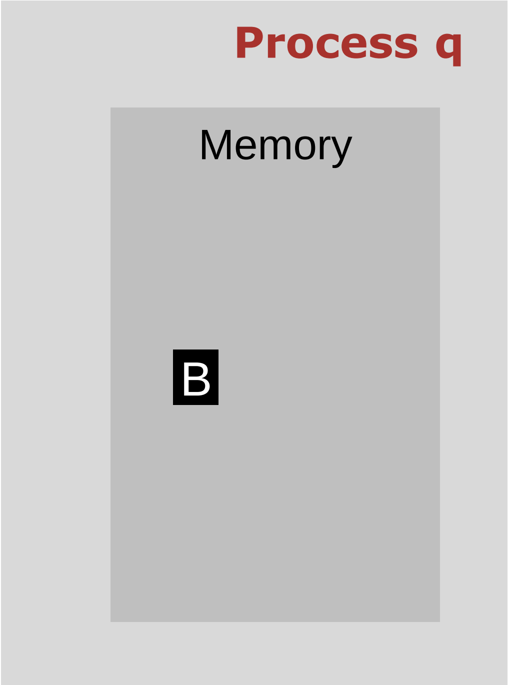
Remote Memory Access (RMA) Programming



Cray
BlueWaters

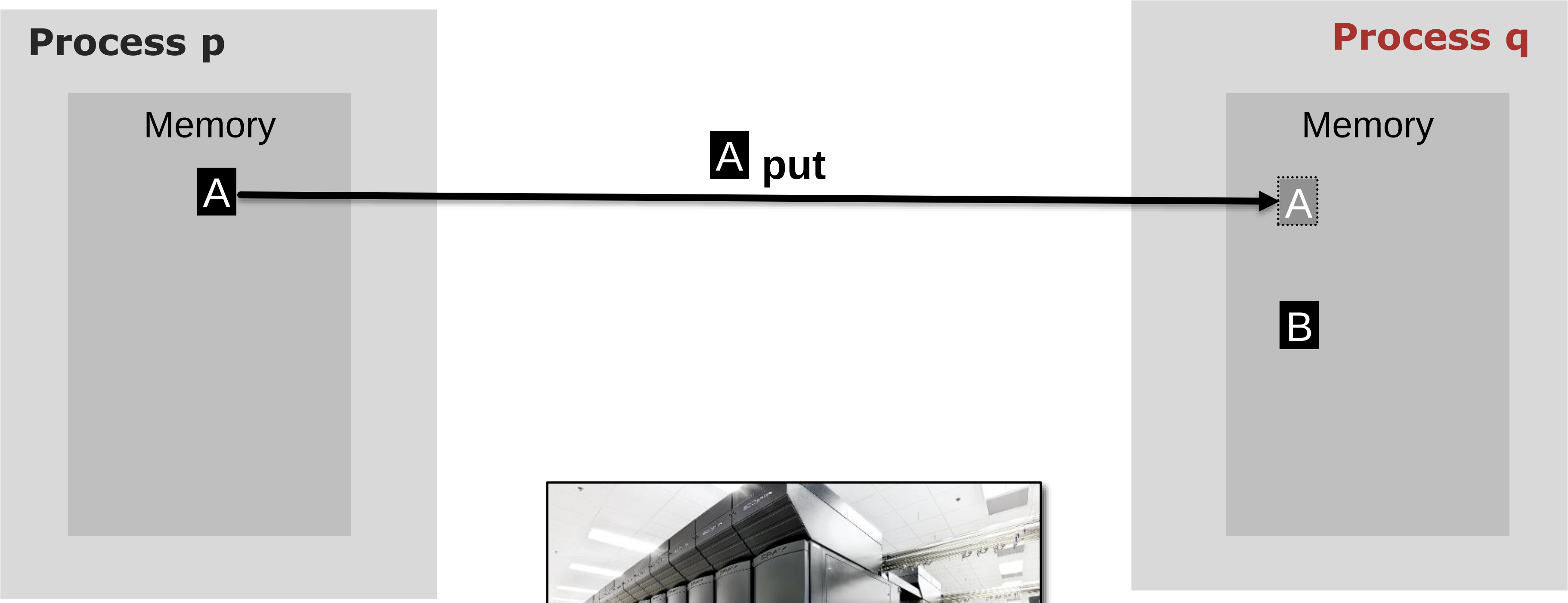
REMOTE MEMORY ACCESS (RMA) PROGRAMMING





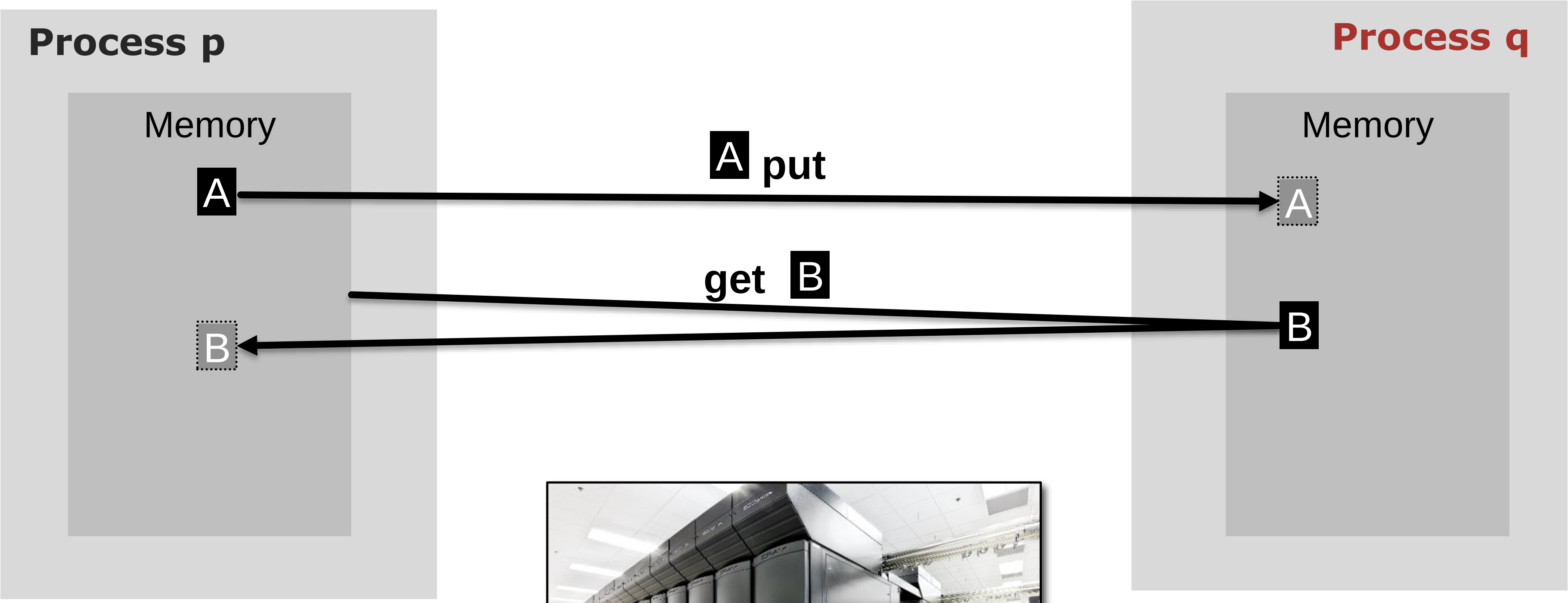
Cray
BlueWaters

Remote Memory Access (RMA) Programming



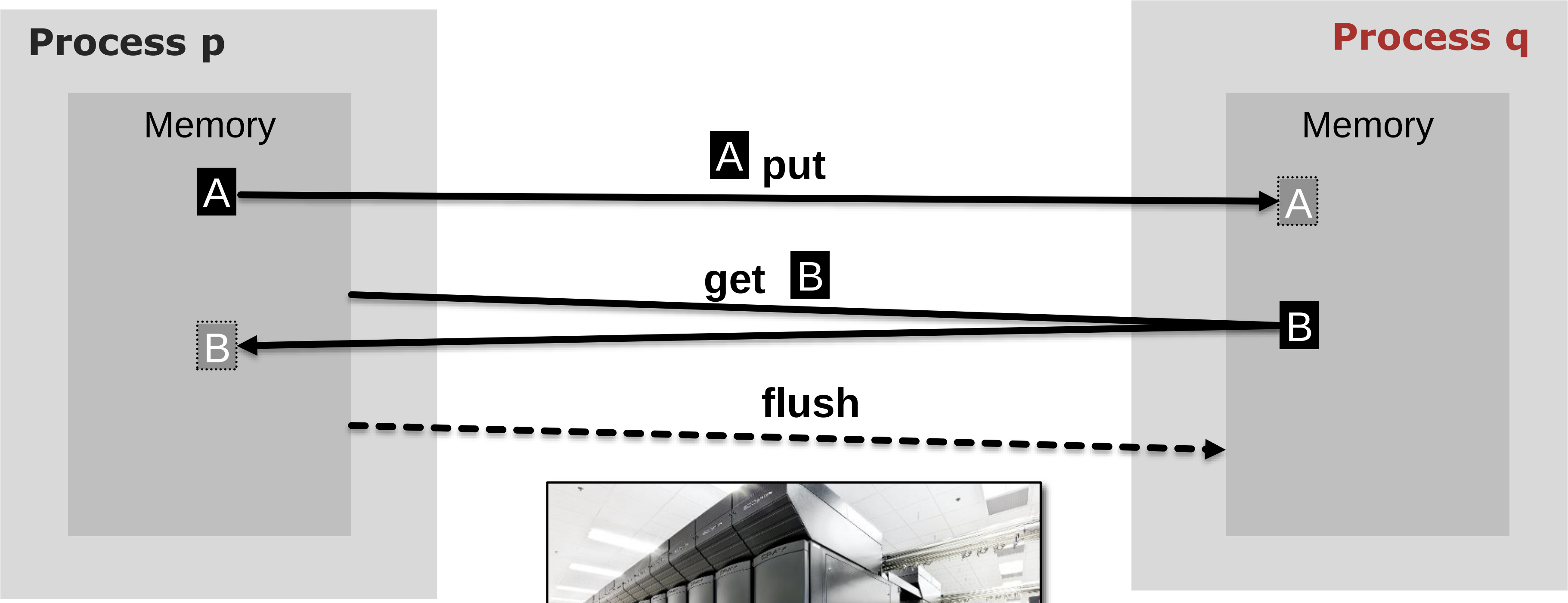
Cray
BlueWaters

Remote Memory Access (RMA) Programming



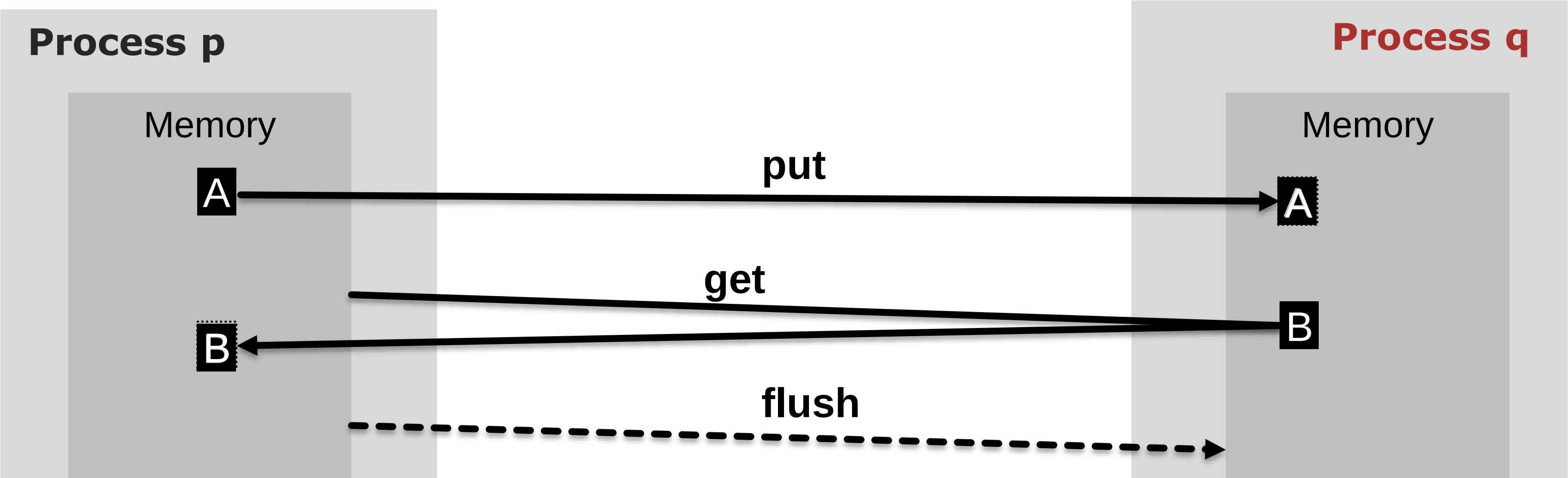
Cray
BlueWaters

Remote Memory Access (RMA) Programming



Cray BlueWaters

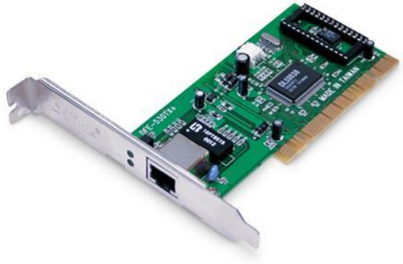
Remote Memory Access (RMA) Programming



Cray
BlueWaters

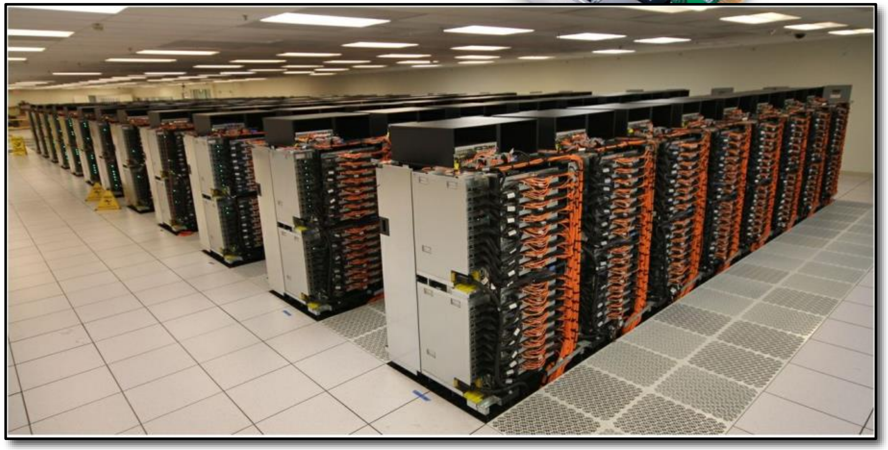
Remote Memory Access Programming

- Implemented in hardware in NICs in the majority of HPC networks (RDMA)



Remote Memory Access Programming

- Implemented in hardware in NICs in the majority of HPC networks (RDMA)

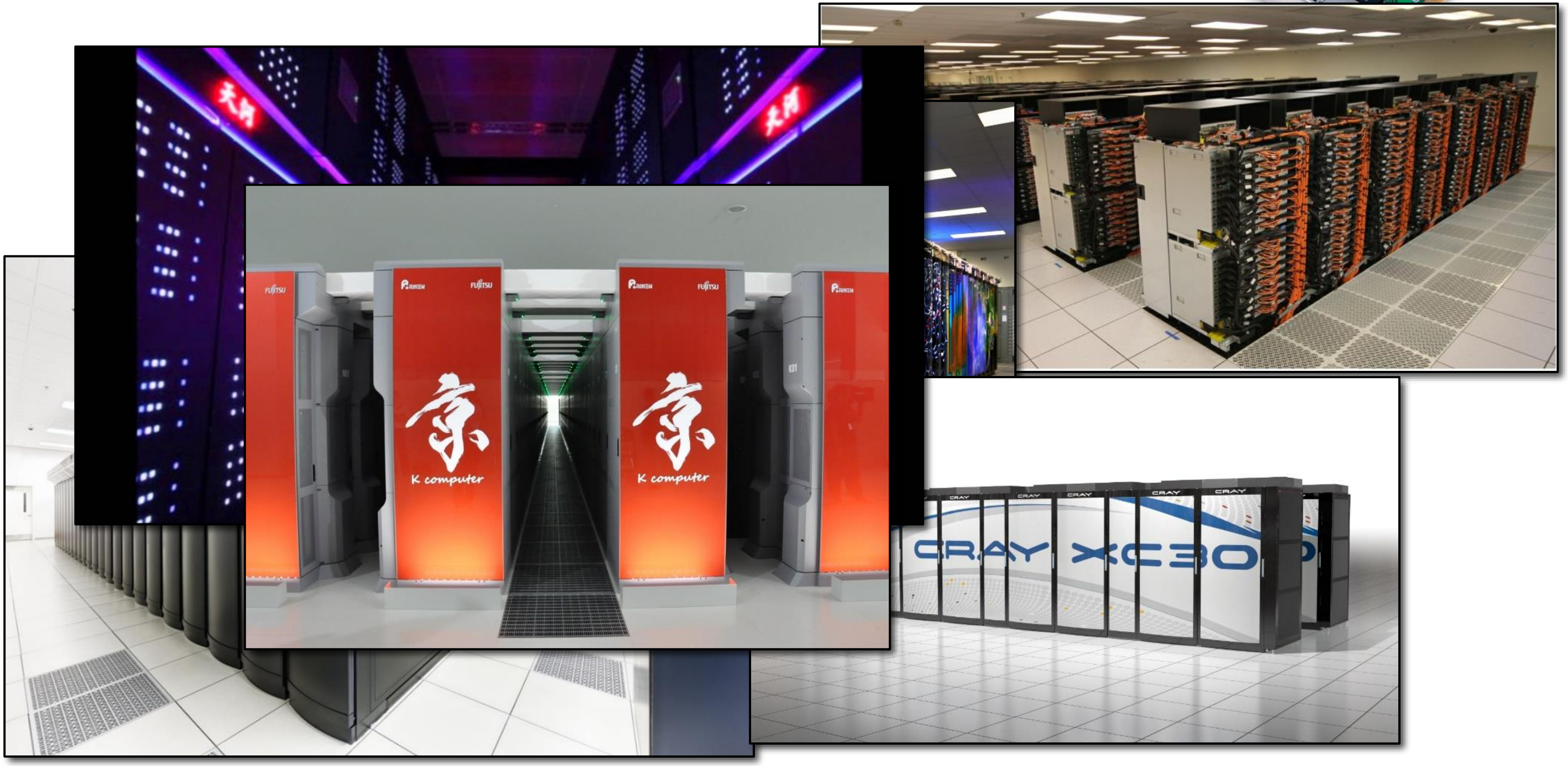


Remote Memory Access Programming

- Implemented in hardware in NICs in the majority of HPC networks (RDMA)

Remote Memory Access Programming

- Implemented in hardware in NICs in the majority of HPC networks (RDMA)



Remote Memory Access Programming

- Implemented in hardware in NICs in the majority of HPC networks (RDMA)

Remote Memory Access Programming

- Supported by many HPC libraries and languages

Remote Memory Access Programming

- Supported by many HPC libraries and languages

Remote Memory Access Programming

- Supported by many HPC libraries and languages

REMOTE MEMORY ACCESS PROGRAMMING

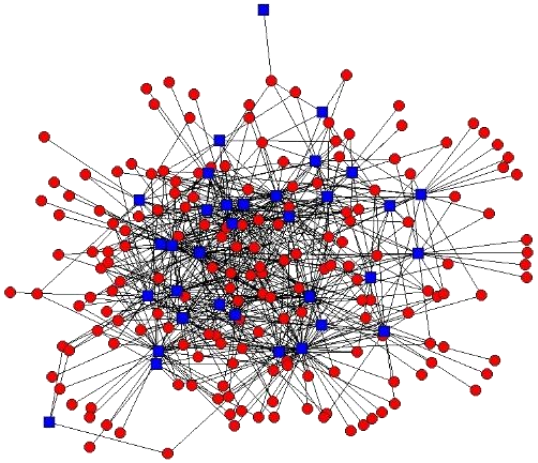
- Enables significant speedups over message passing in many types of applications, e.g.:

[1] R. Gerstenberger et al. Enabling Highly-Scalable Remote Memory Access Programming with MPI-3 One-Sided. SC13

[2] D. Petrovic et al., High-performance RMA-based broadcast on the Intel SCC. SPAA'12

Remote Memory Access Programming

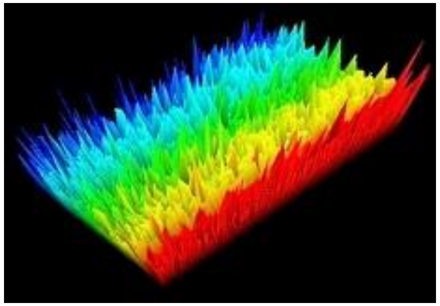
- Enables significant speedups over message passing in many types of applications, e.g.:
  - Speedup of ~1.5 for communication patterns in irregular workloads

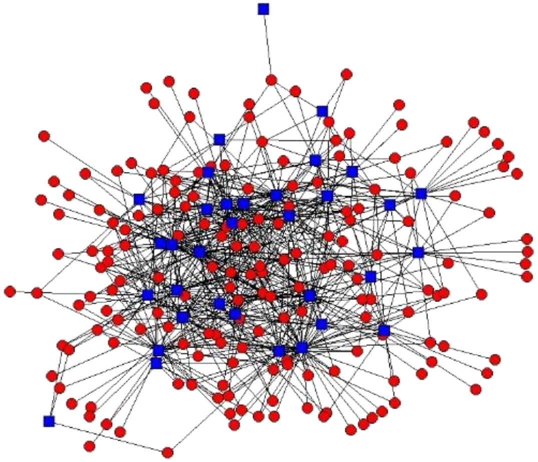


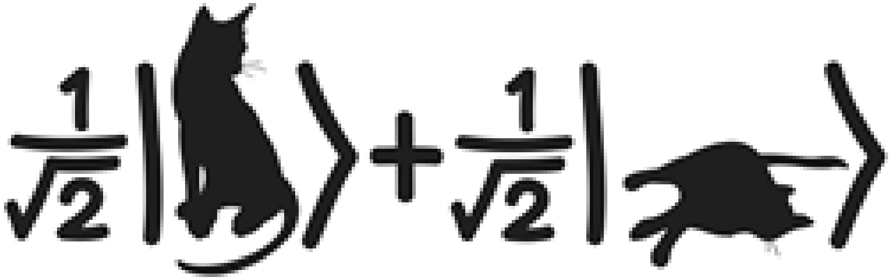
[1] R. Gerstenberger et al. Enabling Highly-Scalable Remote Memory Access Programming with MPI-3 One-Sided. SC13
[2] D. Petrovic et al., High-performance RMA-based broadcast on the Intel SCC. SPAA’12

REMOTE MEMORY ACCESS PROGRAMMING

- Enables significant speedups over message passing in many types of applications, e.g.:
  - Speedup of ~1.5 for communication patterns in irregular workloads
  - Speedup of ~1.4-2 in physics computations





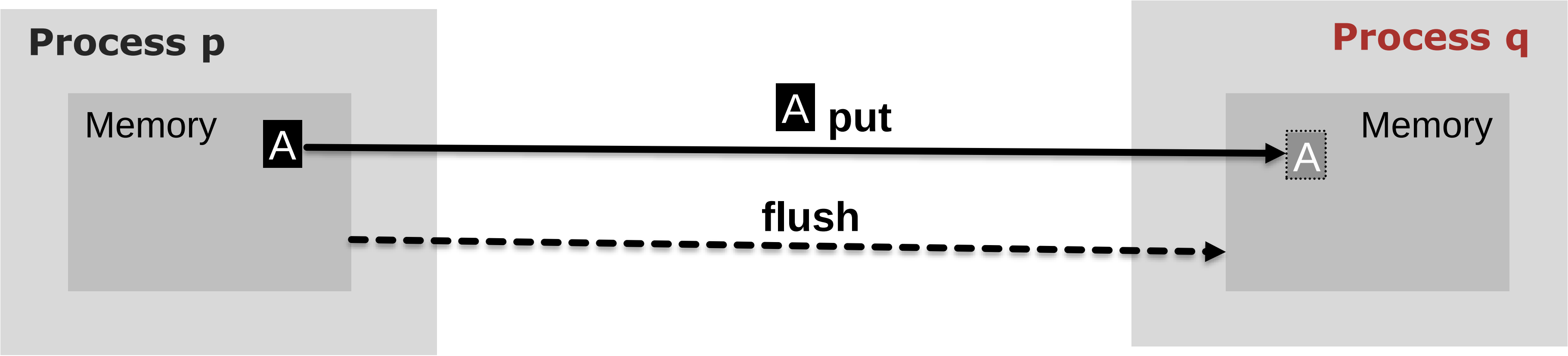

$\frac{1}{\sqrt{2}}|\ \rangle + \frac{1}{\sqrt{2}}|\ \rangle$

[1] R. Gerstenberger et al. Enabling Highly-Scalable Remote Memory Access Programming with MPI-3 One-Sided. SC13

[2] D. Petrovic et al., High-performance RMA-based broadcast on the Intel SCC. SPAA'12

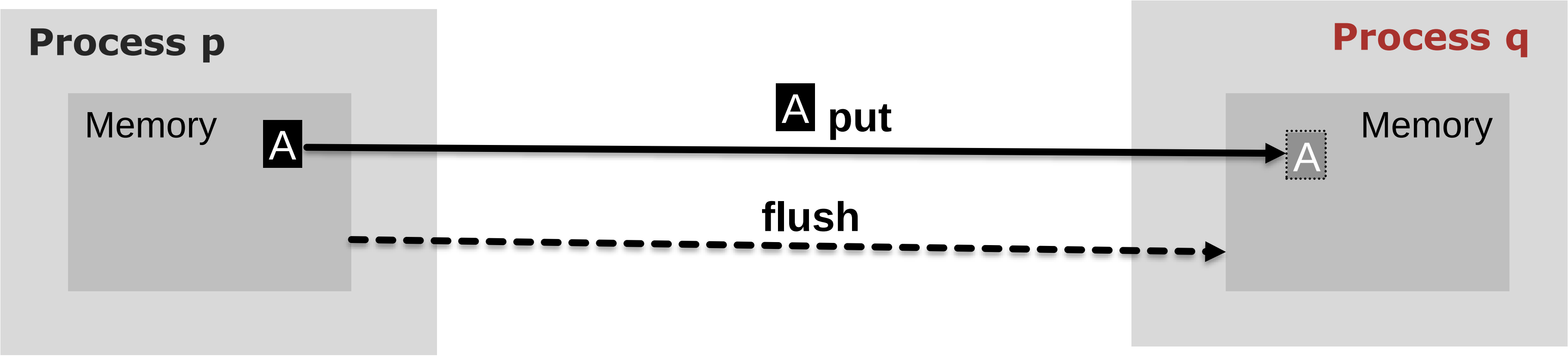
RMA vs. Message Passing

RMA:



RMA VS. MESSAGE PASSING

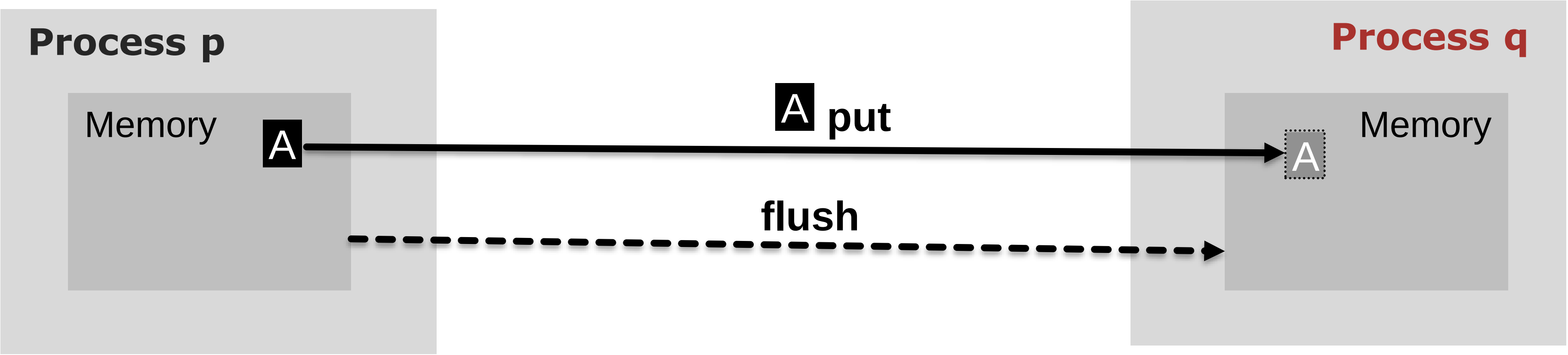
RMA:



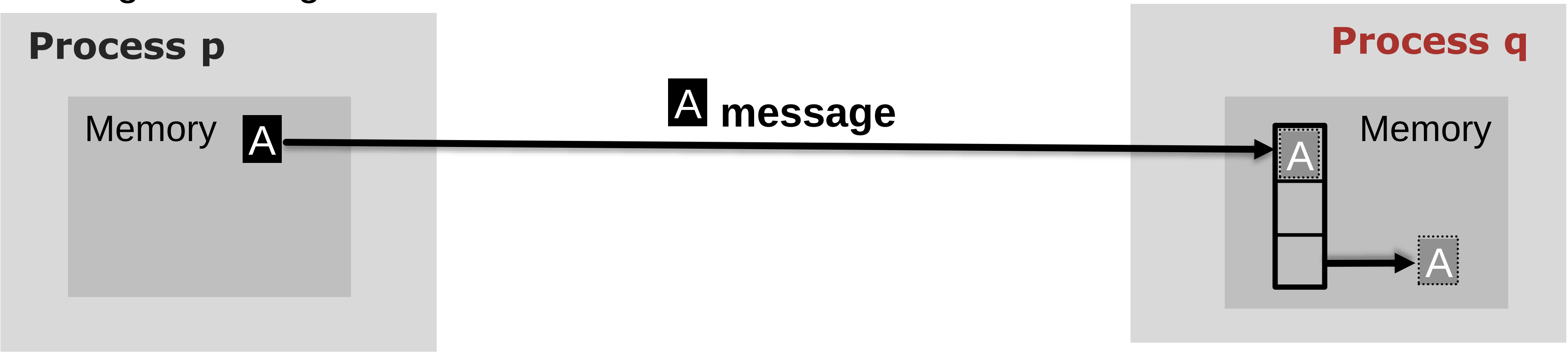
Message Passing:

RMA vs. Message Passing

RMA:



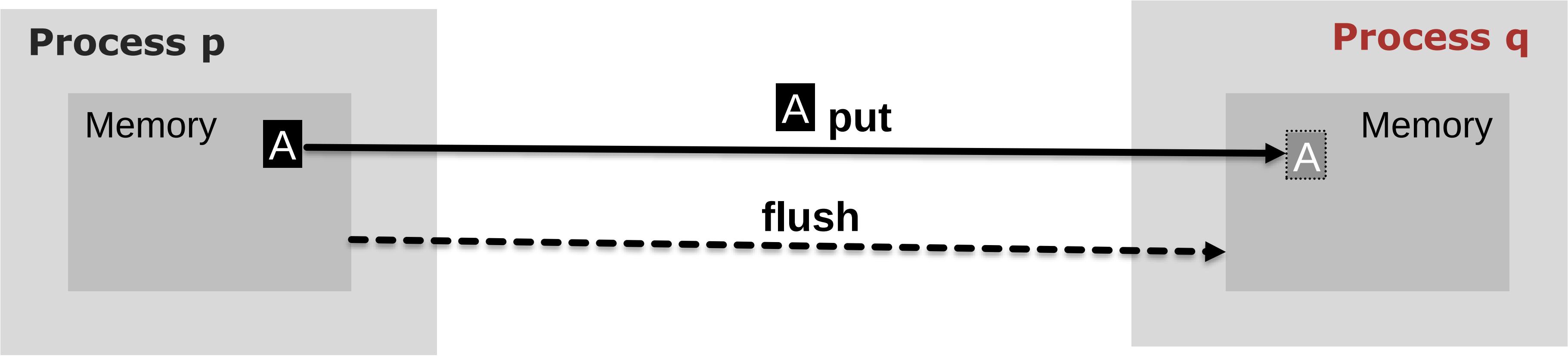
Message Passing:



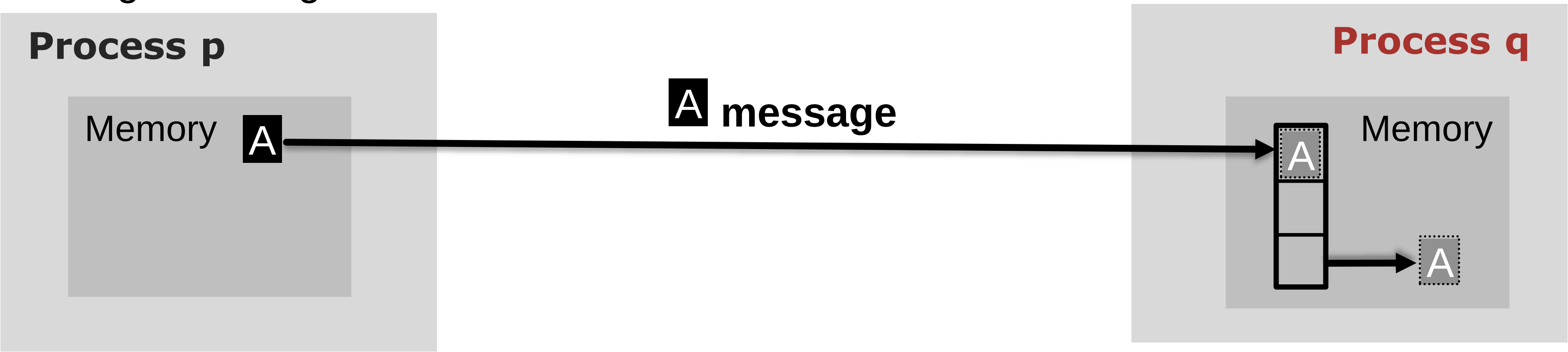
RMA vs. Message Passing

- Communication in RMA is one-sided

RMA:



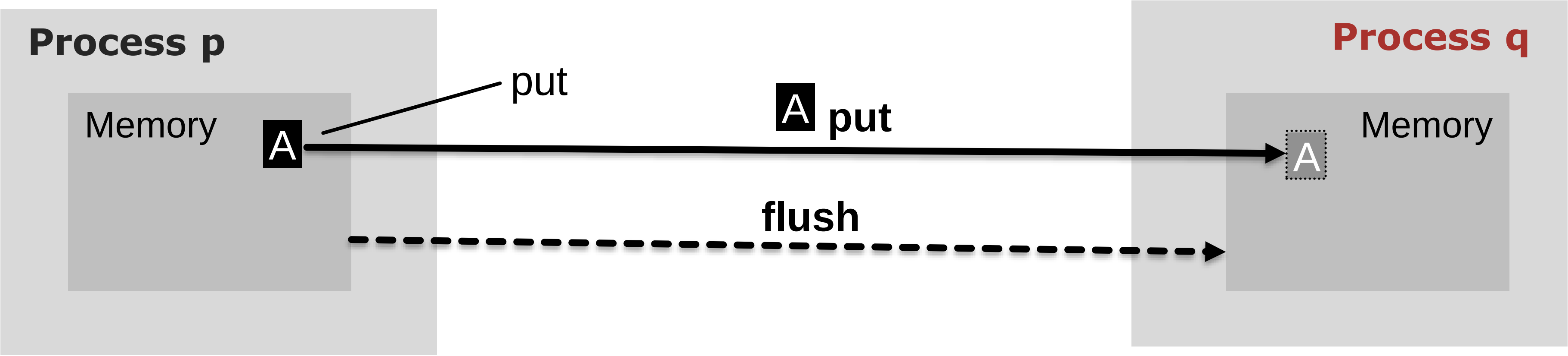
Message Passing:



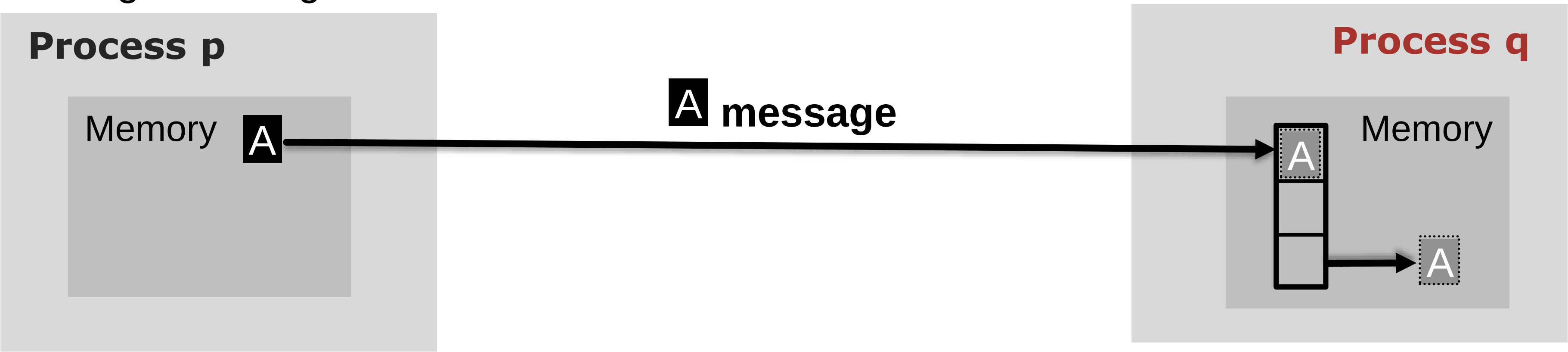
RMA VS. MESSAGE PASSING

- Communication in RMA is one-sided

RMA:



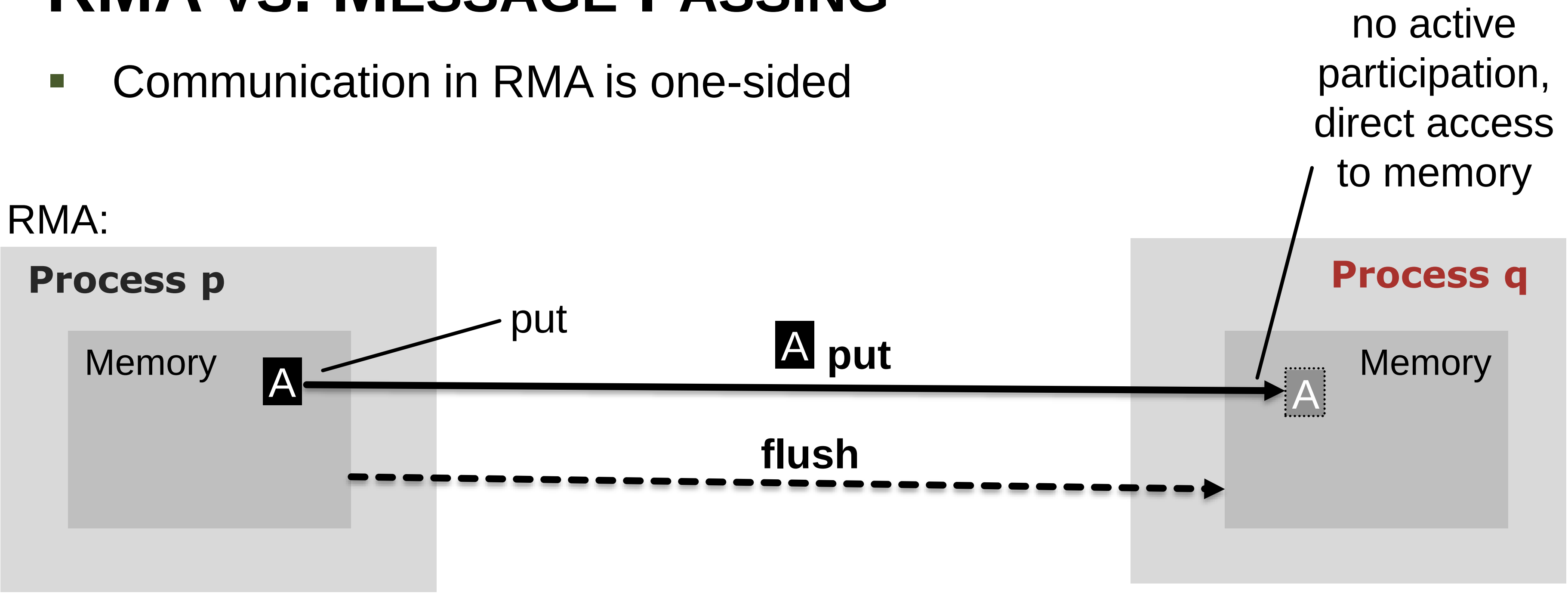
Message Passing:



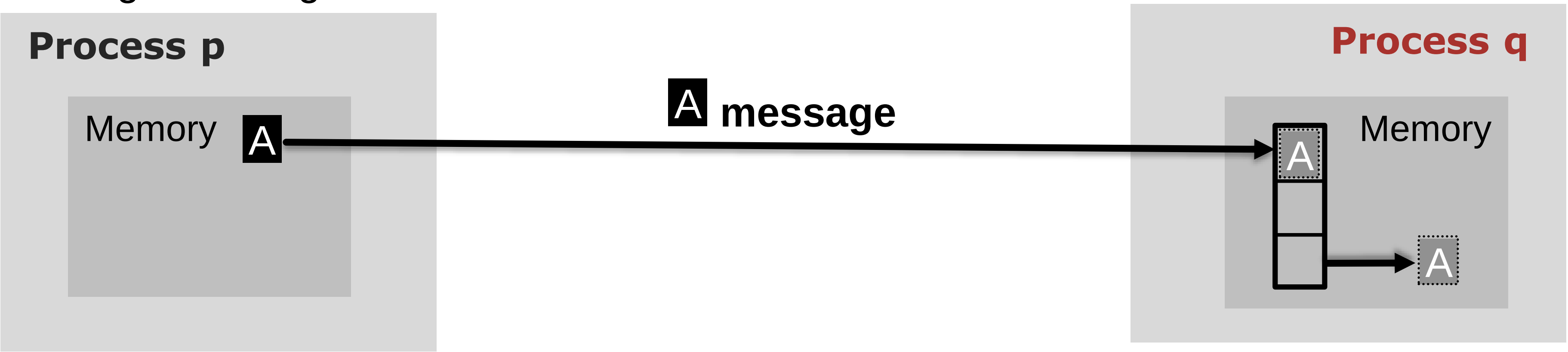
RMA VS. MESSAGE PASSING

- Communication in RMA is one-sided

RMA:

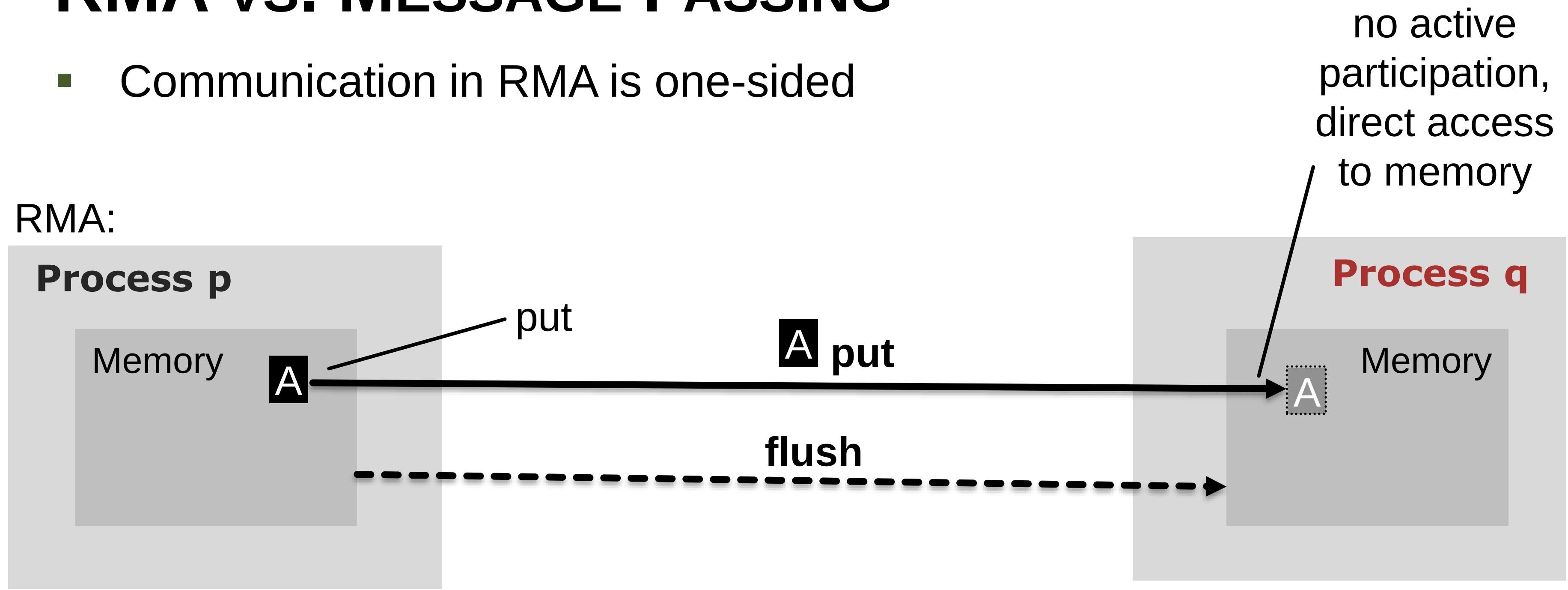


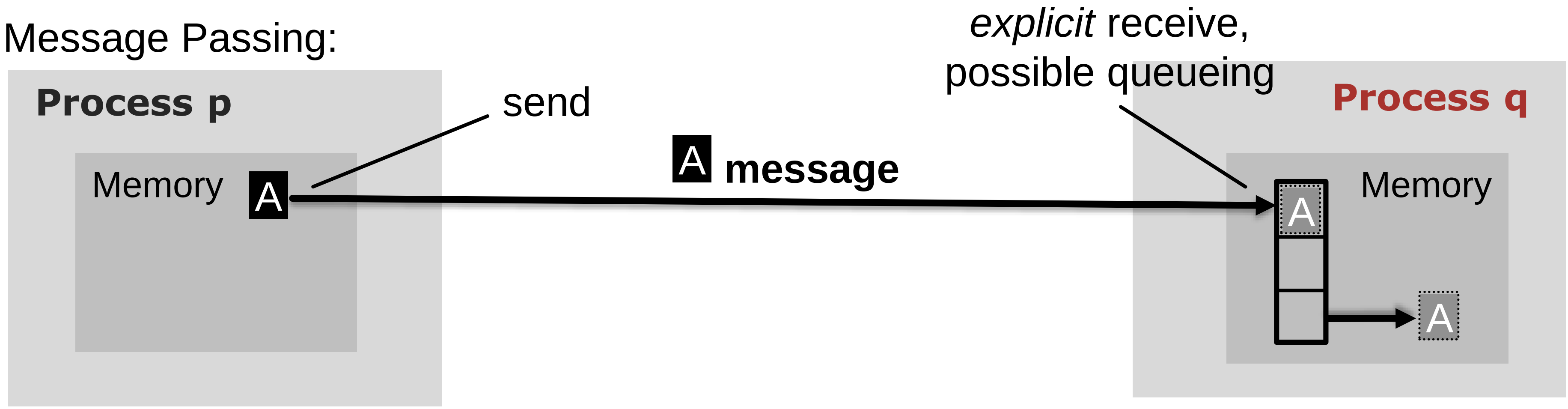
Message Passing:



RMA vs. Message Passing

- Communication in RMA is one-sided





REMOTE MEMORY ACCESS PROGRAMMING

[1] R. Gerstenberger et al. Enabling Highly-Scalable Remote Memory Access Programming with MPI-3 One-Sided. SC13

Remote Memory Access Programming

- Is it ideal?

[1] R. Gerstenberger et al. Enabling Highly-Scalable Remote Memory Access Programming with MPI-3 One-Sided. SC13

Remote Memory Access Programming

- Is it ideal? NO!

[1] R. Gerstenberger et al. Enabling Highly-Scalable Remote Memory Access Programming with MPI-3 One-Sided. SC13

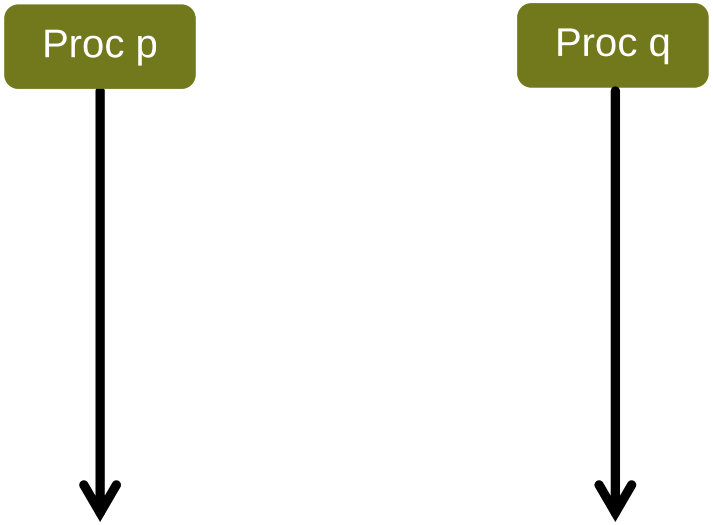
Remote Memory Access Programming

- Is it ideal? NO!
- Consider an insert in a distributed hashtable...

[1] R. Gerstenberger et al. Enabling Highly-Scalable Remote Memory Access Programming with MPI-3 One-Sided. SC13

Remote Memory Access Programming

- Is it ideal? NO!
- Consider an insert in a distributed hashtable...

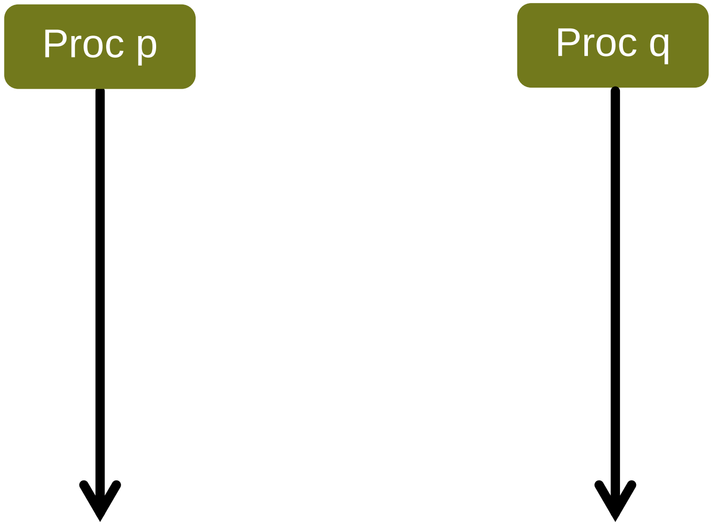


[1] R. Gerstenberger et al. Enabling Highly-Scalable Remote Memory Access Programming with MPI-3 One-Sided. SC13

Remote Memory Access Programming

- Is it ideal? NO!
- Consider an insert in a distributed hashtable...

No hash collision:


Proc p
Proc q

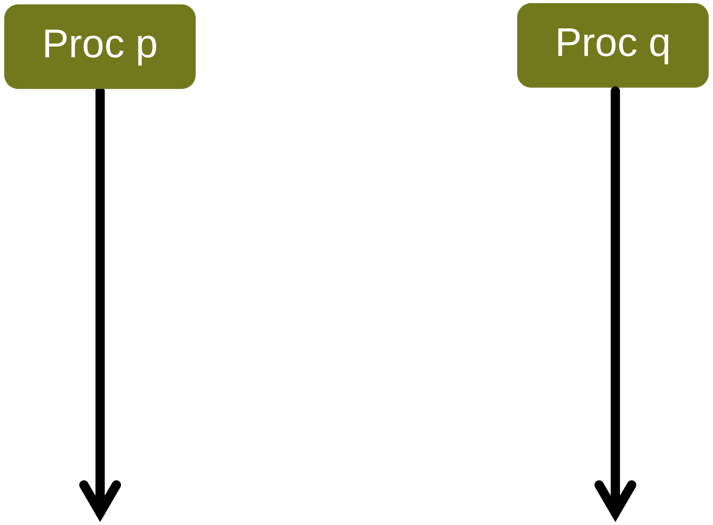
[1] R. Gerstenberger et al. Enabling Highly-Scalable Remote Memory Access Programming with MPI-3 One-Sided. SC13

Remote Memory Access Programming

- Is it ideal? NO!
- Consider an insert in a distributed hashtable...

No hash collision:

➔ 1 remote atomic



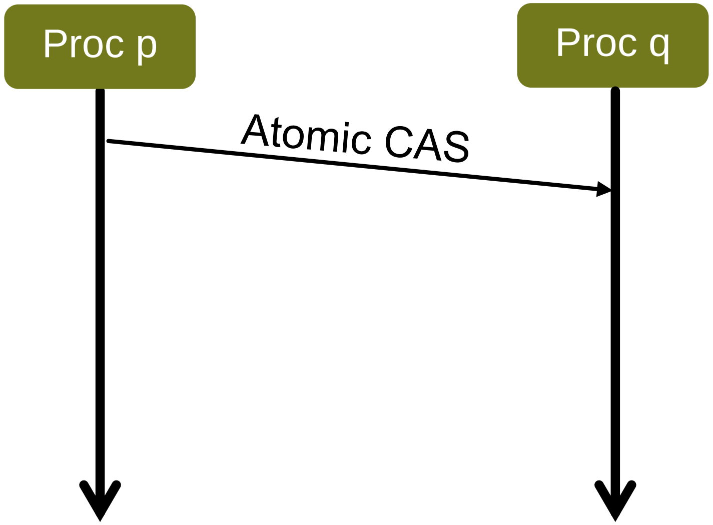
[1] R. Gerstenberger et al. Enabling Highly-Scalable Remote Memory Access Programming with MPI-3 One-Sided. SC13

Remote Memory Access Programming

- Is it ideal? NO!
- Consider an insert in a distributed hashtable...

No hash collision:

➔ 1 remote atomic



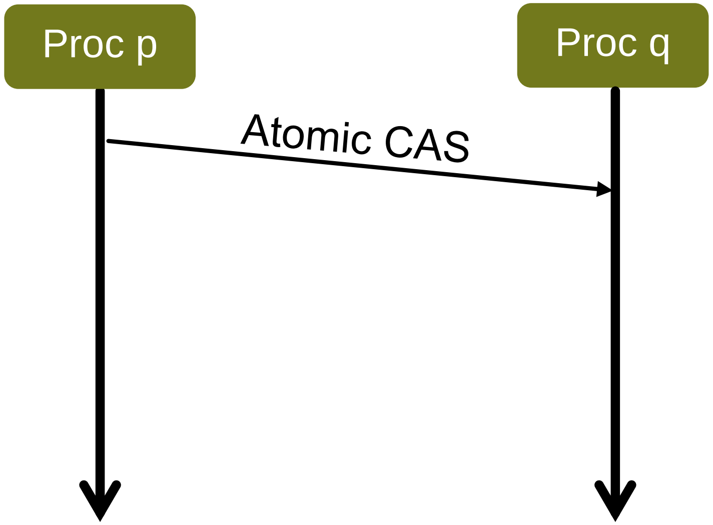
[1] R. Gerstenberger et al. Enabling Highly-Scalable Remote Memory Access Programming with MPI-3 One-Sided. SC13

REMOTE MEMORY ACCESS PROGRAMMING

- Is it ideal? NO!
- Consider an insert in a distributed hashtable...

No hash collision:

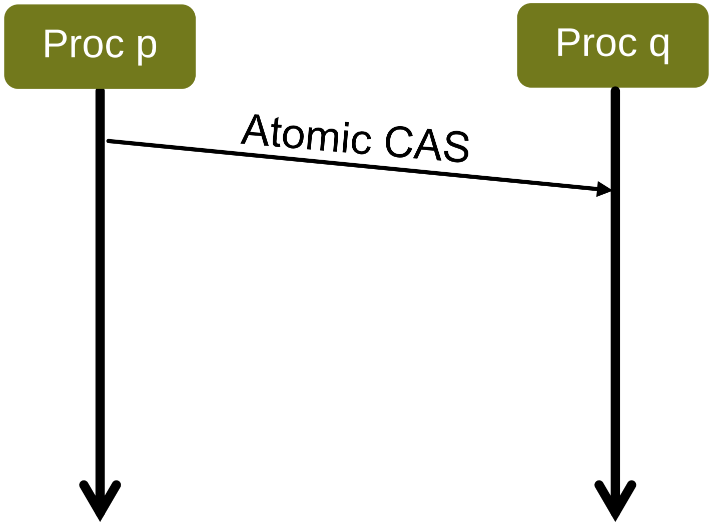
➔ 1 remote atomic
➔ Up to 5x speedup over MP [1]



[1] R. Gerstenberger et al. Enabling Highly-Scalable Remote Memory Access Programming with MPI-3 One-Sided. SC13

Remote Memory Access Programming

- Is it ideal? NO!
- Consider an insert in a distributed hashtable...



No hash collision:

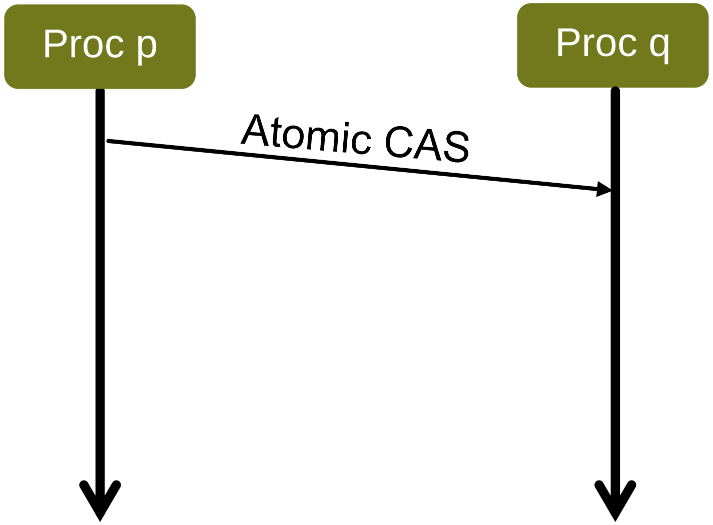
➔ 1 remote atomic
➔ Up to 5x speedup over MP [1]

A hash collision:

[1] R. Gerstenberger et al. Enabling Highly-Scalable Remote Memory Access Programming with MPI-3 One-Sided. SC13

Remote Memory Access Programming

- Is it ideal? NO!
- Consider an insert in a distributed hashtable...



No hash collision:

➔ 1 remote atomic
➔ Up to 5x speedup over MP [1]

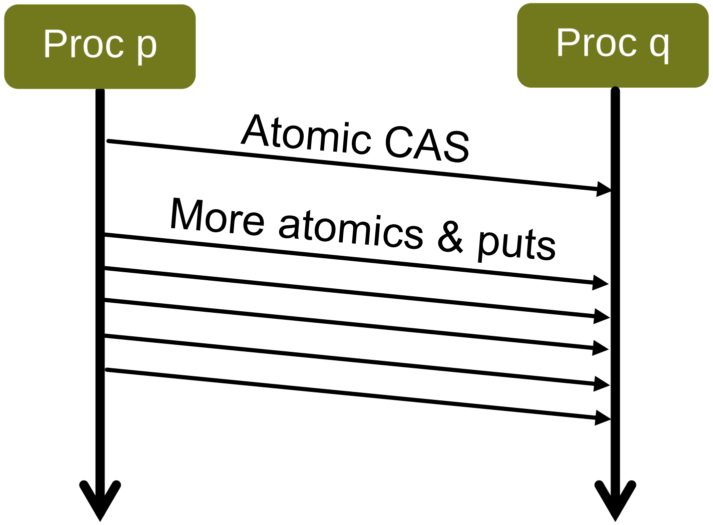
A hash collision:

➔ 4 remote atomics + 2 remote puts

[1] R. Gerstenberger et al. Enabling Highly-Scalable Remote Memory Access Programming with MPI-3 One-Sided. SC13

REMOTE MEMORY ACCESS PROGRAMMING

- Is it ideal? NO!
- Consider an insert in a distributed hashtable...



No hash collision:

➔ 1 remote atomic
➔ Up to 5x speedup over MP [1]

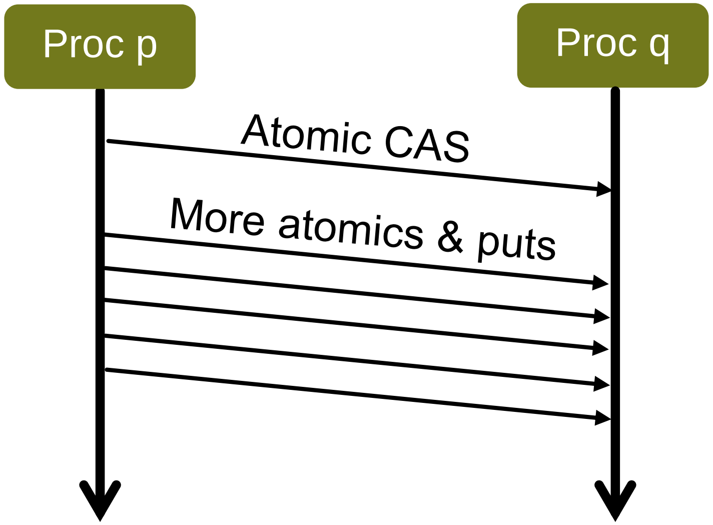
A hash collision:

➔ 4 remote atomics + 2 remote puts

[1] R. Gerstenberger et al. Enabling Highly-Scalable Remote Memory Access Programming with MPI-3 One-Sided. SC13

Remote Memory Access Programming

- Is it ideal? NO!
- Consider an insert in a distributed hashtable...

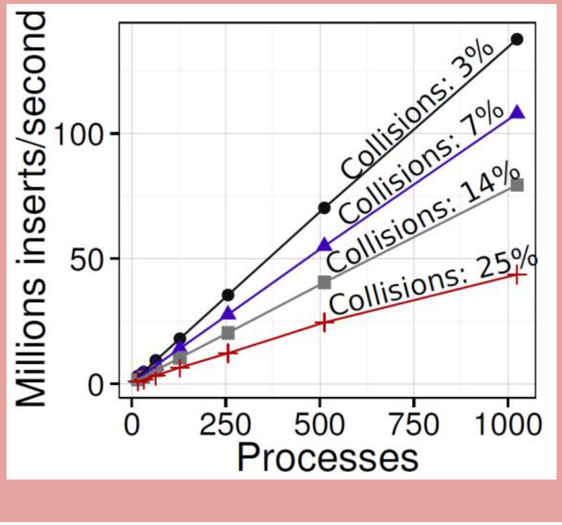


No hash collision:

➔ 1 remote atomic
➔ Up to 5x speedup over MP [1]

A hash collision:

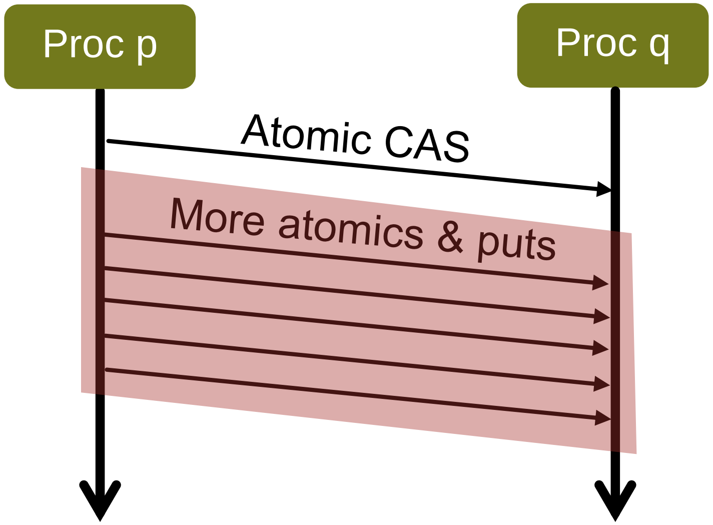
➔ 4 remote atomics + 2 remote puts
➔ Significant performance drops



[1] R. Gerstenberger et al. Enabling Highly-Scalable Remote Memory Access Programming with MPI-3 One-Sided. SC13

Remote Memory Access Programming

- Is it ideal? NO!
- Consider an insert in a distributed hashtable...

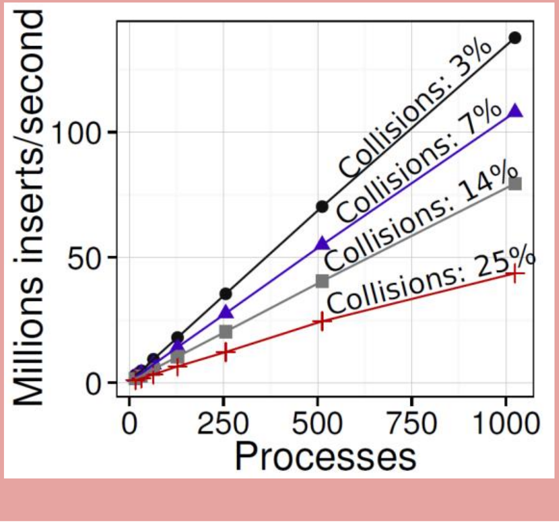


No hash collision:

➔ 1 remote atomic
➔ Up to 5x speedup over MP [1]

A hash collision:

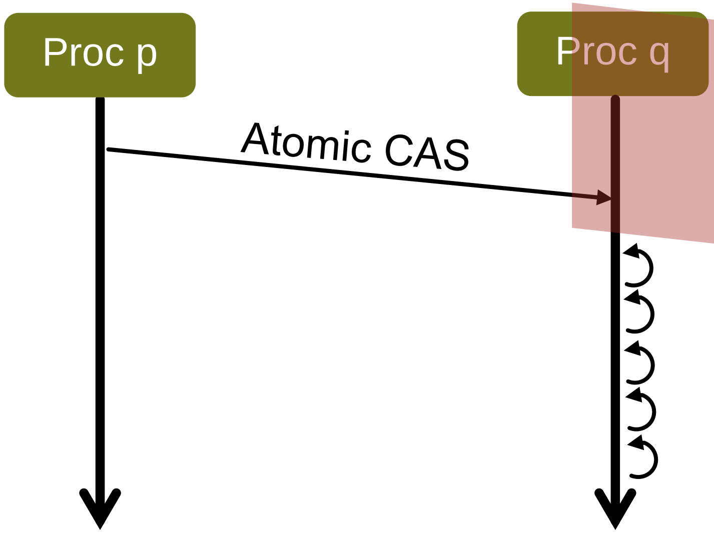
➔ 4 remote atomics + 2 remote puts
➔ Significant performance drops



[1] R. Gerstenberger et al. Enabling Highly-Scalable Remote Memory Access Programming with MPI-3 One-Sided. SC13

Remote Memory Access Programming

- Is it ideal? NO!
- Consider an insert in a distributed hashtable...

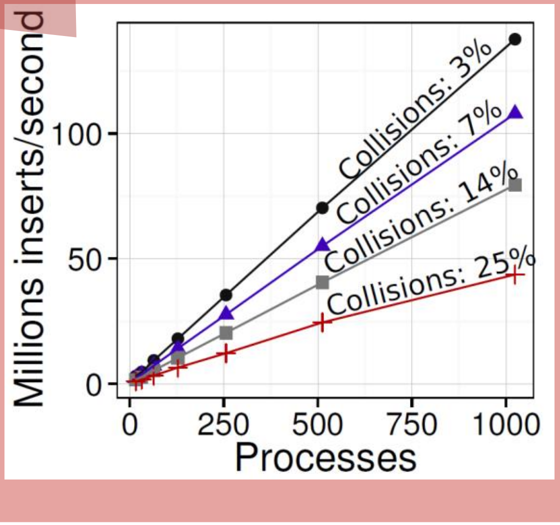


No hash collision:

➔ 1 remote atomic
➔ Up to 5x speedup over MP [1]

A hash collision:

➔ 4 remote atomics + 2 remote puts
➔ Significant performance drops



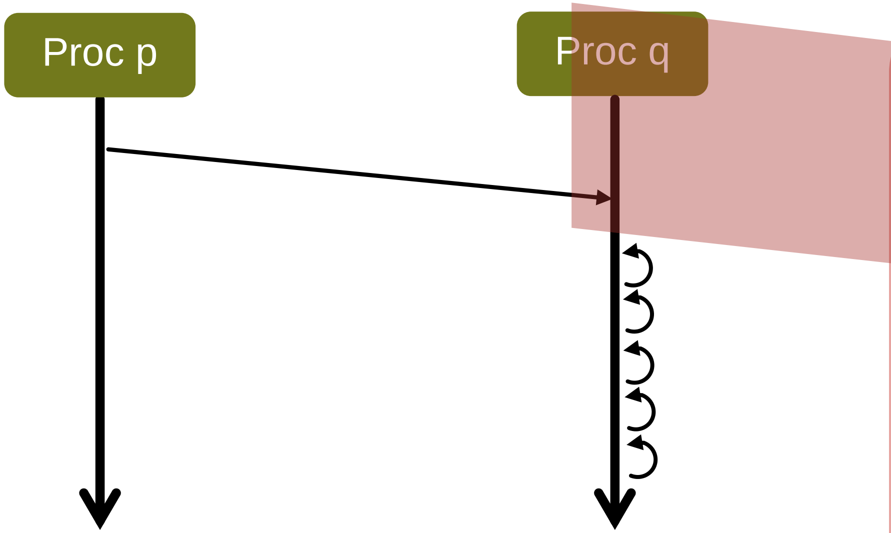
[1] R. Gerstenberger et al. Enabling Highly-Scalable Remote Memory Access Programming with MPI-3 One-Sided. SC13

Remote Memory Access Programming

- Is it ideal? NO!
- Consider an insert in a distributed hashtable...

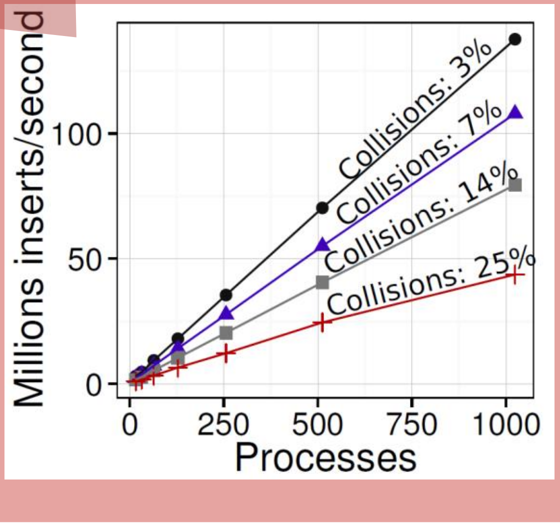
No hash collision:

➔ 1 remote atomic
➔ Up to 5x speedup over MP [1]



A hash collision:

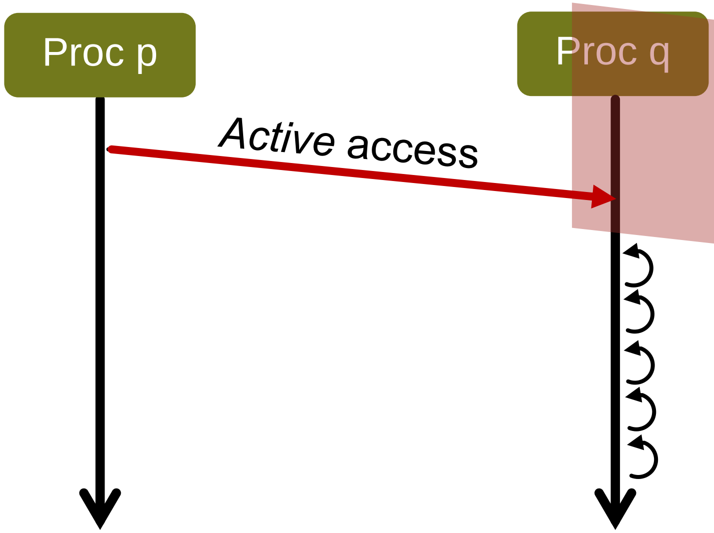
➔ 4 remote atomics + 2 remote puts
➔ Significant performance drops



[1] R. Gerstenberger et al. Enabling Highly-Scalable Remote Memory Access Programming with MPI-3 One-Sided. SC13

Remote Memory Access Programming

- Is it ideal? NO!
- Consider an insert in a distributed hashtable...

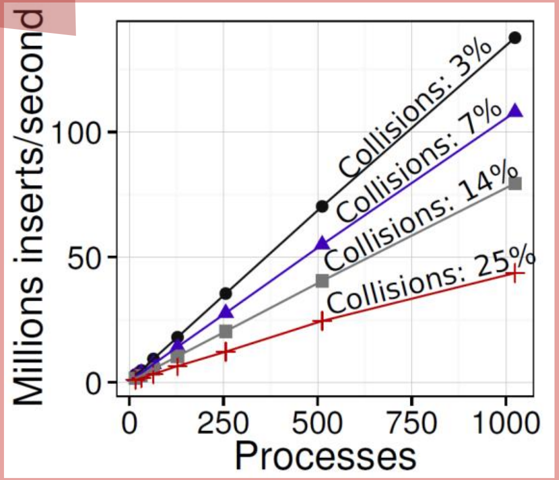


No hash collision:

➔ 1 remote atomic
➔ Up to 5x speedup over MP [1]

A hash collision:

➔ 4 remote atomics + 2 remote puts
➔ Significant performance drops



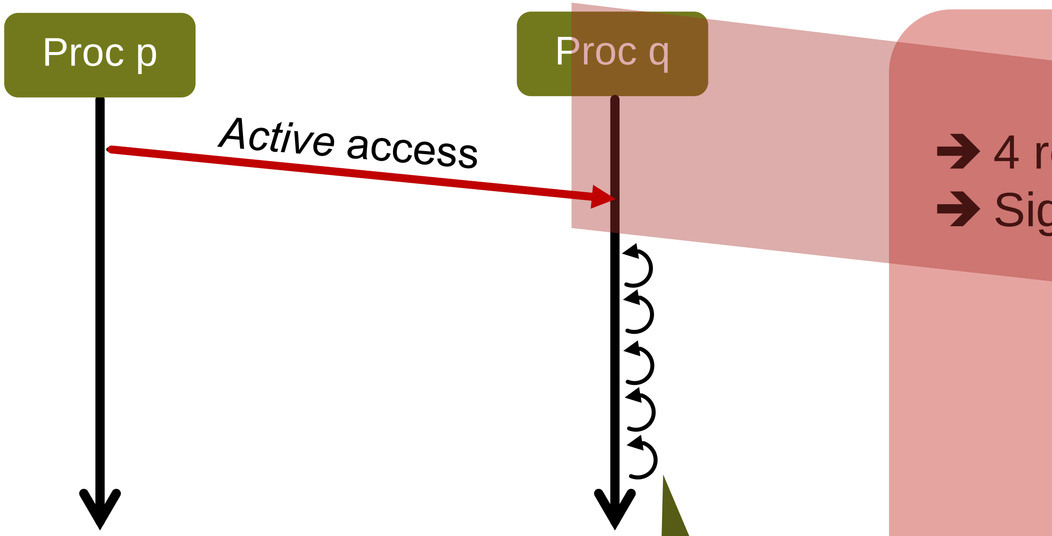
[1] R. Gerstenberger et al. Enabling Highly-Scalable Remote Memory Access Programming with MPI-3 One-Sided. SC13

REMOTE MEMORY ACCESS PROGRAMMING

- Is it ideal? NO!
- Consider an insert in a distributed hashtable...

No hash collision:

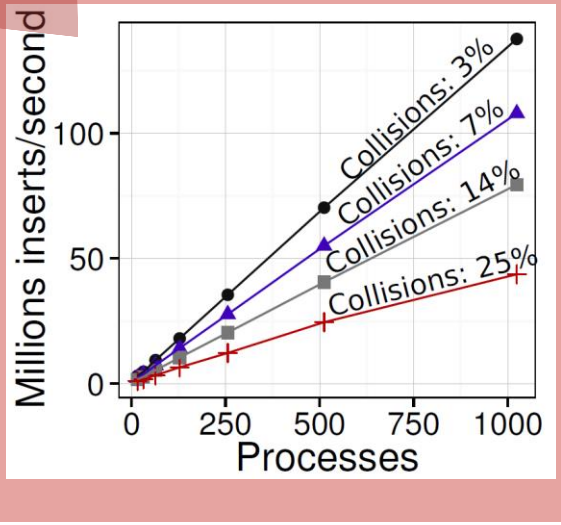
➔ 1 remote atomic
➔ Up to 5x speedup over MP [1]



Local execution; triggered by an active access. In RMA?

A hash collision:

➔ 4 remote atomics + 2 remote puts
➔ Significant performance drops



[1] R. Gerstenberger et al. Enabling Highly-Scalable Remote Memory Access Programming with MPI-3 One-Sided. SC13

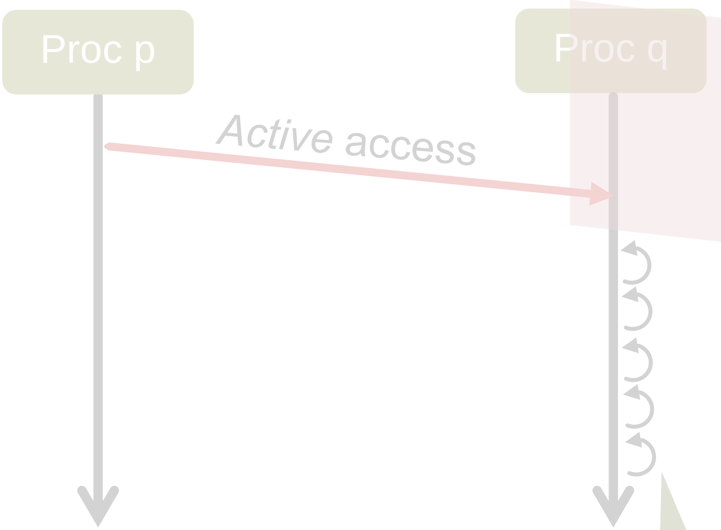
Remote Memory Access Programming

How to enable it?

- Is i
- Co
distributed hashtable...

No hash collision:

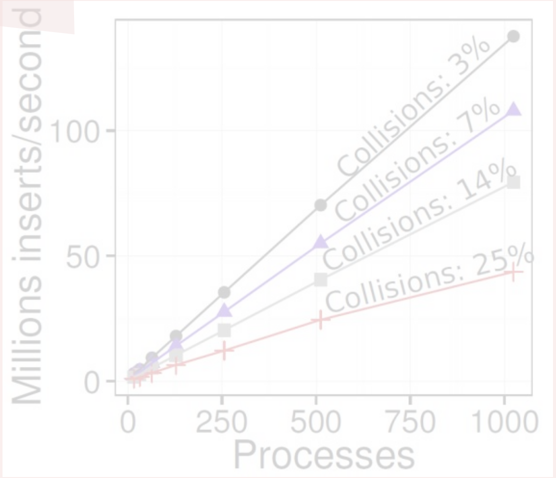
➔ 1 remote atomic
➔ Up to 5x speedup over MP [1]



Local execution; triggered by an active *access*. In RMA?

A hash collision:

➔ 4 remote atomics + 2 remote puts
➔ Significant performance drops



[1] R. Gerstenberger et al. Enabling Highly-Scalable Remote Memory Access Programming with MPI-3 One-Sided. SC13

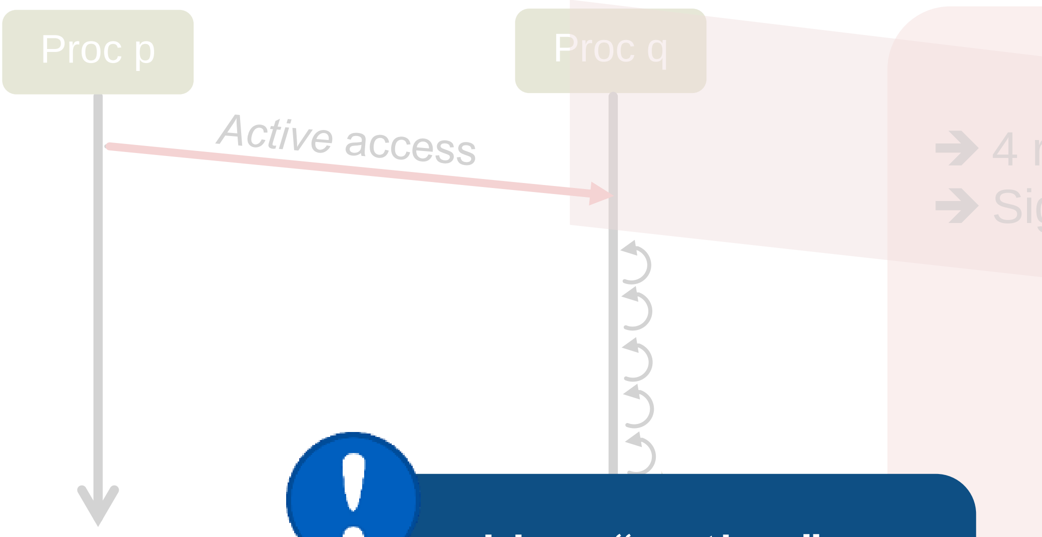
REMOTE MEMORY ACCESS PROGRAMMING

- Is i
- Co
distributed hashtable...

How to enable it?

No hash collision:

➔ 1 remote atomic
➔ Up to 5x speedup over MP [1]

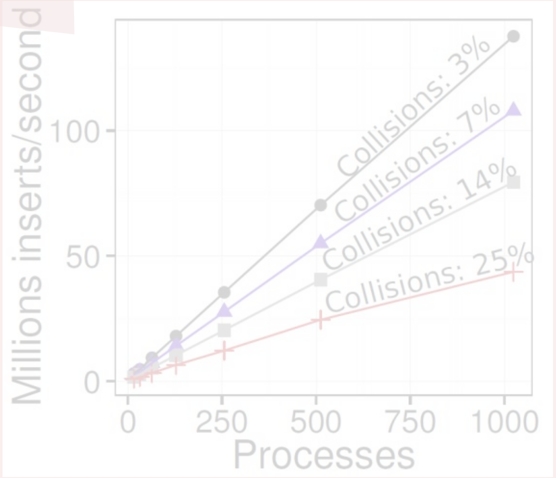


Local executi
active access. In RMA?

Use “active” semantics

A hash collision:

➔ 4 remote atomics + 2 remote puts
➔ Significant performance drops



[1] R. Gerstenberger et al. Enabling Highly-Scalable Remote Memory Access Programming with MPI-3 One-Sided. SC13

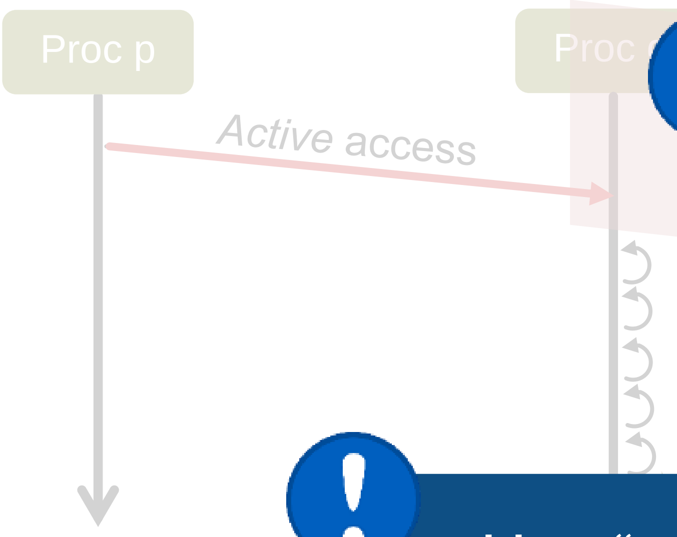
Remote Memory Access Programming

- Is i
- Co
distributed hashtable...

How to enable it?

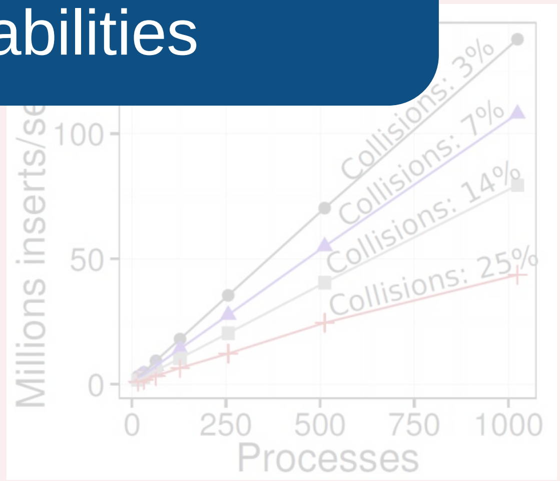
No hash collision:

➔ 1 remote atomic
➔ Up to 5x speedup over MP [1]



A hash collision:

emote puts
e drops



Use and extend I/O MMUs and their paging capabilities

Local executi
active access. In RMA?

Use “active” semantics

[1] R. Gerstenberger et al. Enabling Highly-Scalable Remote Memory Access Programming with MPI-3 One-Sided. SC13

“Active” Remote Memory Access Programming

“Active” Remote Memory Access Programming

Local execution; triggered by an active *access*. In RMA?

NO!

"Active" Remote Memory Access Programming

Local execution; triggered by an active *access*. In RMA?

How to enable it?

“ACTIVE” REMOTE MEMORY ACCESS PROGRAMMING

Local execution; triggered by an active *access*. In RMA?

How to enable it?

Use “active” semantics

"Active" Remote Memory Access Programming

Local execution; triggered by an active *access*. In RMA?

? How to enable it?

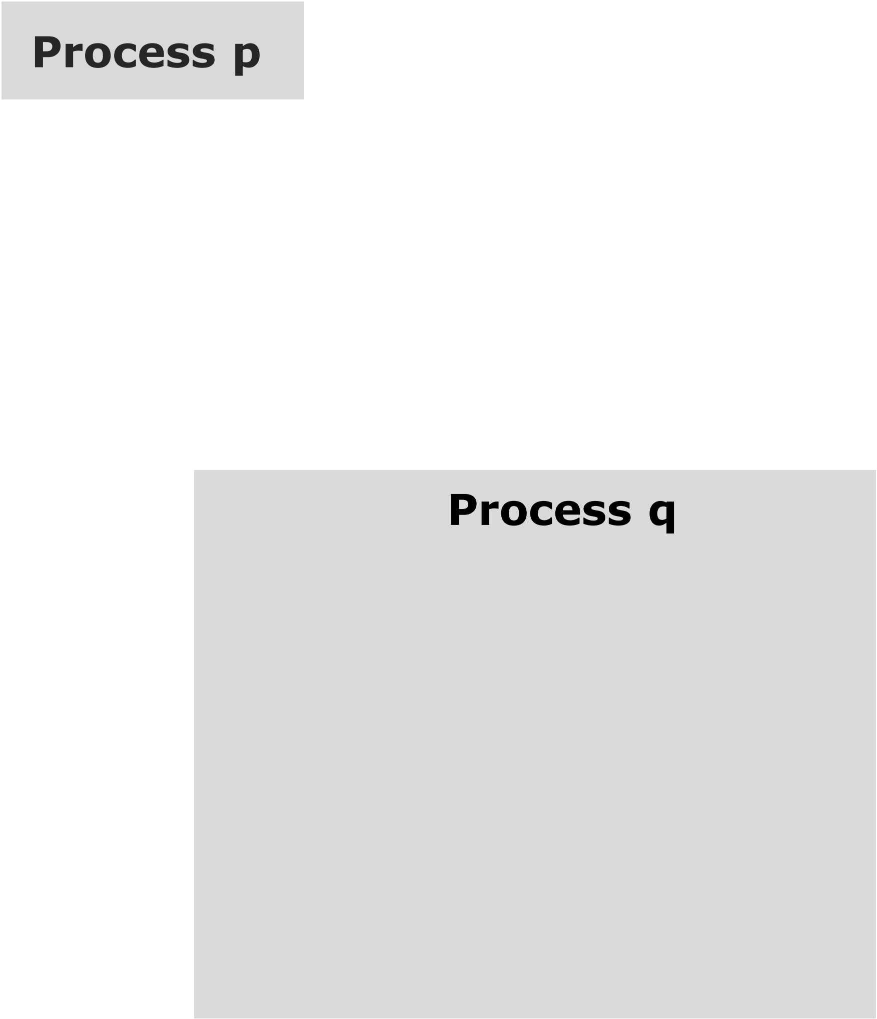
! Use "active" semantics

! Use and extend I/O MMUs and their paging capabilities

USE SEMANTICS FROM ACTIVE MESSAGES (AM) [1]

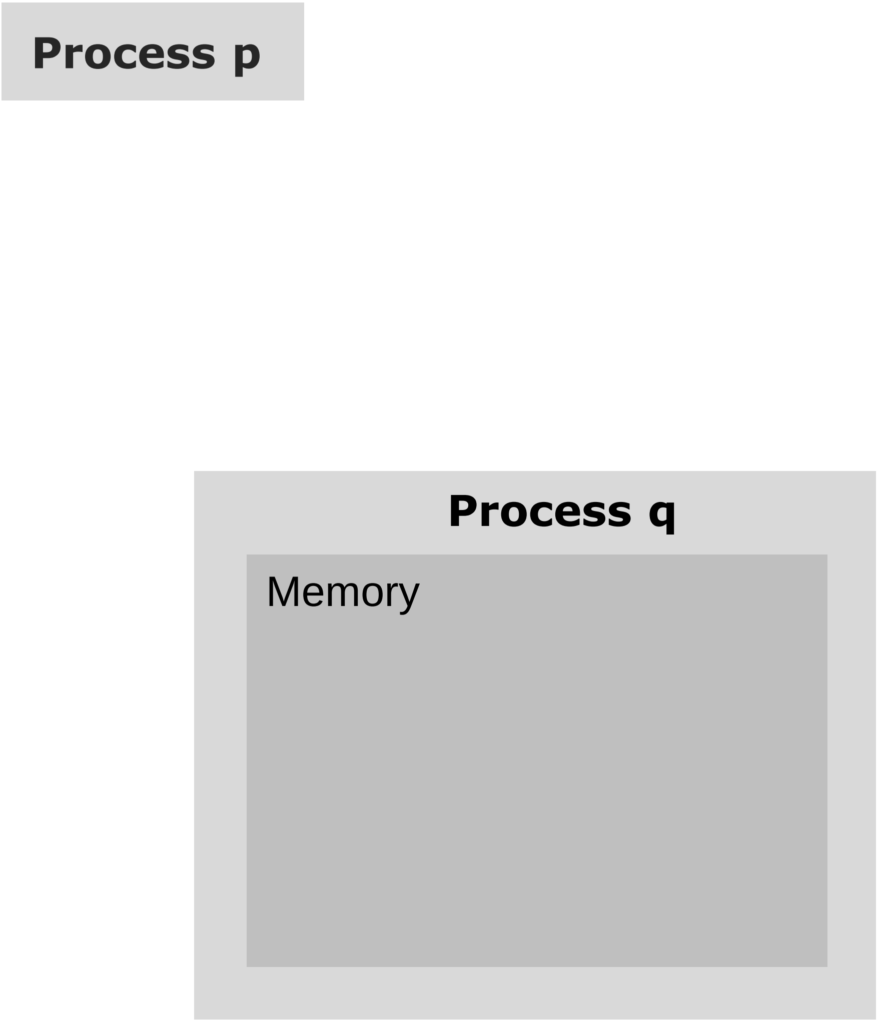
[1] T. von Eicken et al. Active messages: a mechanism for integrated communication and computation. ISCA’92.

USE SEMANTICS FROM ACTIVE MESSAGES (AM) [1]



[1] T. von Eicken et al. Active messages: a mechanism for integrated communication and computation. ISCA’92.

USE SEMANTICS FROM ACTIVE MESSAGES (AM) [1]



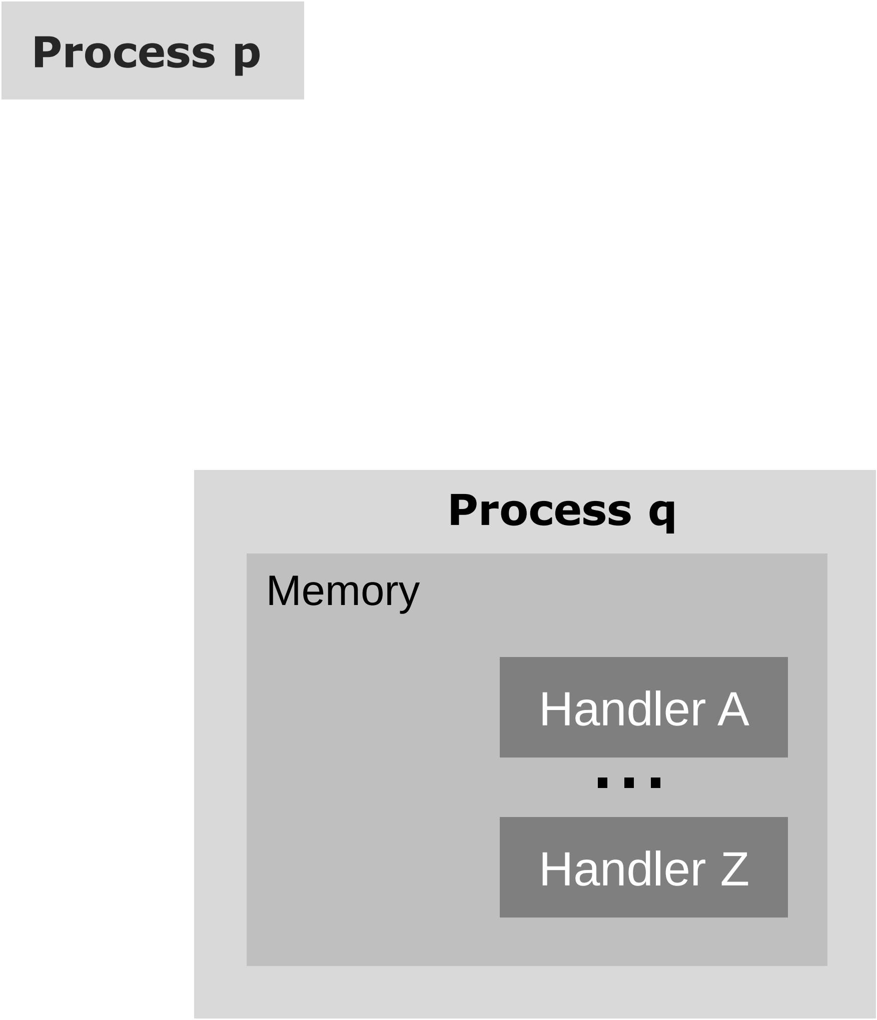
Process p

Process q

Memory

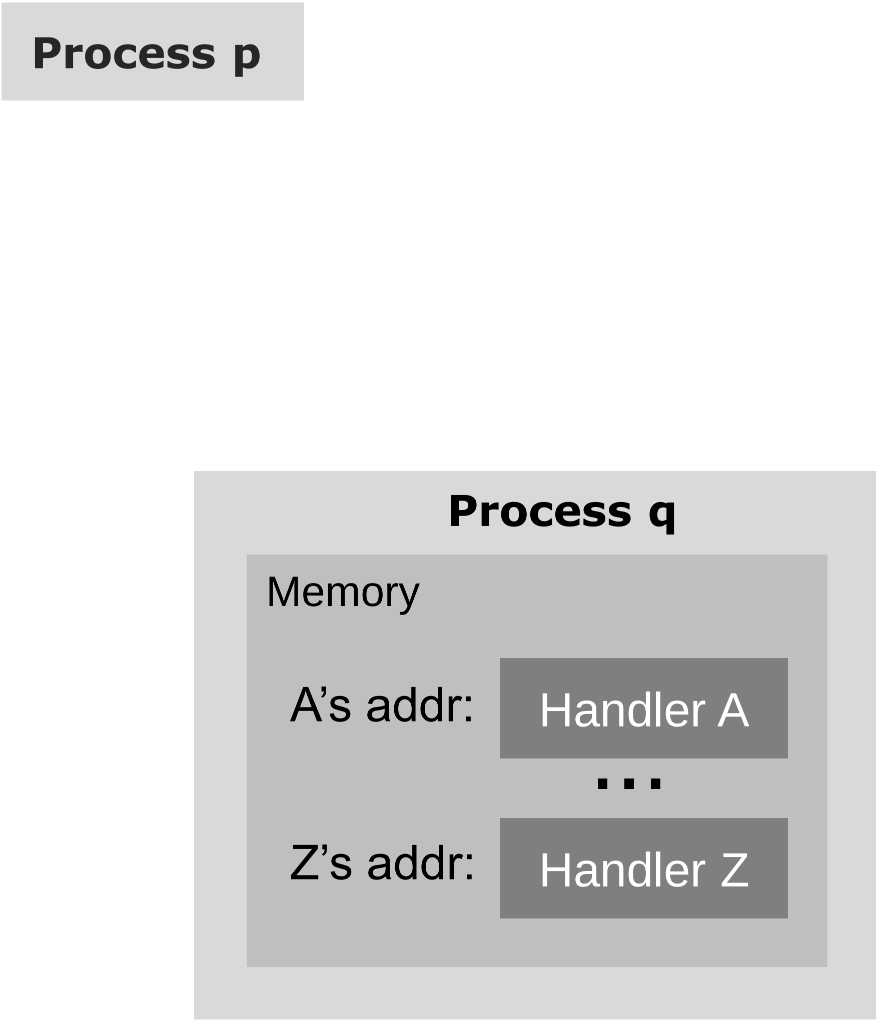
[1] T. von Eicken et al. Active messages: a mechanism for integrated communication and computation. ISCA’92.

USE SEMANTICS FROM ACTIVE MESSAGES (AM) [1]



[1] T. von Eicken et al. Active messages: a mechanism for integrated communication and computation. ISCA'92.

USE SEMANTICS FROM ACTIVE MESSAGES (AM) [1]



Process p

Process q

Memory

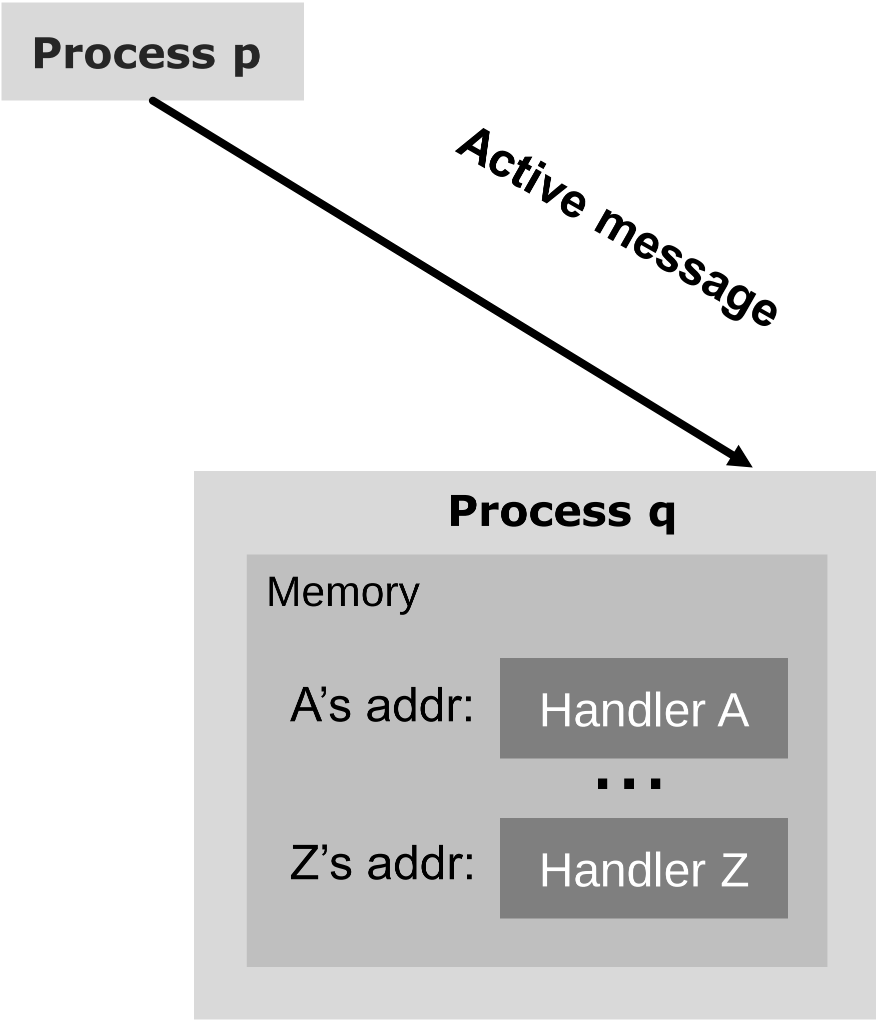
A's addr: Handler A

…

Z's addr: Handler Z

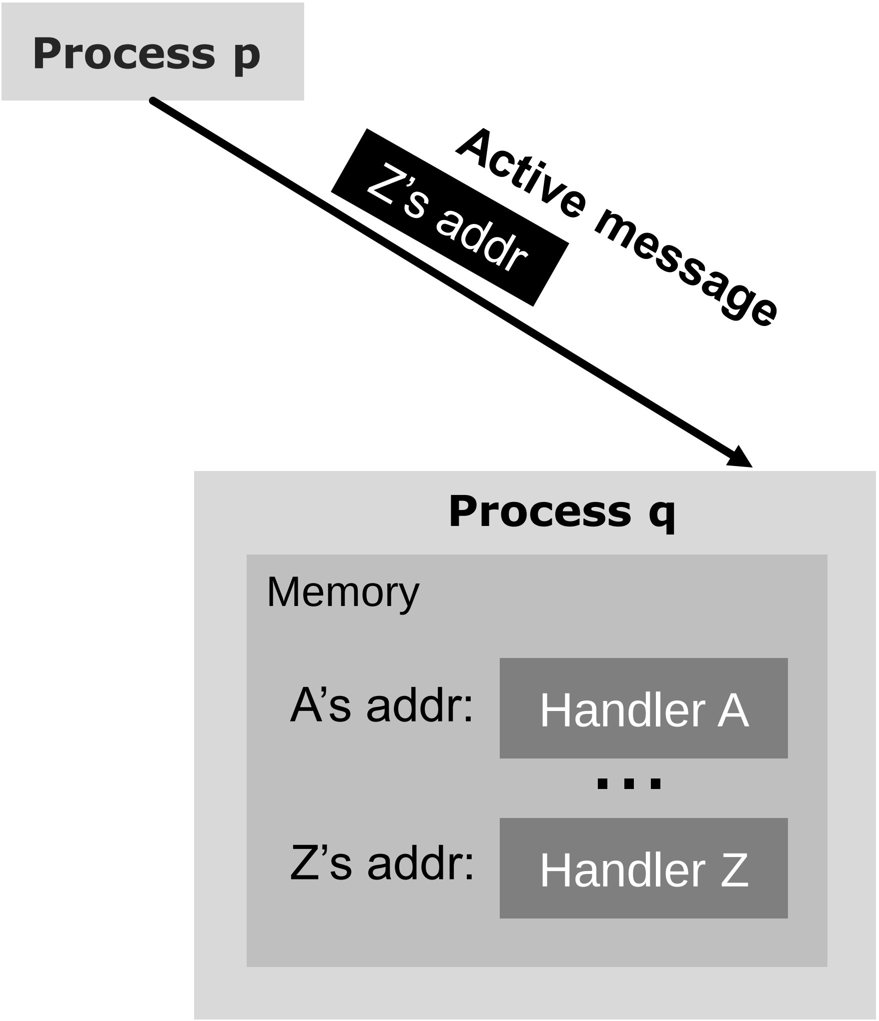
[1] T. von Eicken et al. Active messages: a mechanism for integrated communication and computation. ISCA'92.

USE SEMANTICS FROM ACTIVE MESSAGES (AM) [1]



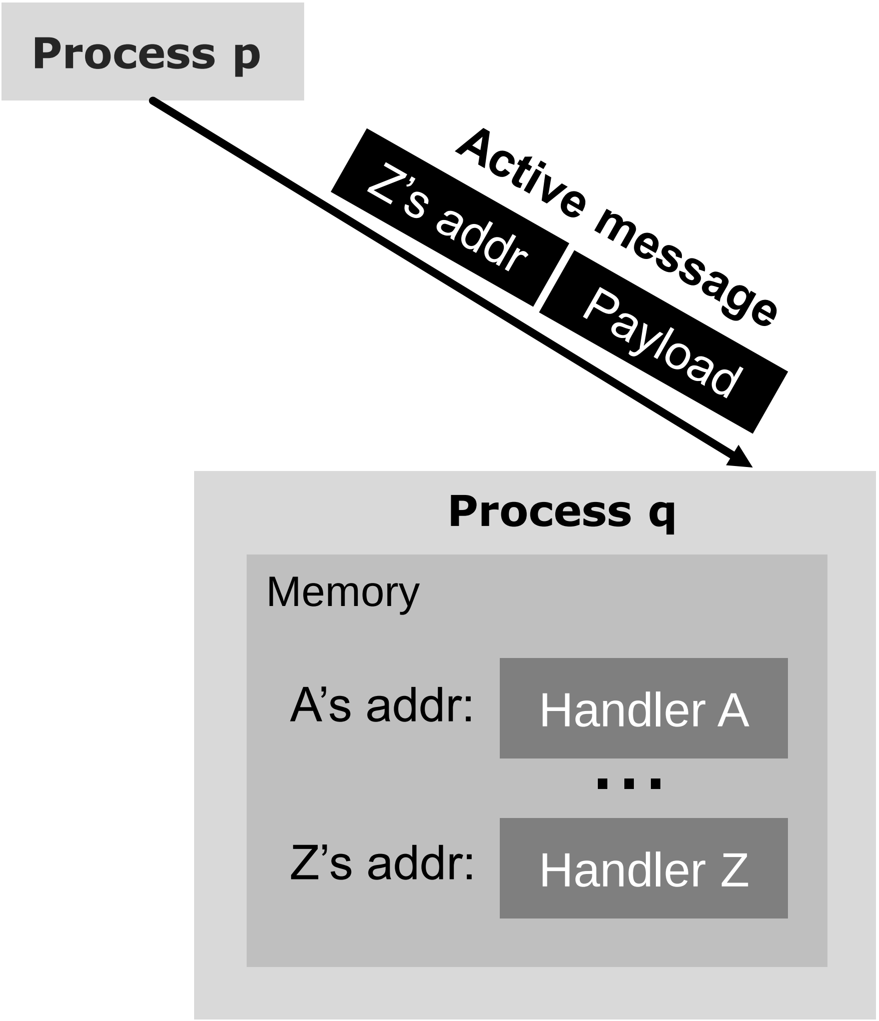
[1] T. von Eicken et al. Active messages: a mechanism for integrated communication and computation. ISCA'92.

USE SEMANTICS FROM ACTIVE MESSAGES (AM) [1]



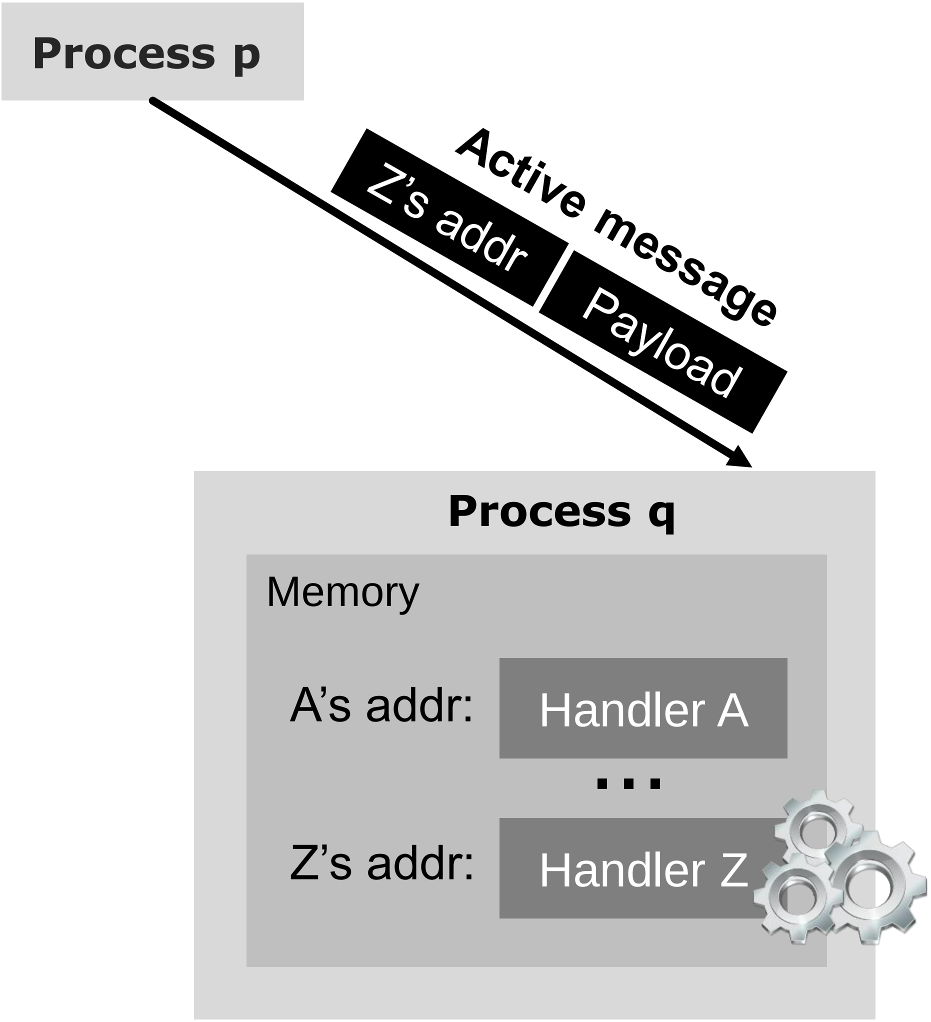
[1] T. von Eicken et al. Active messages: a mechanism for integrated communication and computation. ISCA'92.

USE SEMANTICS FROM ACTIVE MESSAGES (AM) [1]



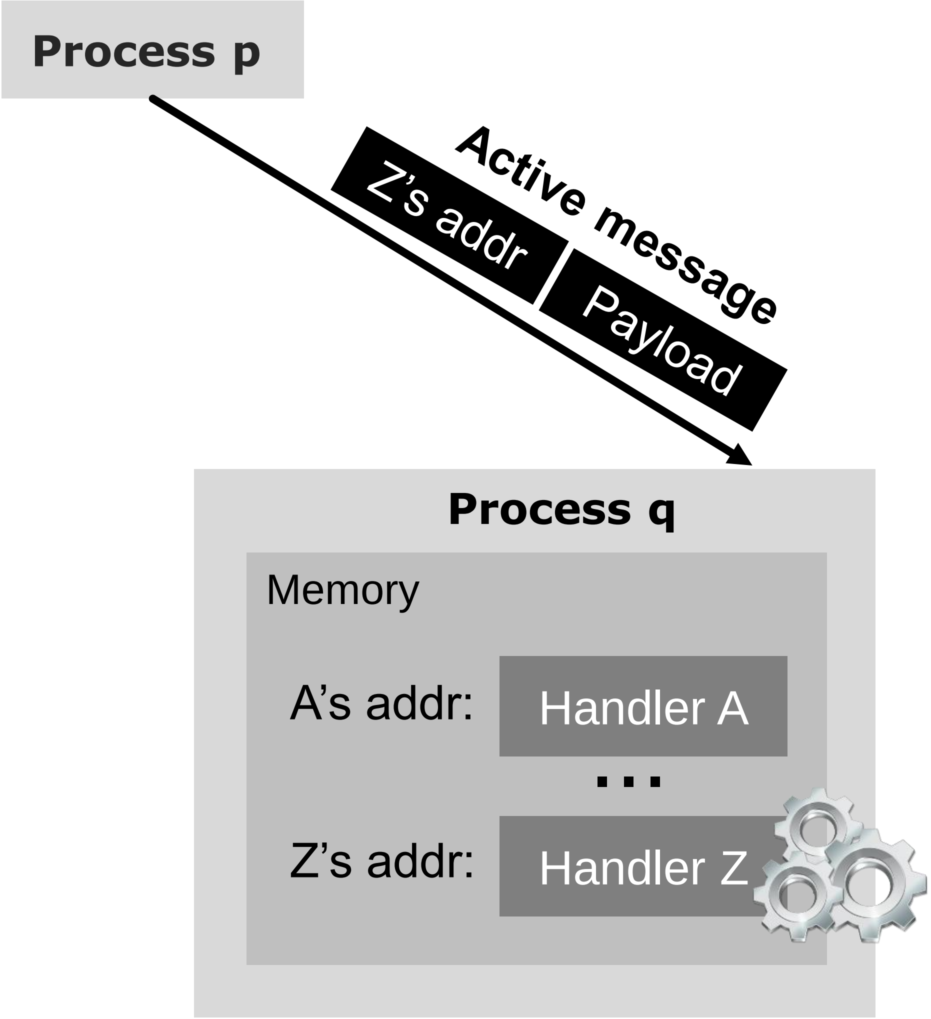
[1] T. von Eicken et al. Active messages: a mechanism for integrated communication and computation. ISCA'92.

USE SEMANTICS FROM ACTIVE MESSAGES (AM) [1]



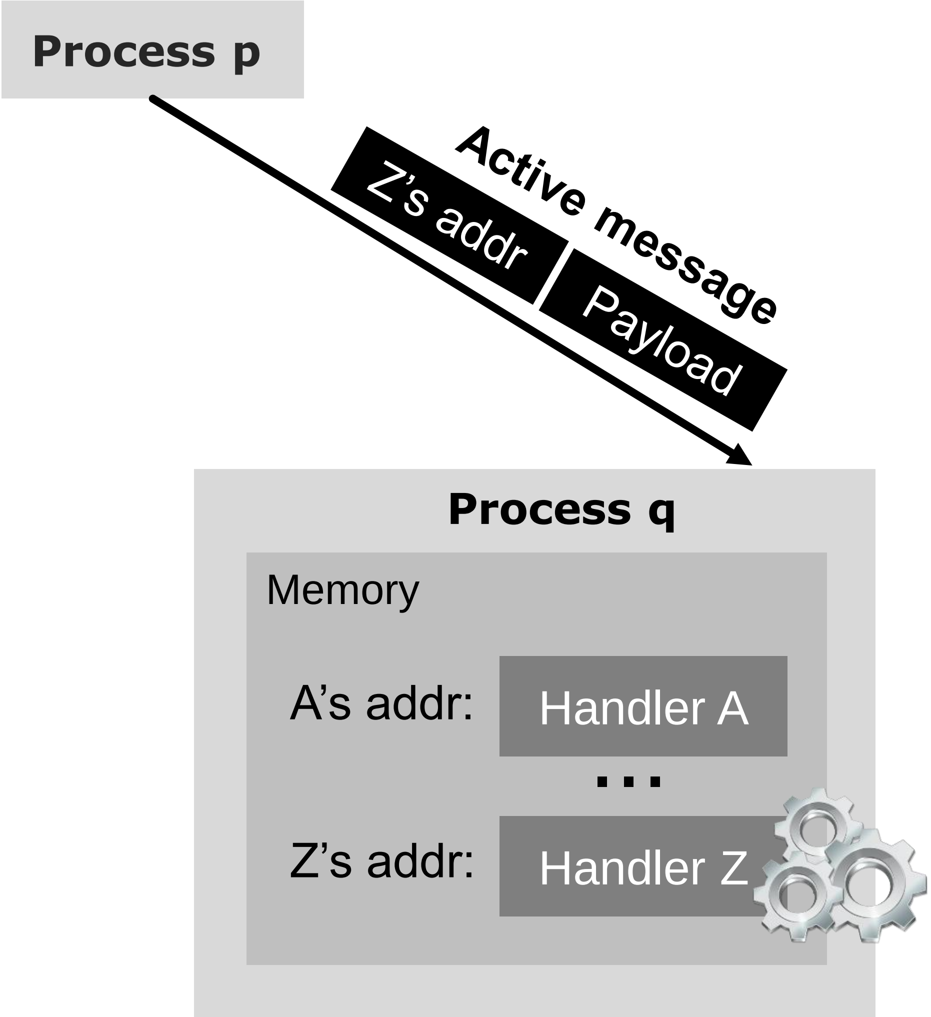
[1] T. von Eicken et al. Active messages: a mechanism for integrated communication and computation. ISCA'92.

USE SEMANTICS FROM ACTIVE MESSAGES (AM) [1]



[1] T. von Eicken et al. Active messages: a mechanism for integrated communication and computation. ISCA'92.

USE SEMANTICS FROM ACTIVE MESSAGES (AM) [1]



[1] T. von Eicken et al. Active messages: a mechanism for integrated communication and computation. ISCA'92.

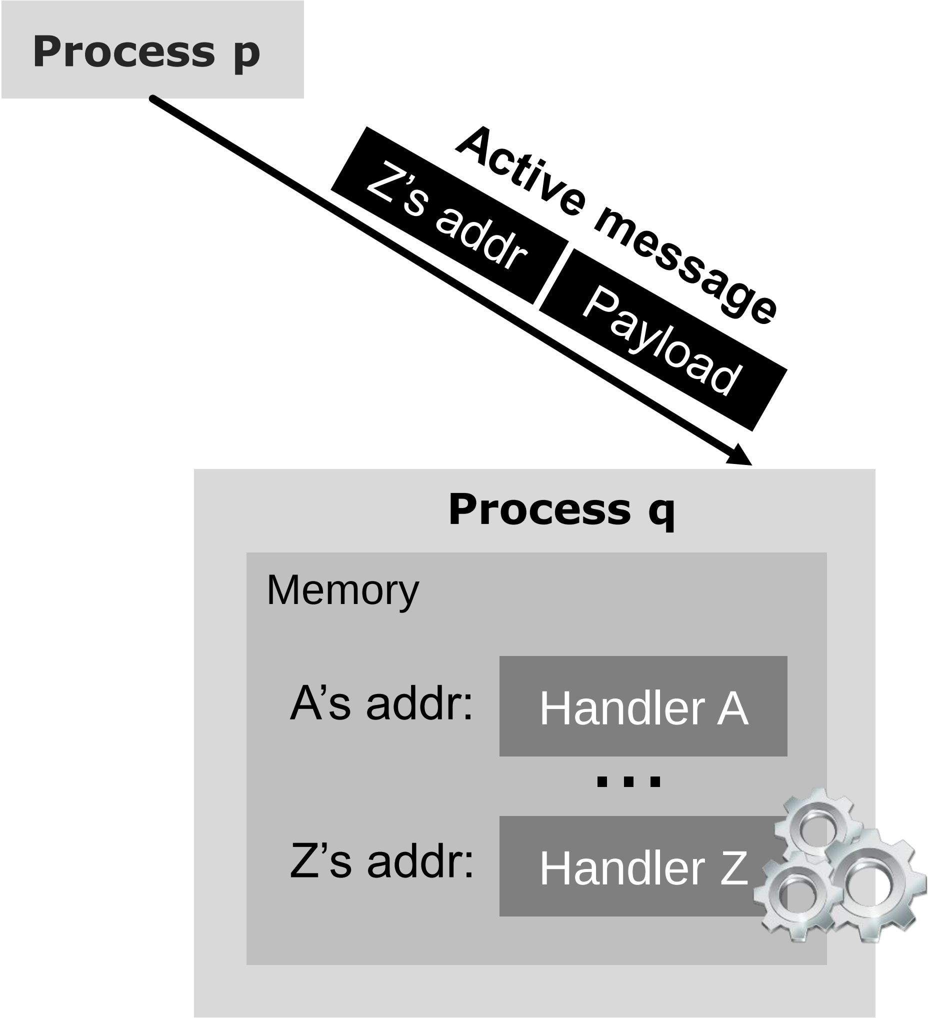
USE SEMANTICS FROM ACTIVE MESSAGES (AM) [1]

IBM

Myricom

AM++[2]

GASNet [3]



Process p

Active message

Z's addr

Payload

Process q

Memory

A's addr: Handler A

...

Z's addr: Handler Z

[1] T. von Eicken et al. Active messages: a mechanism for integrated communication and computation. ISCA'92.

[2] J. J. Willcock et al. AM++: A generalized active message framework. PACT'10.

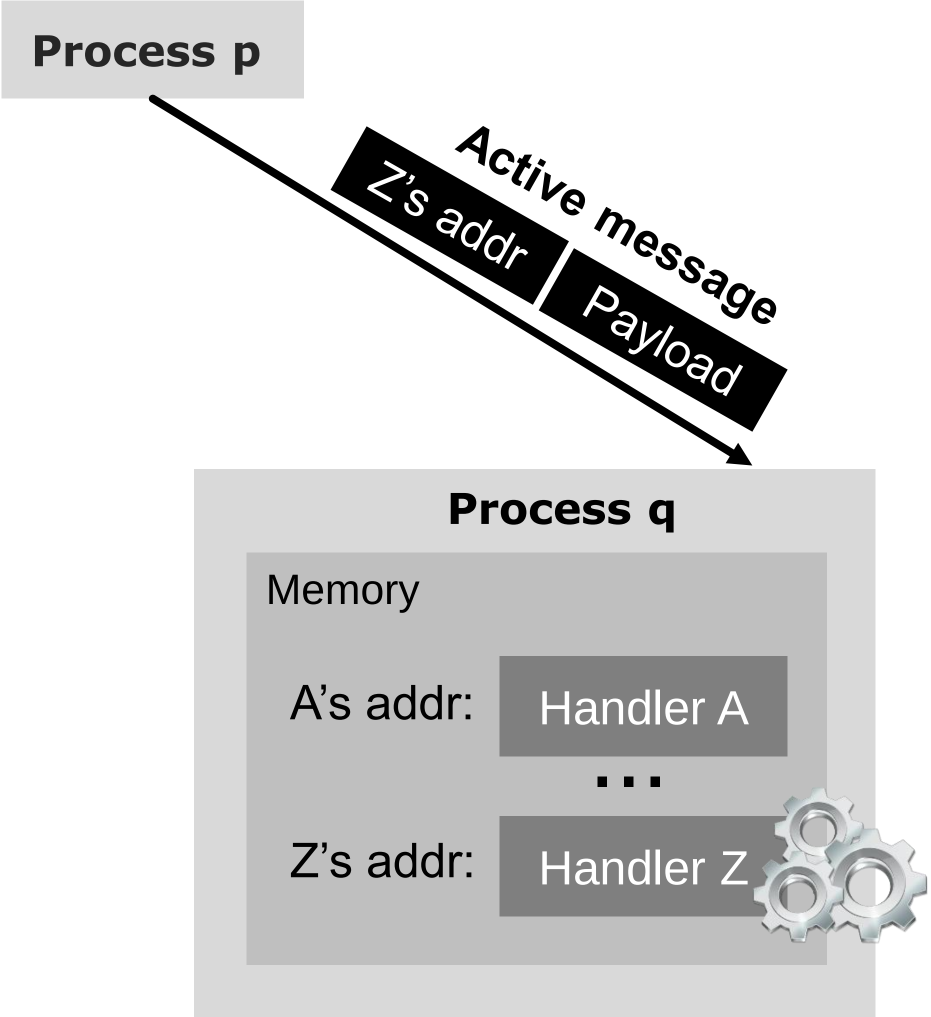
[3] D. Bonachea, GASNet Specification, v1.1. Berkeley Technical Report. 2002.

USE SEMANTICS FROM ACTIVE MESSAGES (AM) [1]

AM++[2]
GASNet [3]

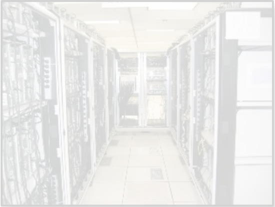
We need *active* puts/gets:

- Invoke a handler upon accessing a given page
- Preserve one-sided RMA behavior



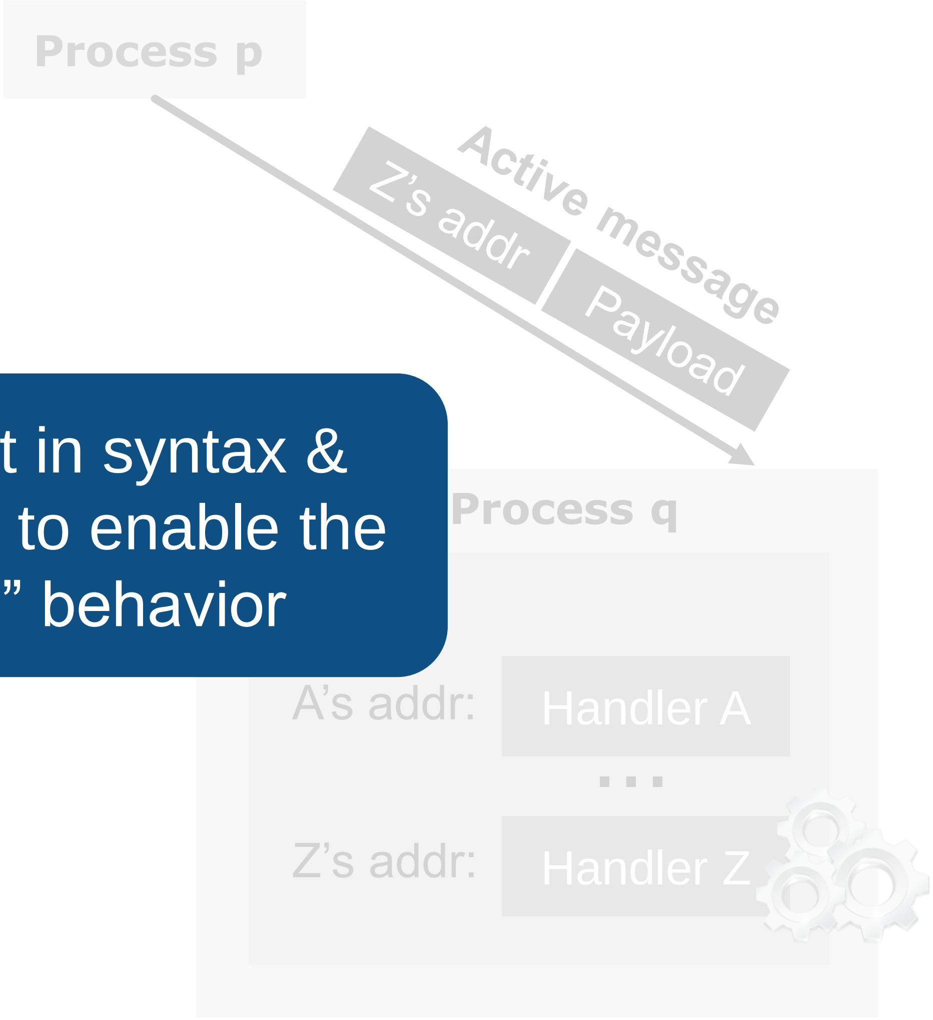
[1] T. von Eicken et al. Active messages: a mechanism for integrated communication and computation. ISCA'92.
[2] J. J. Willcock et al. AM++: A generalized active message framework. PACT'10.
[3] D. Bonachea, GASNet Specification, v1.1. Berkeley Technical Report. 2002.

USE SEMANTICS FROM ACTIVE MESSAGES (AM) [1]



A

GAS



We use it in syntax & semantics to enable the "active" behavior

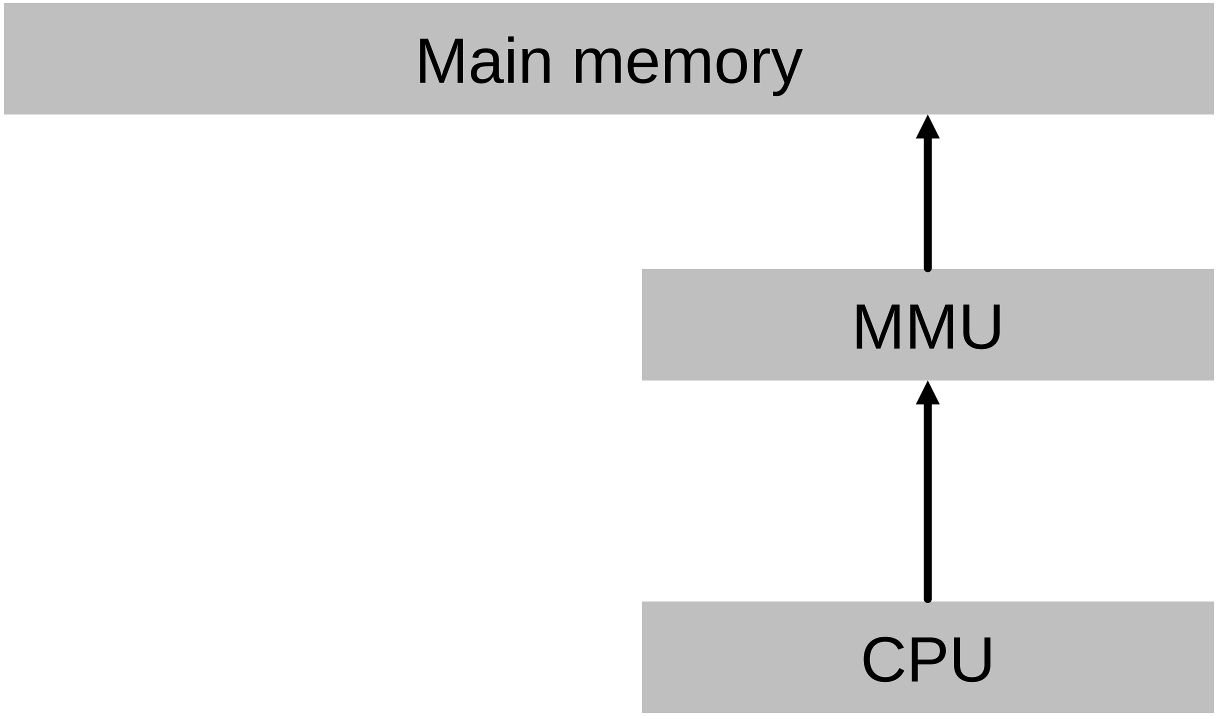
We need *active* puts/gets:

- Invoke a handler upon accessing a given page
- Preserve one-sided RMA behavior

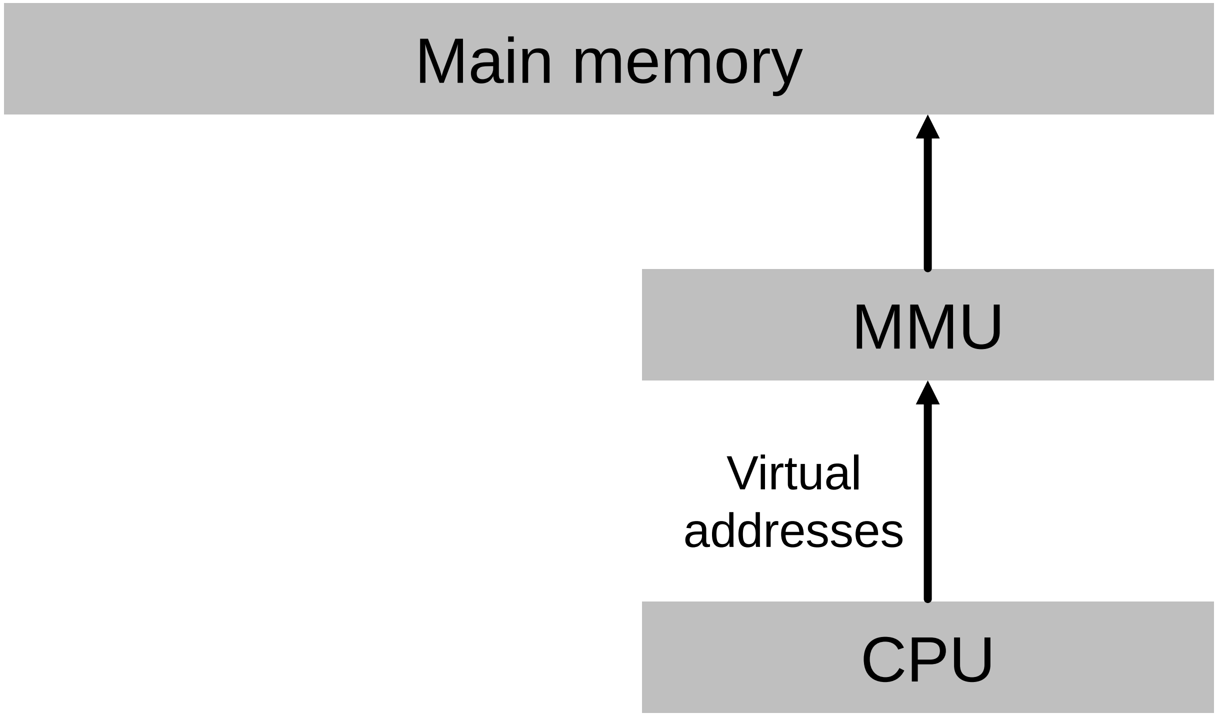
[1] T. von Eicken et al. Active messages: a mechanism for integrated communication and computation. ISCA'92.
[2] J. J. Willcock et al. AM++: A generalized active message framework. PACT'10.
[3] D. Bonachea, GASNet Specification, v1.1. Berkeley Technical Report. 2002.

Use Input/Output Memory Management Units

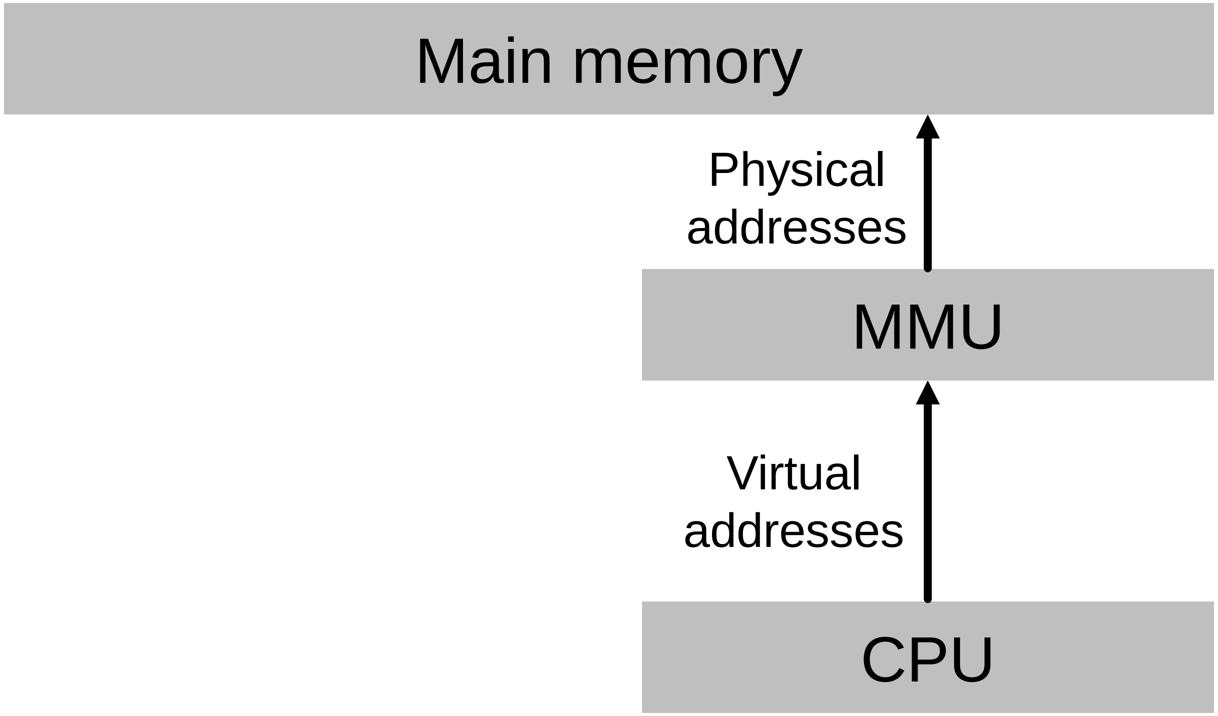
Use Input/Output Memory Management Units



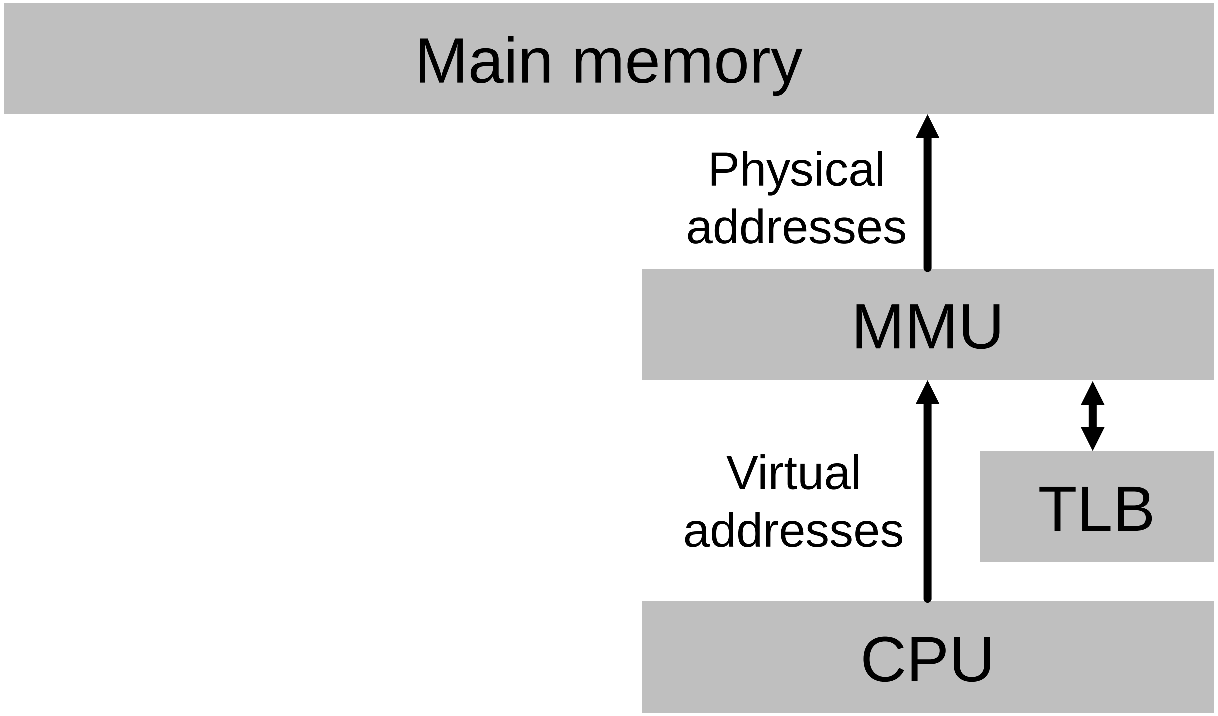
USE INPUT/OUTPUT MEMORY MANAGEMENT UNITS



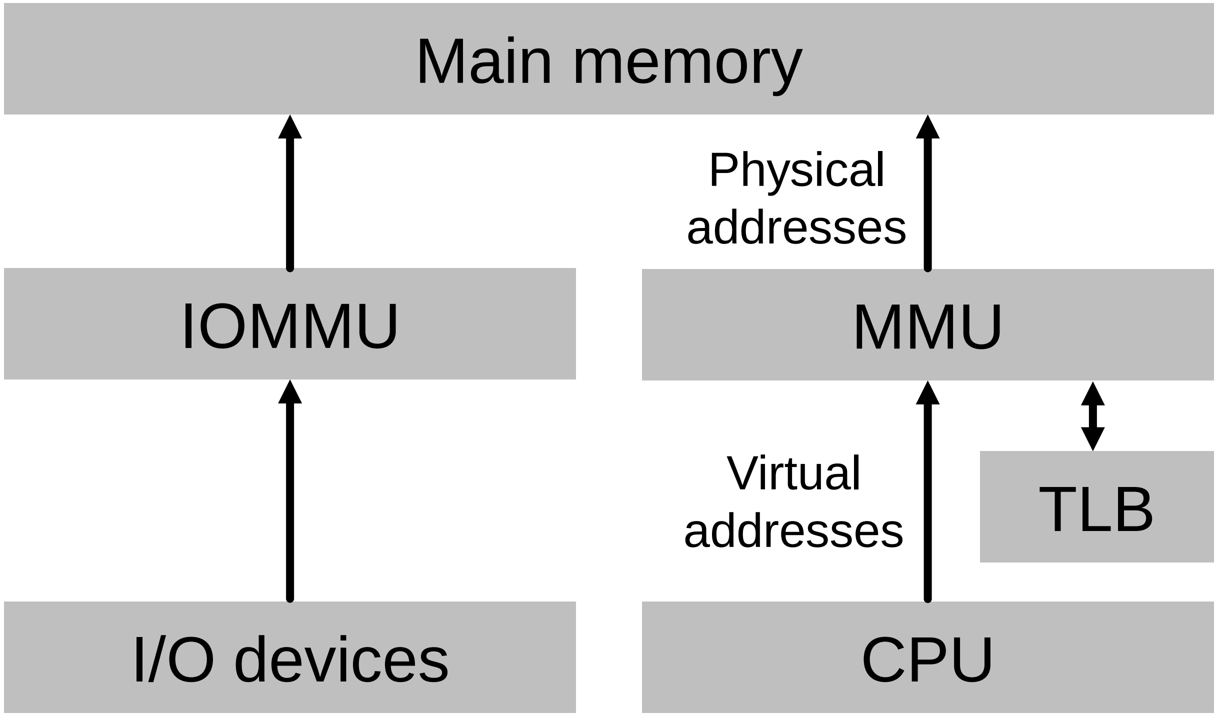
Use Input/Output Memory Management Units



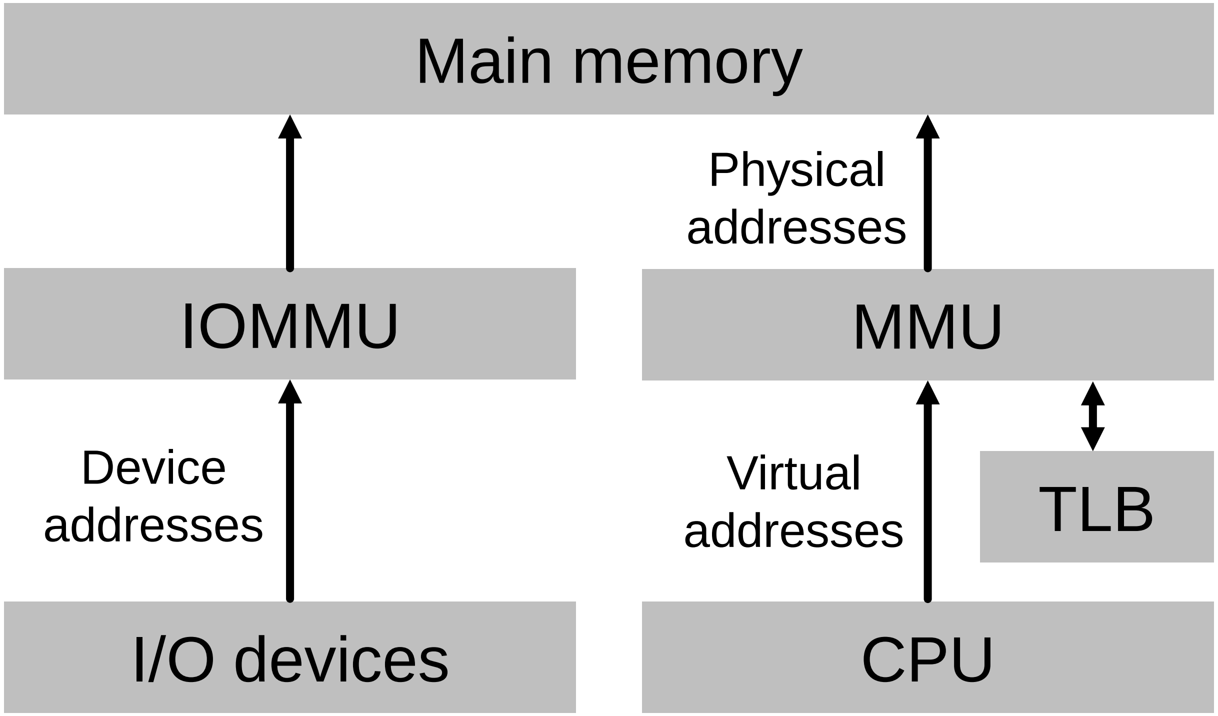
Use Input/Output Memory Management Units



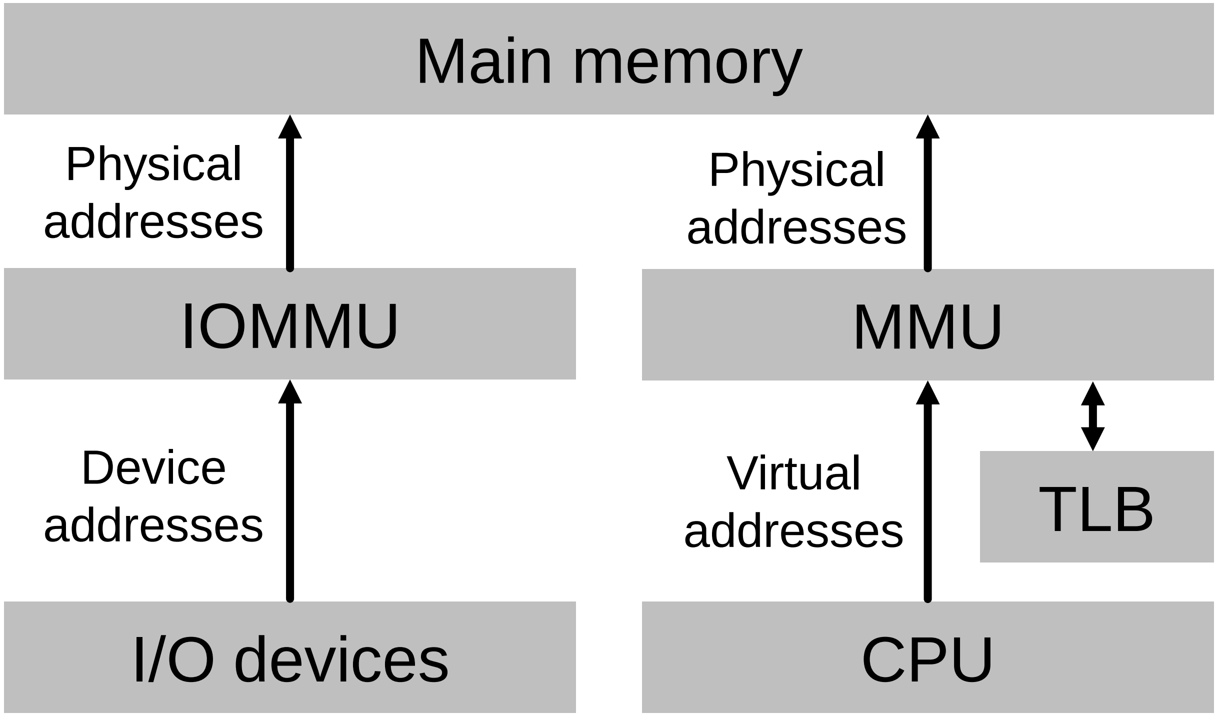
Use Input/Output Memory Management Units



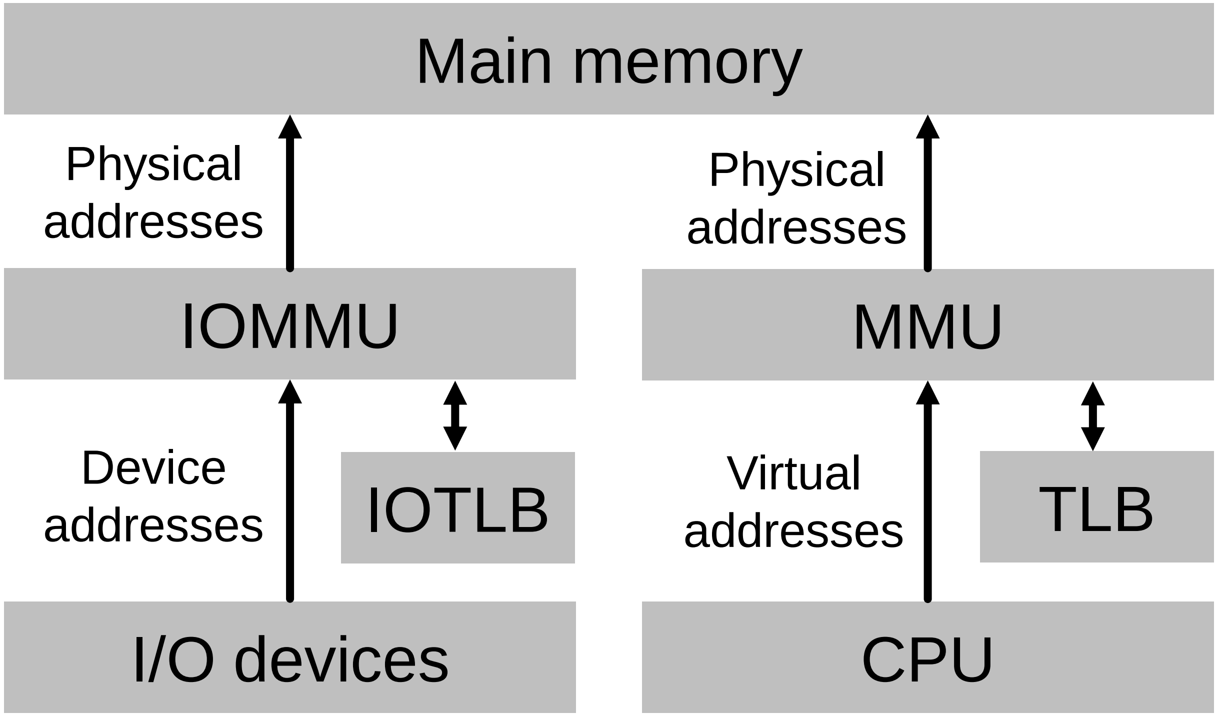
Use Input/Output Memory Management Units



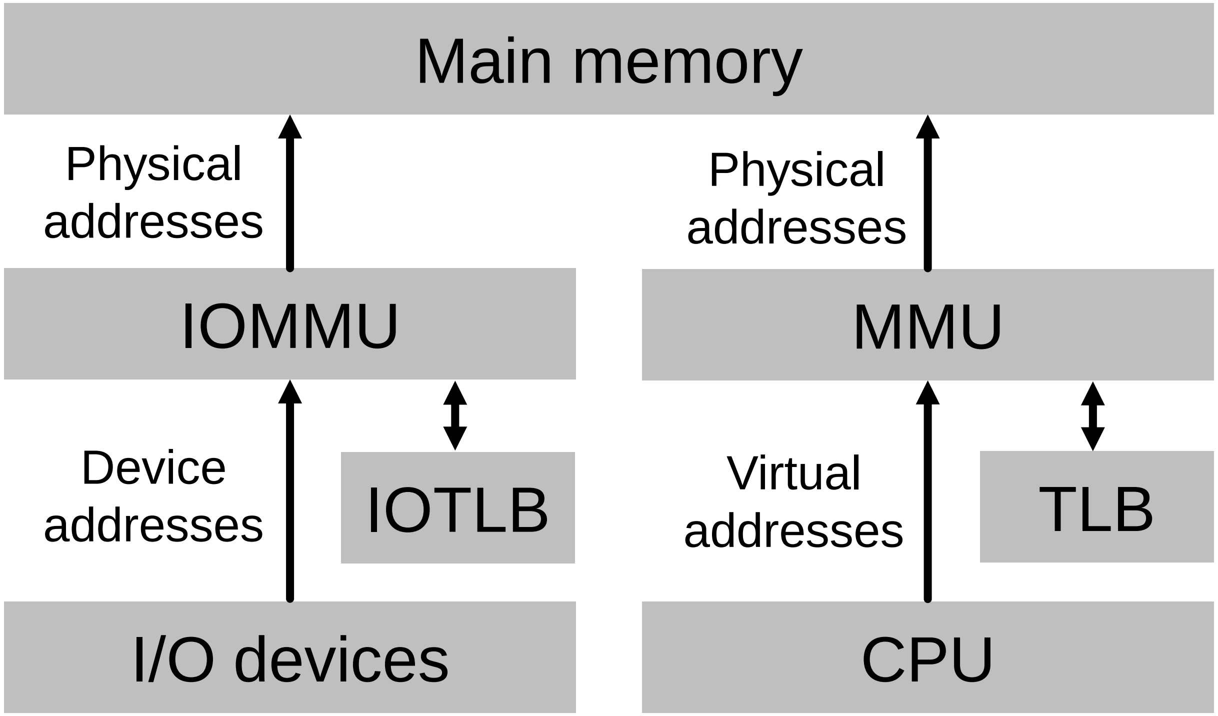
Use Input/Output Memory Management Units



Use Input/Output Memory Management Units



Use Input/Output Memory Management Units



AMD

IBM

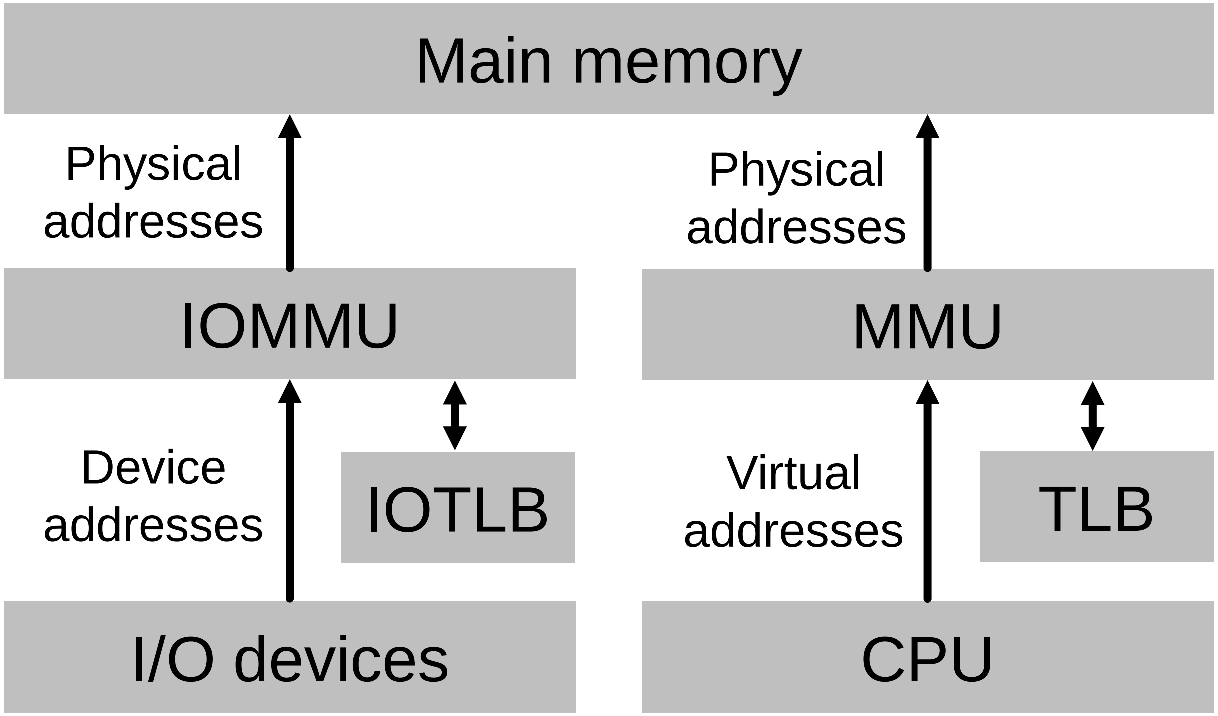
ARM®

Sun microsystems

SOLARFLARE®

intel®

Use Input/Output Memory Management Units



AMD

IBM

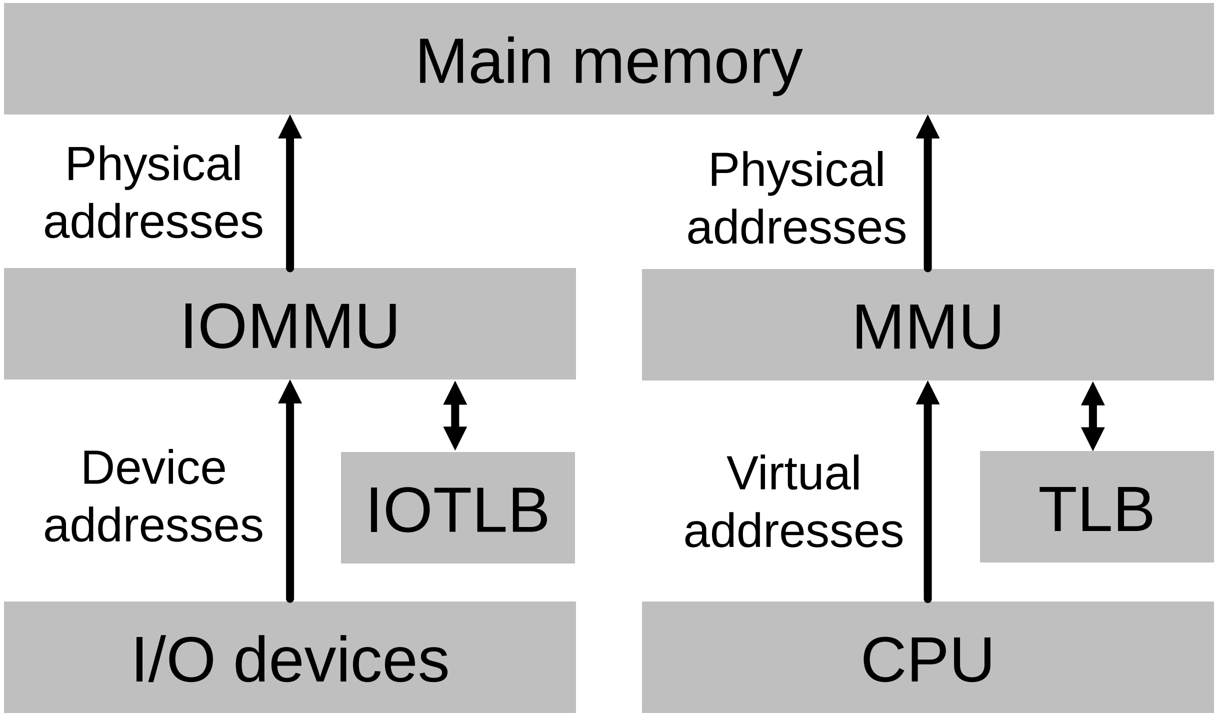
ARM®

Sun microsystems

SOLARFLARE®

intel®

Use Input/Output Memory Management Units



Main memory

Physical addresses

Physical addresses

IOMMU

MMU

Device addresses

IOTLB

Virtual addresses

TLB

I/O devices

CPU

AMD

IBM

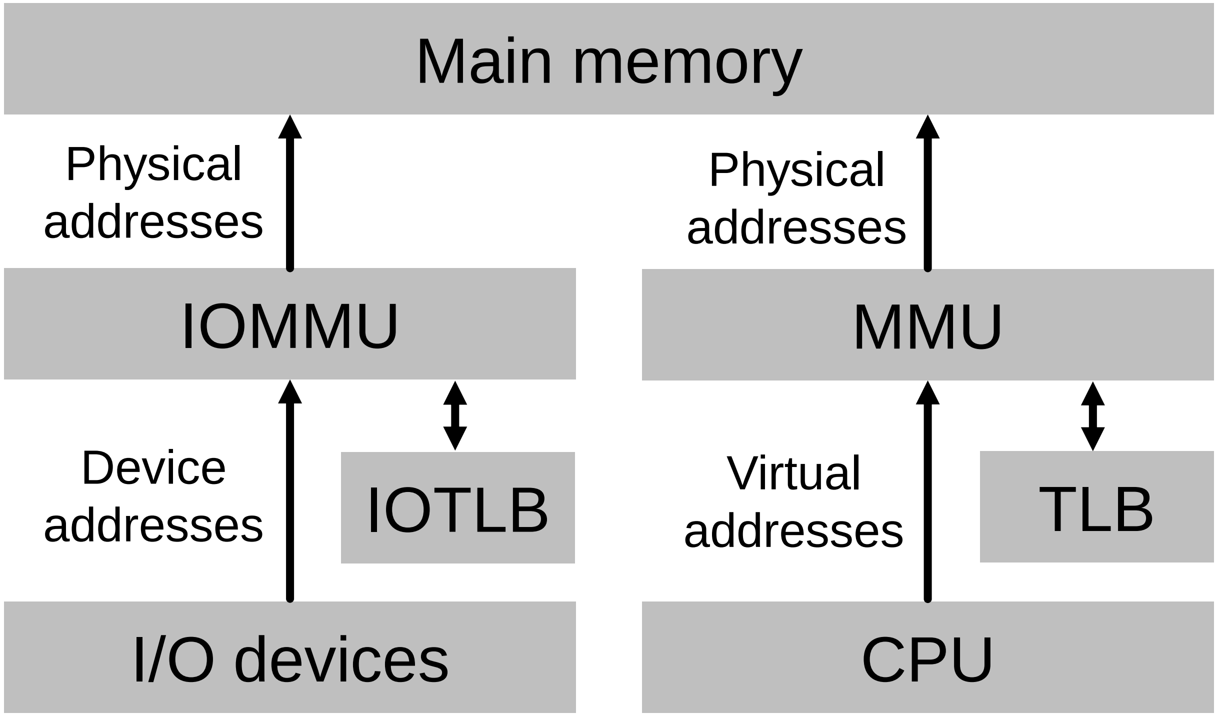
ARM®

Sun microsystems

SOLARFLARE®

intel® +

Use Input/Output Memory Management Units



AMD

IBM

ARM®

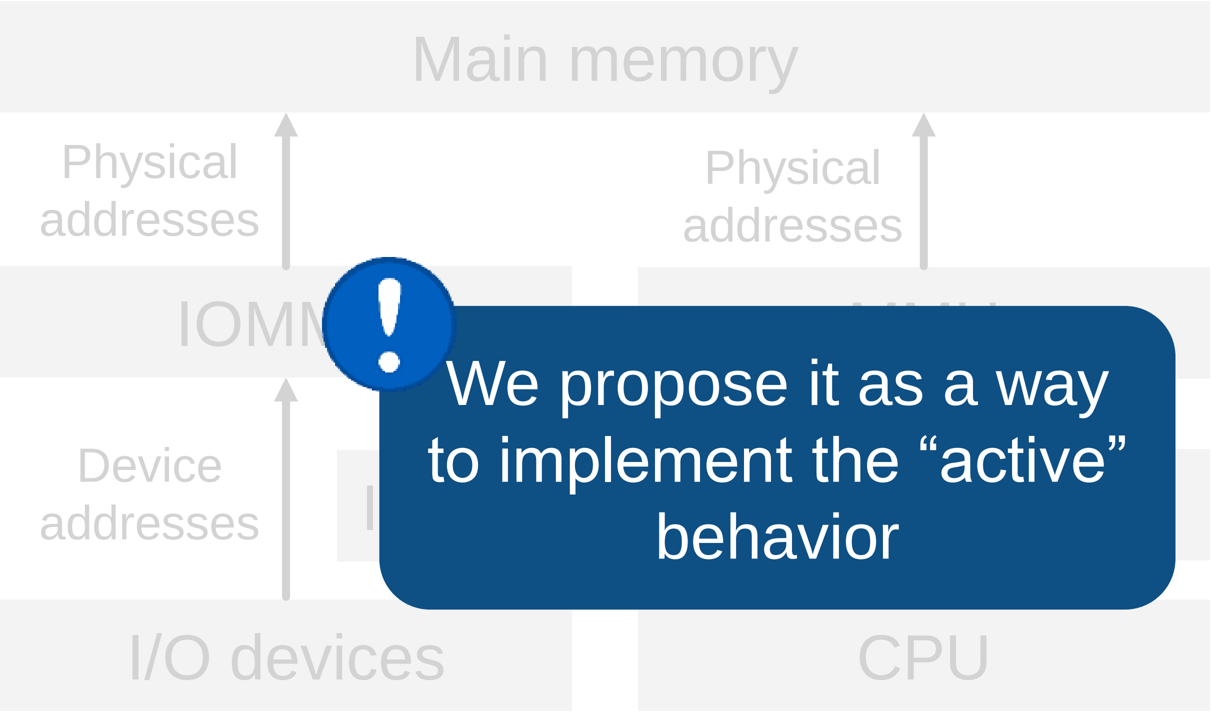
Sun microsystems

SOLARFLARE®

intel®

+ PCI EXPRESS®

Use Input/Output Memory Management Units



We propose it as a way to implement the "active" behavior

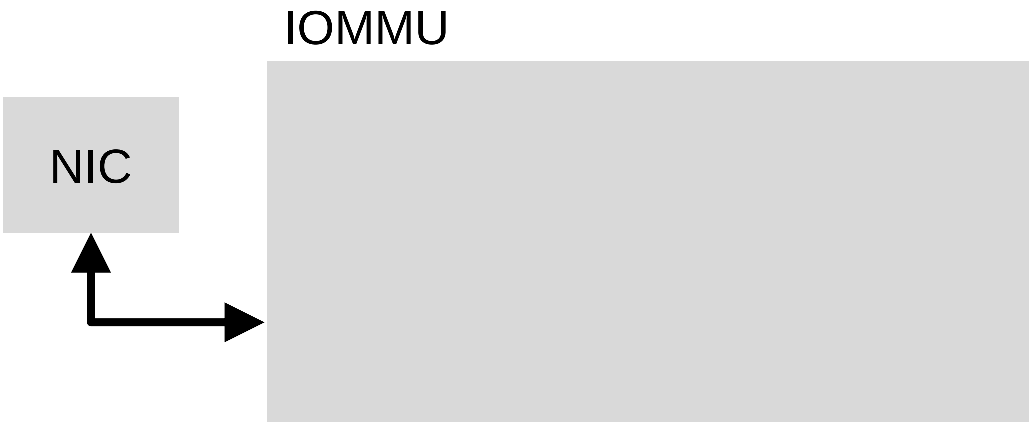
AMD
IBM
ARM®
Sun microsystems
SOLARFLARE®
intel®
+ PCI EXPRESS®

IOMMUs and RMA

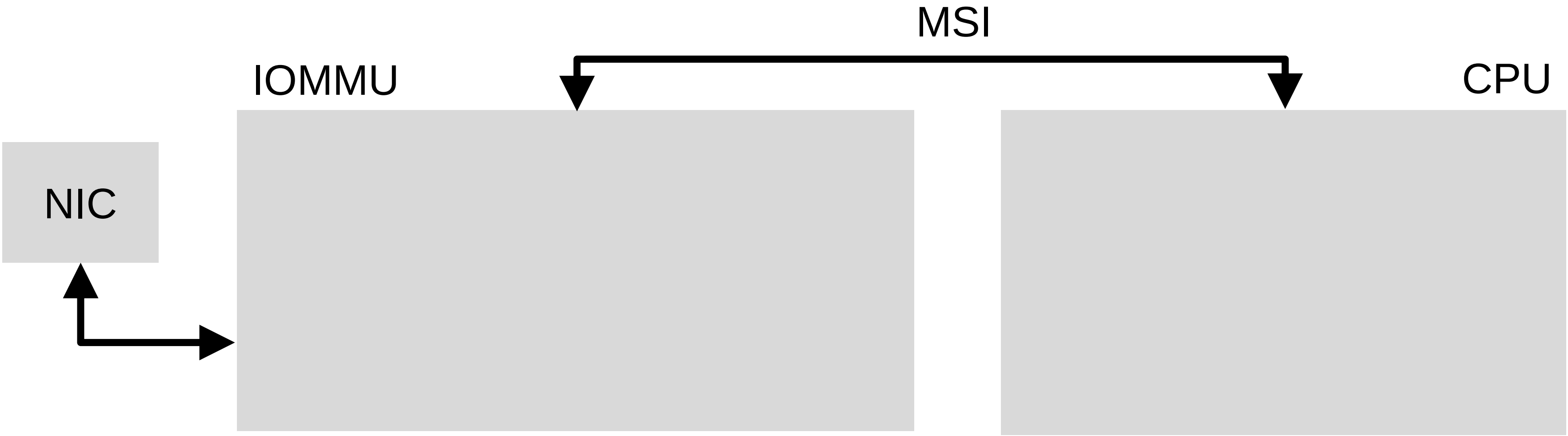
IOMMUs and RMA

NIC

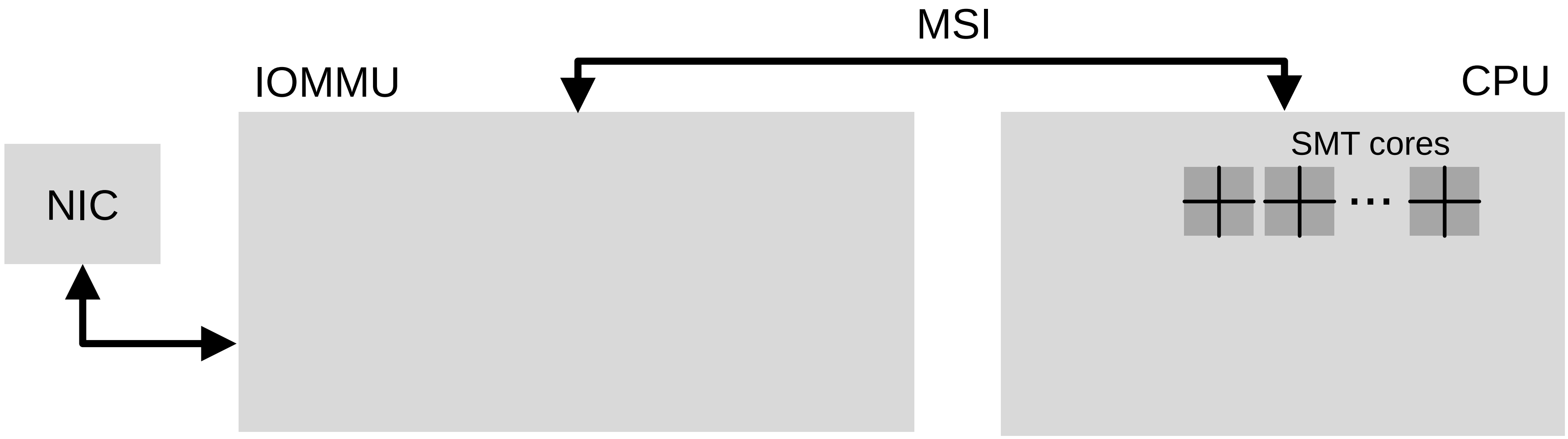
IOMMUs and RMA



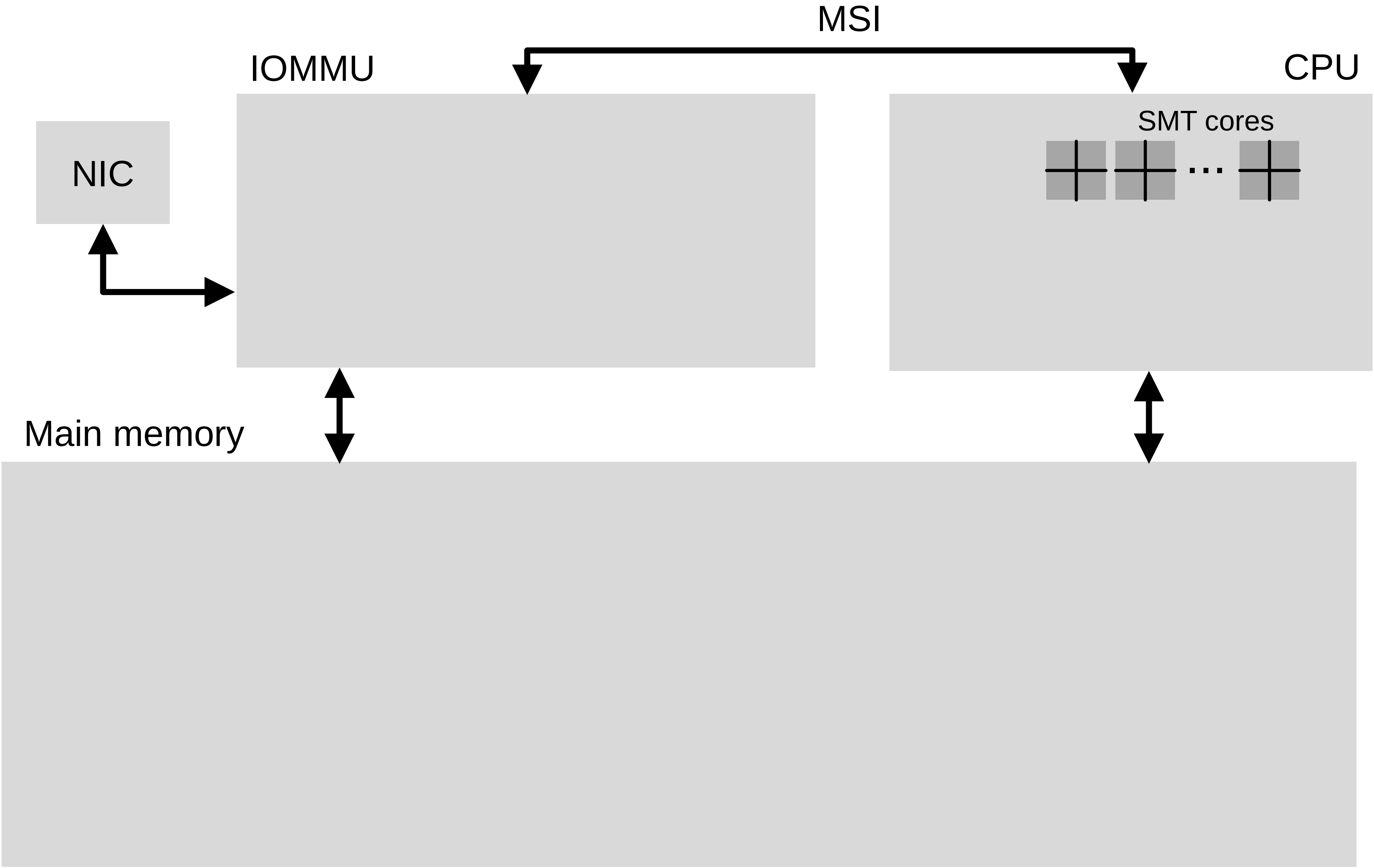
IOMMUs and RMA



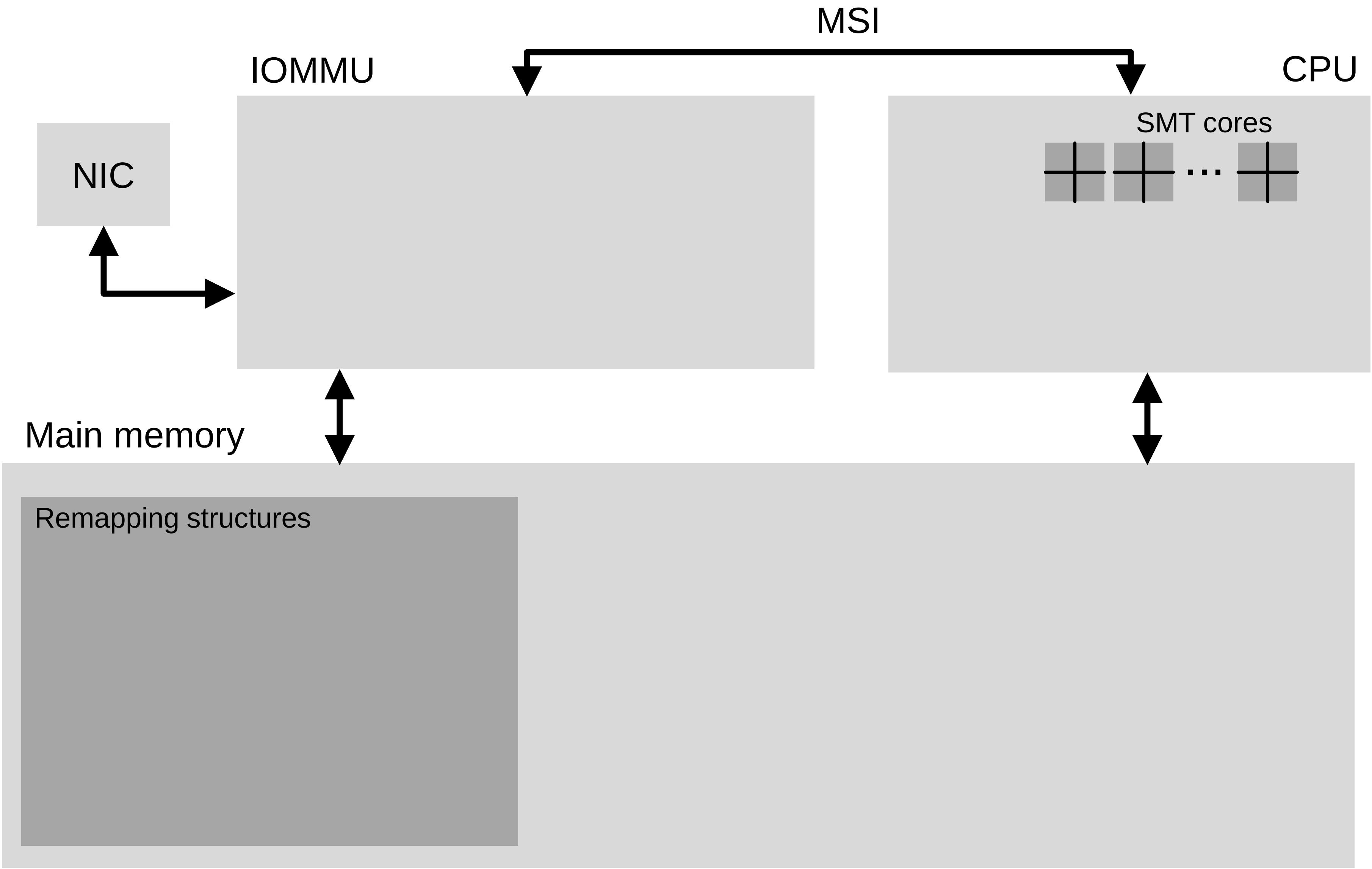
IOMMUs and RMA



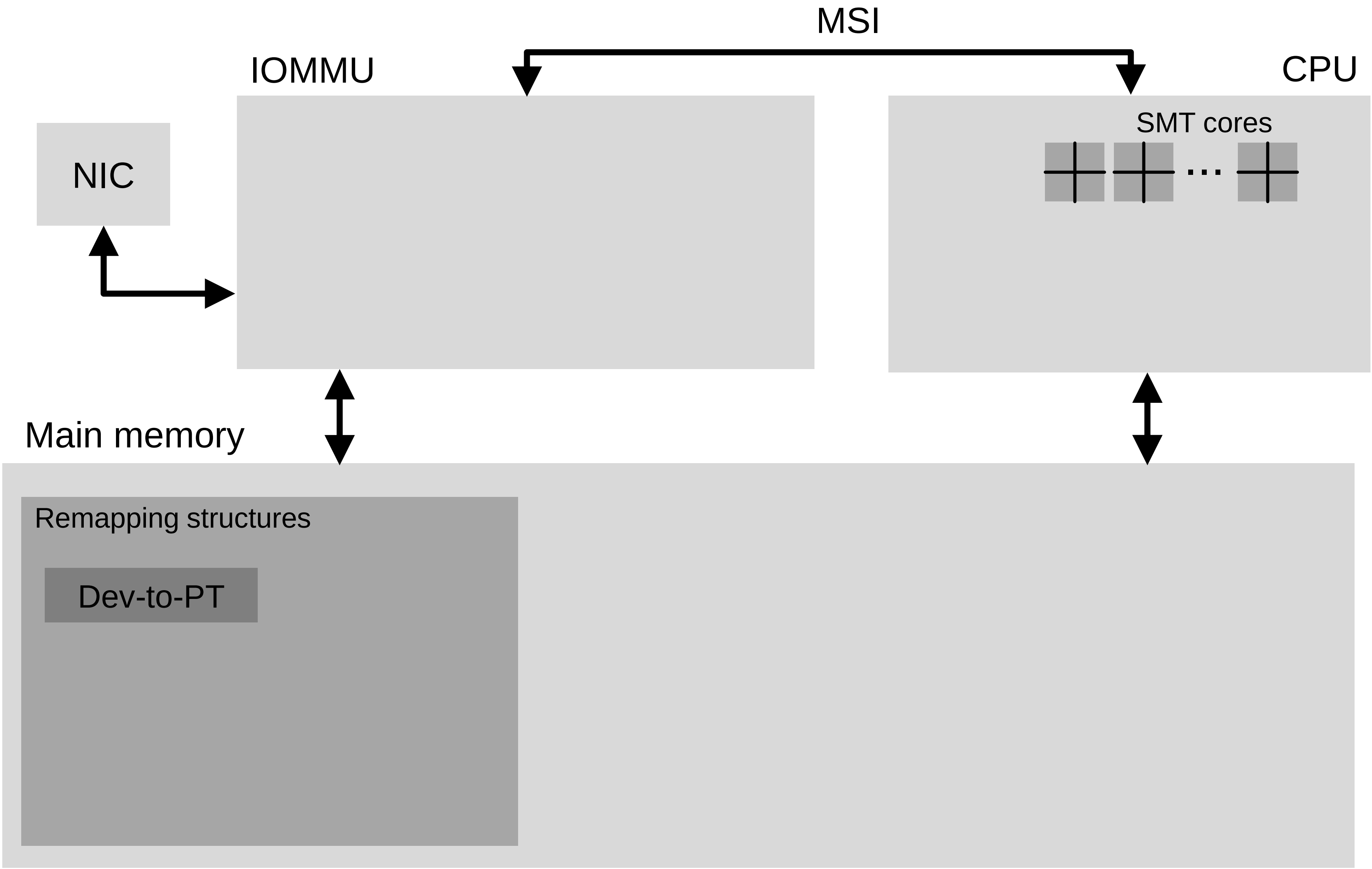
IOMMUs and RMA



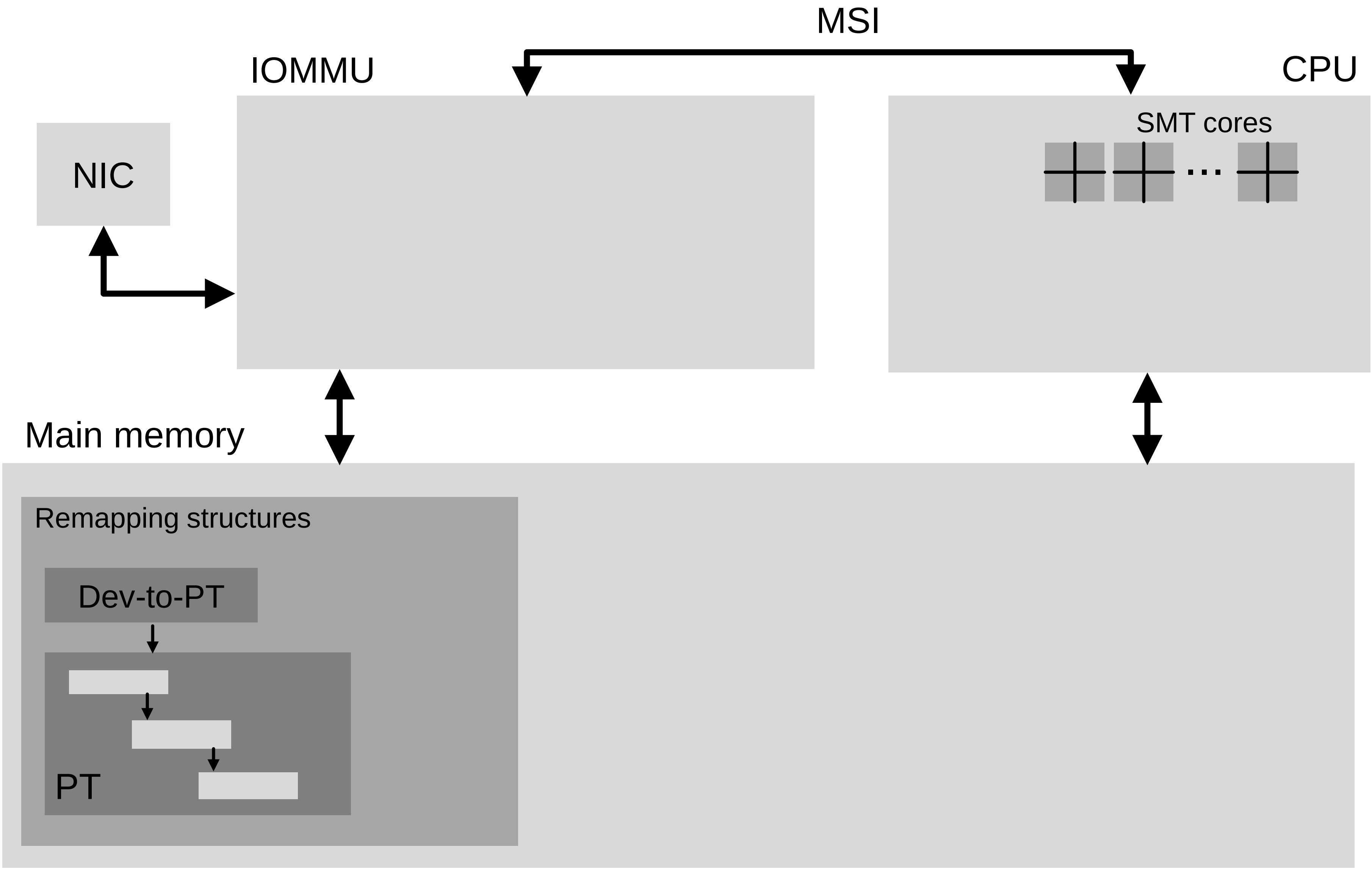
IOMMUs and RMA



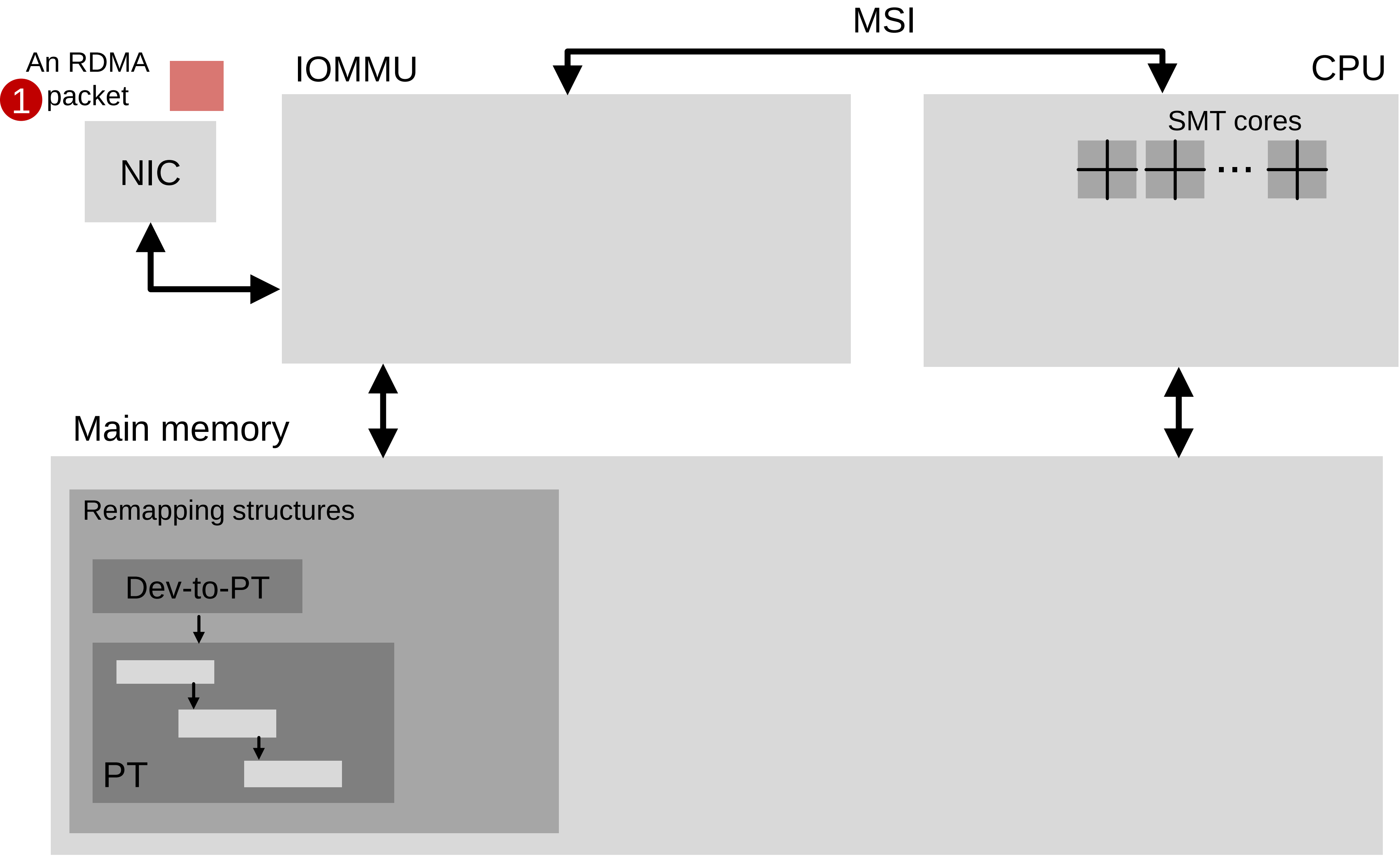
IOMMUs and RMA



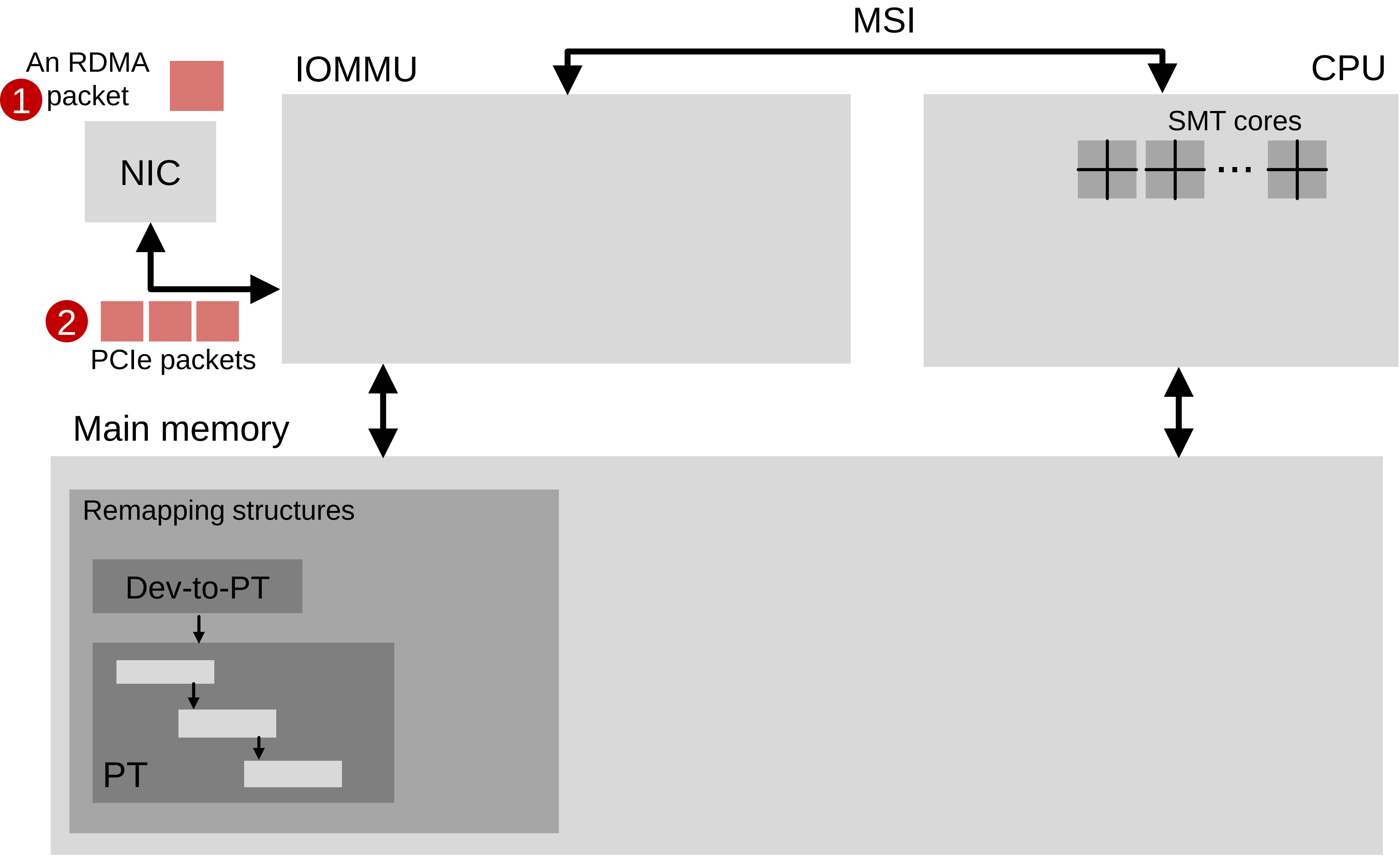
IOMMUs and RMA



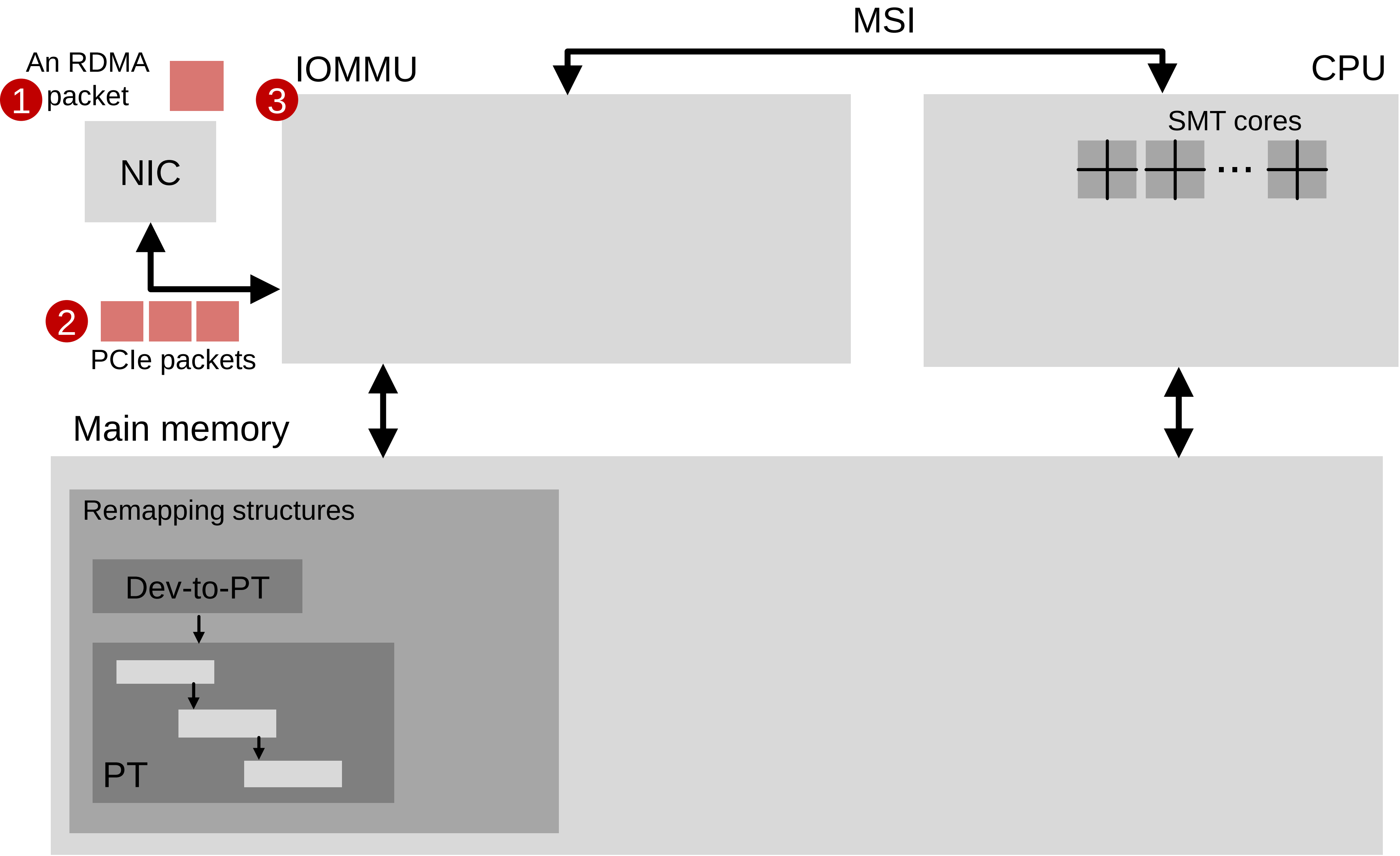
IOMMUs and RMA



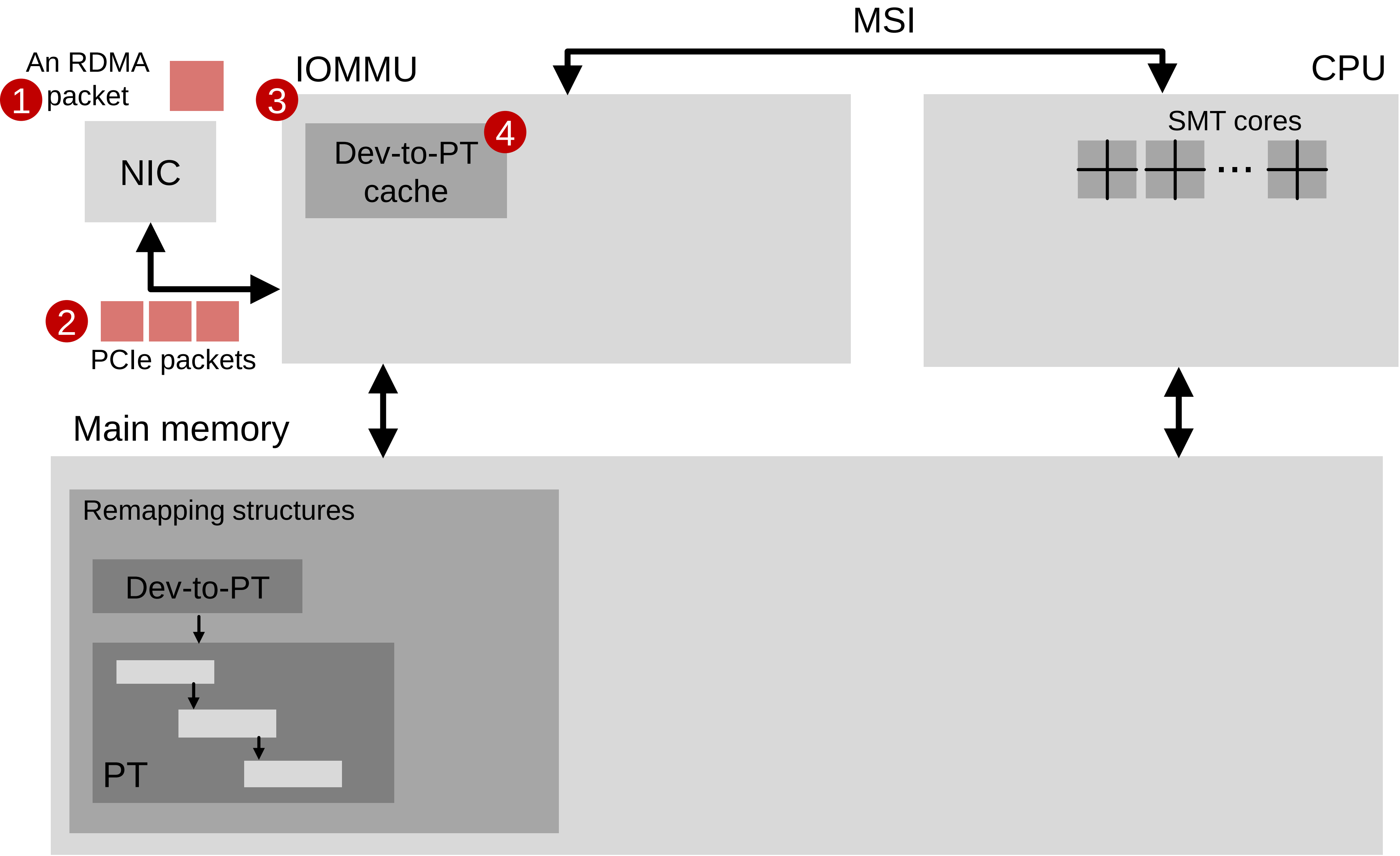
IOMMUs and RMA



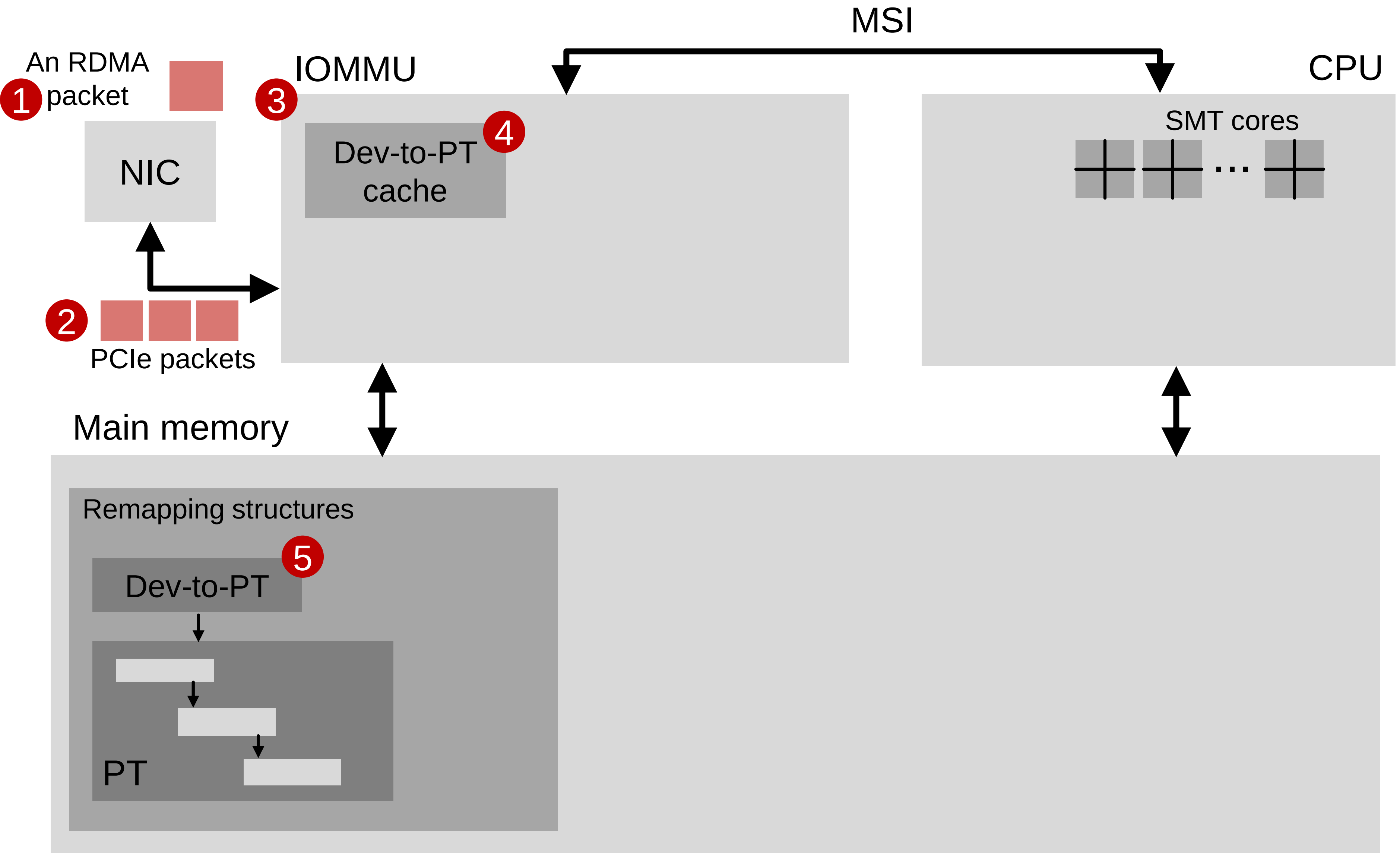
IOMMUs and RMA



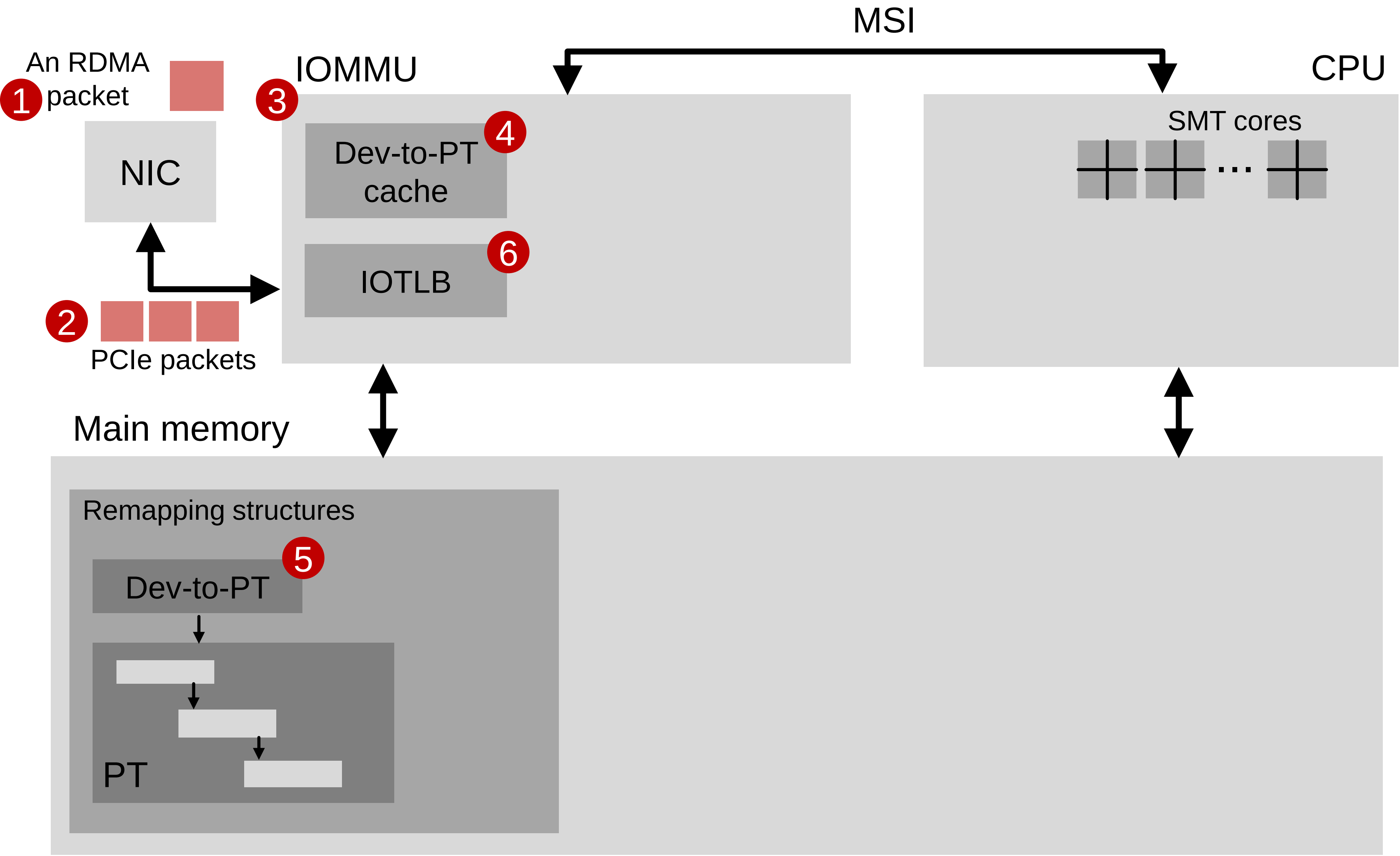
IOMMUs and RMA



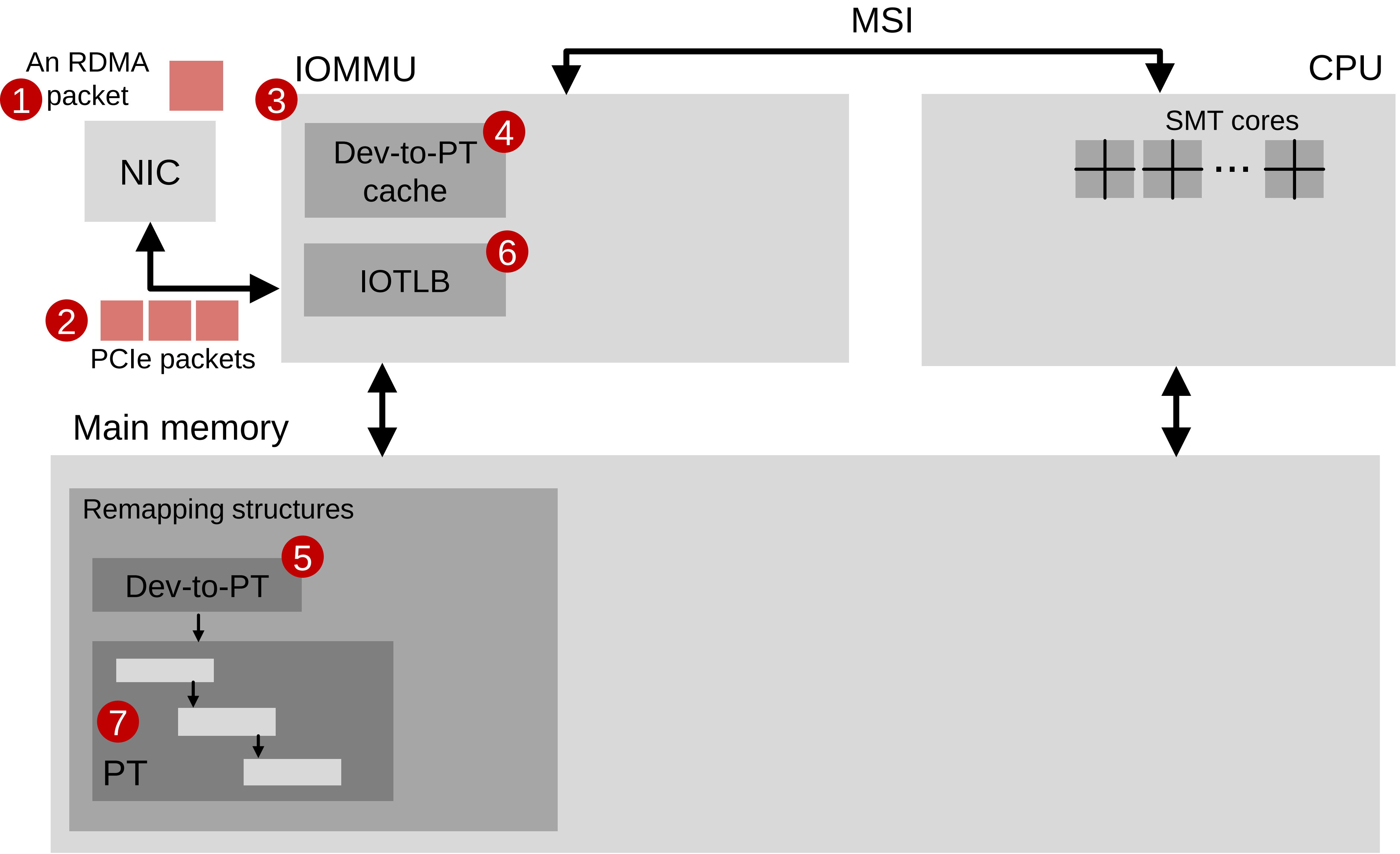
IOMMUs and RMA



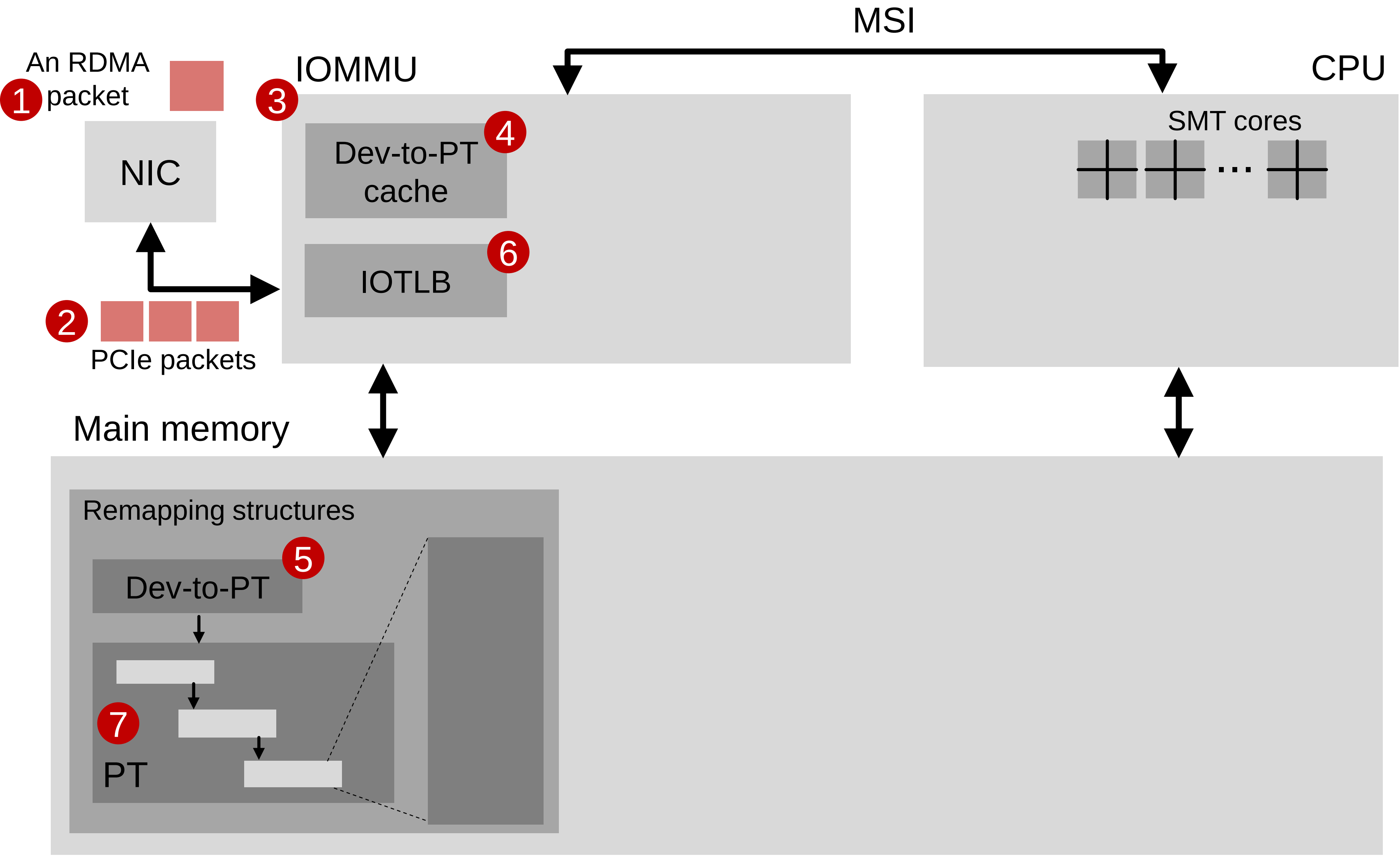
IOMMUs and RMA



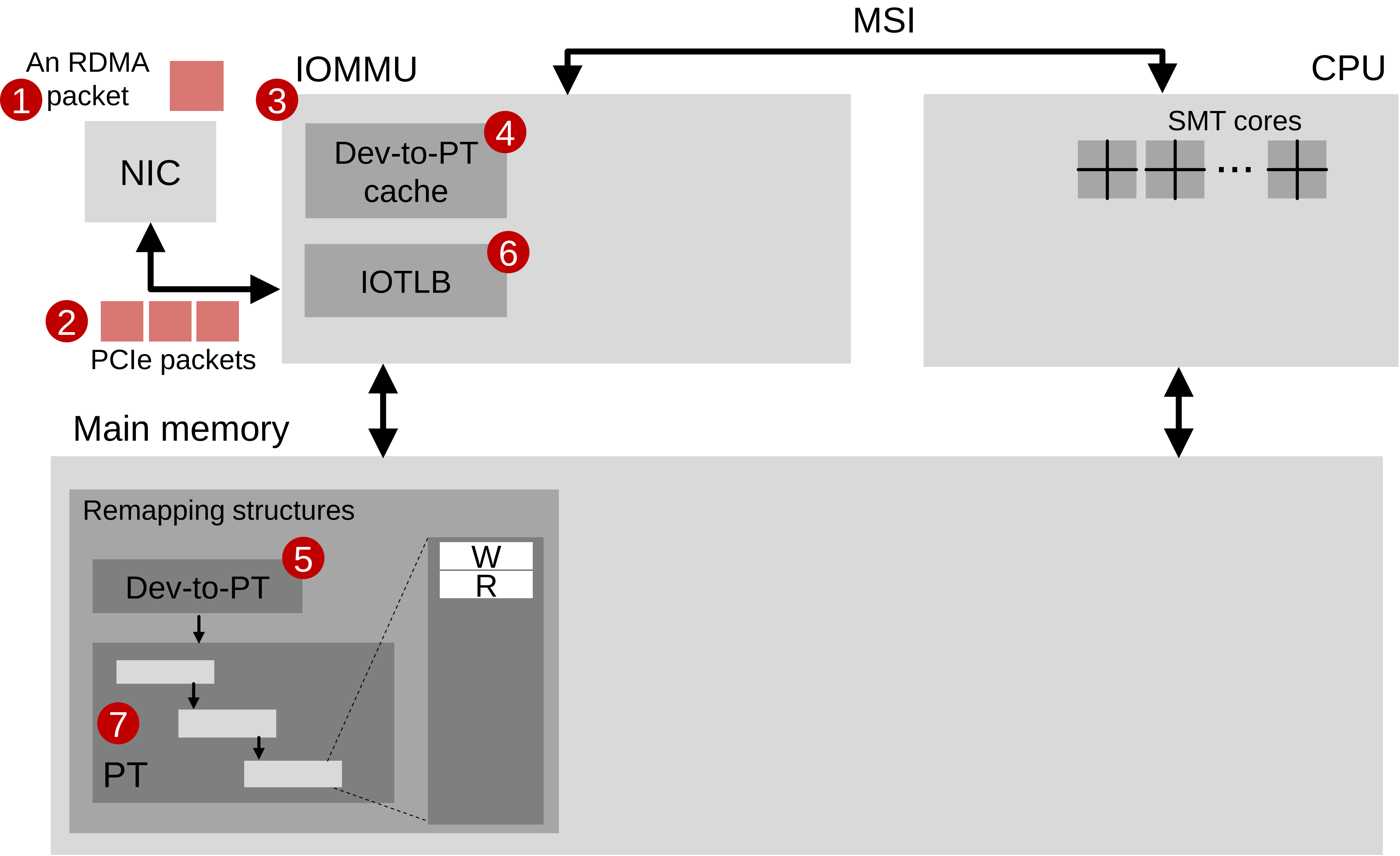
IOMMUs and RMA



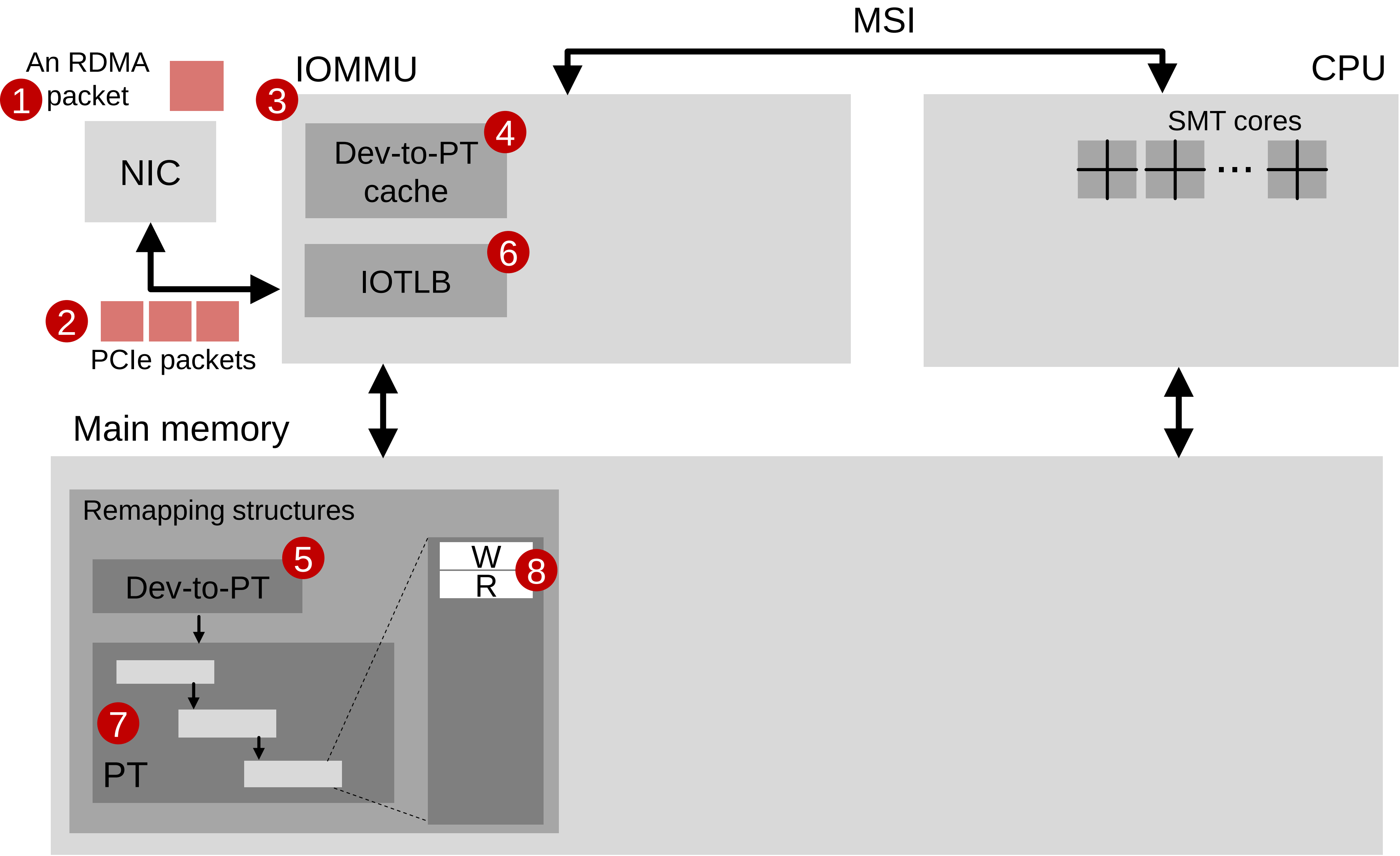
IOMMUs and RMA



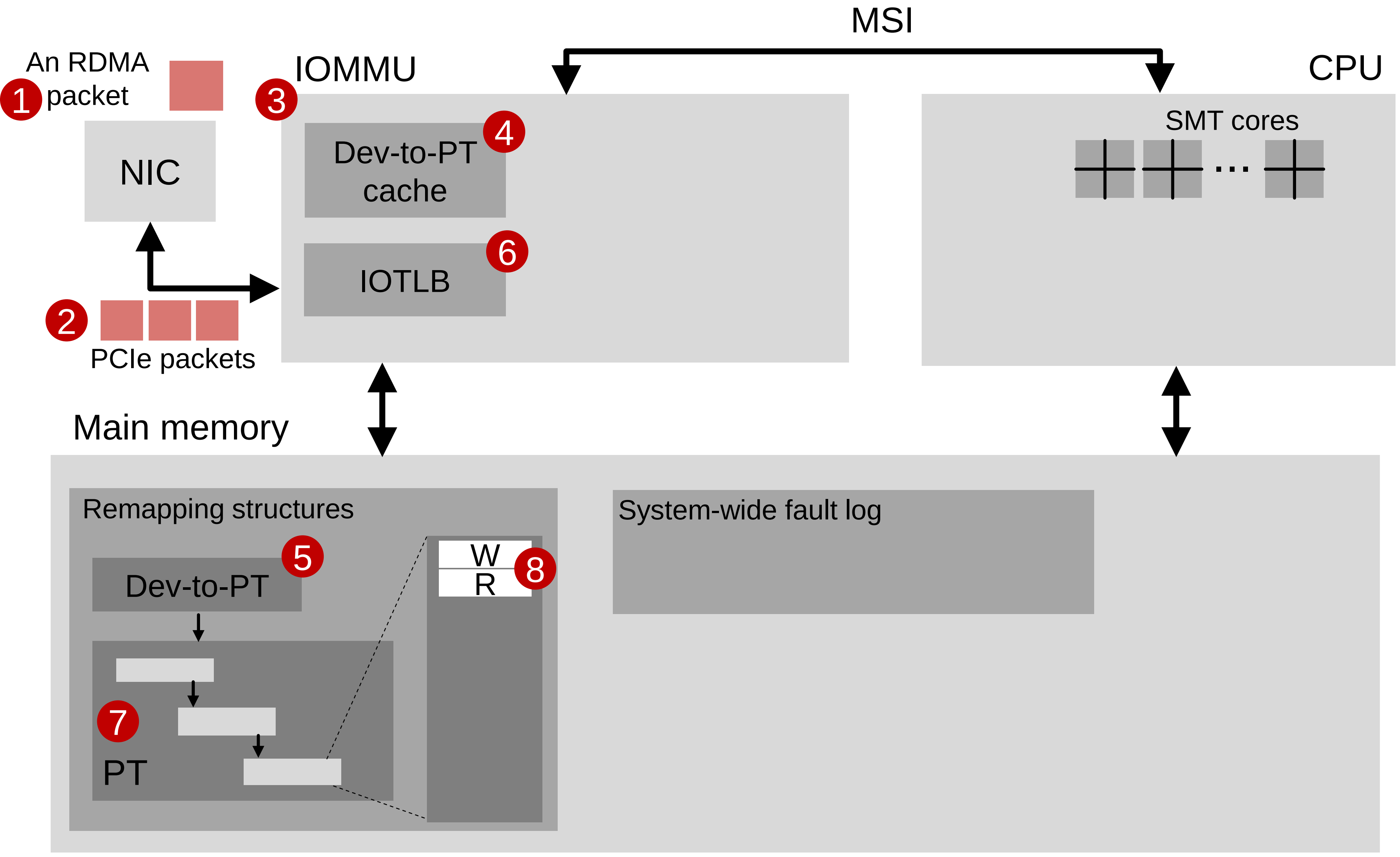
IOMMUs and RMA



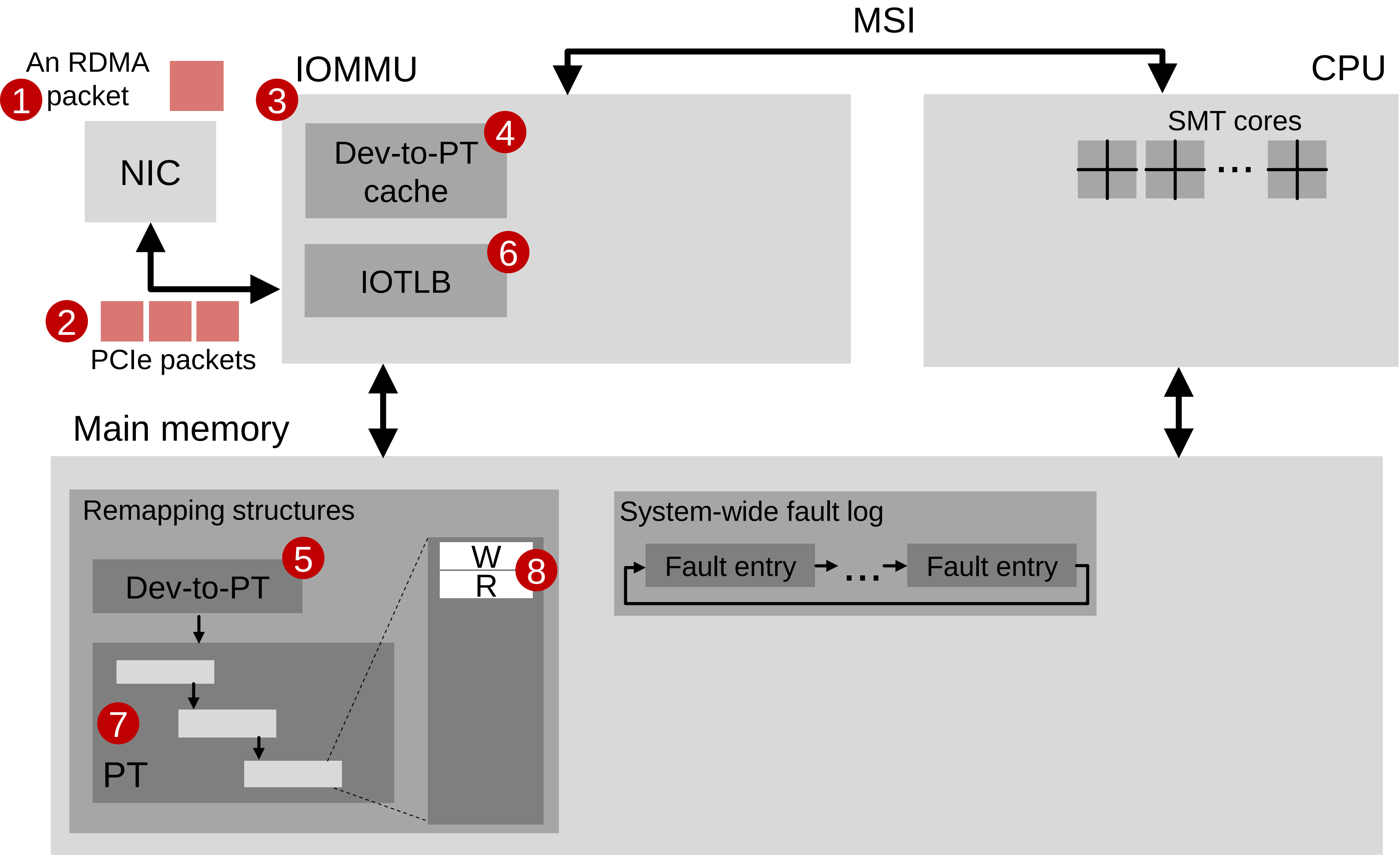
IOMMUs and RMA



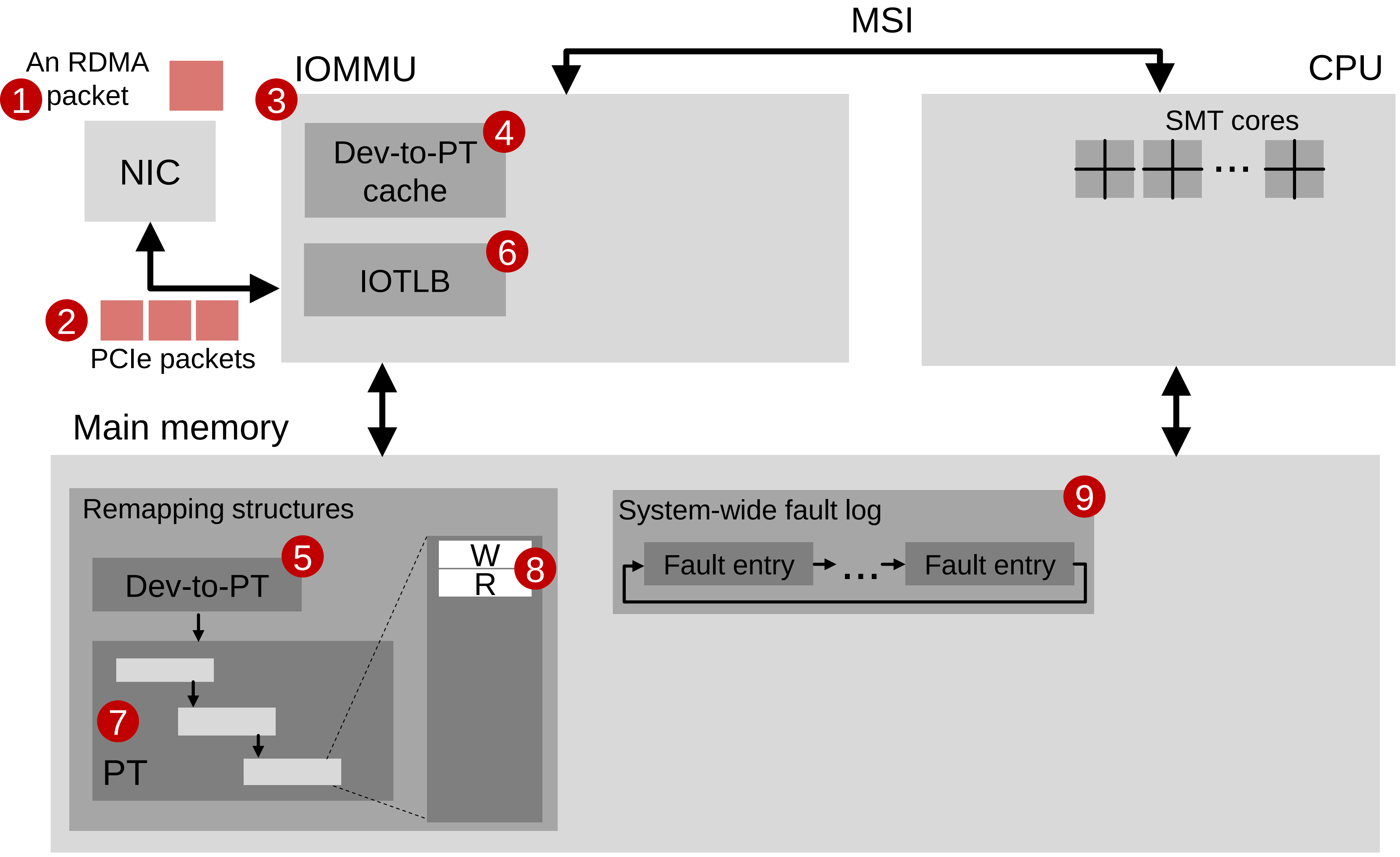
IOMMUs and RMA



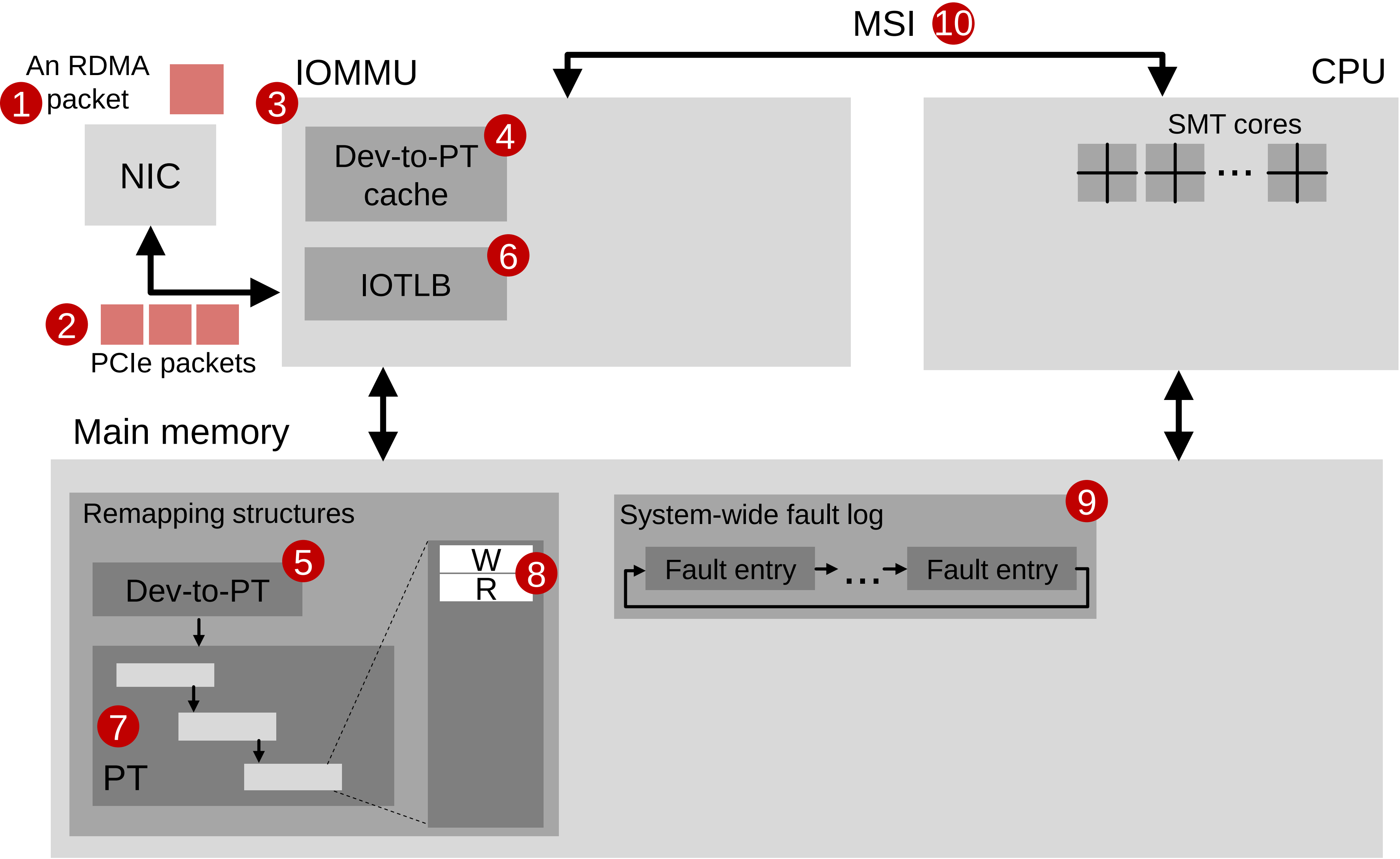
IOMMUs and RMA



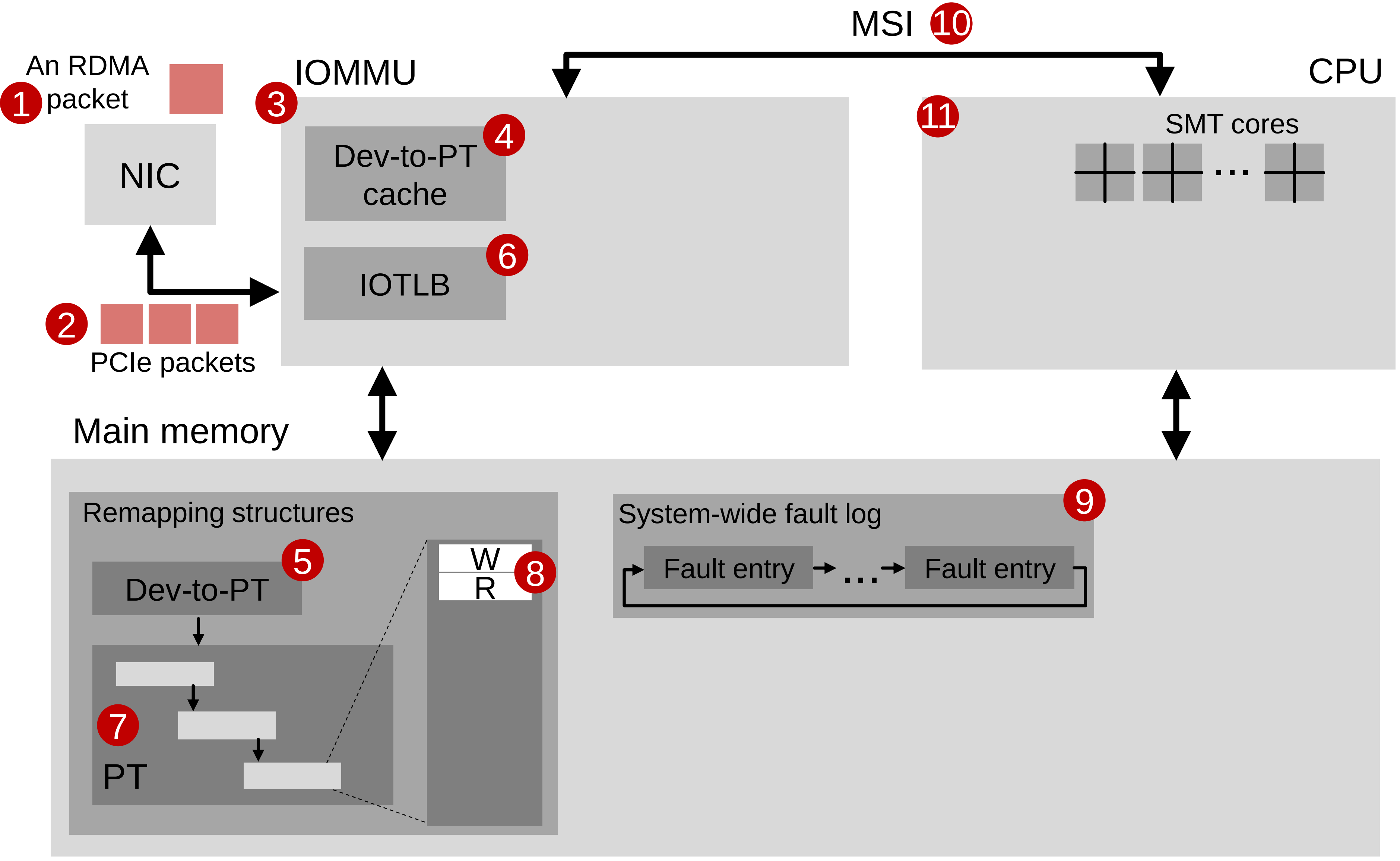
IOMMUs and RMA



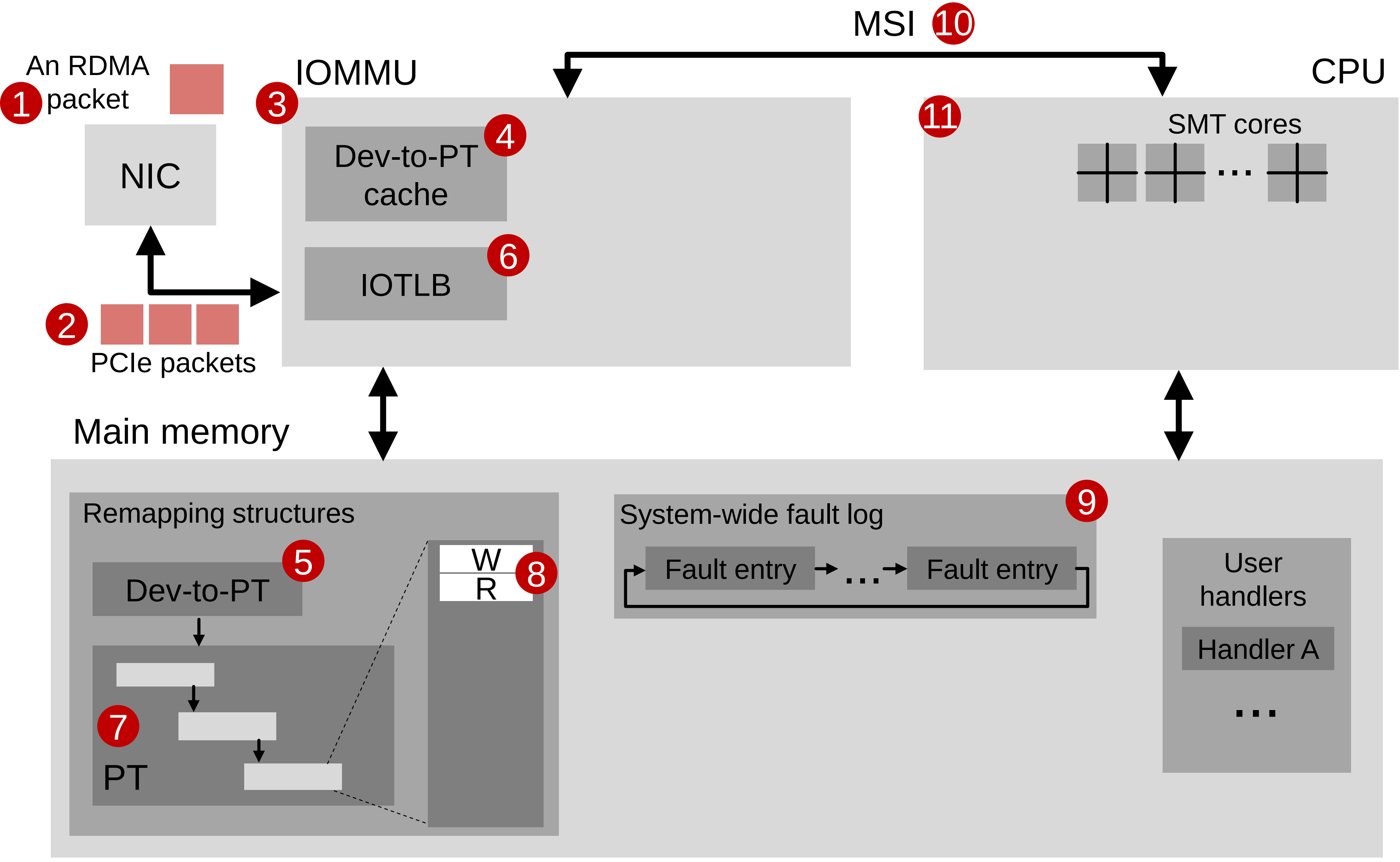
IOMMUs and RMA



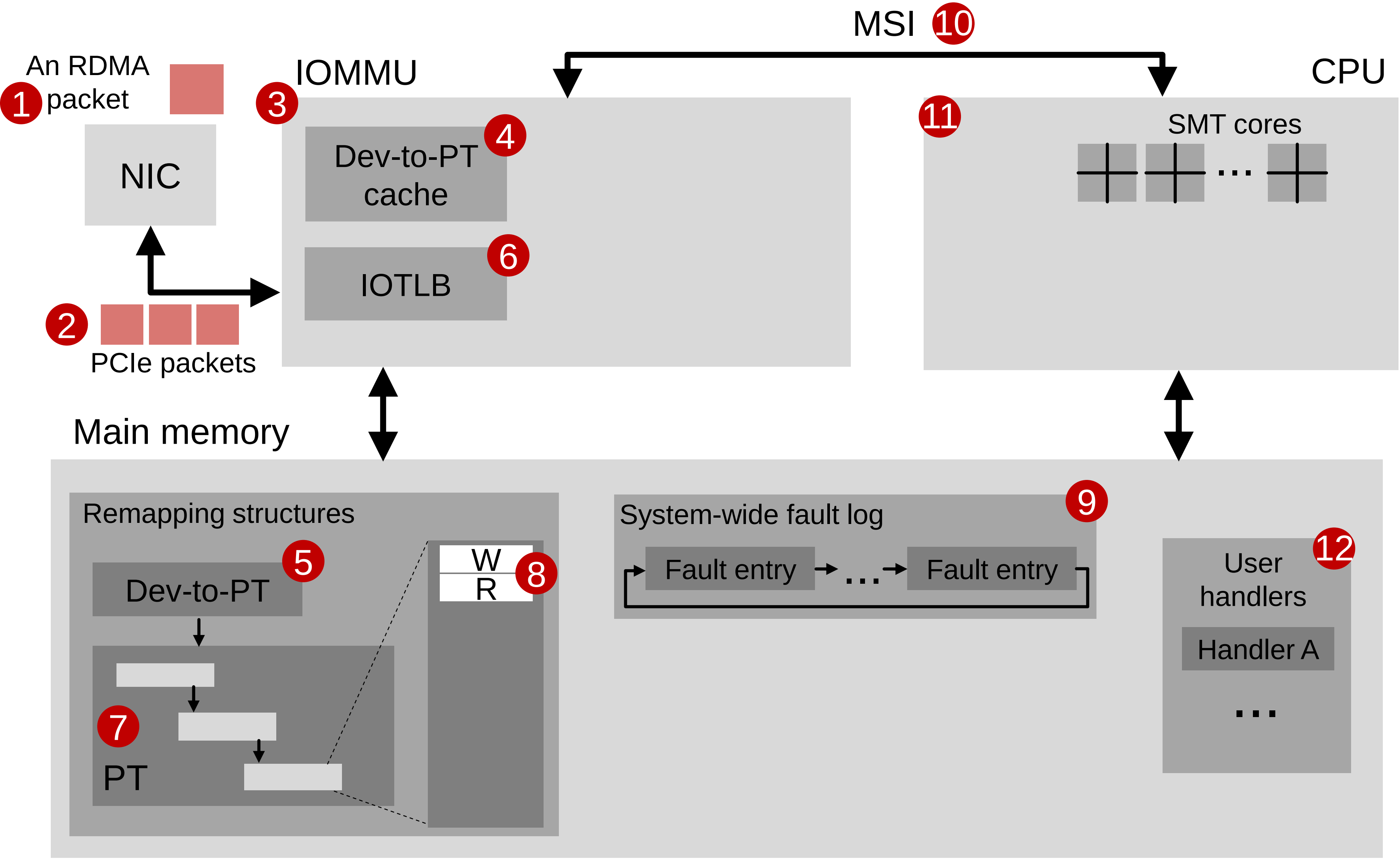
IOMMUs and RMA



IOMMUs and RMA

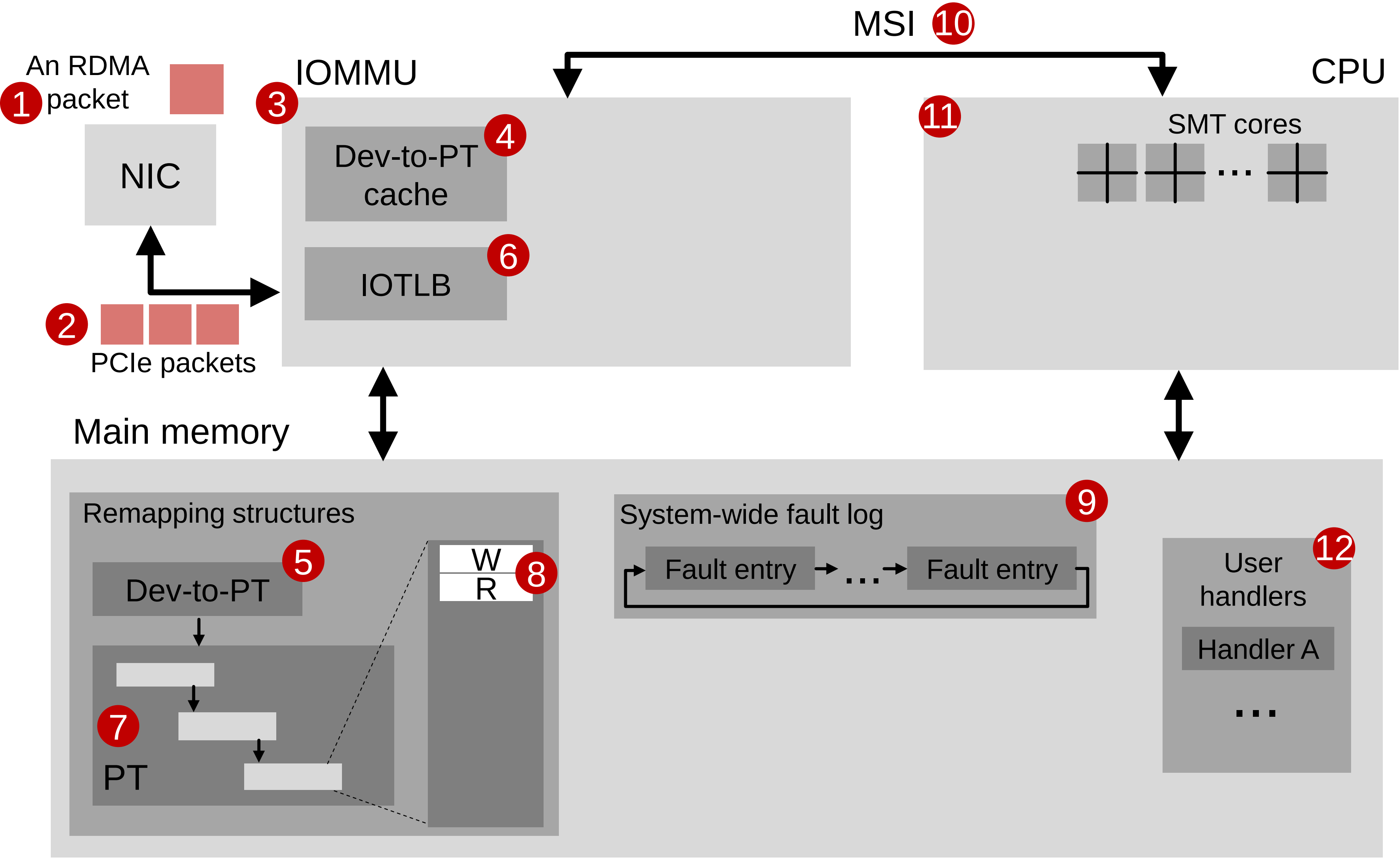


IOMMUs and RMA



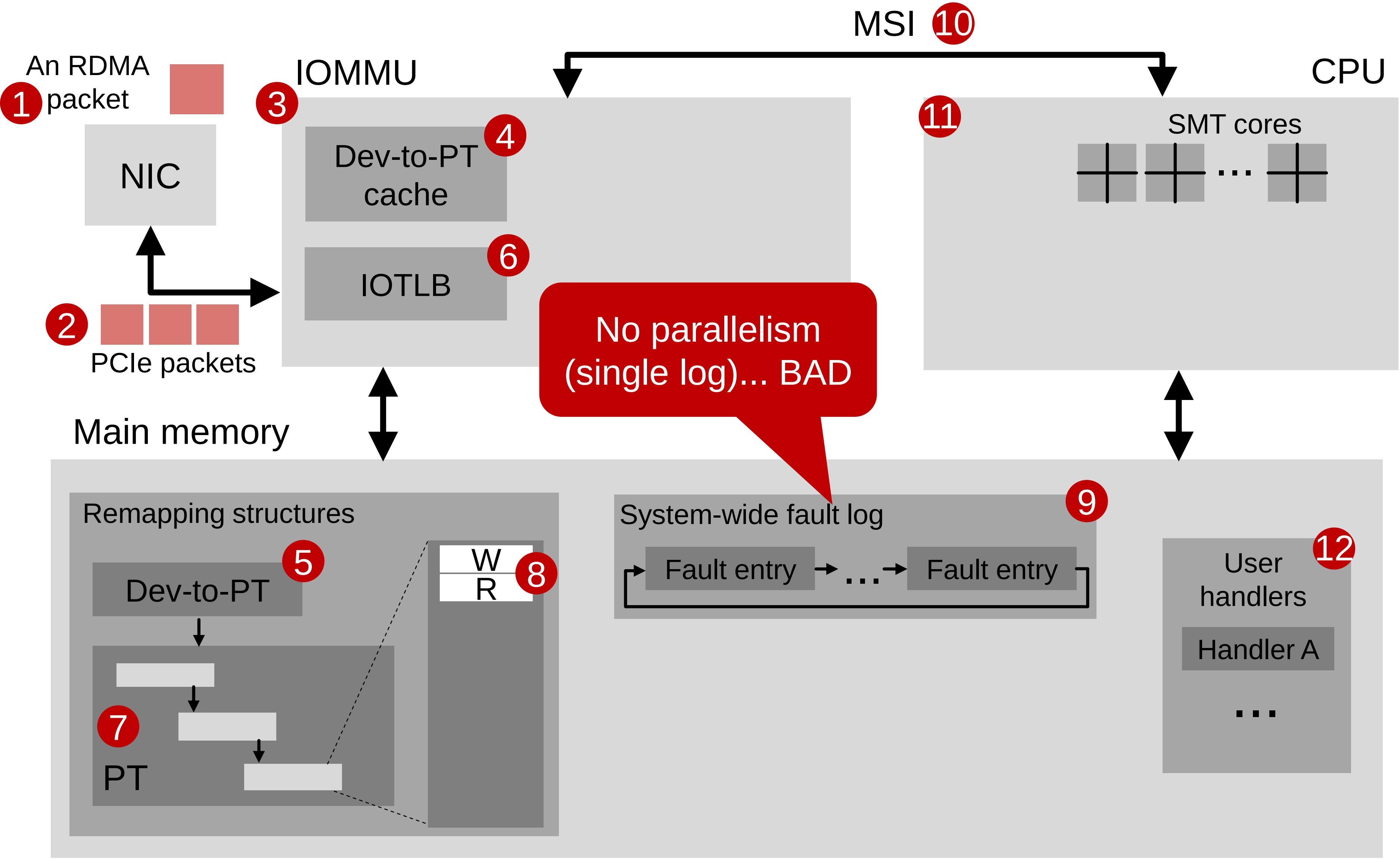
IOMMUs and RMA

We could use it somehow. But…



IOMMUs and RMA

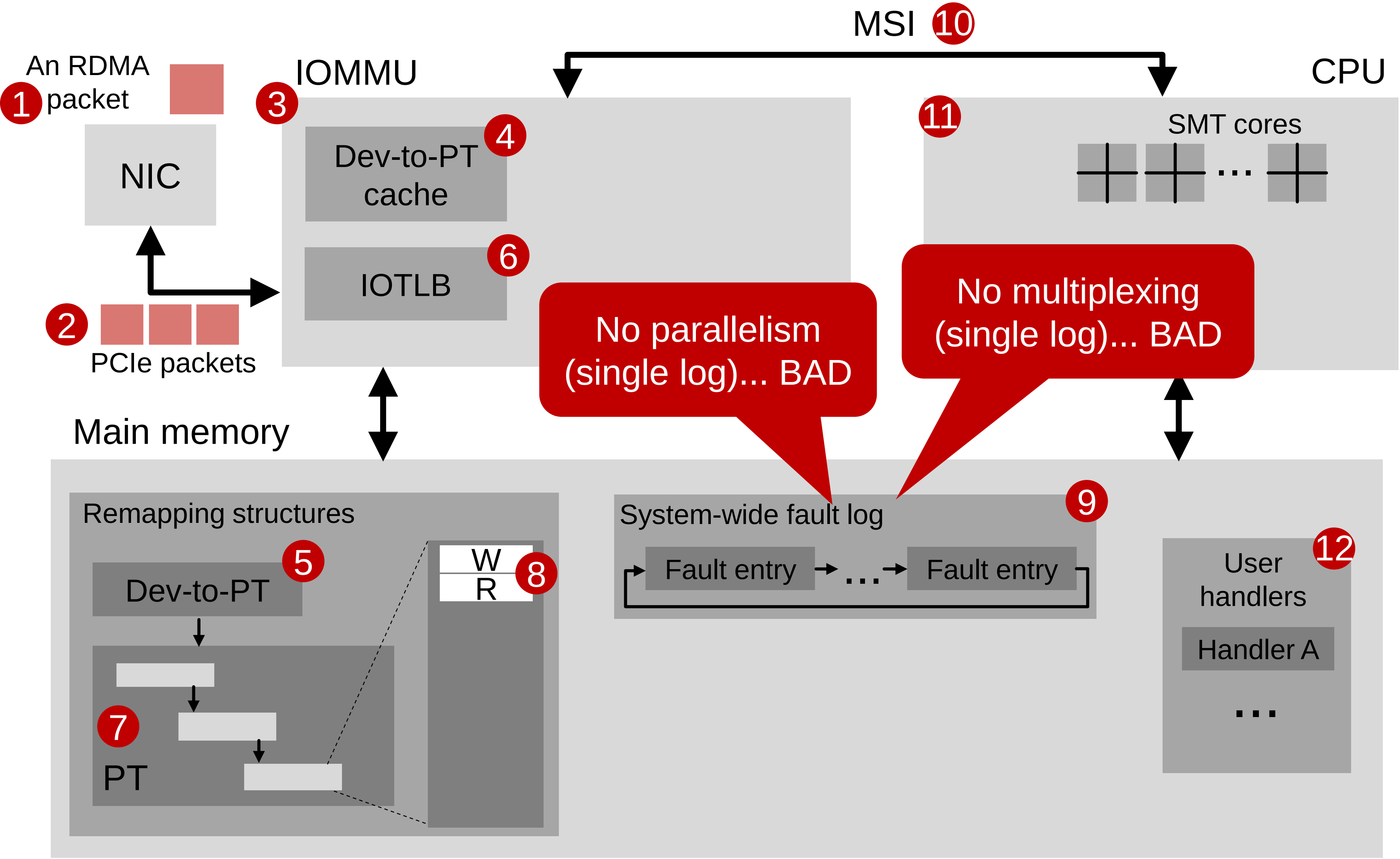
We could use it somehow. But…



No parallelism (single log)... BAD

IOMMUs and RMA

We could use it somehow. But…

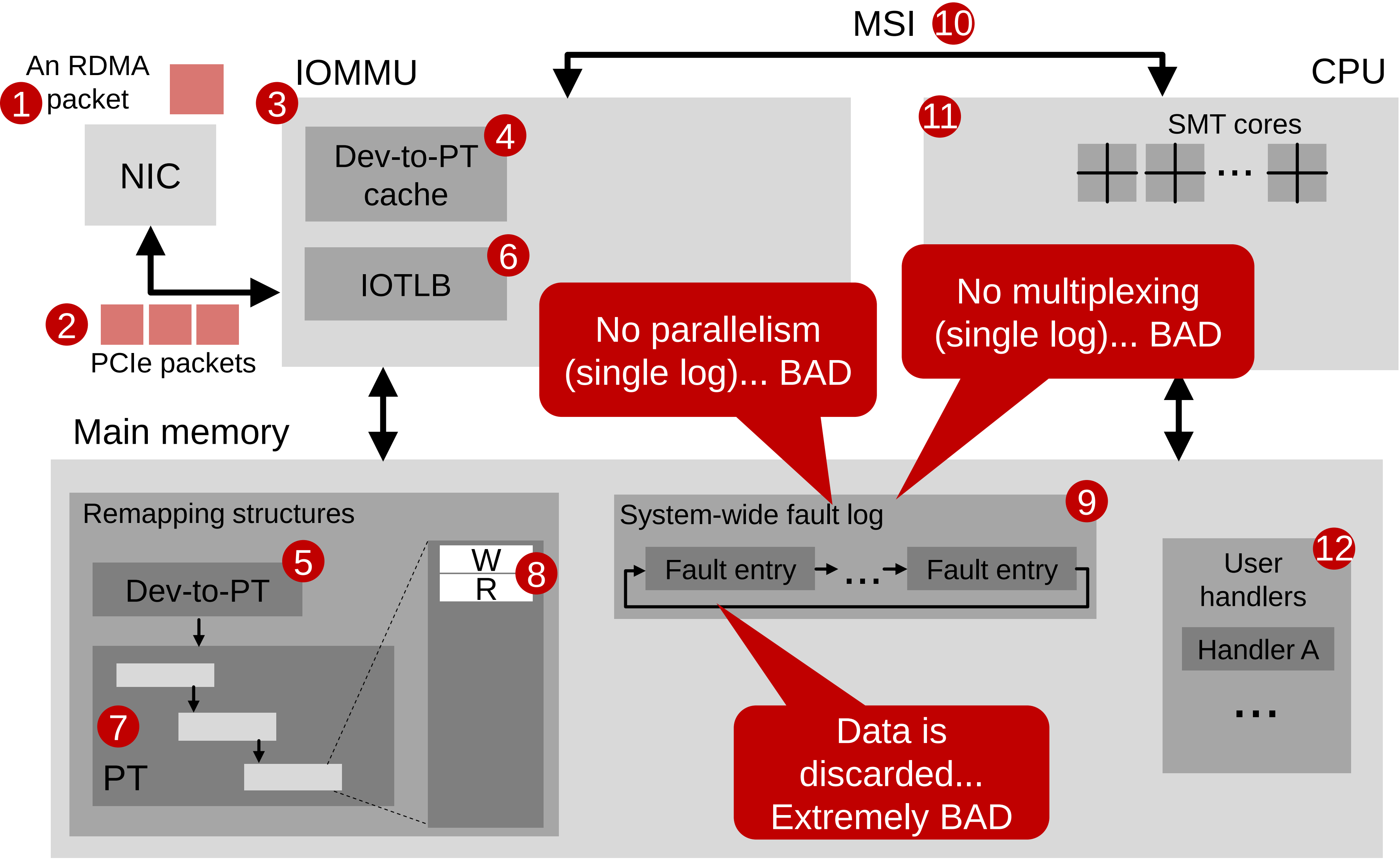


No parallelism (single log)... BAD

No multiplexing (single log)... BAD

IOMMUs and RMA

We could use it somehow. But…

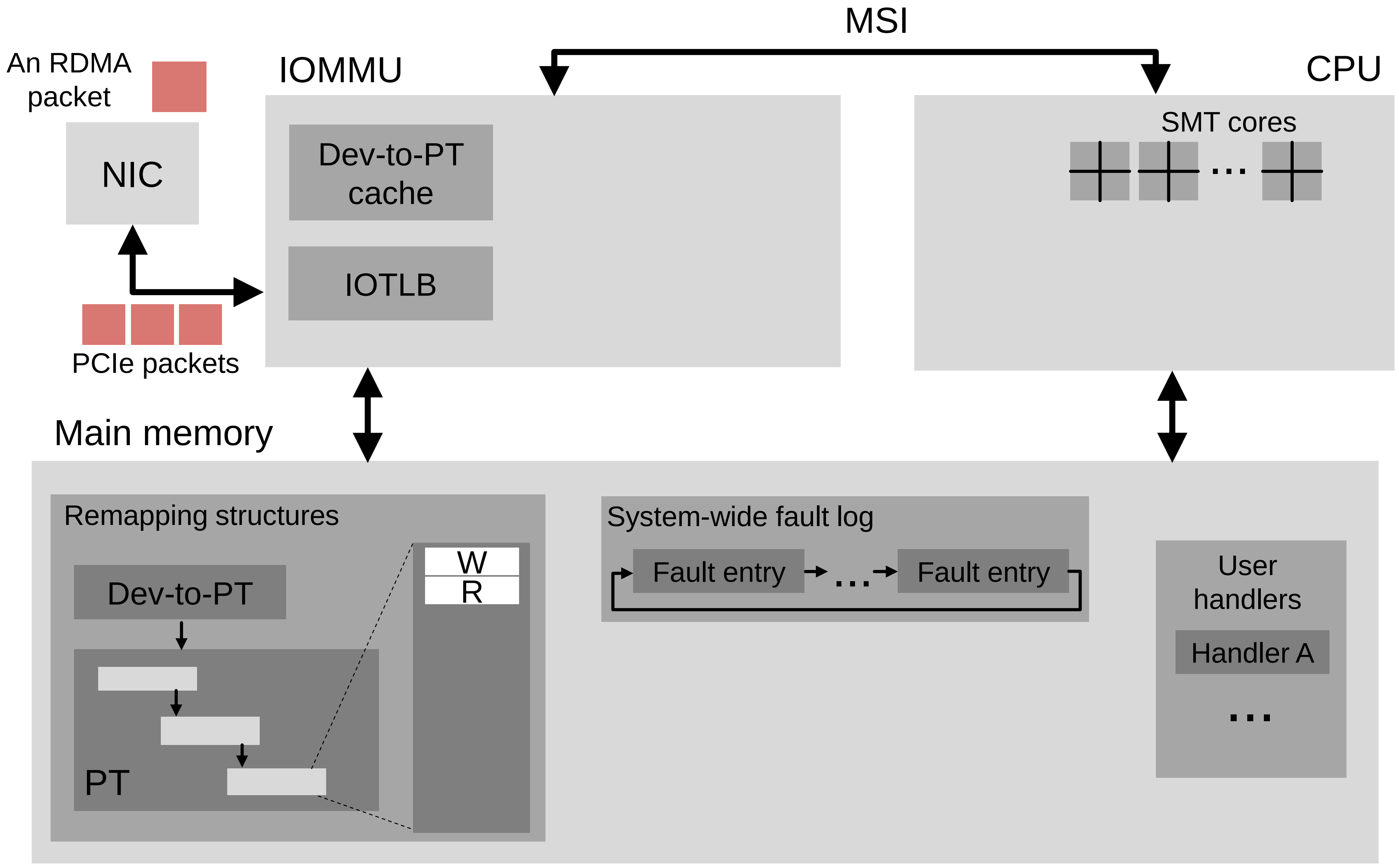


No parallelism (single log)... BAD

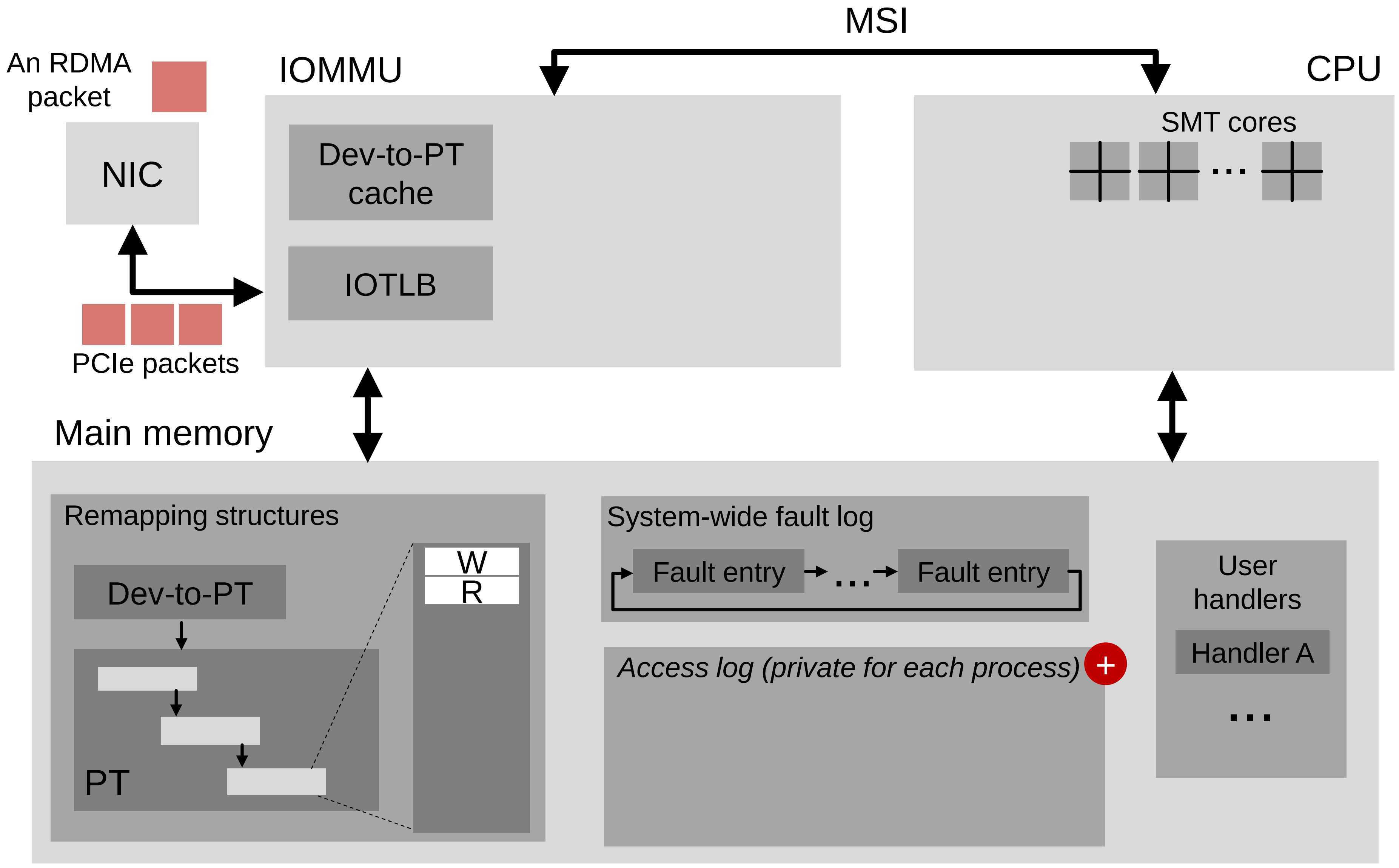
No multiplexing (single log)... BAD

Data is discarded... Extremely BAD

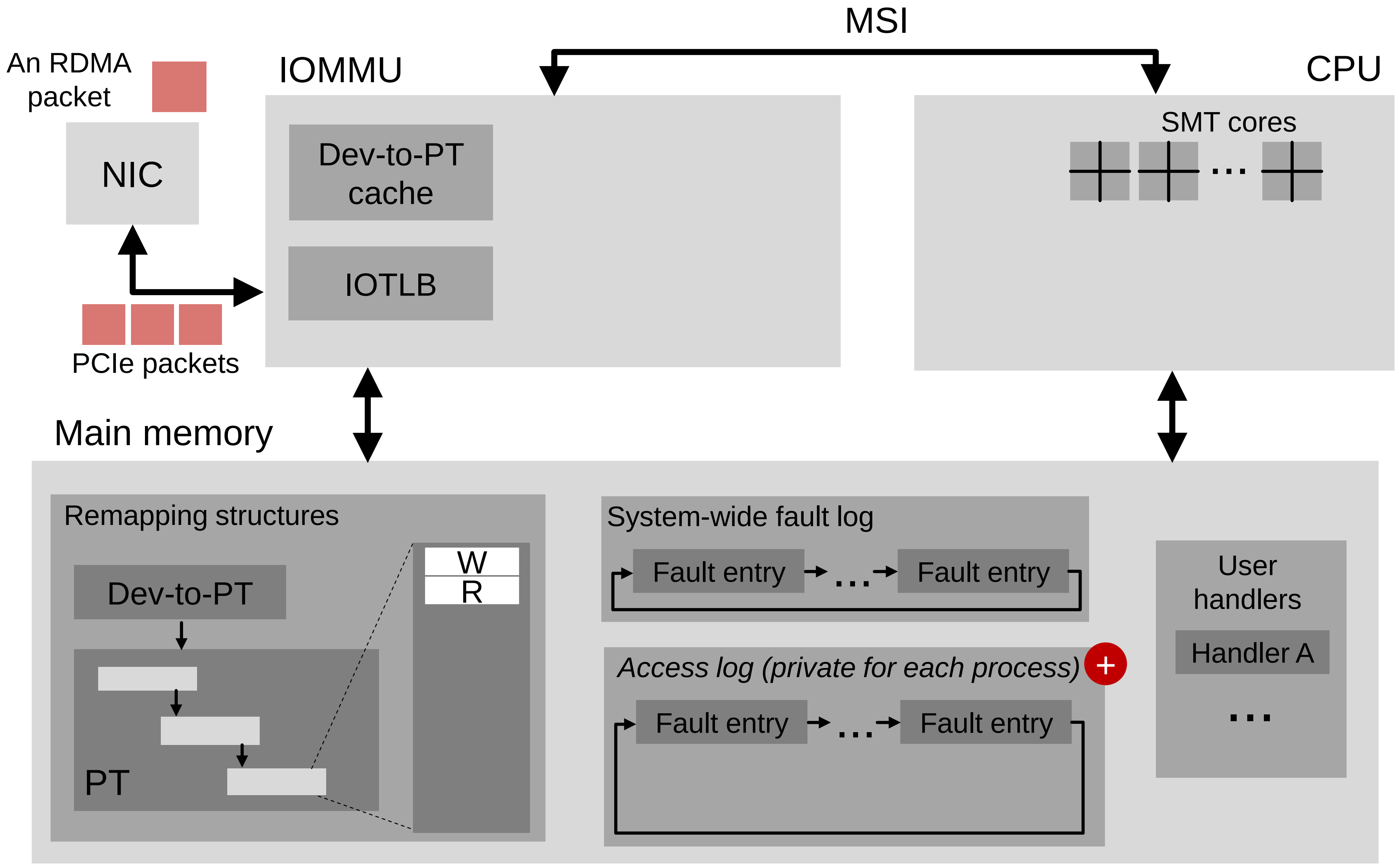
Active Puts


MSI
An RDMA packet
IOMMU
CPU
NIC
Dev-to-PT cache
IOTLB
SMT cores
...
PCIe packets
Main memory
Remapping structures
Dev-to-PT
W
R
PT
System-wide fault log
Fault entry
...
Fault entry
User handlers
Handler A
...

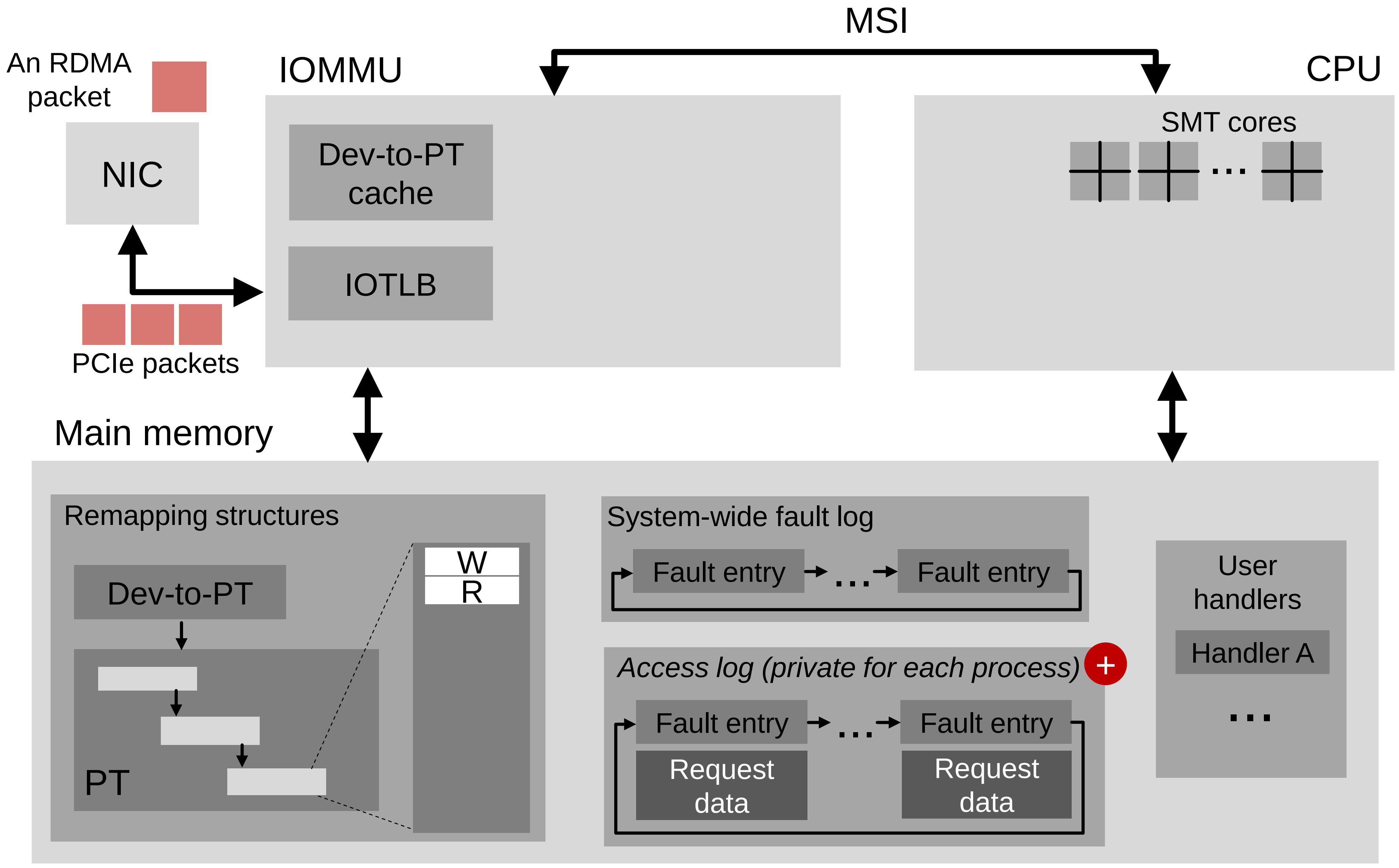
Active Puts


MSI
An RDMA packet
NIC
PCIe packets
IOMMU
Dev-to-PT cache
IOTLB
CPU
SMT cores
...
Main memory
Remapping structures
Dev-to-PT
PT
W
R
System-wide fault log
Fault entry
...
Fault entry
Access log (private for each process)
+
User handlers
Handler A
...

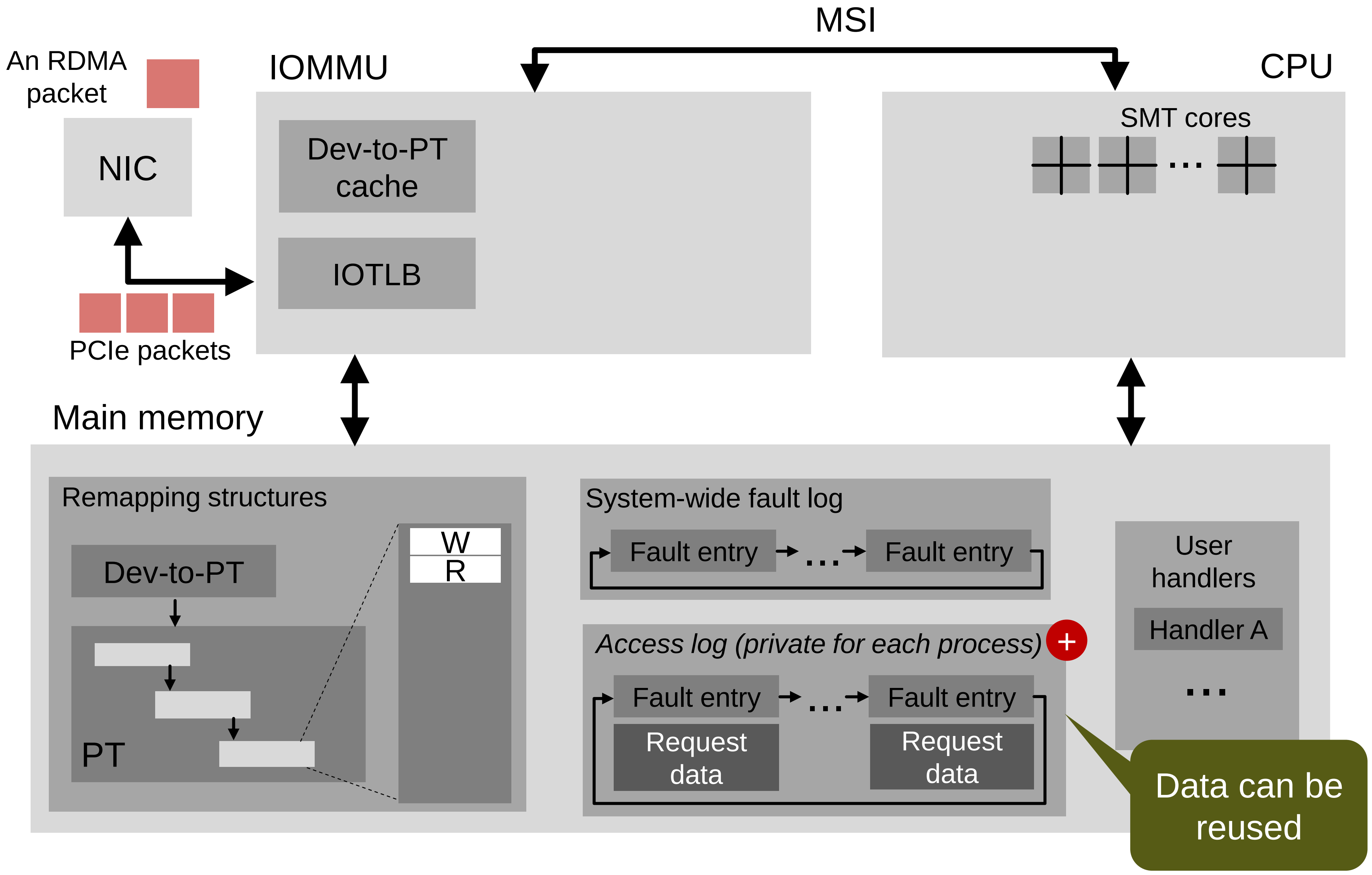
ACTIVE PUTS


MSI
An RDMA packet
IOMMU
CPU
NIC
Dev-to-PT cache
IOTLB
SMT cores
...
PCIe packets
Main memory
Remapping structures
Dev-to-PT
W
R
PT
System-wide fault log
Fault entry
...
Fault entry
Access log (private for each process)
+
Fault entry
...
Fault entry
User handlers
Handler A
...

Active Puts

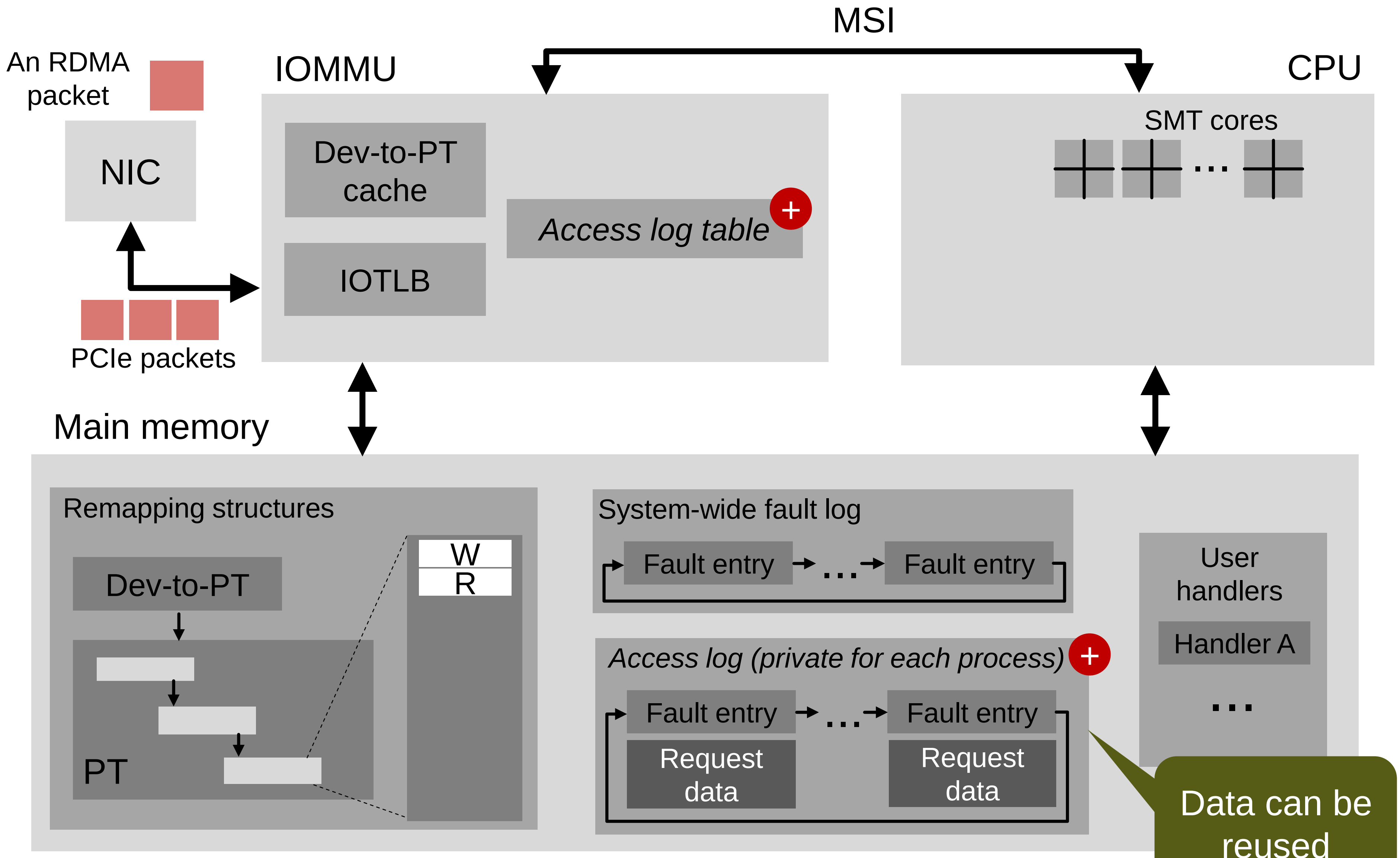

MSI
An RDMA packet
IOMMU
CPU
NIC
Dev-to-PT cache
IOTLB
SMT cores
...
PCIe packets
Main memory
Remapping structures
W
R
Dev-to-PT
PT
System-wide fault log
Fault entry
...
Fault entry
Access log (private for each process)
+
Fault entry
...
Fault entry
Request data
Request data
User handlers
Handler A
...

Active Puts



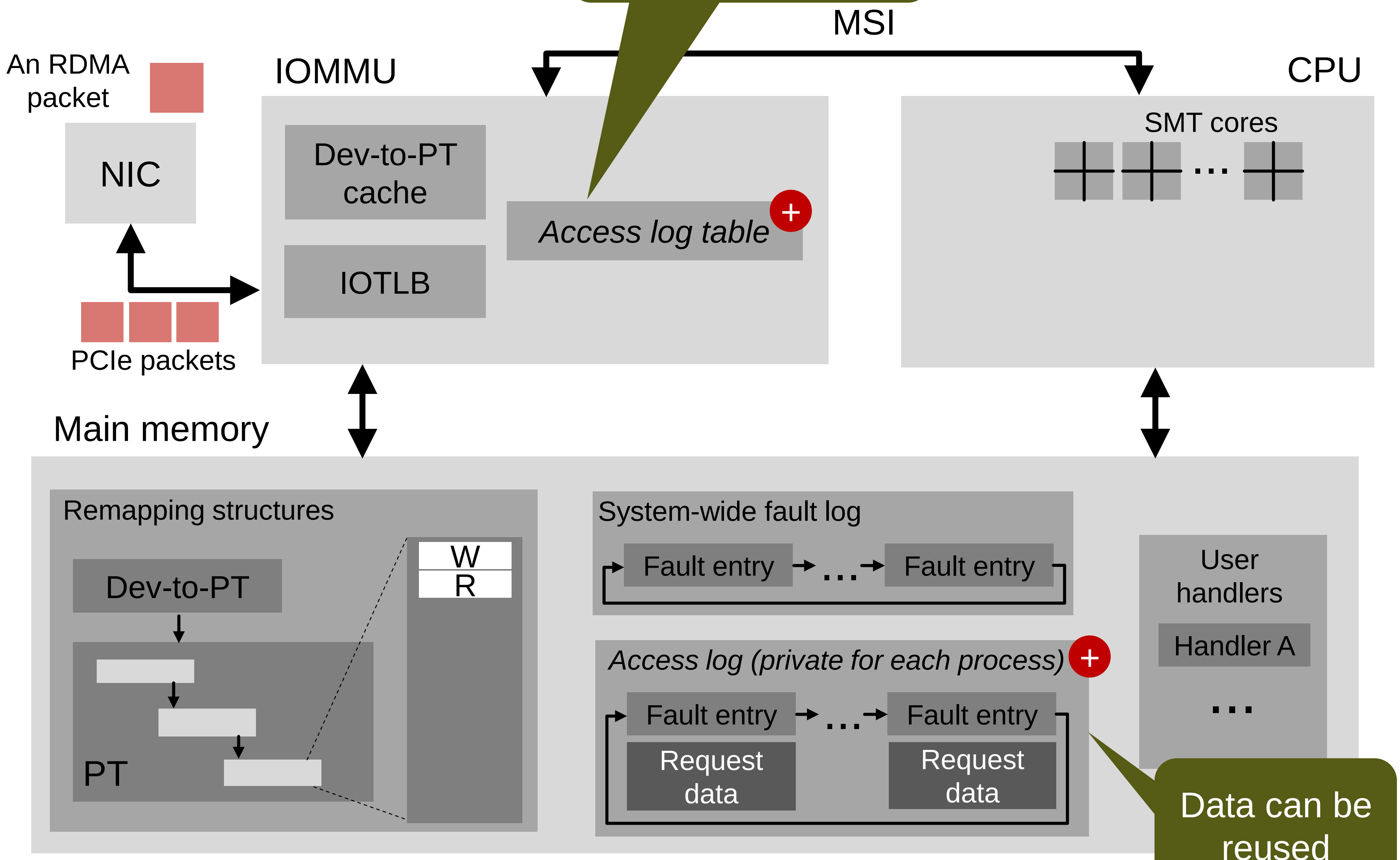
Data can be reused

Active Puts



Data can be reused

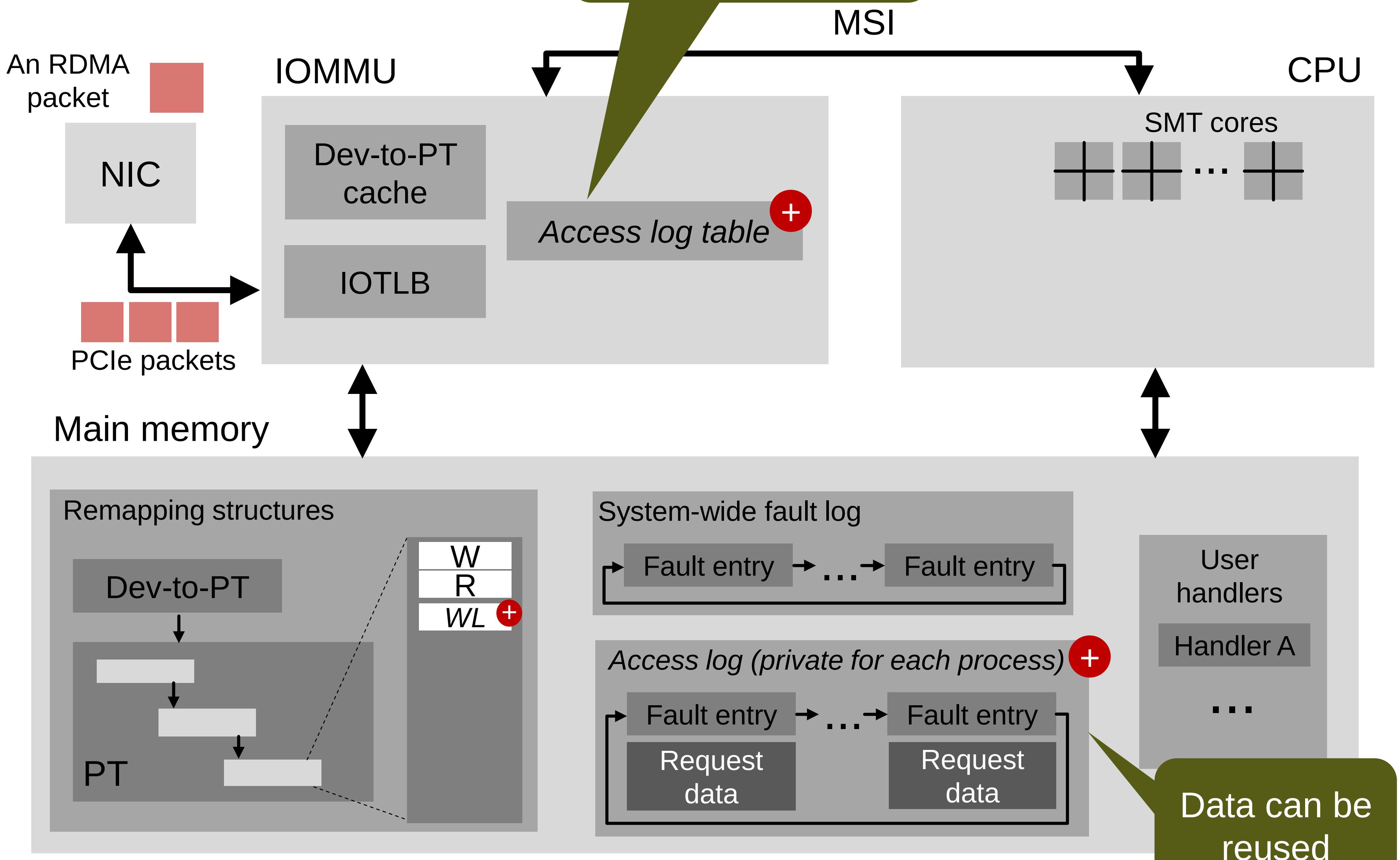
Active Puts



Stores addresses of each access log

Data can be reused

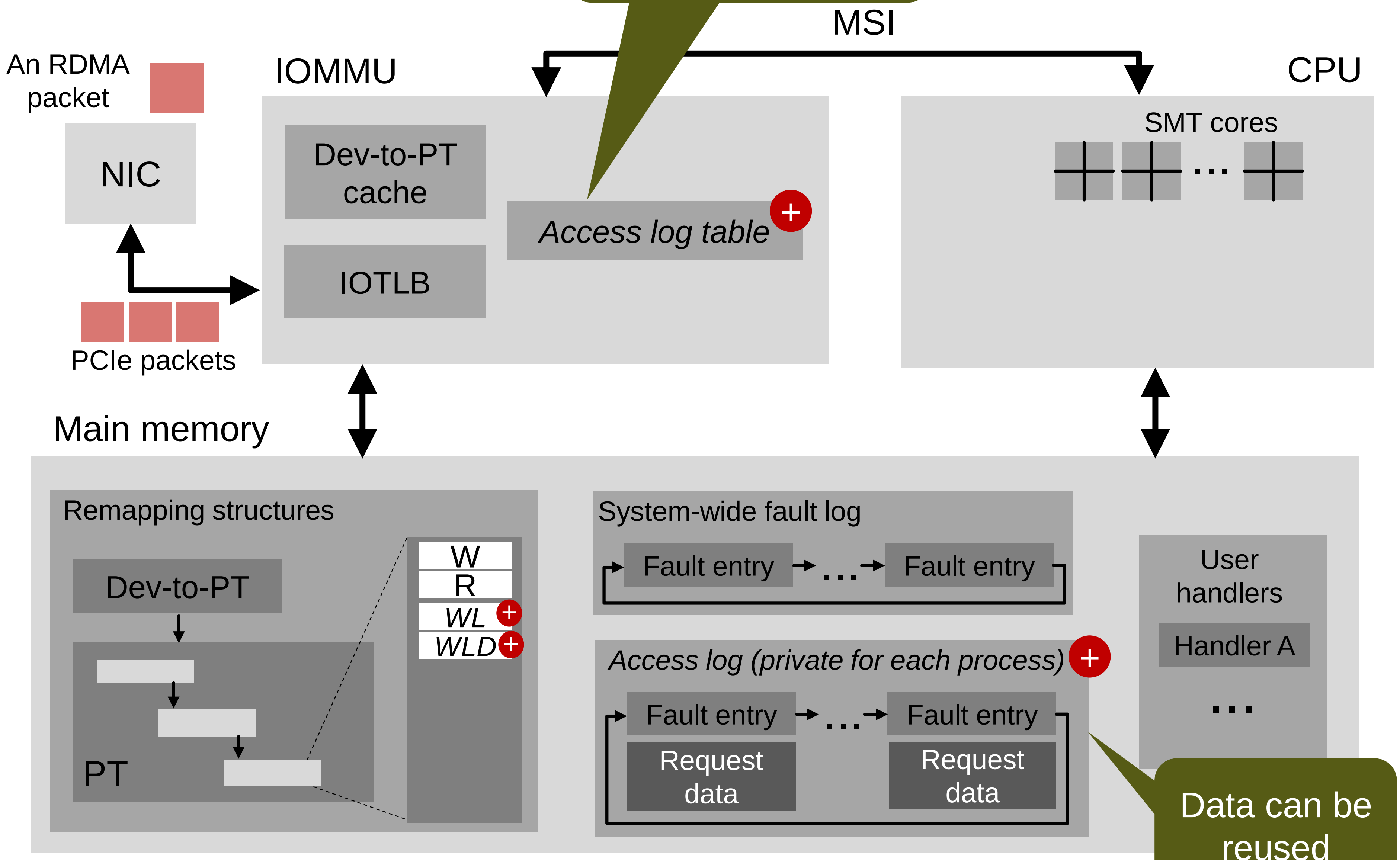
Active Puts



Stores addresses of each access log

Data can be reused

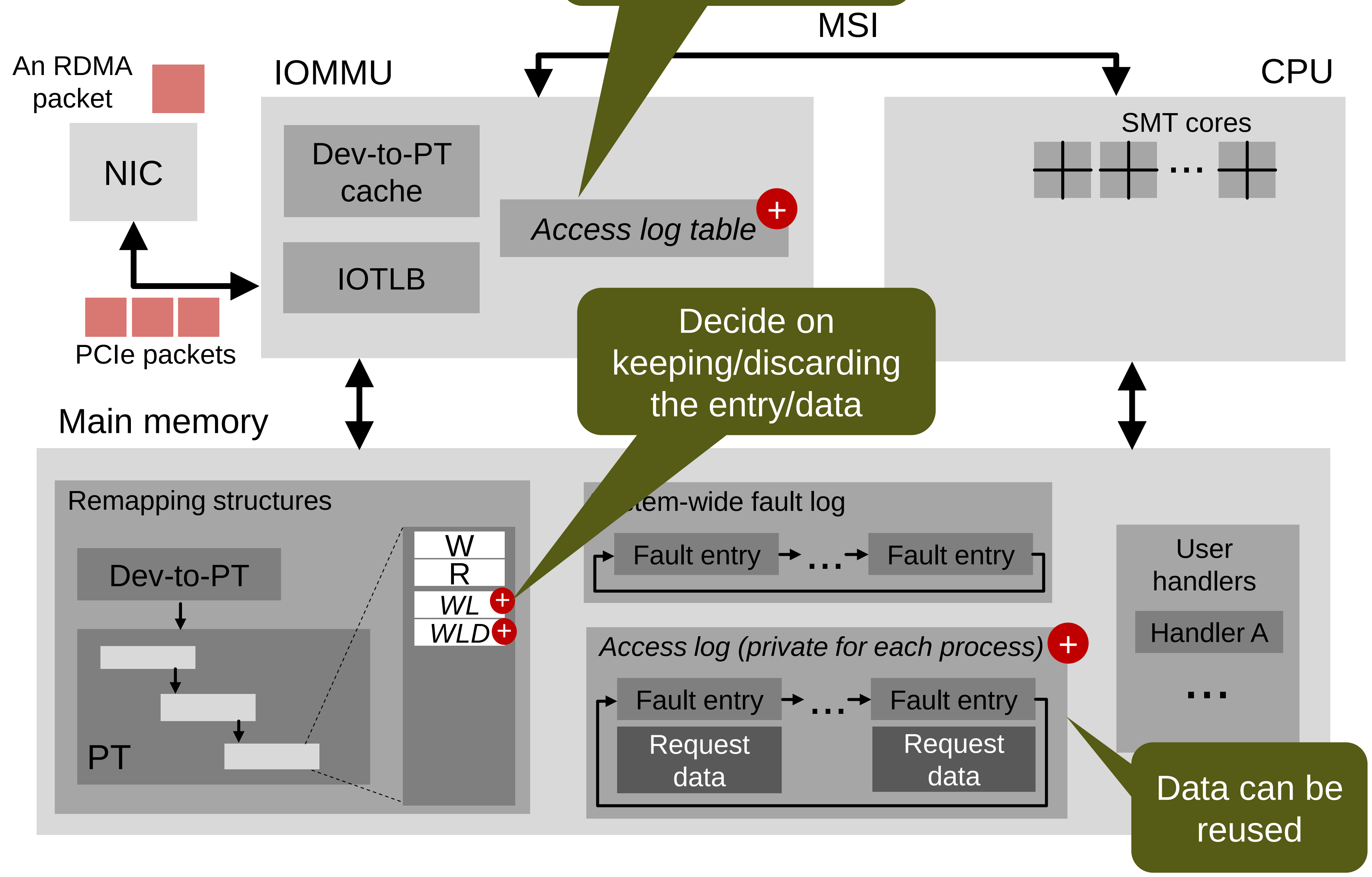
Active Puts



Stores addresses of each access log

Data can be reused

ACTIVE PUTS

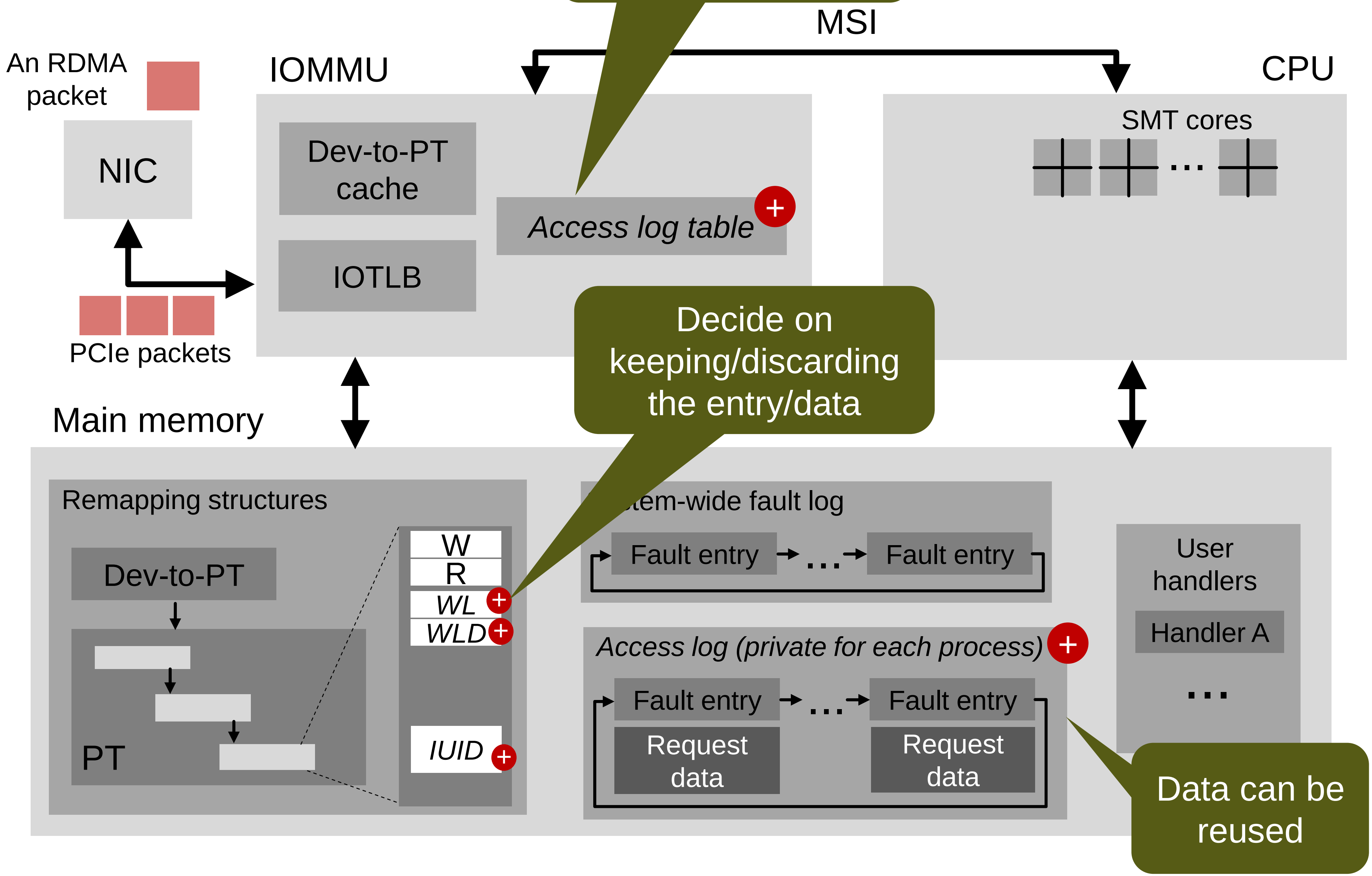


Stores addresses of each access log

Decide on keeping/discarding the entry/data

Data can be reused

Active Puts

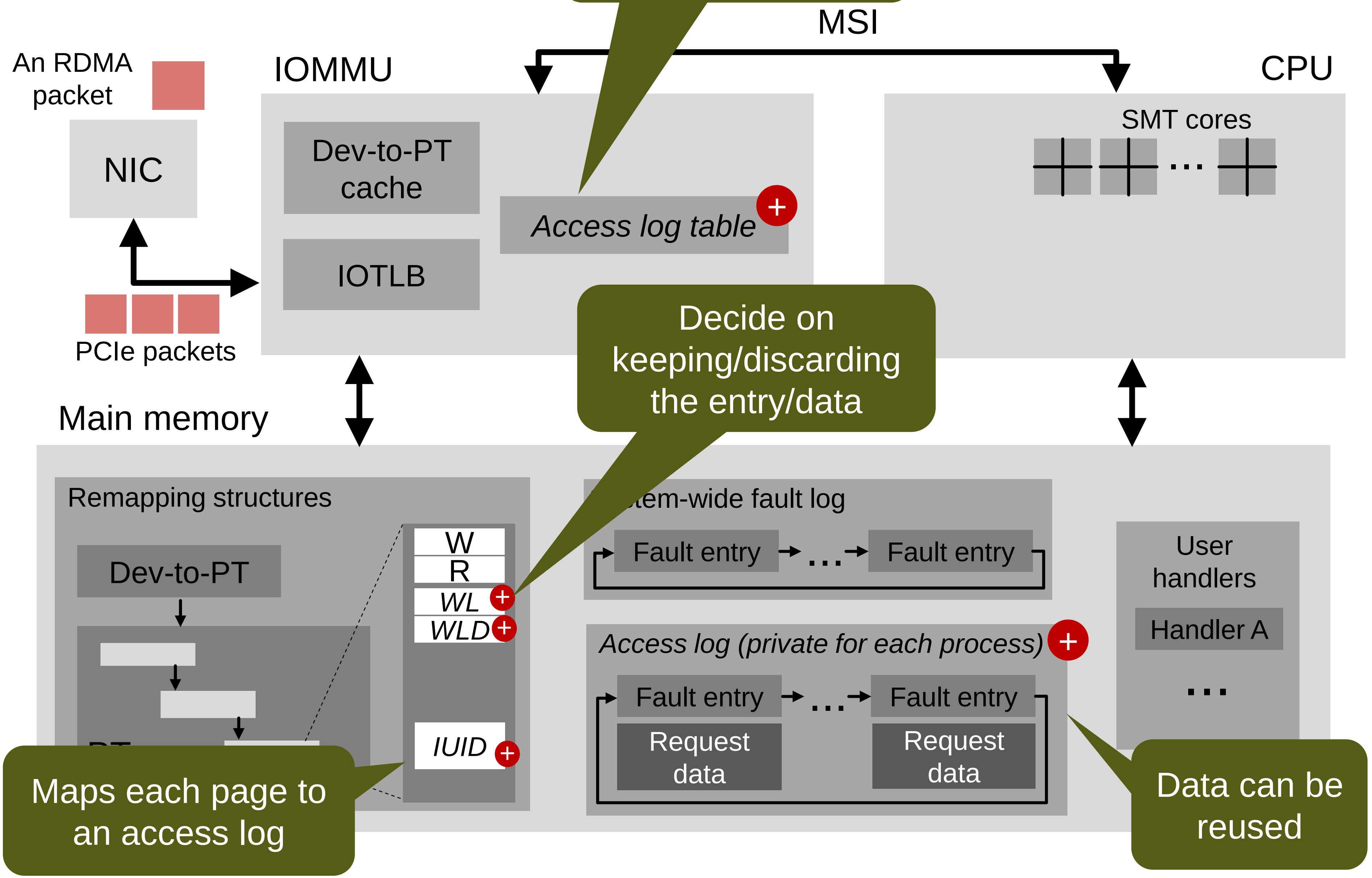


Stores addresses of each access log

Decide on keeping/discarding the entry/data

Data can be reused

Active Puts



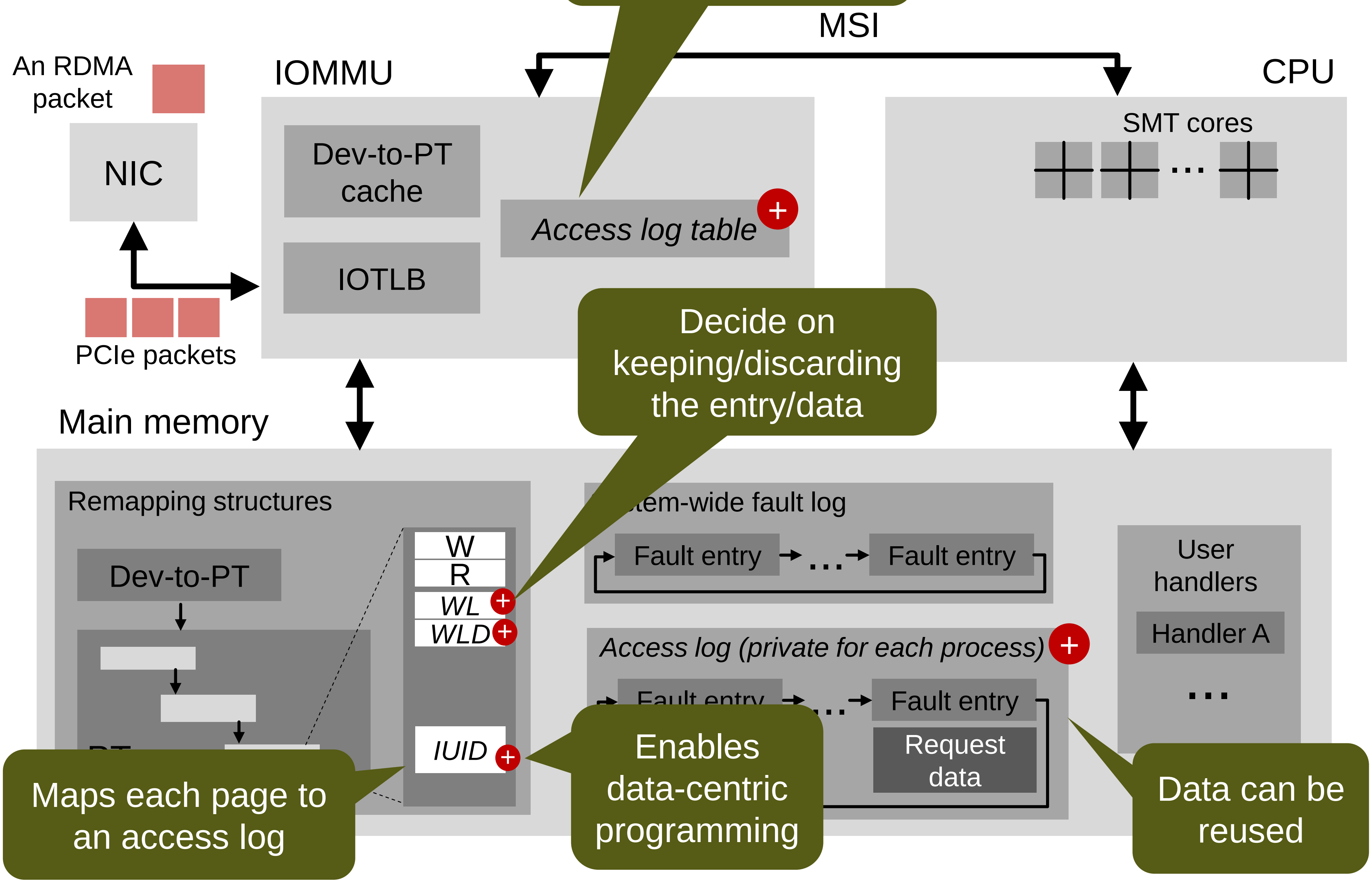
Stores addresses of each access log

Decide on keeping/discarding the entry/data

Maps each page to an access log

Data can be reused

Active Puts



Stores addresses of each access log

Decide on keeping/discarding the entry/data

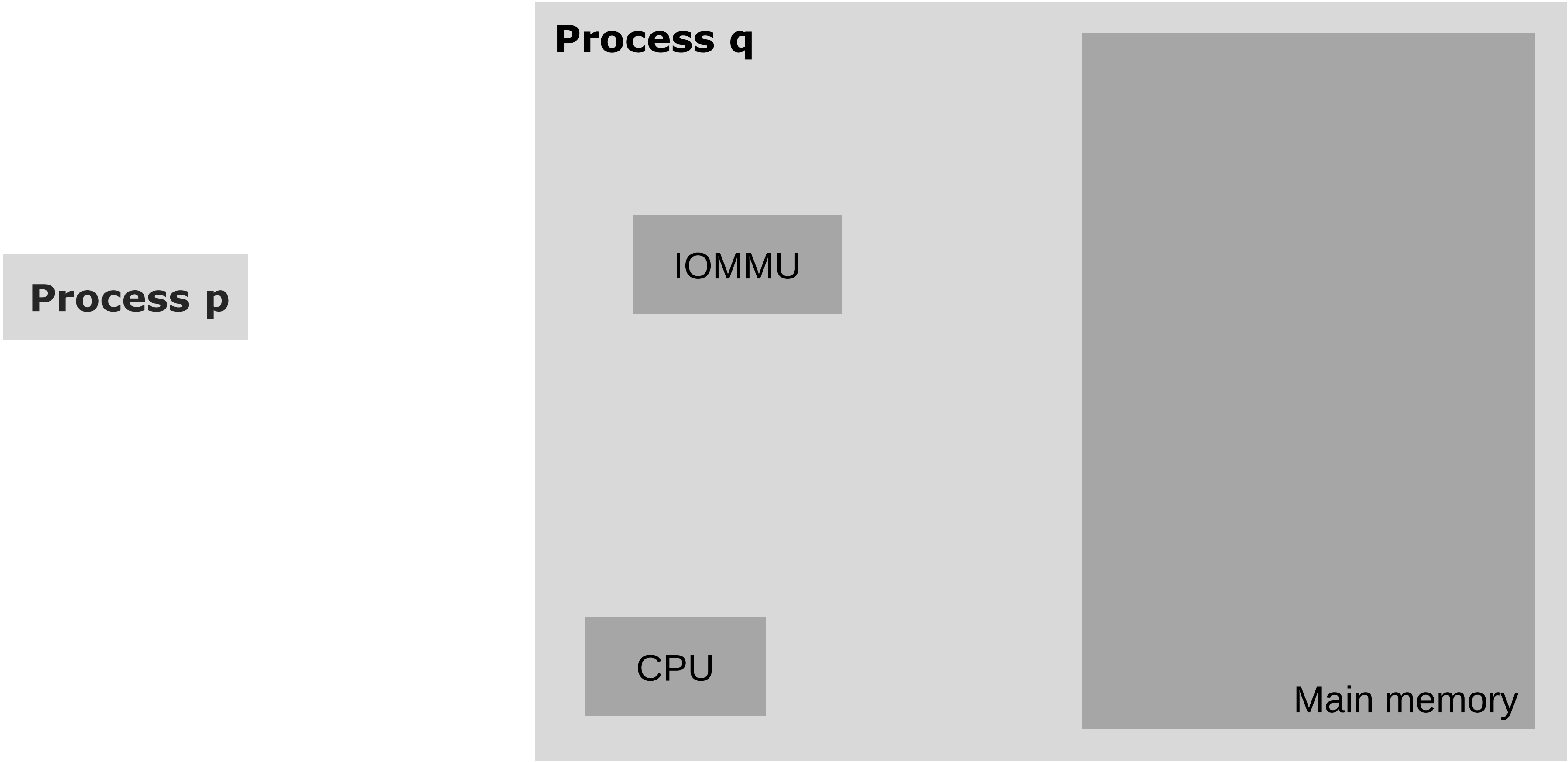
Maps each page to an access log

Enables data-centric programming

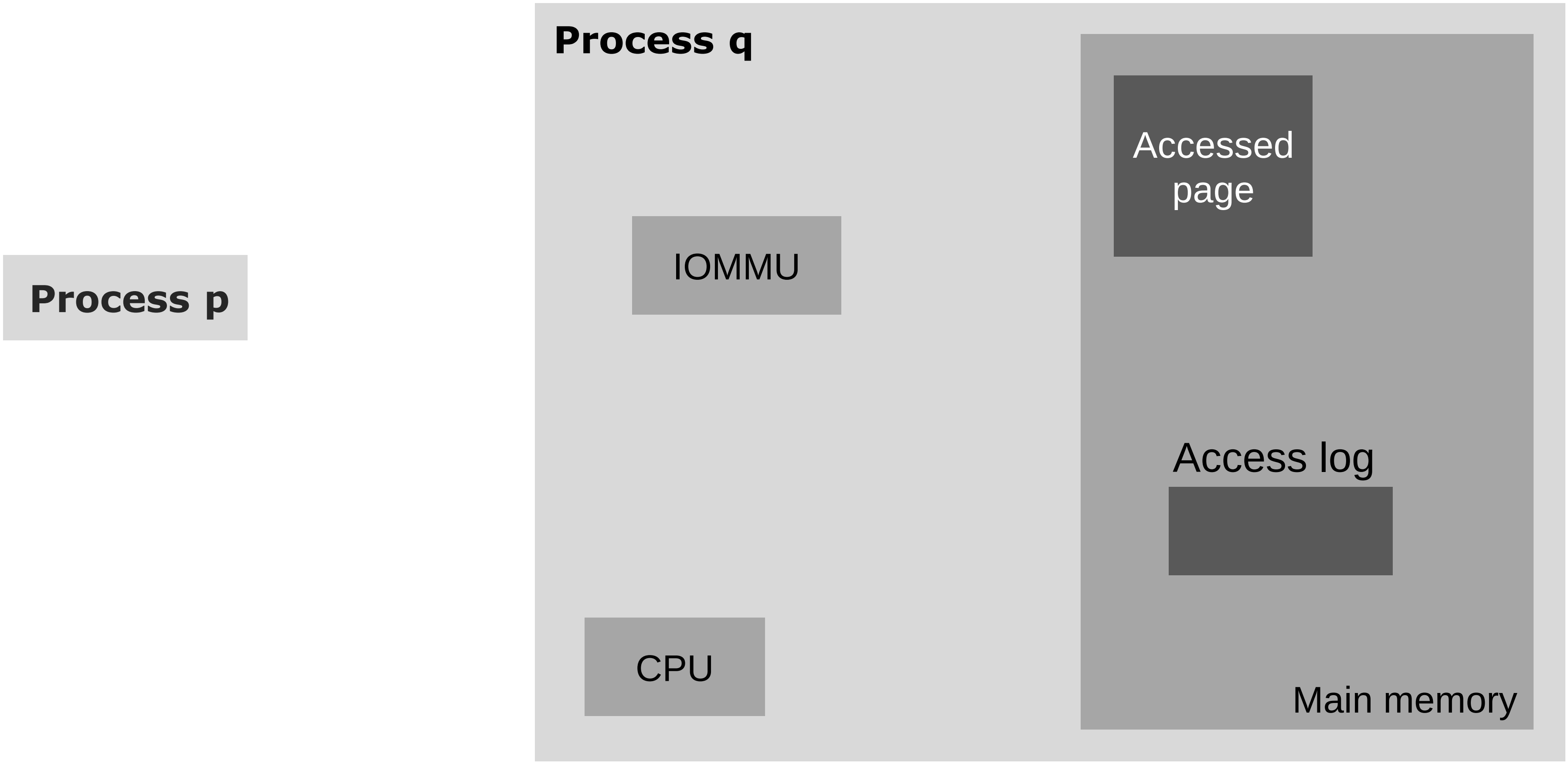
Data can be reused

ACTIVE PUTS

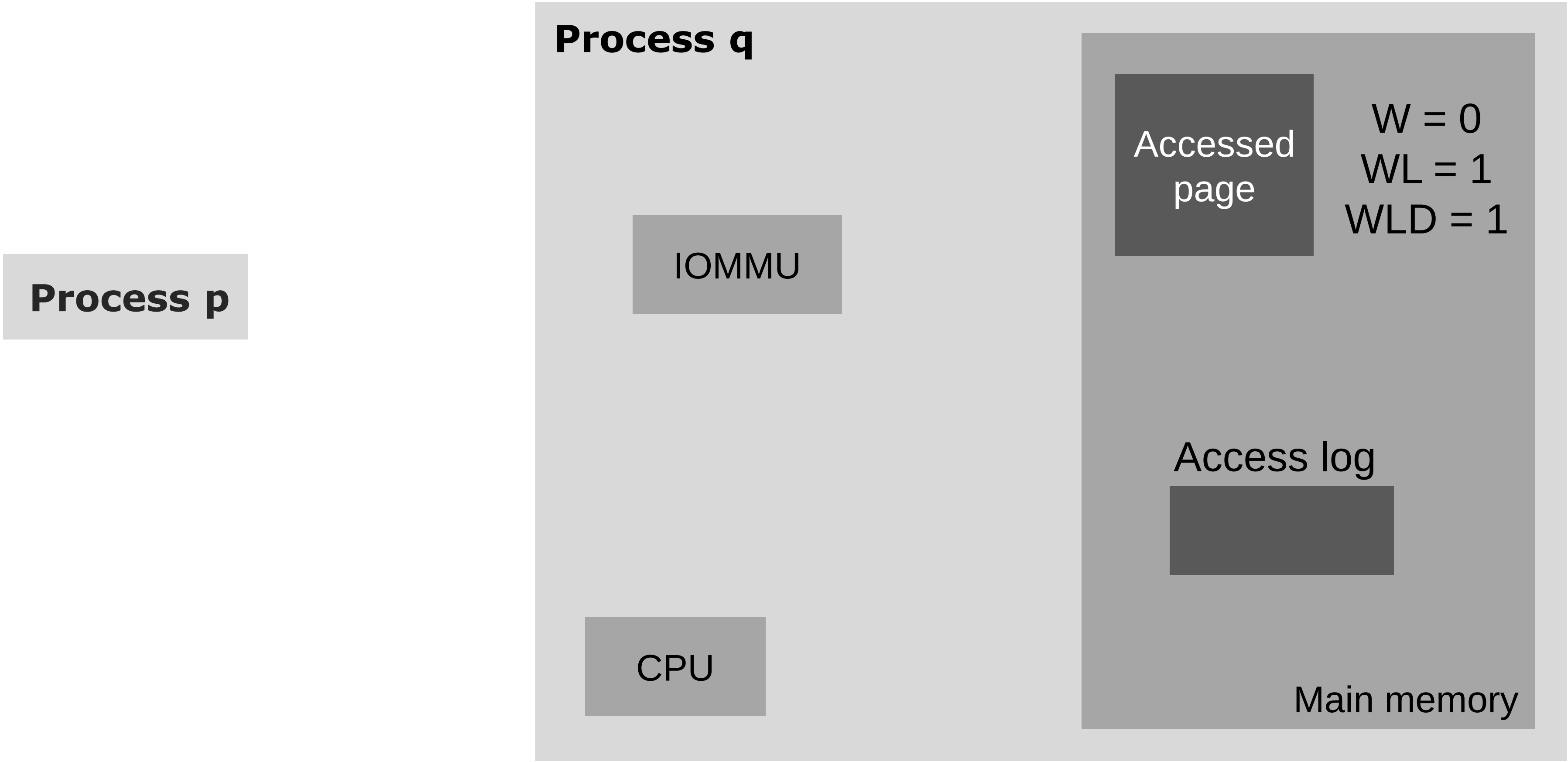
Active Puts



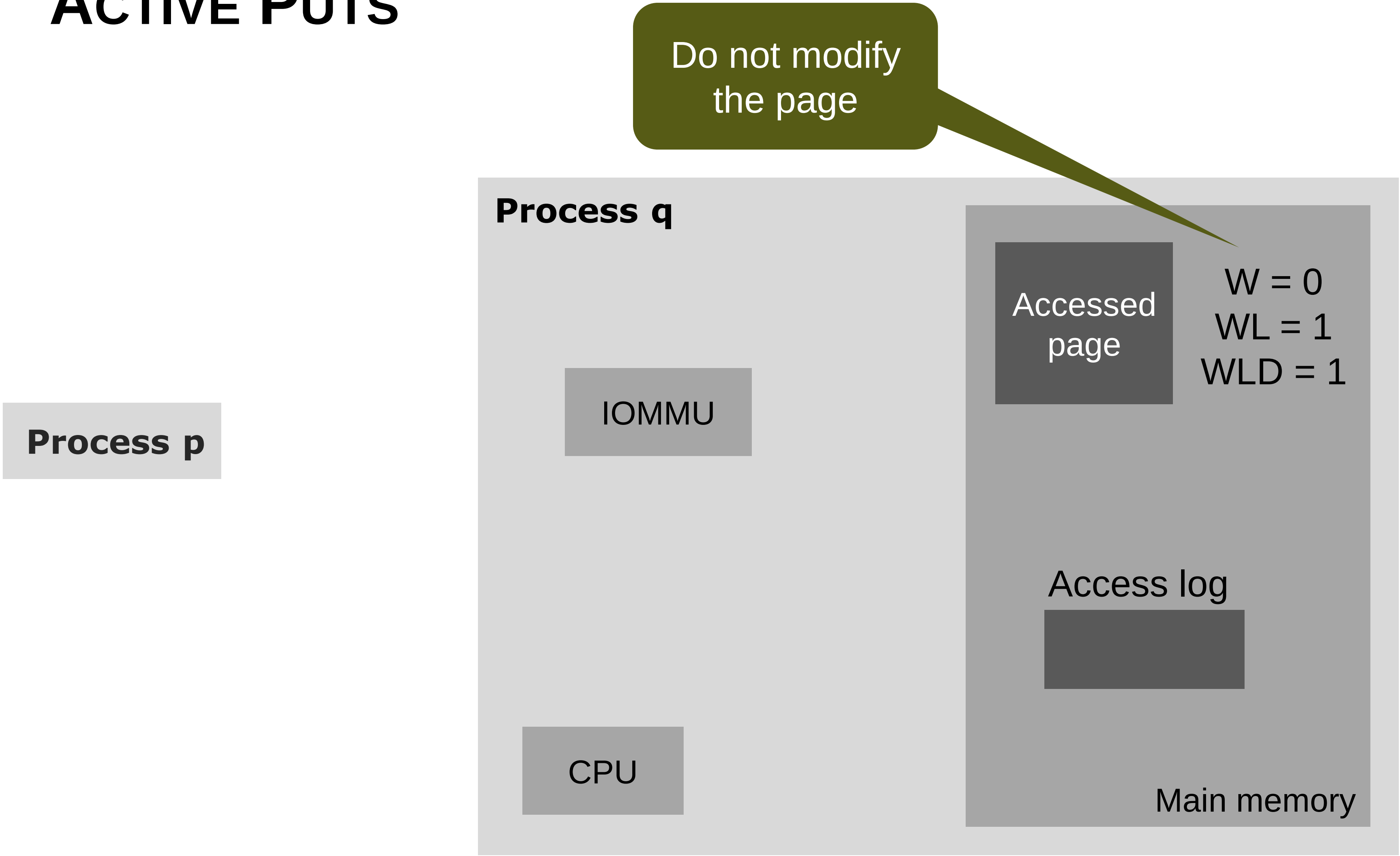
Active Puts


Process p
Process q
IOMMU
CPU
Accessed page
Access log
Main memory

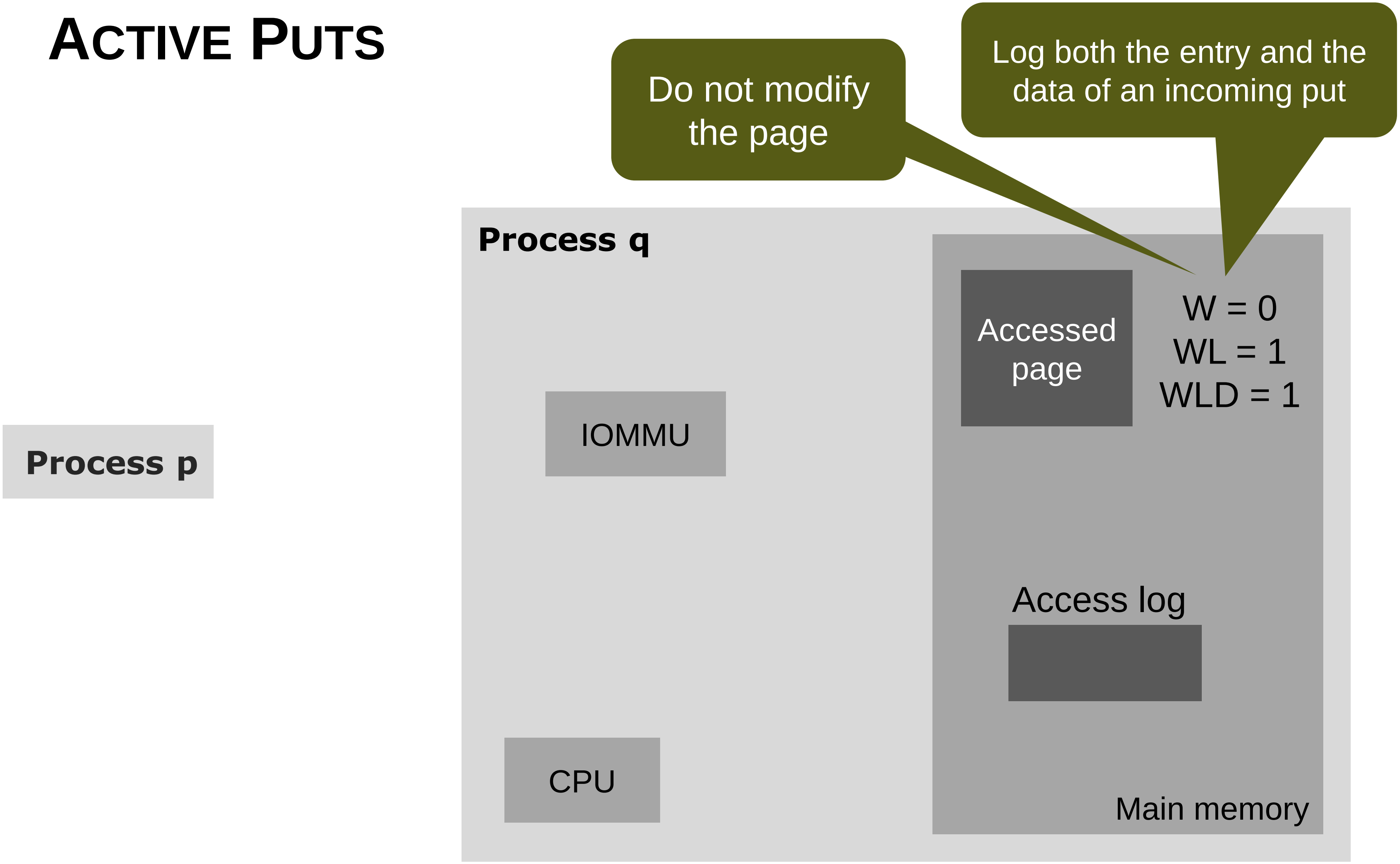
Active Puts


Process p
Process q
IOMMU
CPU
Accessed page
W = 0
WL = 1
WLD = 1
Access log
Main memory

Active Puts

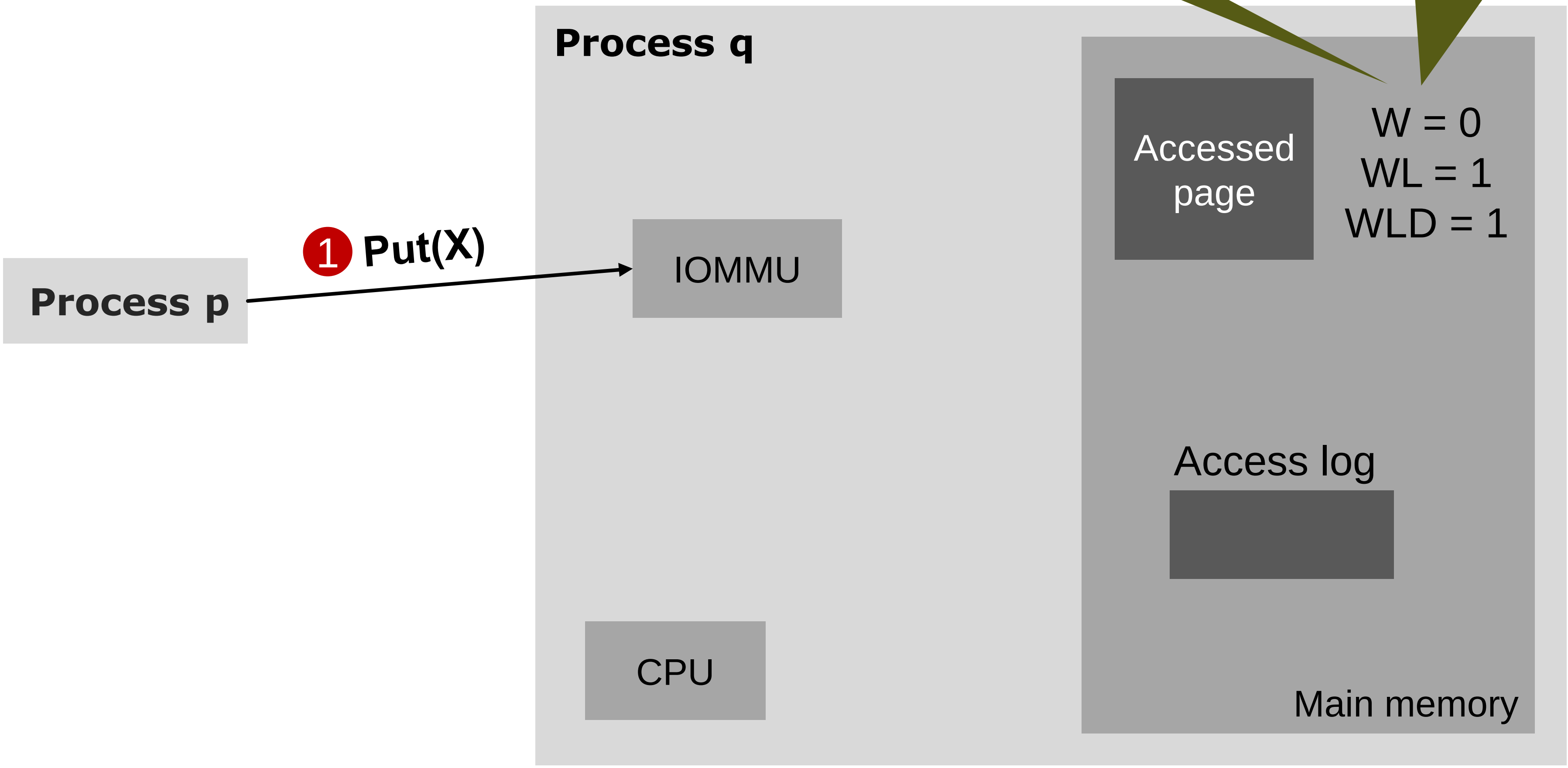

Do not modify the page
Process q
Accessed page
W = 0
WL = 1
WLD = 1
IOMMU
Process p
Access log
CPU
Main memory

Active Puts


Do not modify the page

Log both the entry and the data of an incoming put

Active Puts



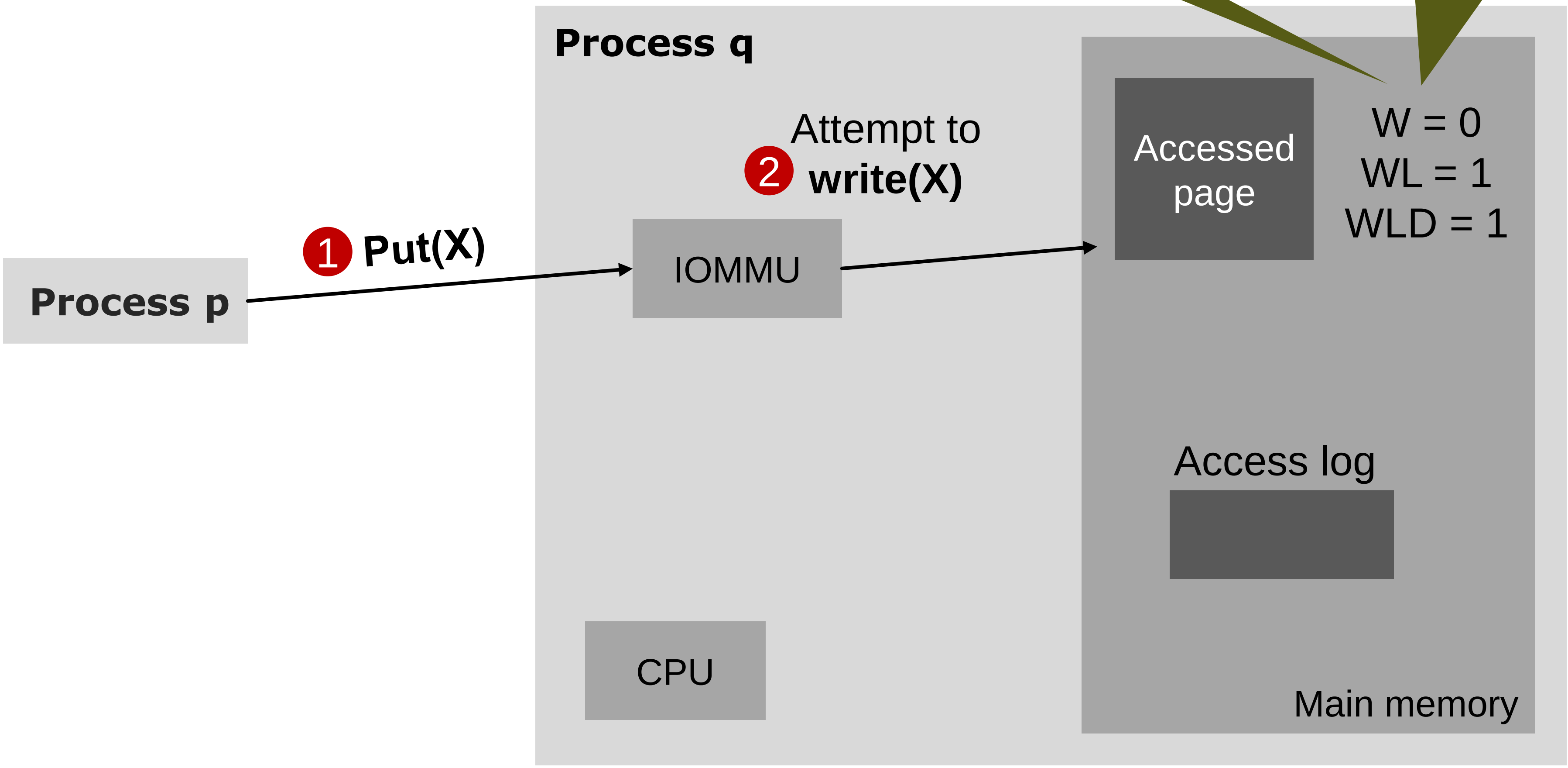
Do not modify the page

Log both the entry and the data of an incoming put

ACTIVE PUTS

Do not modify the page

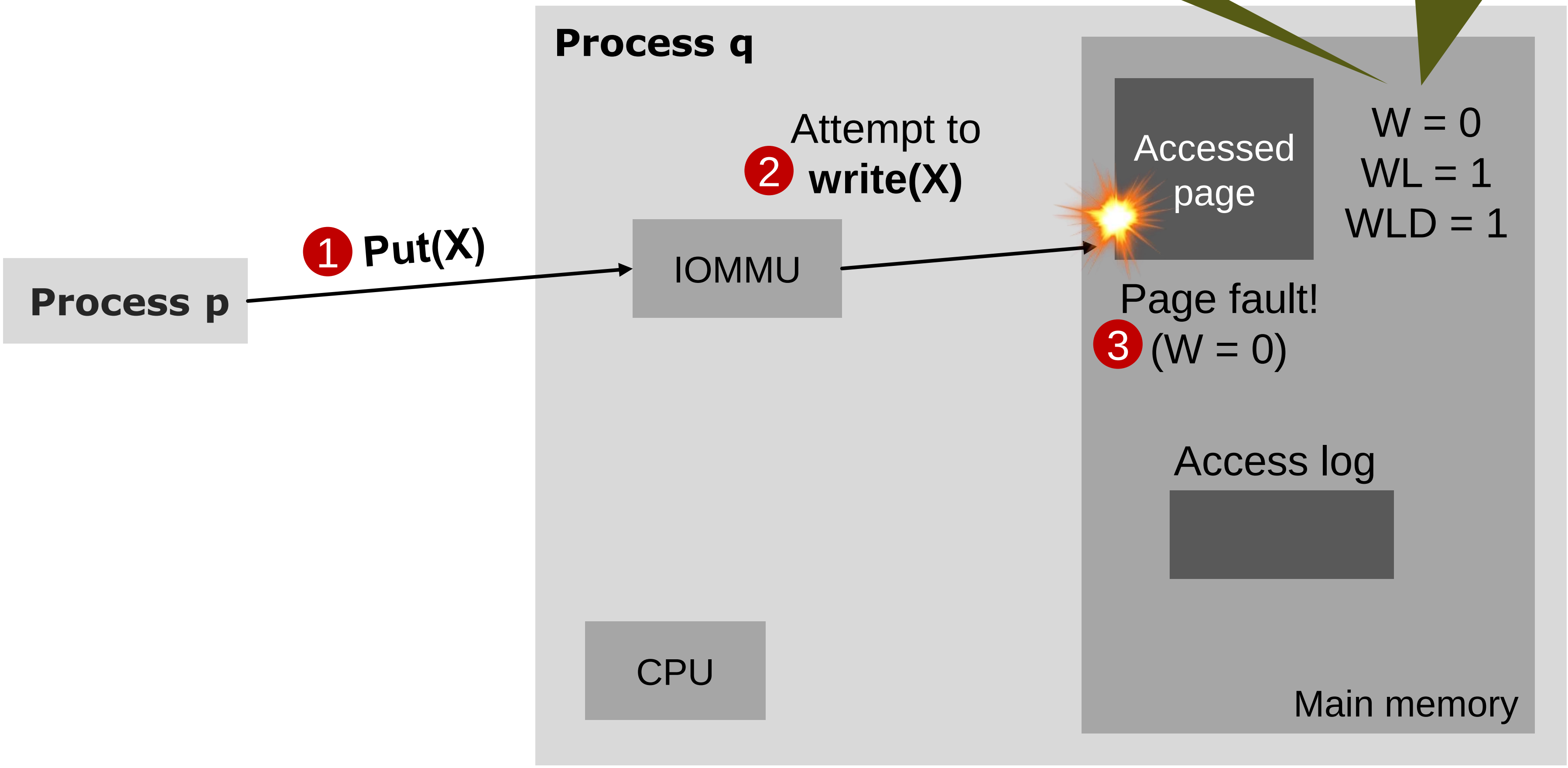
Log both the entry and the data of an incoming put



Active Puts

Do not modify the page

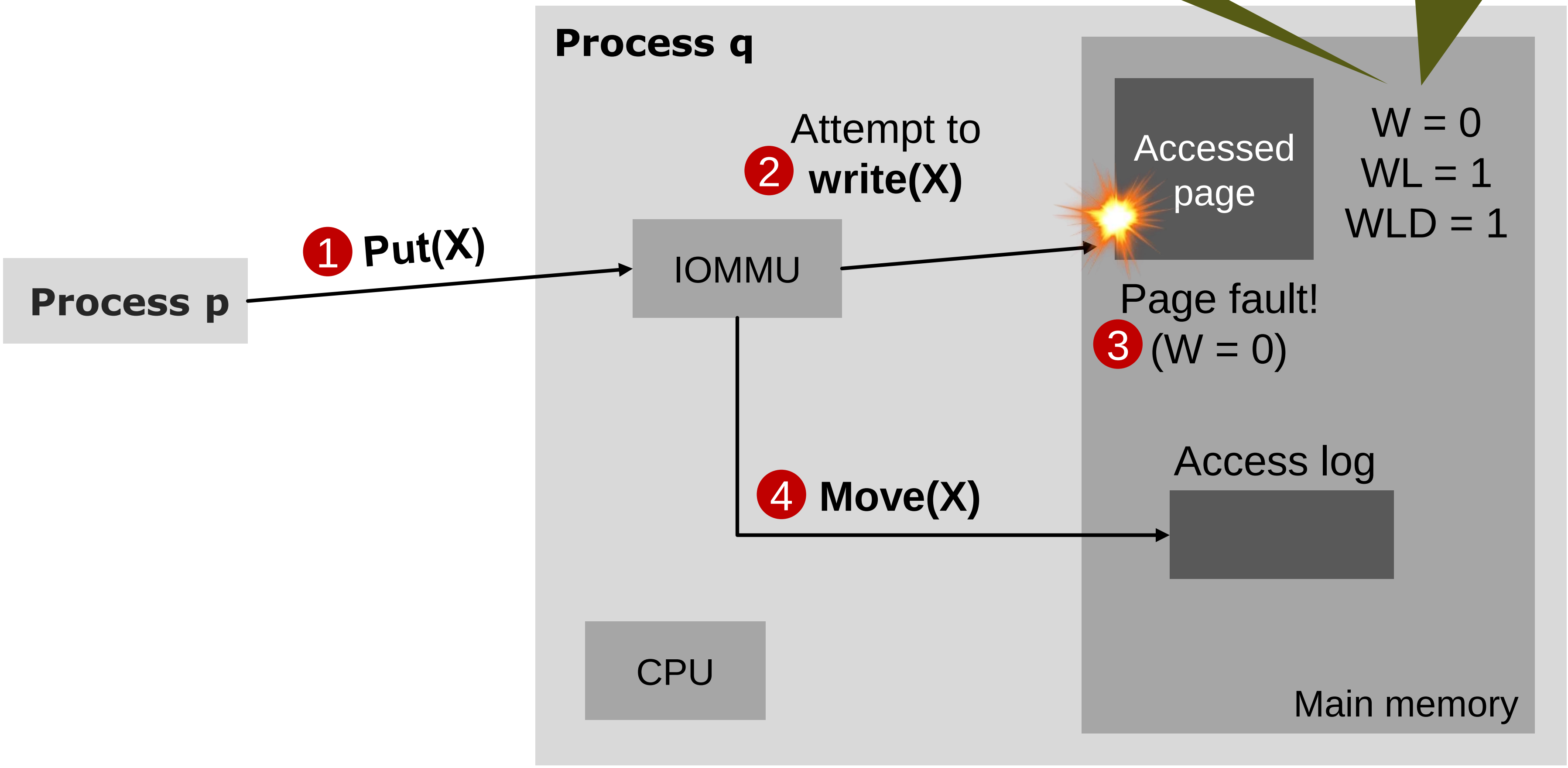
Log both the entry and the data of an incoming put



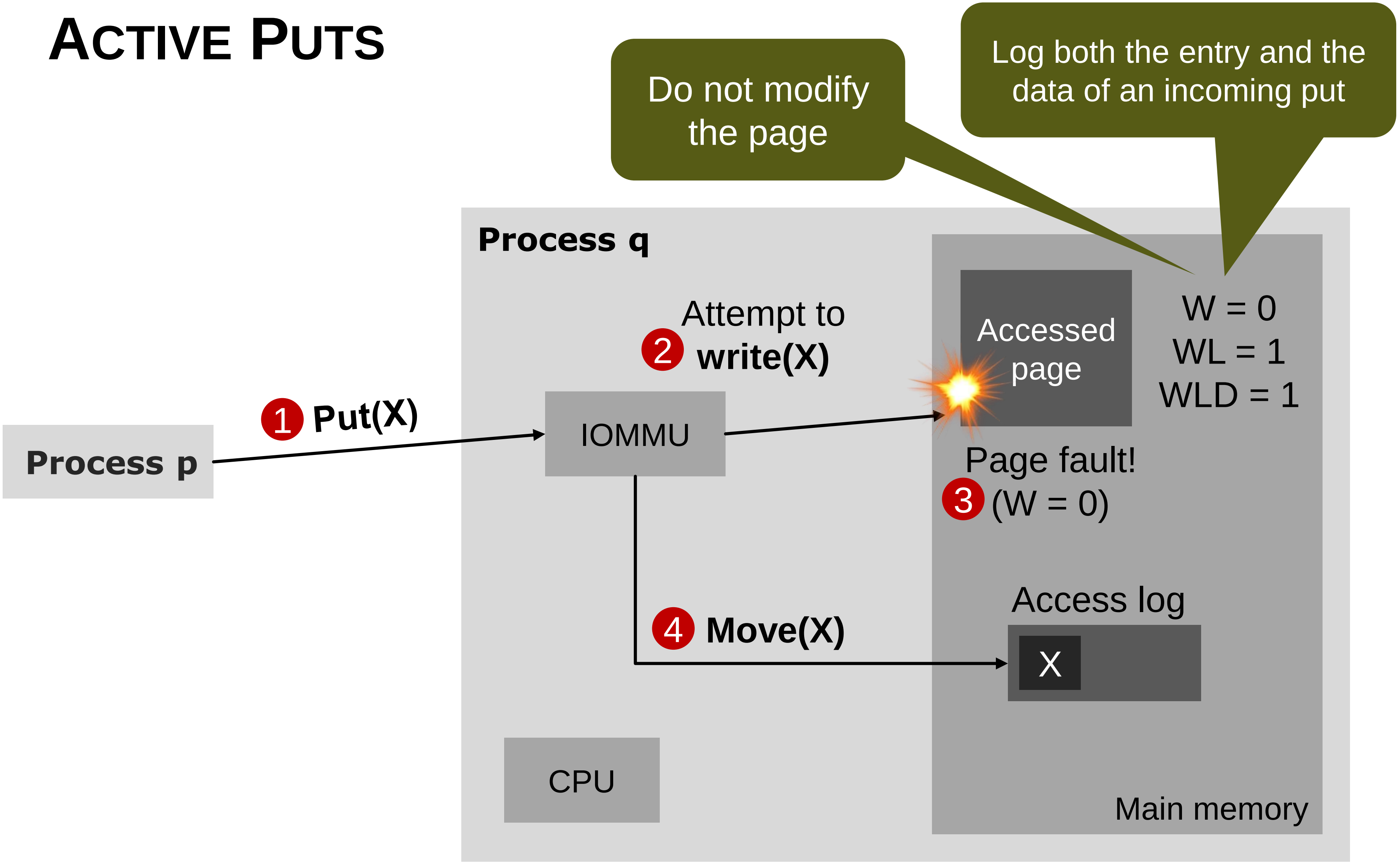
Active Puts

Do not modify the page

Log both the entry and the data of an incoming put



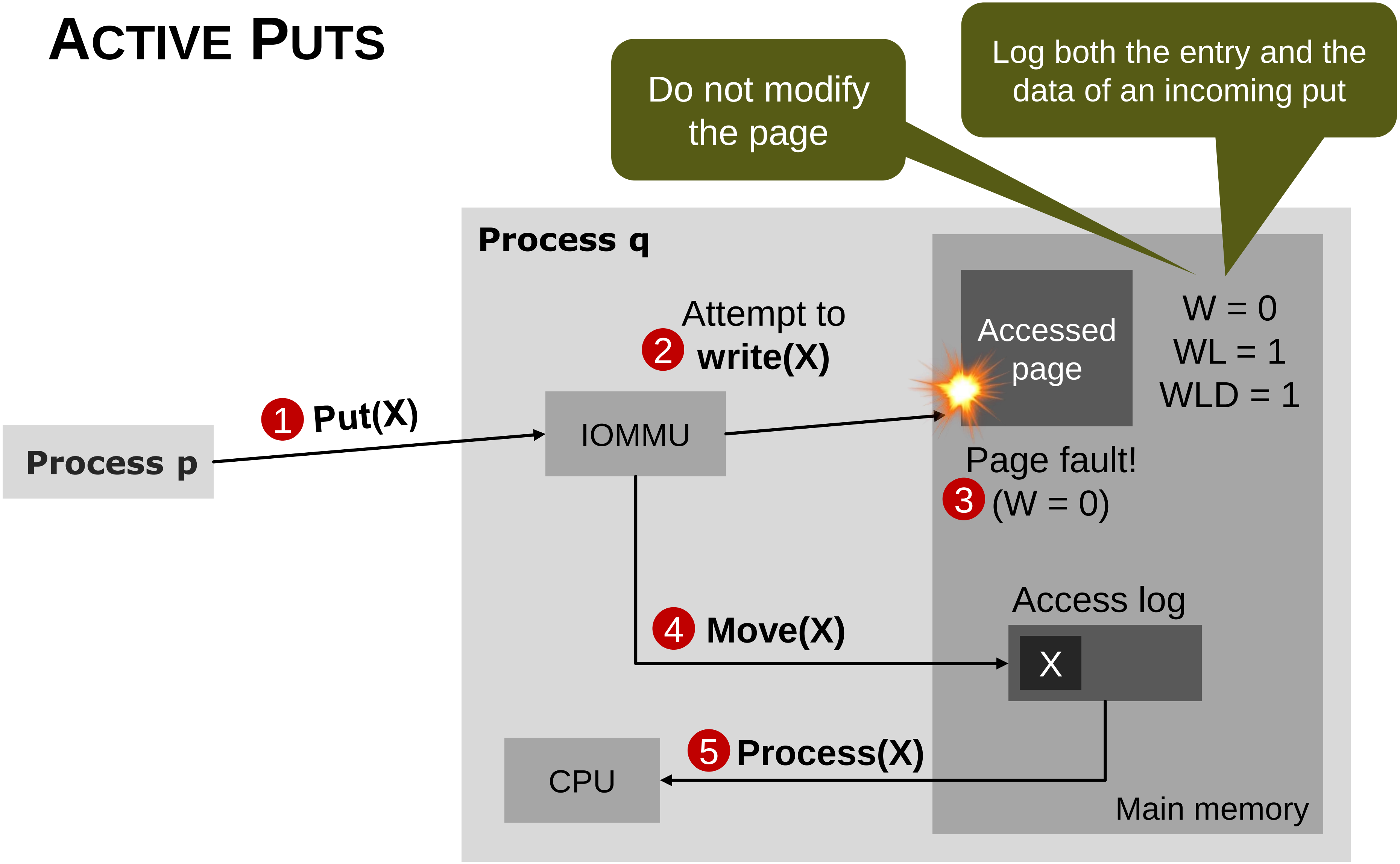
Active Puts



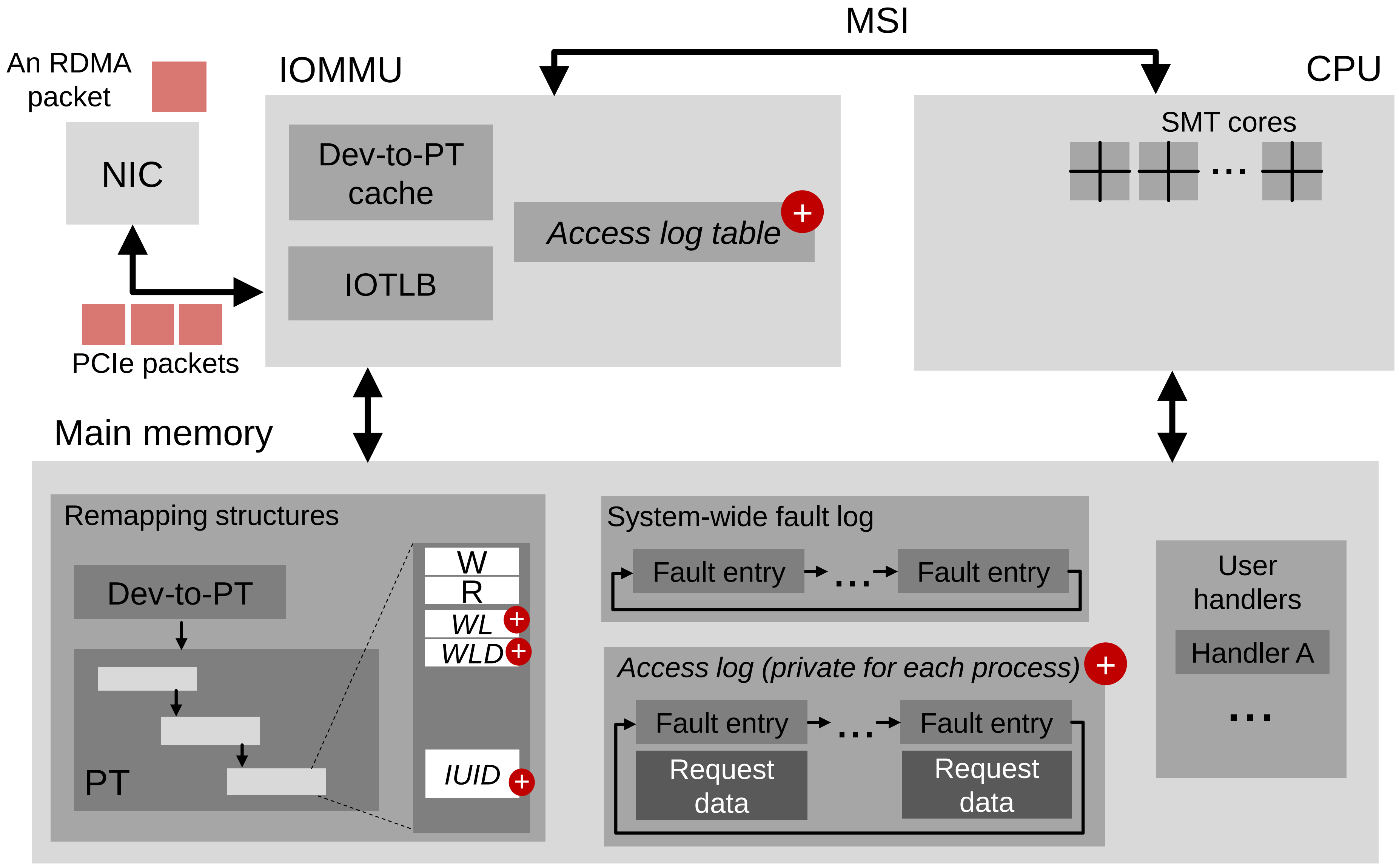
Do not modify the page

Log both the entry and the data of an incoming put

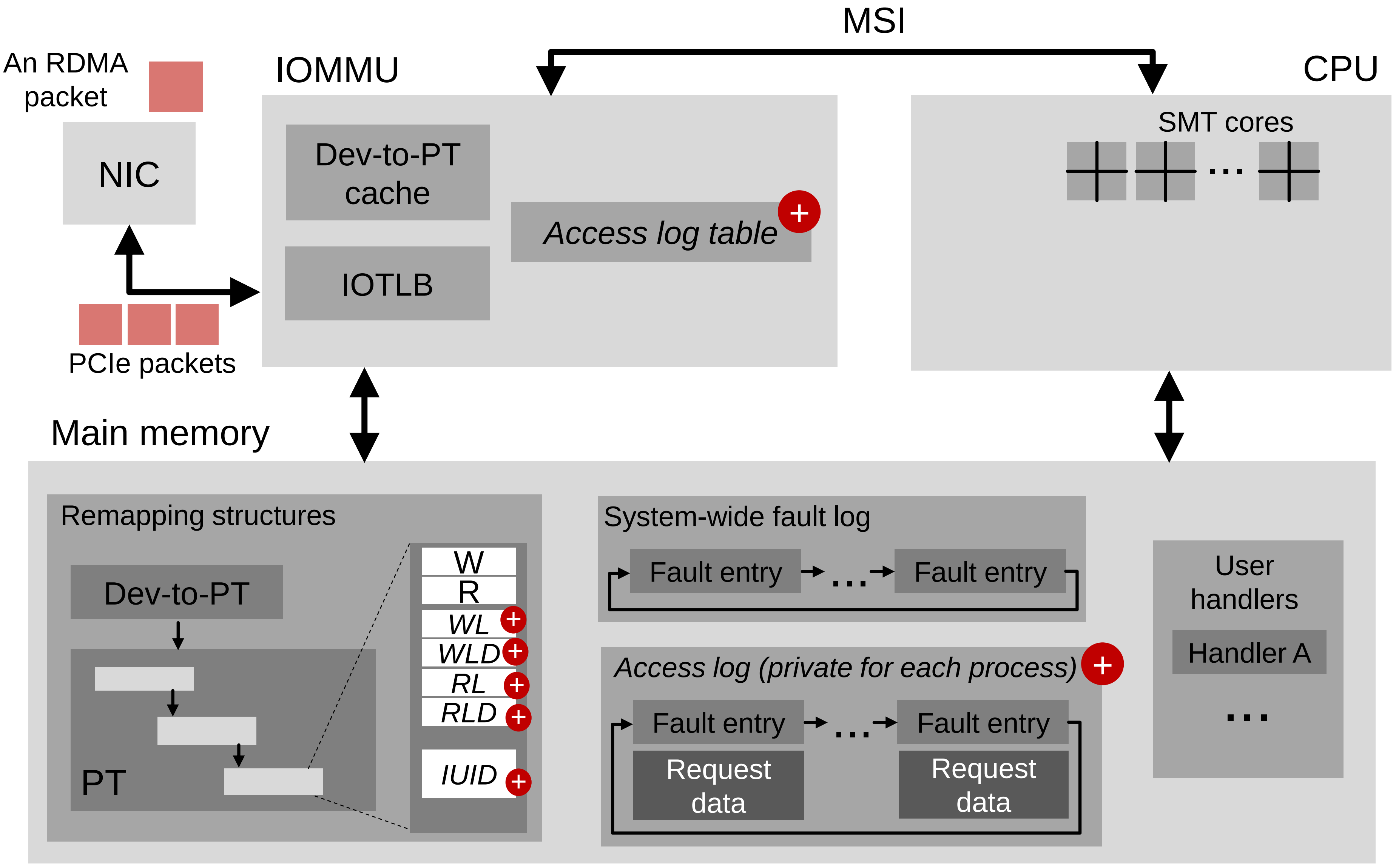
ACTIVE PUTS


Do not modify the page
Log both the entry and the data of an incoming put
Process q
Process p
1 Put(X)
Attempt to
2 write(X)
IOMMU
Accessed page
W = 0
WL = 1
WLD = 1
Page fault!
3 (W = 0)
Access log
X
4 Move(X)
5 Process(X)
CPU
Main memory

Active Gets

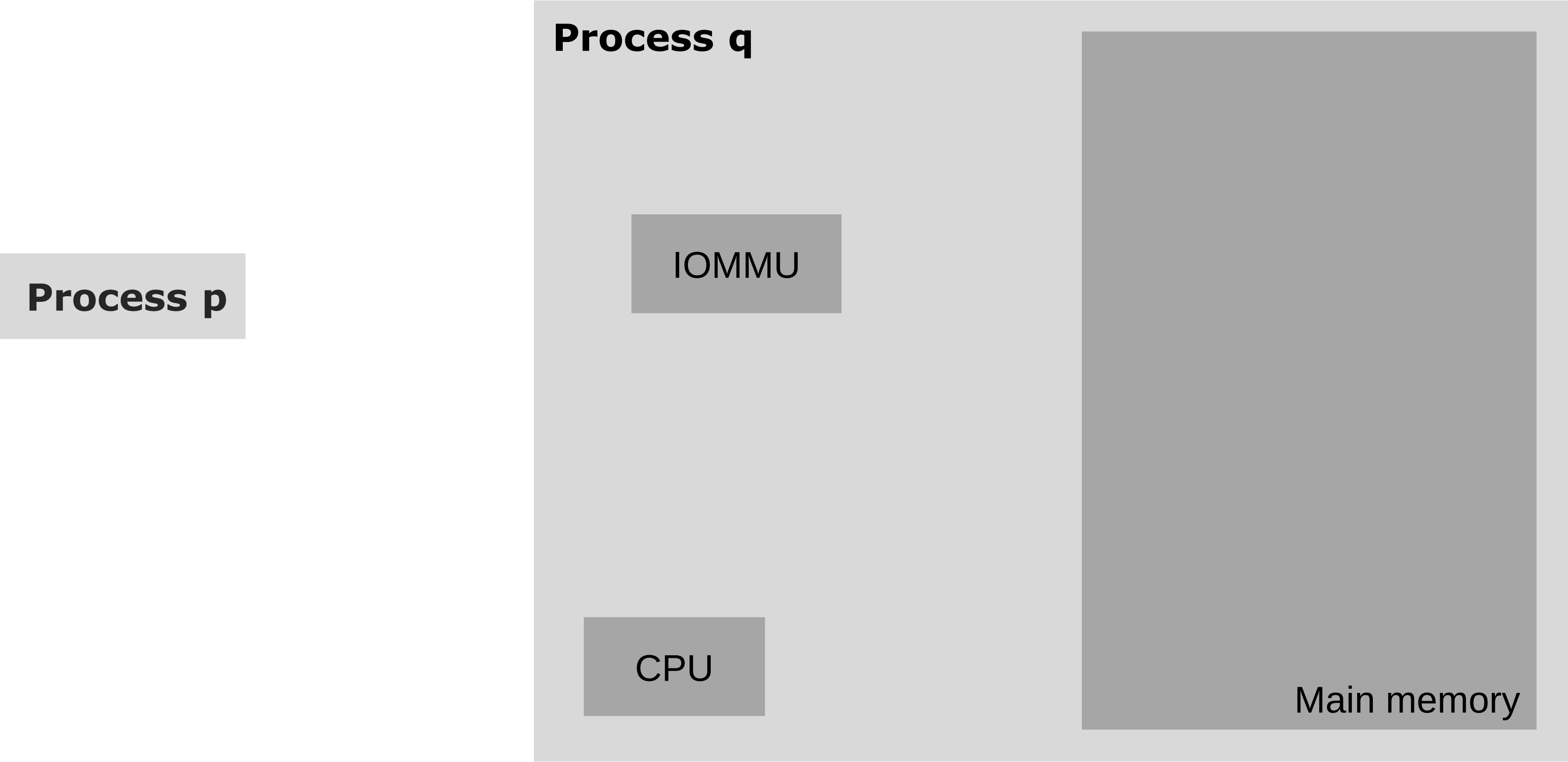

MSI
An RDMA packet
IOMMU
CPU
NIC
Dev-to-PT cache
IOTLB
Access log table
+
SMT cores
...
PCIe packets
Main memory
Remapping structures
Dev-to-PT
PT
W
R
WL
+
WLD
+
IUID
+
System-wide fault log
Fault entry
...
Fault entry
Access log (private for each process)
+
Fault entry
...
Fault entry
Request data
Request data
User handlers
Handler A
...

ACTIVE GETS

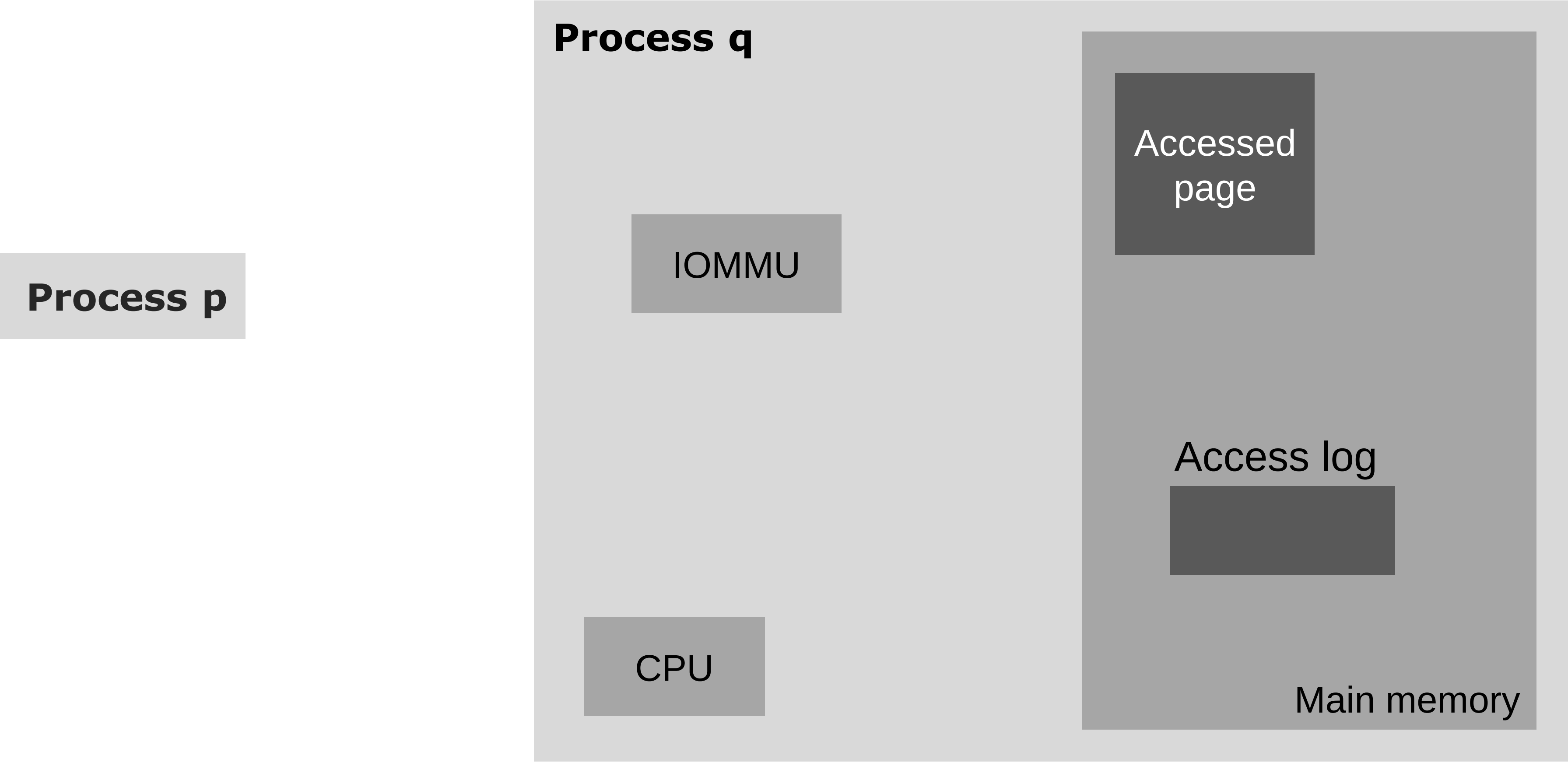

MSI
An RDMA packet
IOMMU
CPU
NIC
Dev-to-PT cache
IOTLB
Access log table +
SMT cores
...
PCIe packets
Main memory
Remapping structures
Dev-to-PT
PT
W
R
WL +
WLD +
RL +
RLD +
IUID +
System-wide fault log
Fault entry
...
Fault entry
Access log (private for each process) +
Fault entry
...
Fault entry
Request data
Request data
User handlers
Handler A
...

ACTIVE GETS

Active Gets



Active Gets



Process p

Process q

IOMMU

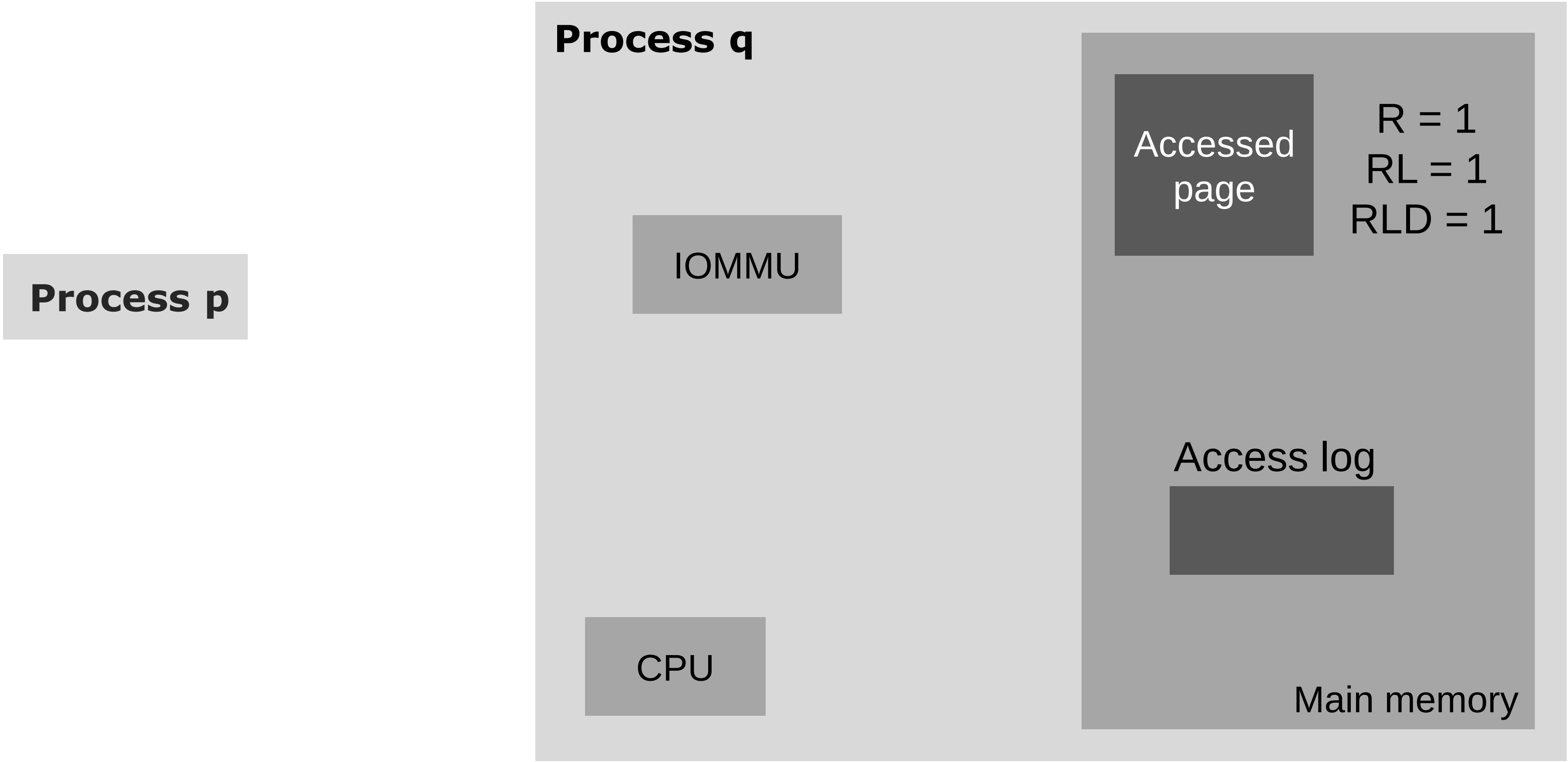
CPU

Accessed page

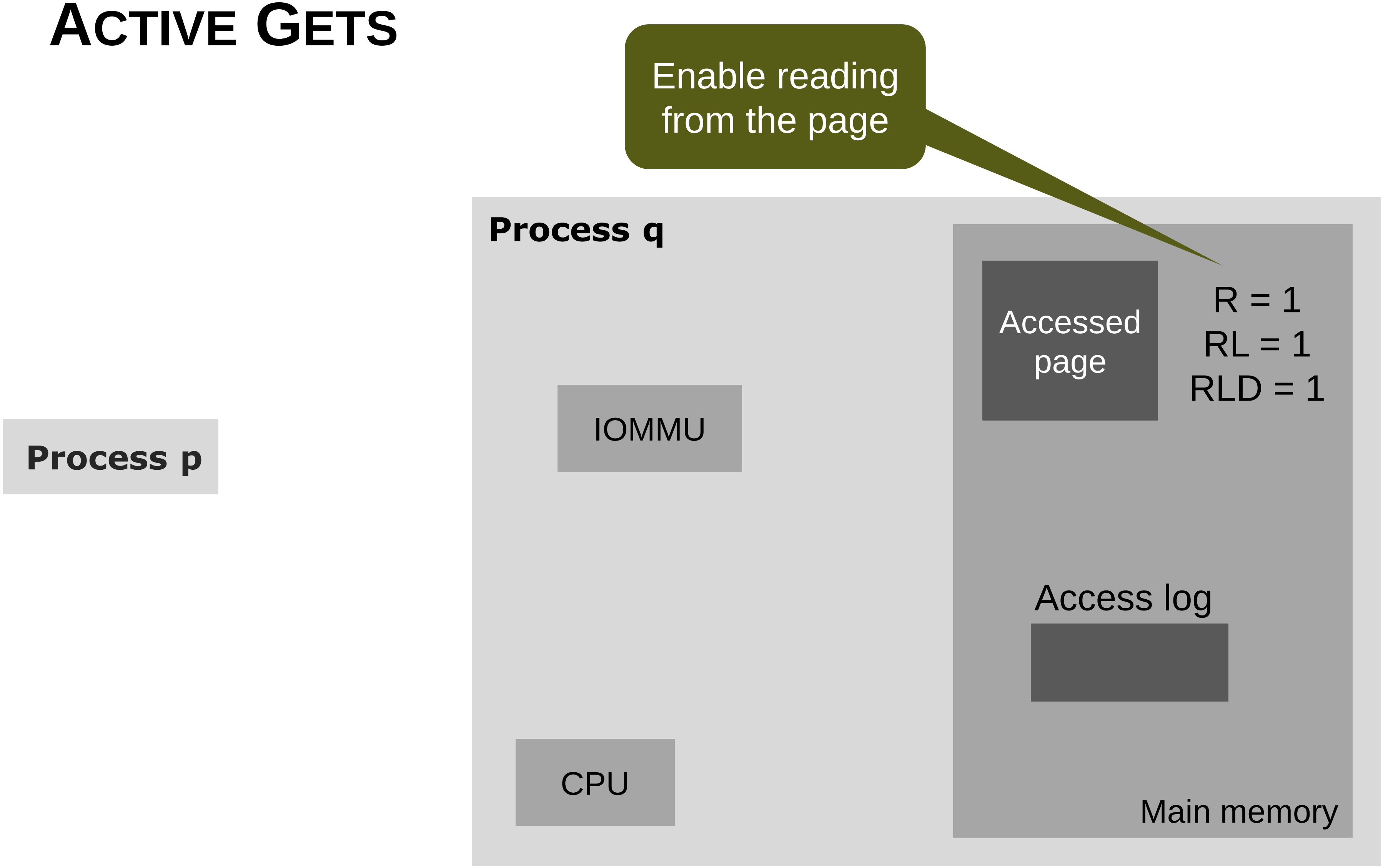
Access log

Main memory

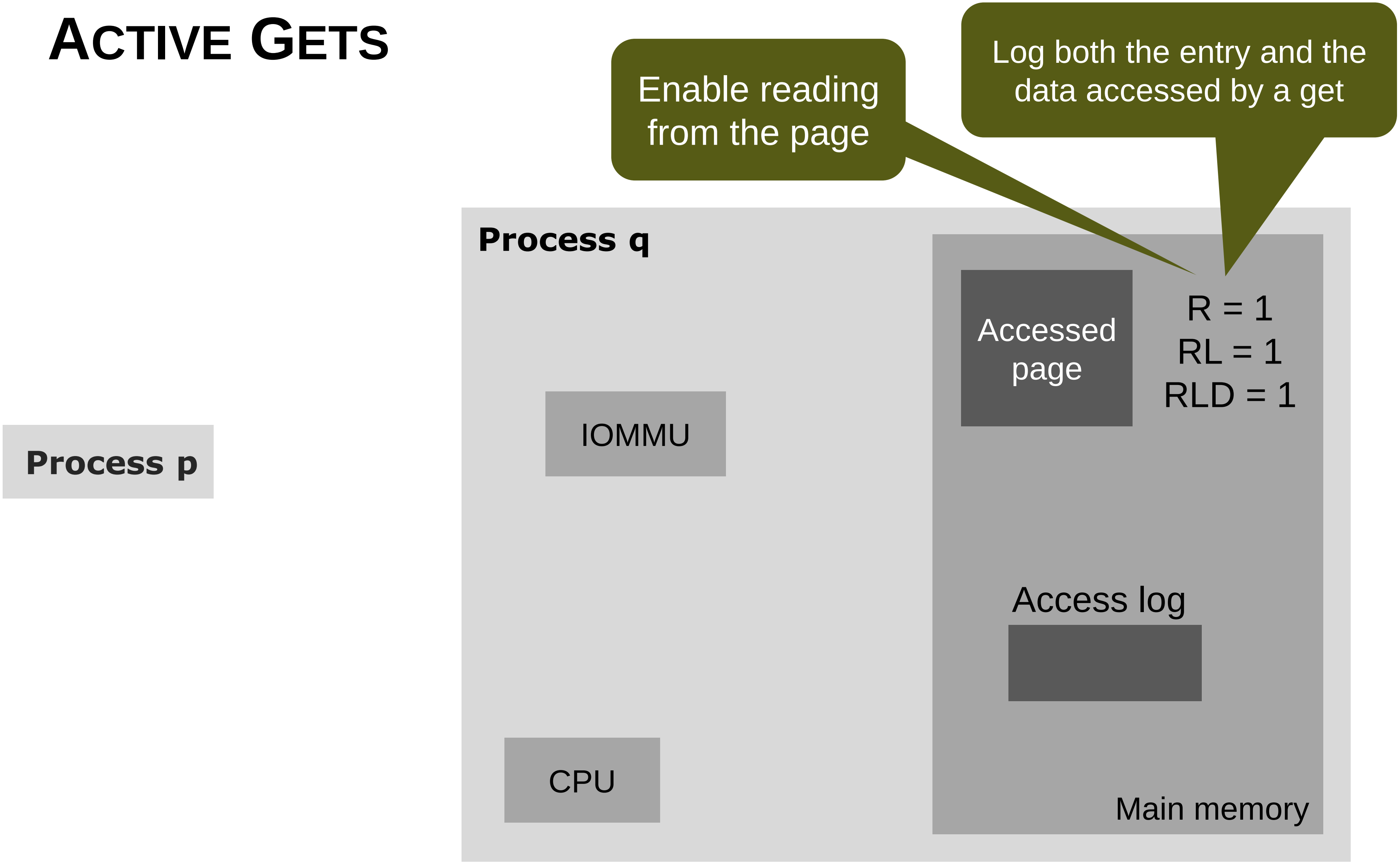
ACTIVE GETS


Process p
Process q
IOMMU
CPU
Accessed page
R = 1
RL = 1
RLD = 1
Access log
Main memory

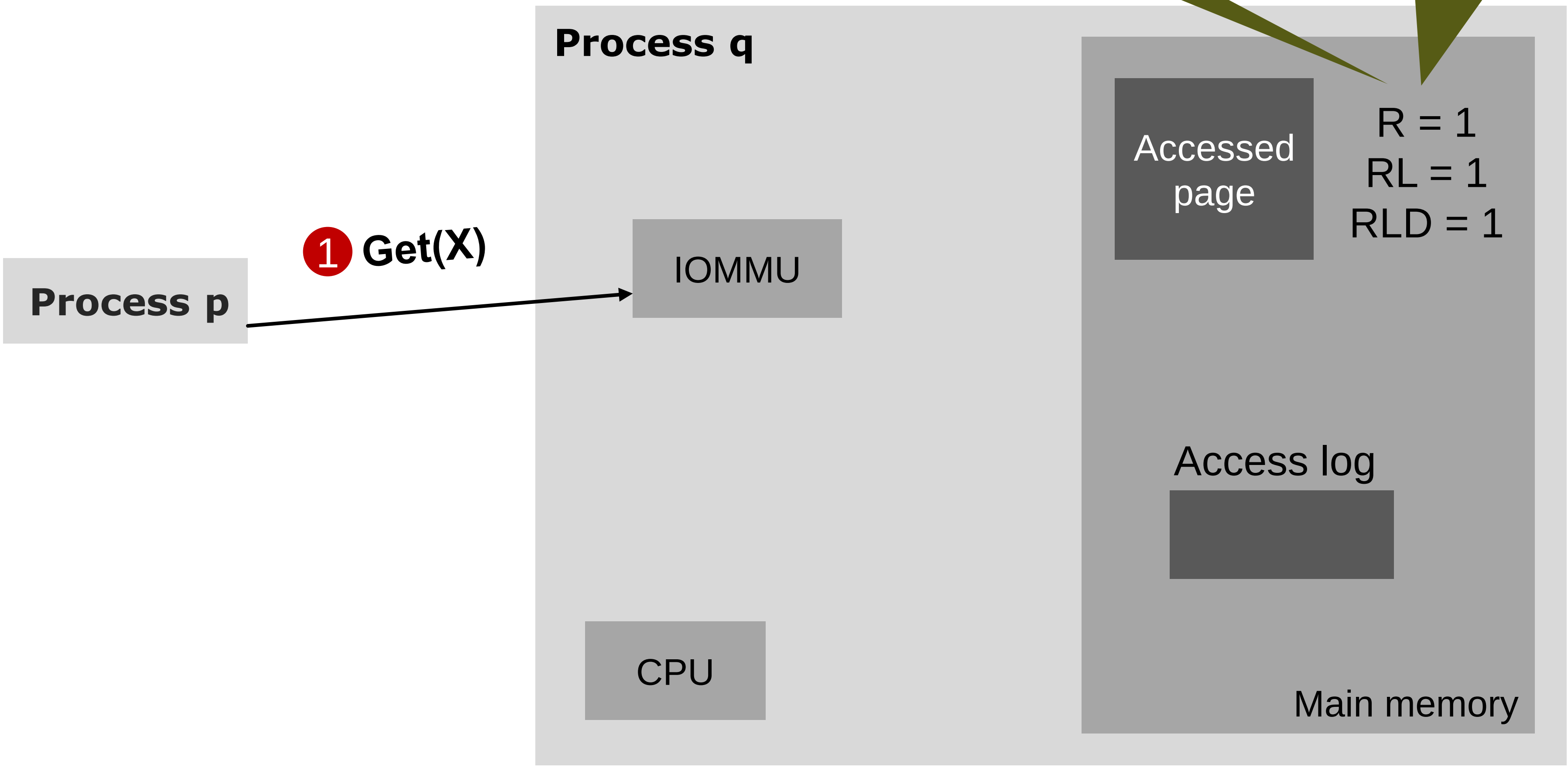
Active Gets


Enable reading from the page

ACTIVE GETS


Enable reading from the page
Log both the entry and the data accessed by a get
Process q
Process p
IOMMU
CPU
Accessed page
R = 1
RL = 1
RLD = 1
Access log
Main memory

ACTIVE GETS



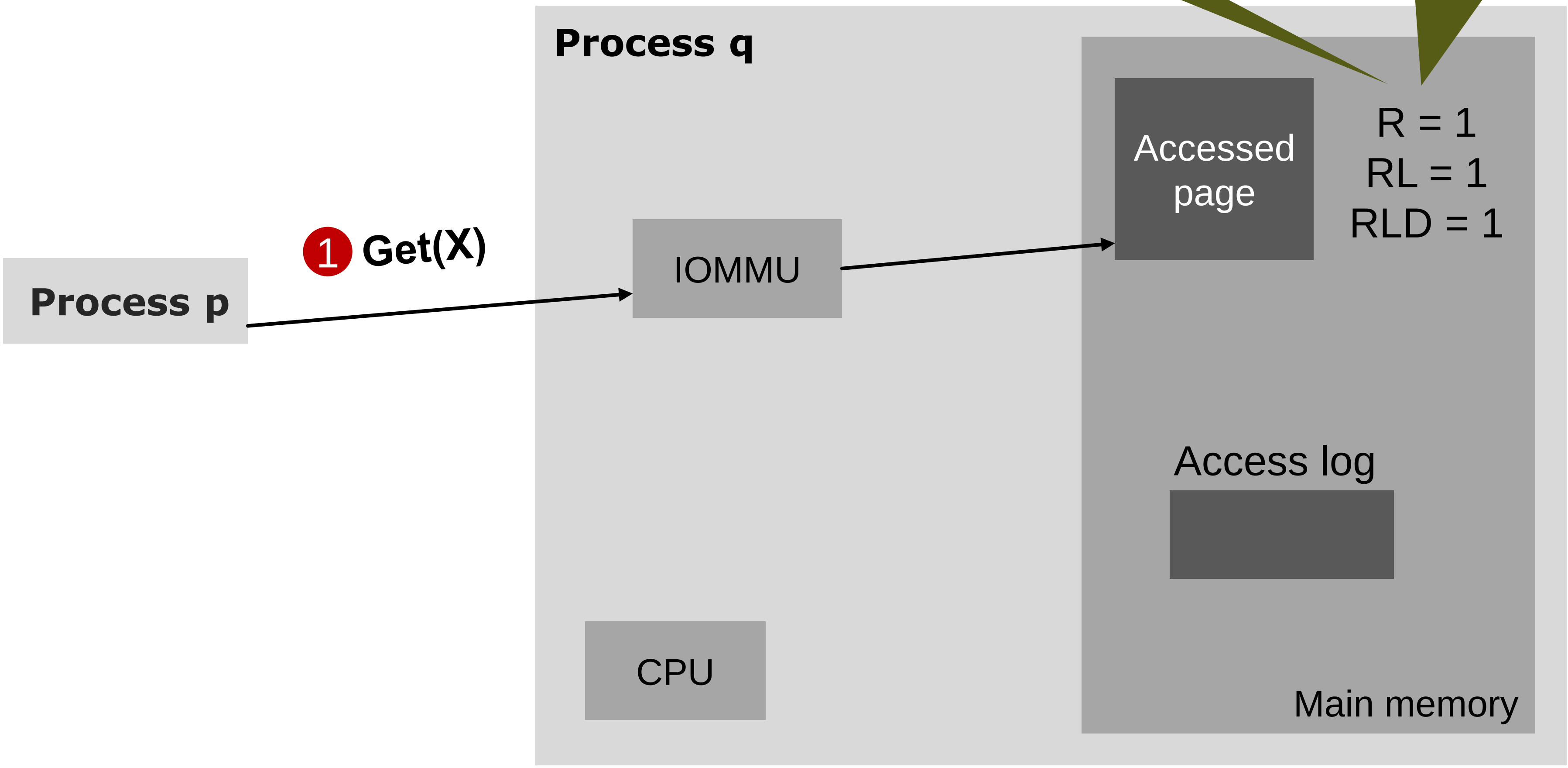
Enable reading from the page

Log both the entry and the data accessed by a get

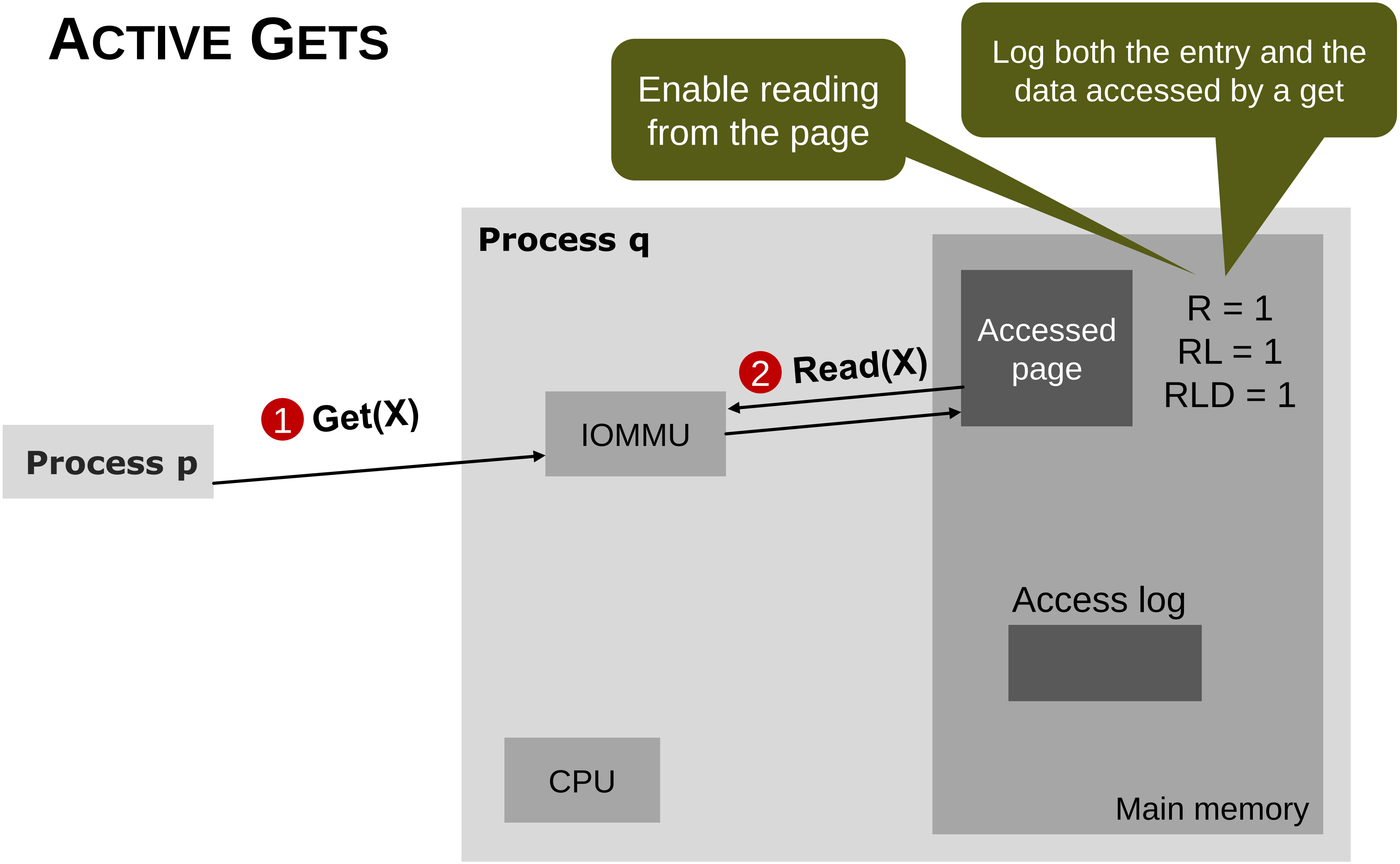
ACTIVE GETS

Enable reading from the page

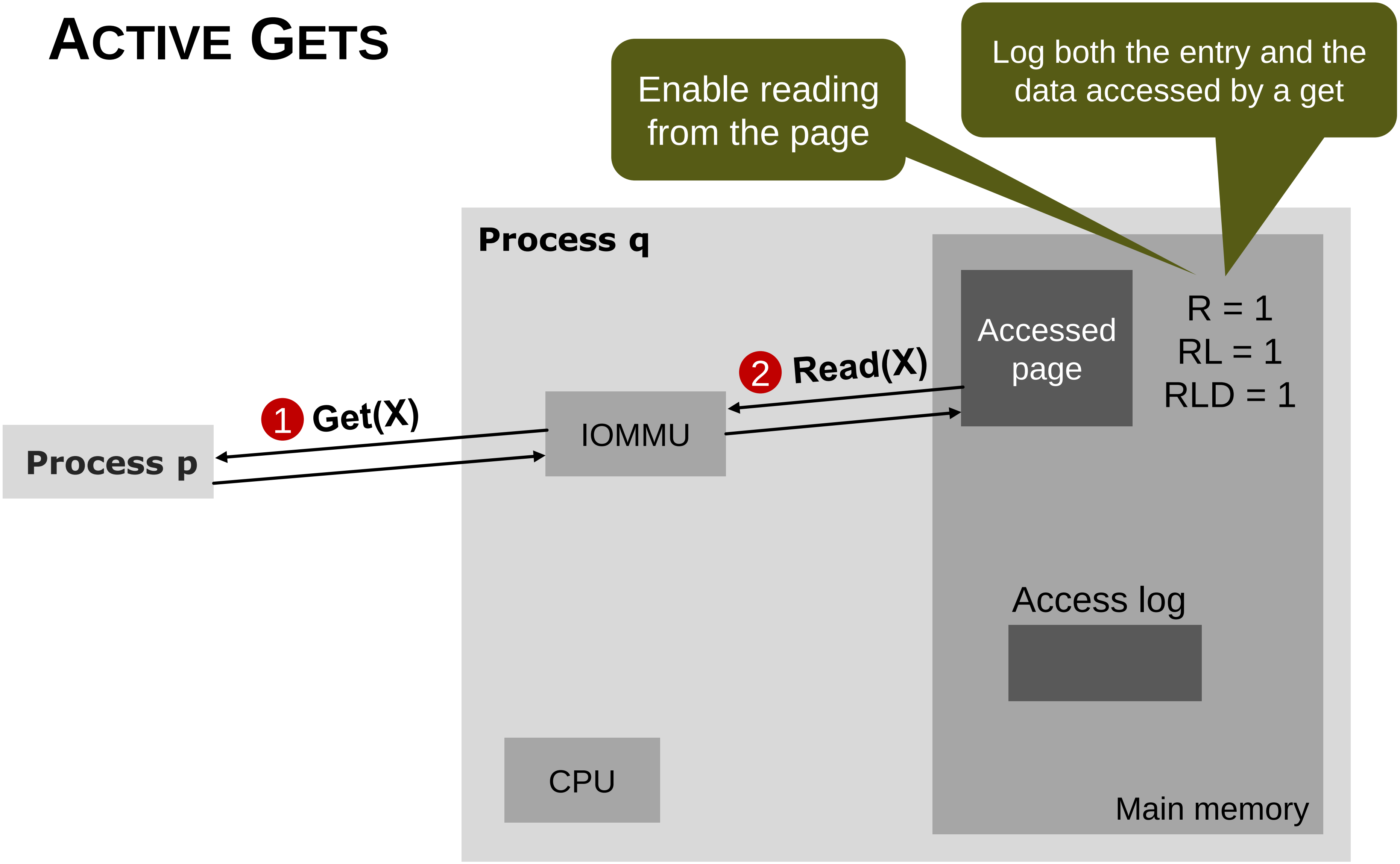
Log both the entry and the data accessed by a get



Active Gets


Enable reading from the page
Log both the entry and the data accessed by a get
Process q
Accessed page
R = 1
RL = 1
RLD = 1
2 Read(X)
1 Get(X)
Process p
IOMMU
Access log
CPU
Main memory

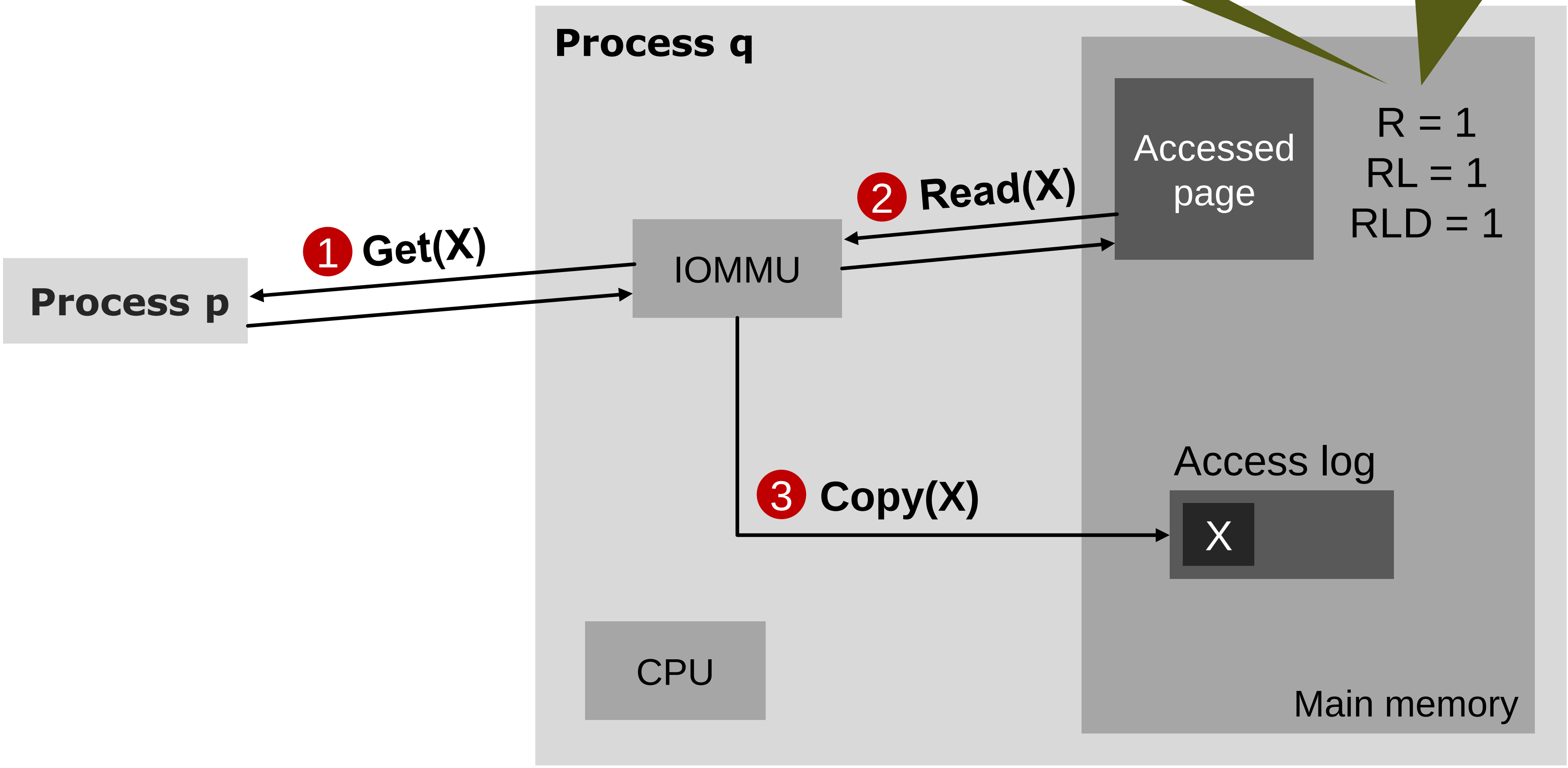
Active Gets


Enable reading from the page
Log both the entry and the data accessed by a get
Process q
R = 1
RL = 1
RLD = 1
Accessed page
2 Read(X)
1 Get(X)
IOMMU
Process p
Access log
CPU
Main memory

ACTIVE GETS

Enable reading from the page

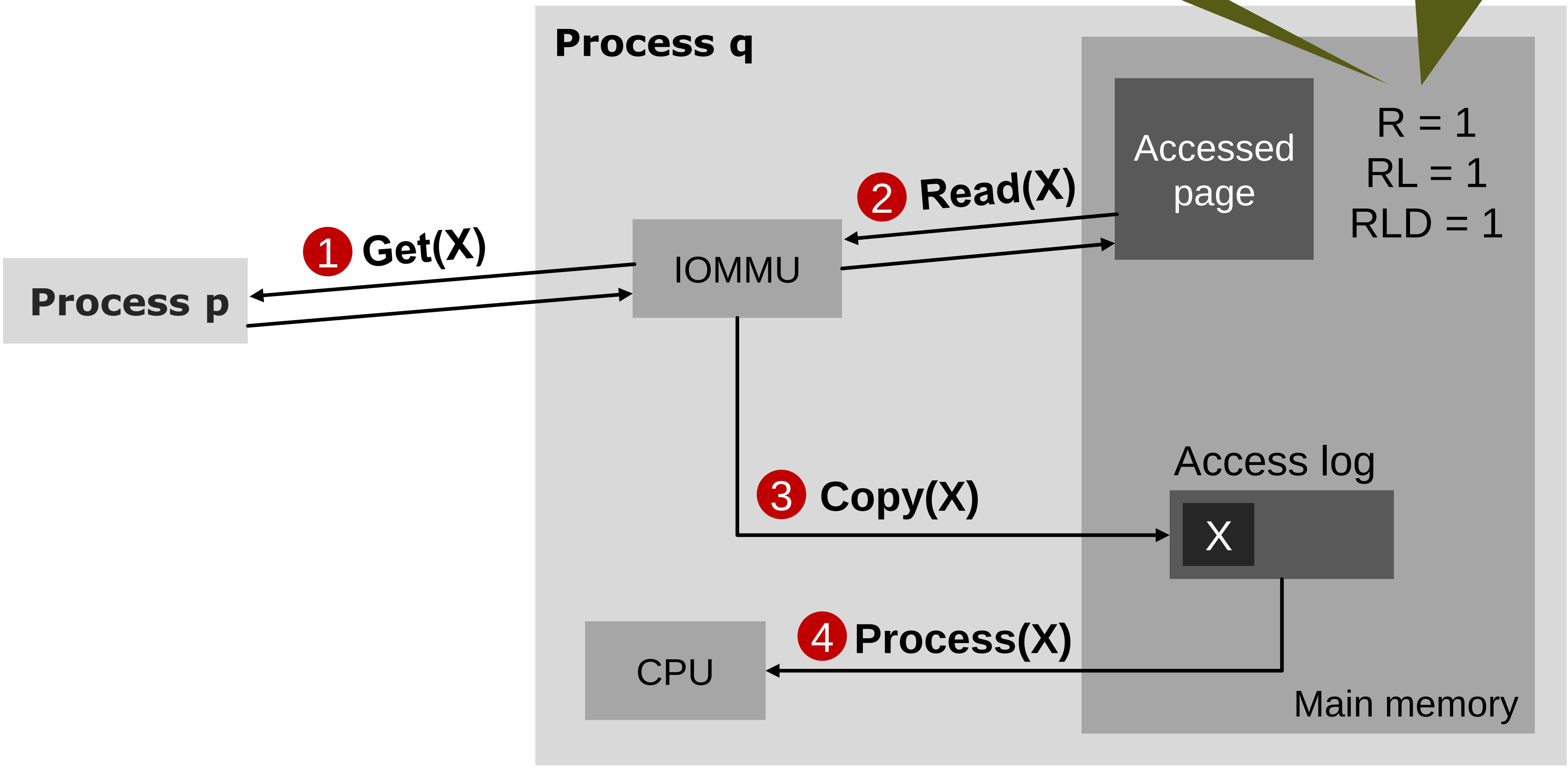
Log both the entry and the data accessed by a get



Active Gets

Enable reading from the page

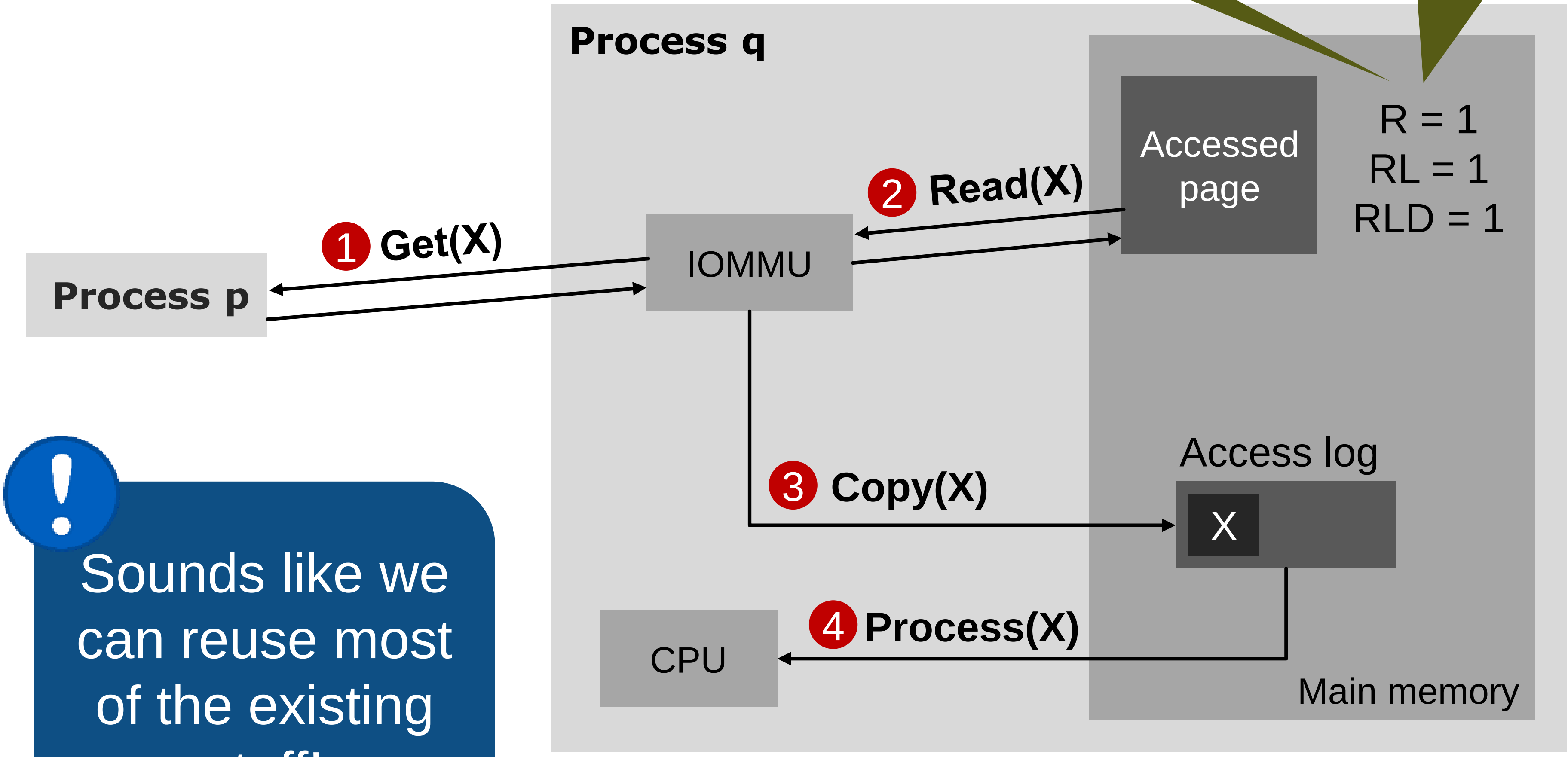
Log both the entry and the data accessed by a get



Active Gets

Enable reading from the page

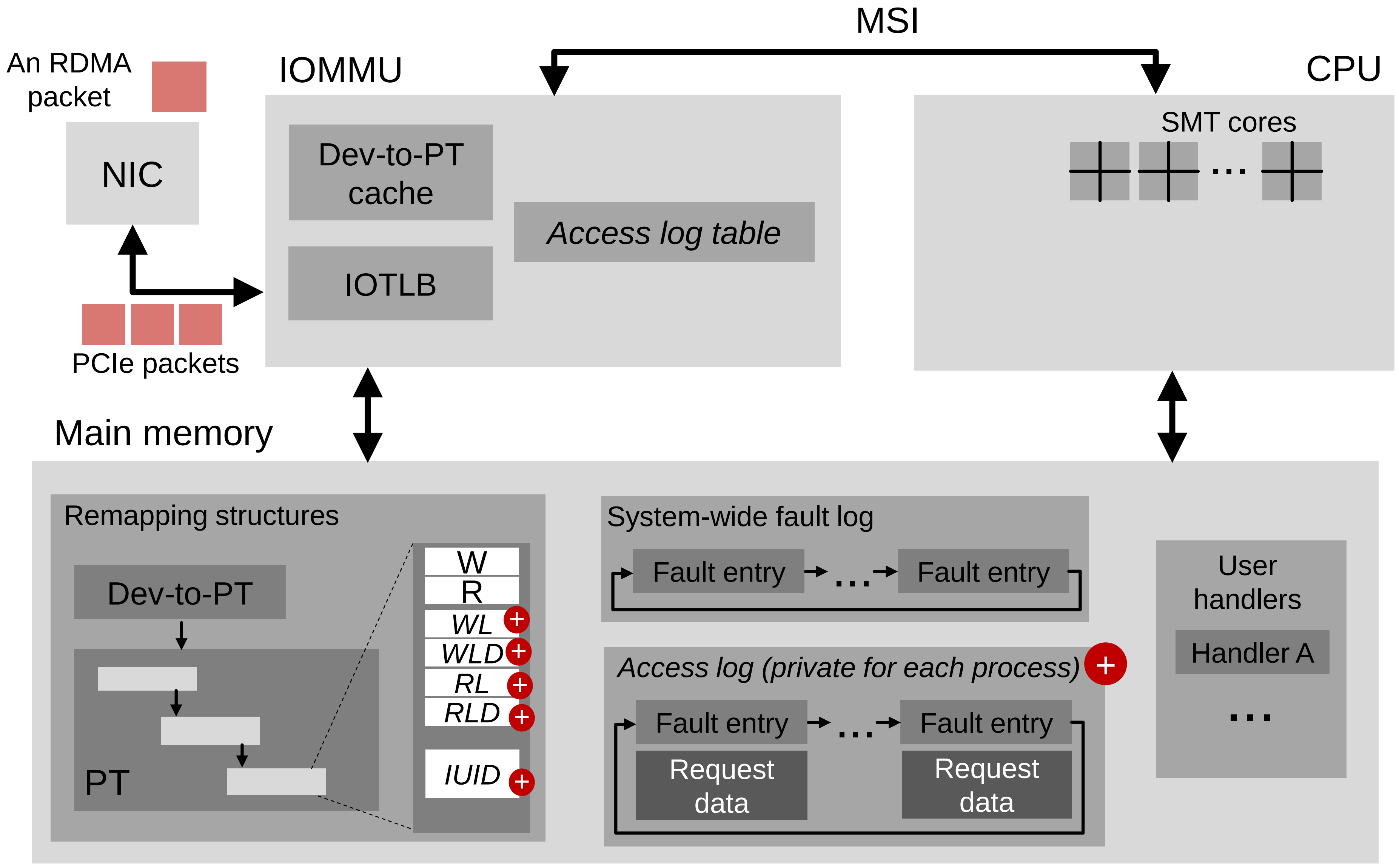
Log both the entry and the data accessed by a get



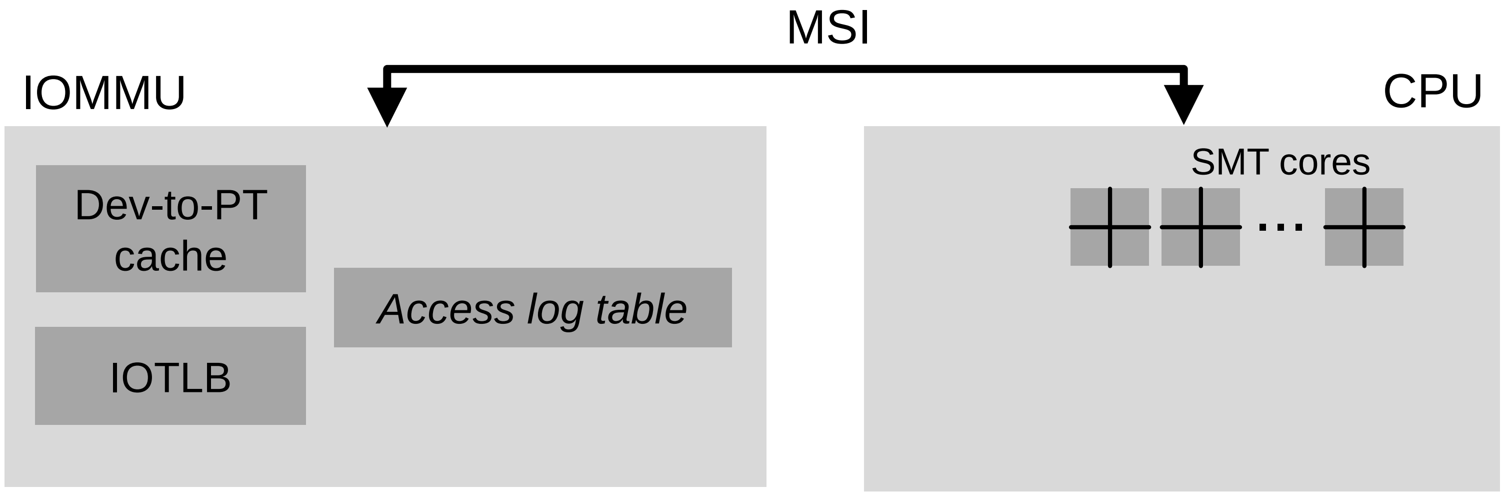
Sounds like we can reuse most of the existing stuff!

intel + PCI EXPRESS

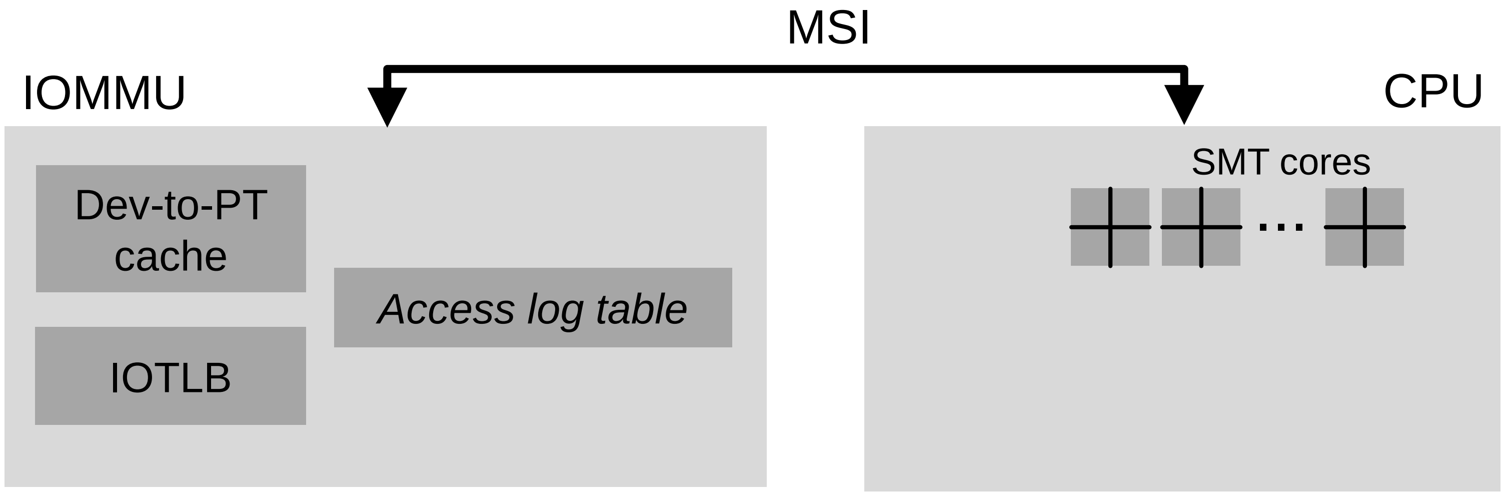
Interactions with the CPU



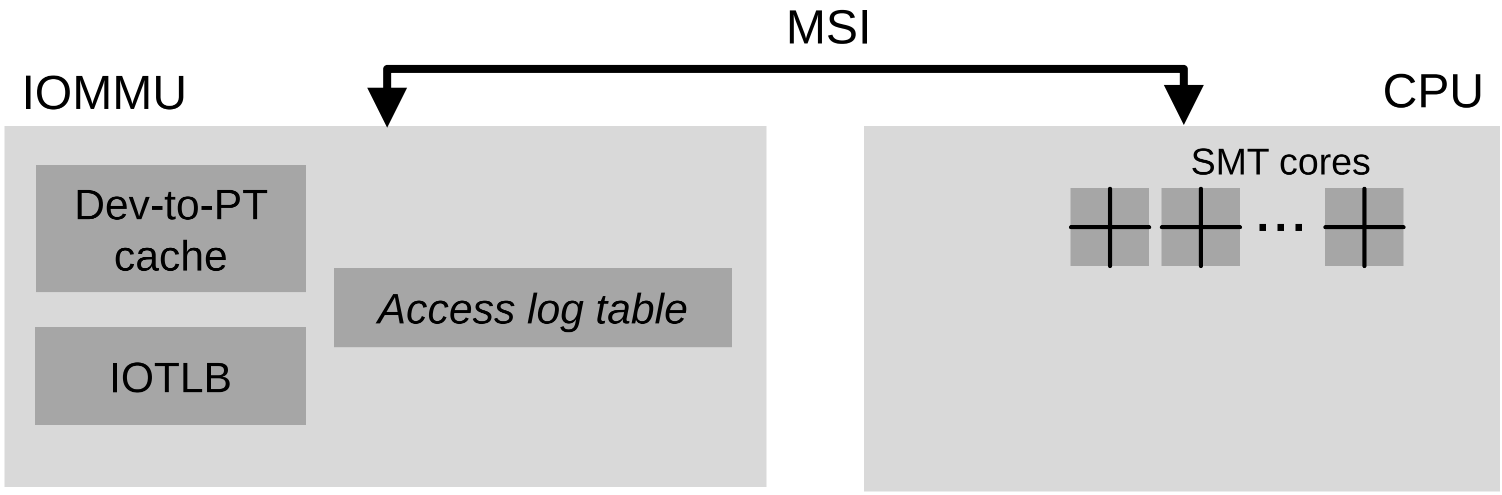
INTERACTIONS WITH THE CPU



INTERACTIONS WITH THE CPU

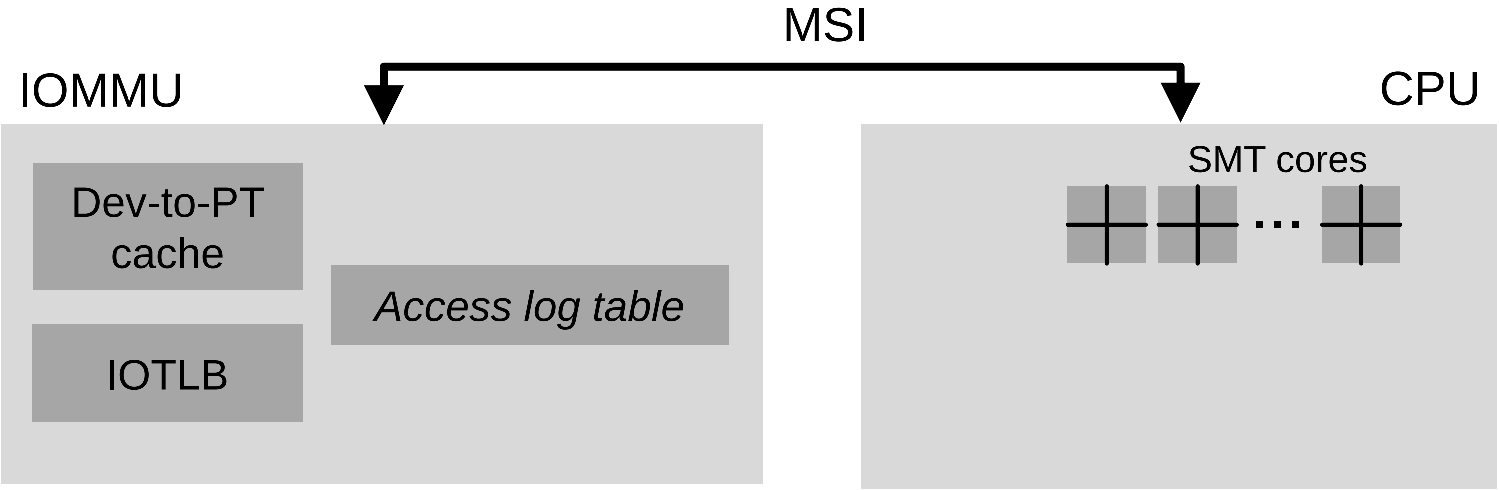


INTERACTIONS WITH THE CPU



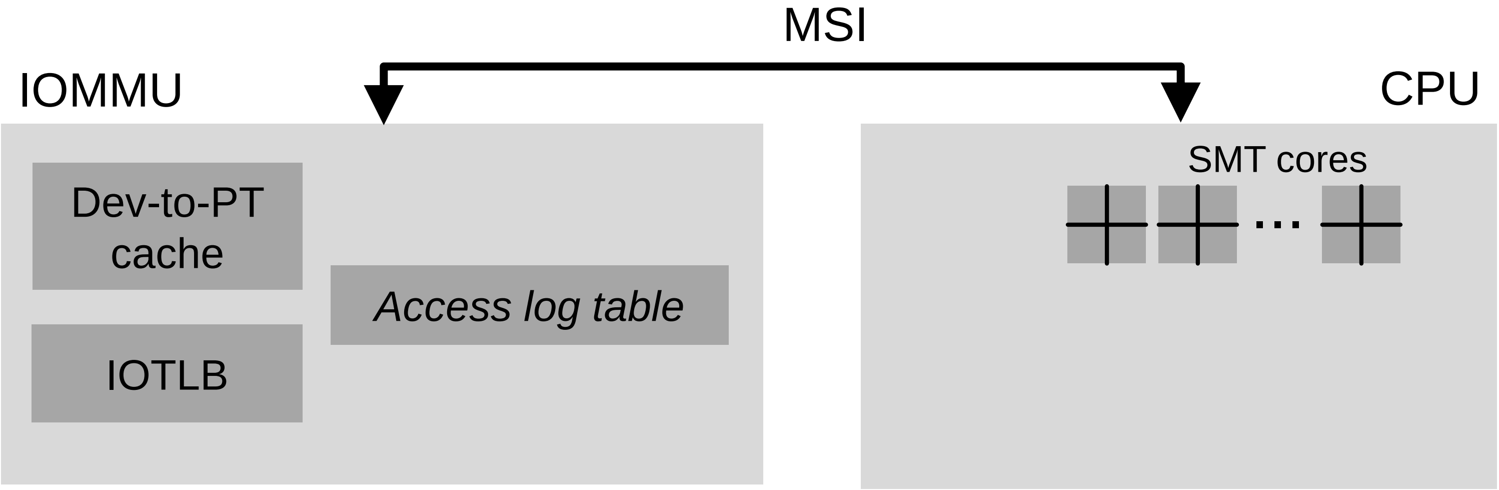
- Interrupts

INTERACTIONS WITH THE CPU



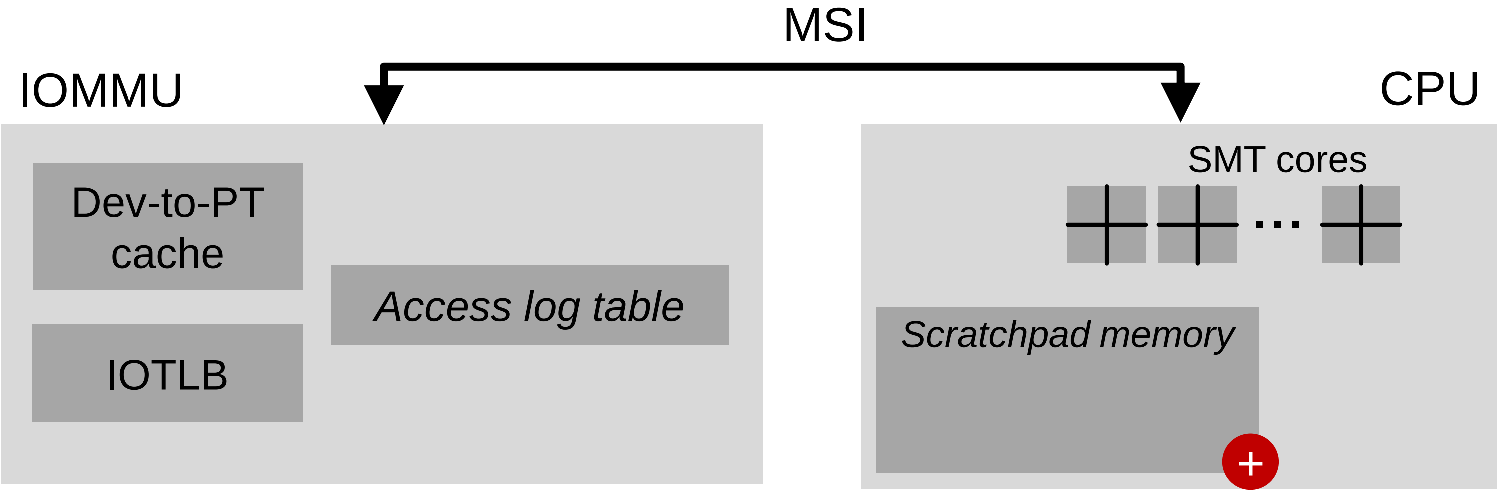
- Interrupts
- Polling

INTERACTIONS WITH THE CPU



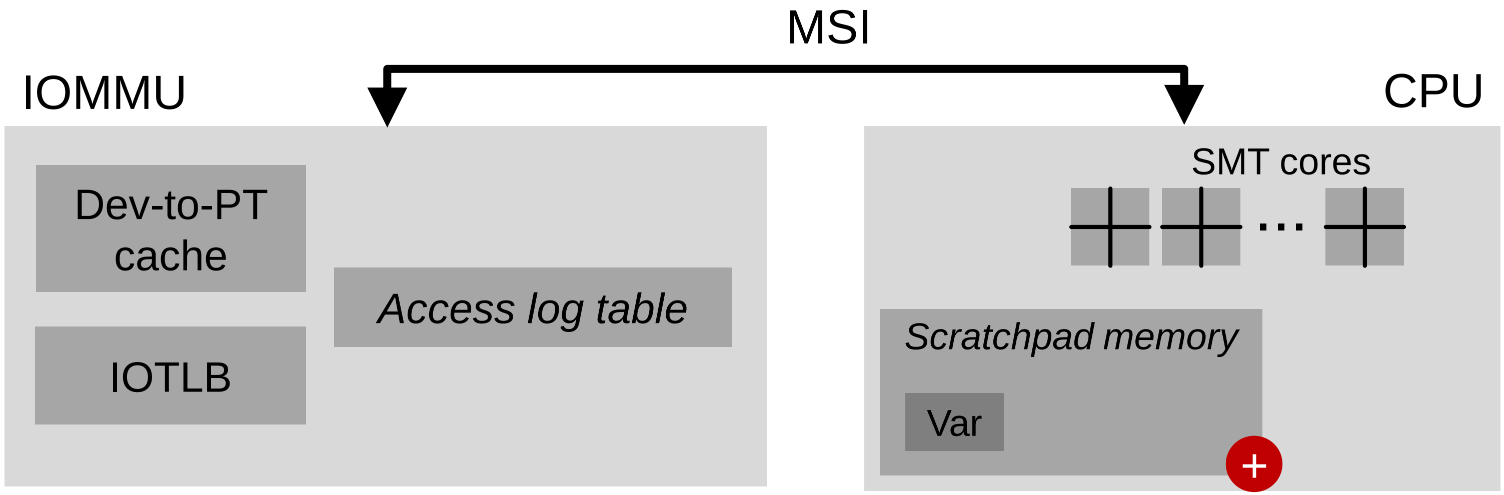
- Interrupts
- Polling
- Direct notifications via scratchpads

Interactions with the CPU



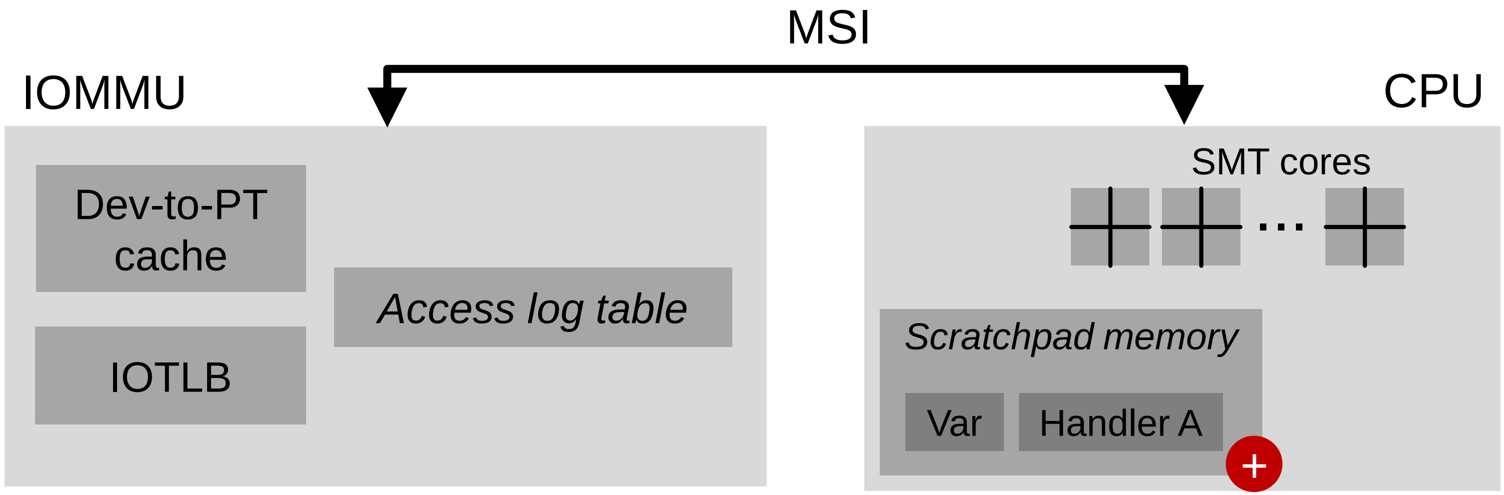
- Interrupts
- Polling
- Direct notifications via scratchpads

INTERACTIONS WITH THE CPU



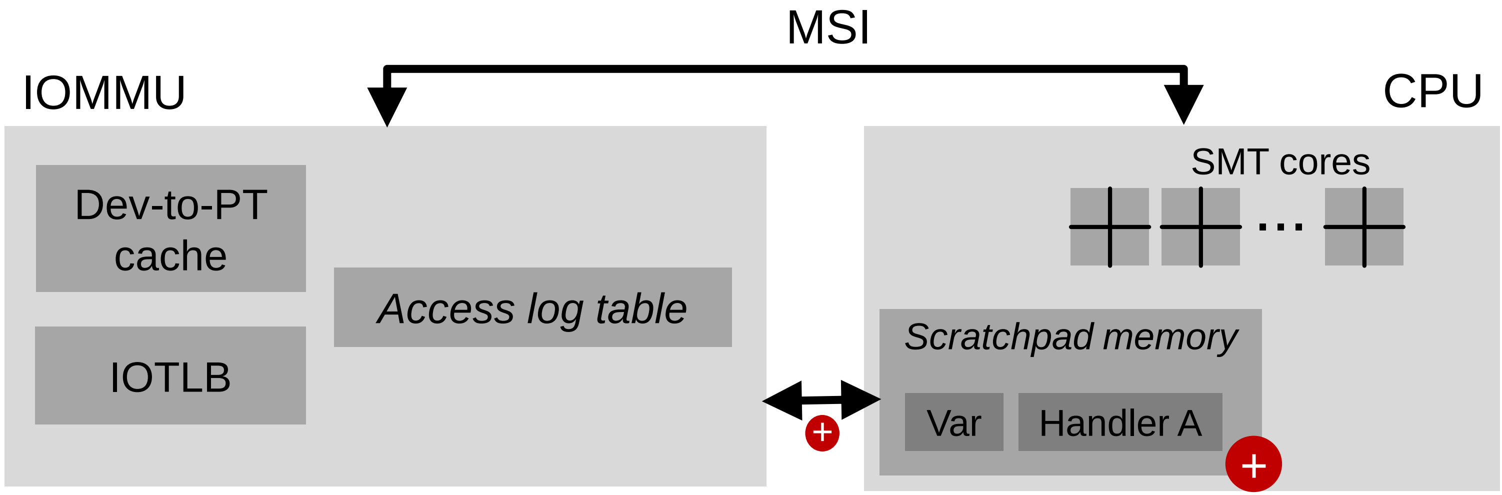
- Interrupts
- Polling
- Direct notifications via scratchpads

INTERACTIONS WITH THE CPU



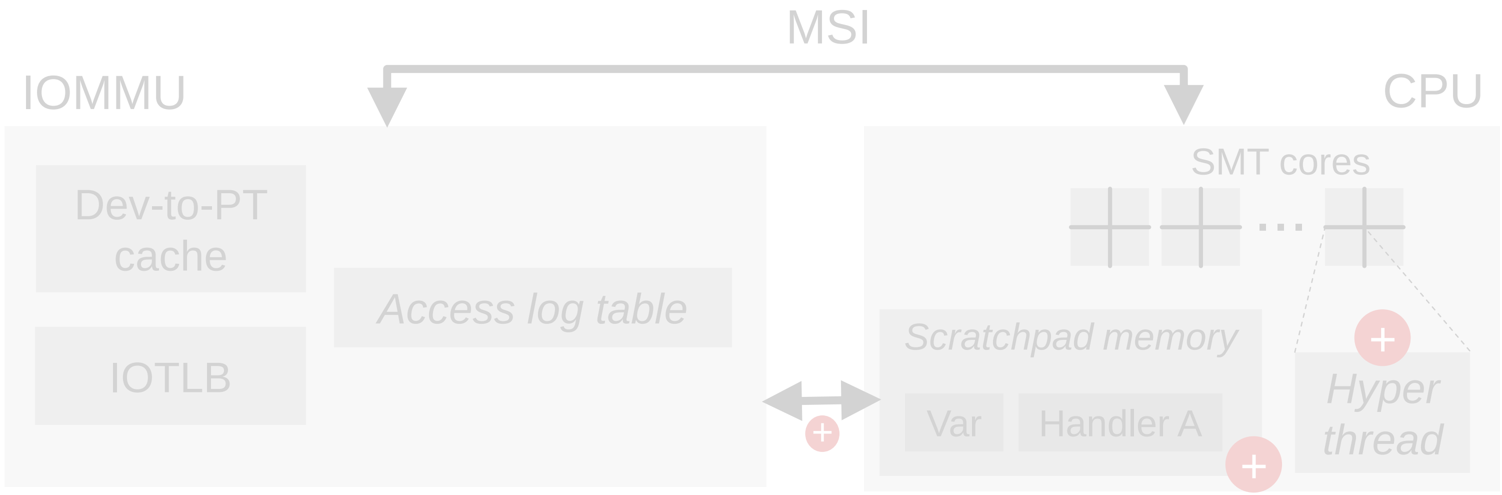
- Interrupts
- Polling
- Direct notifications via scratchpads

INTERACTIONS WITH THE CPU



- Interrupts
- Polling
- Direct notifications via scratchpads

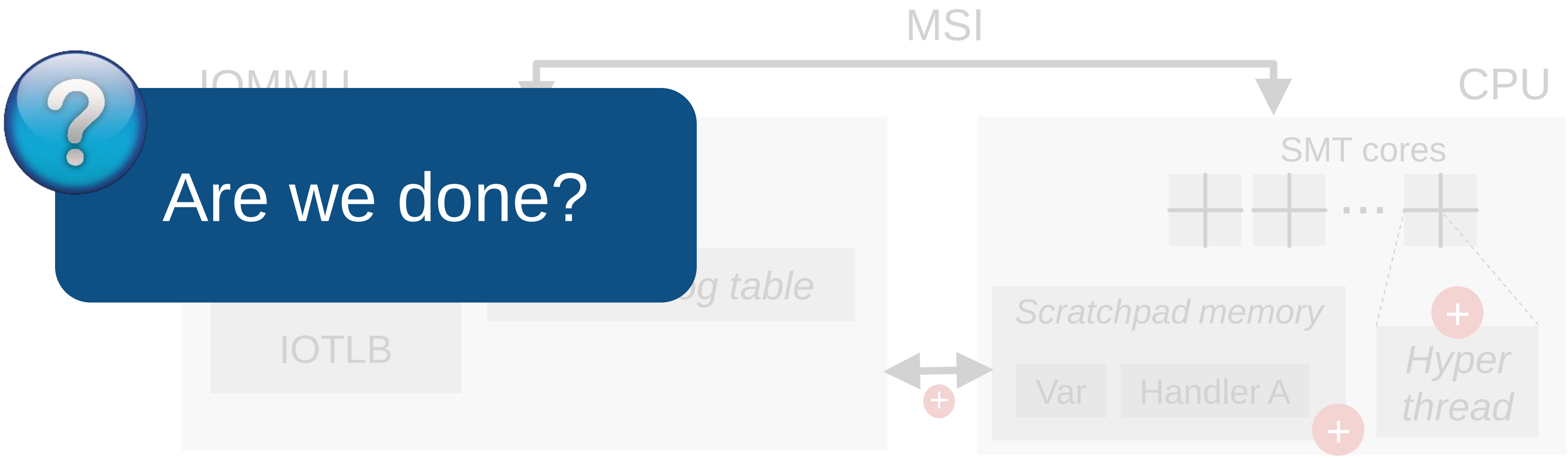
Interactions with the CPU



- Interrupts
- Polling
- Direct notifications via scratchpads

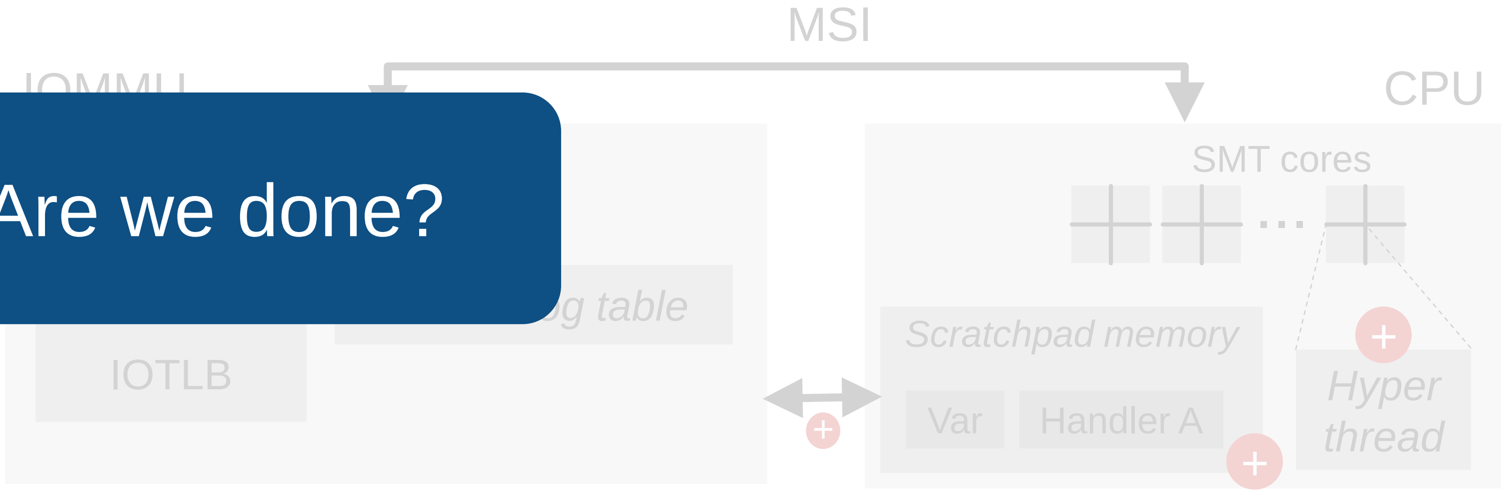
Interactions with the CPU

Are we done?



- Interrupts
- Polling
- Direct notifications via scratchpads

INTERACTIONS WITH THE CPU



Are we done?

- Interrupts
- Polling
- Direct notifications via sc

Well…

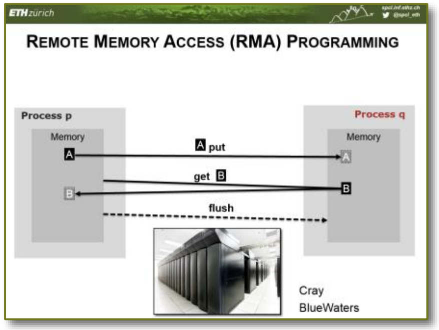
CONSISTENCY

CONSISTENCY

- A weak consistency model [1]

[1] K. Gharachorloo et al. Memory consistency and event ordering in scalable shared-memory multiprocessors. ISCA '90

CONSISTENCY

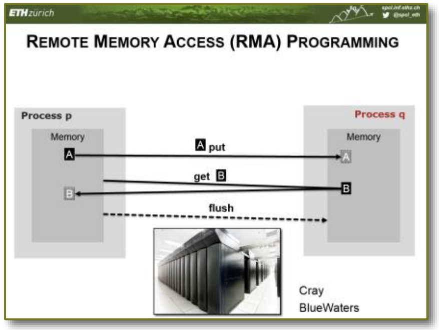


REMOTE MEMORY ACCESS (RMA) PROGRAMMING

- A weak consistency model [1]
    - Consistency on-demand

[1] K. Gharachorloo et al. Memory consistency and event ordering in scalable shared-memory multiprocessors. ISCA '90

CONSISTENCY

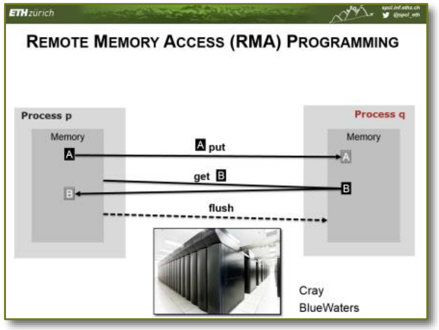


REMOTE MEMORY ACCESS (RMA) PROGRAMMING

- A weak consistency model [1]
  - Consistency on-demand
- active_flush(int target_id)

[1] K. Gharachorloo et al. Memory consistency and event ordering in scalable shared-memory multiprocessors. ISCA '90

CONSISTENCY

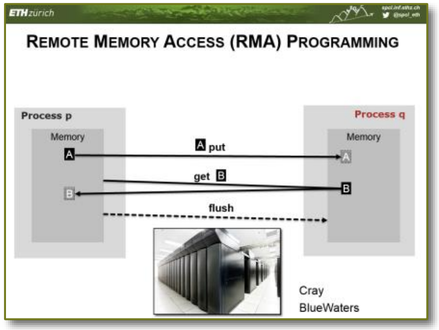


REMOTE MEMORY ACCESS (RMA) PROGRAMMING

- A weak consistency model [1]
  - Consistency on-demand
- active_flush(int target_id)
  - Enforces the completion of active accesses issued by the calling process and targeted at target_id

[1] K. Gharachorloo et al. Memory consistency and event ordering in scalable shared-memory multiprocessors. ISCA '90

CONSISTENCY

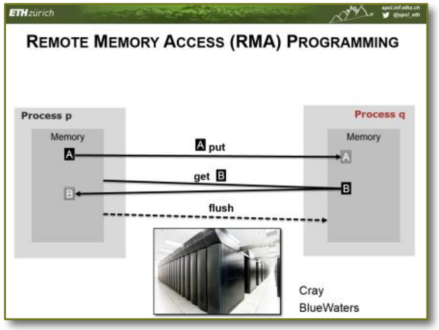


REMOTE MEMORY ACCESS (RMA) PROGRAMMING

- A weak consistency model [1]
  - Consistency on-demand
- active_flush(int target_id)
  - Enforces the completion of active accesses issued by the calling process and targeted at target_id
  - Implemented with an active get issued at a special *flushing page*

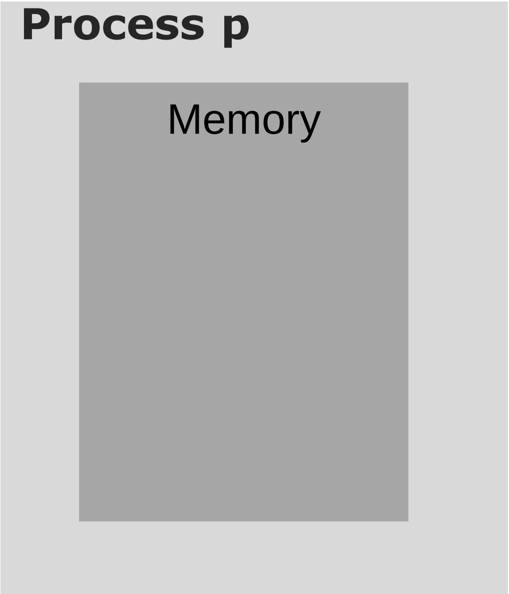
[1] K. Gharachorloo et al. Memory consistency and event ordering in scalable shared-memory multiprocessors. ISCA '90

CONSISTENCY



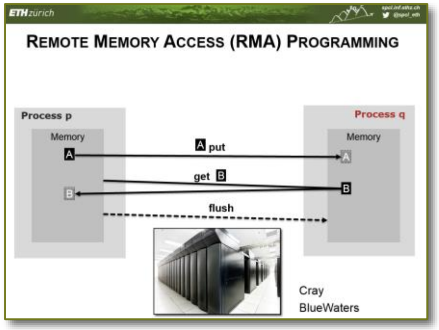
REMOTE MEMORY ACCESS (RMA) PROGRAMMING

- A weak consistency model [1]
  - Consistency on-demand
- active_flush(int target_id)
  - Enforces the completion of active accesses issued by the calling process and targeted at target_id
  - Implemented with an active get issued at a special *flushing page*



[1] K. Gharachorloo et al. Memory consistency and event ordering in scalable shared-memory multiprocessors. ISCA '90

Consistency



ETH zürich

spcl.inf.ethz.ch
@spcl_eth

Remote Memory Access (RMA) Programming

Process p

Memory

A

put

get B

B

flush

Process q

Memory

A

B

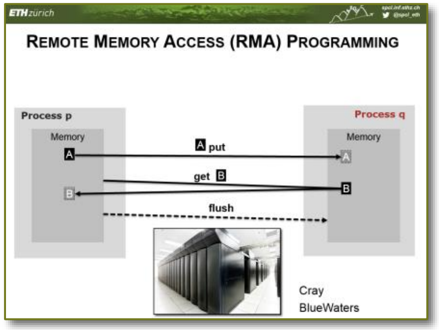
Cray
BlueWaters

- A weak consistency model [1]
  - Consistency on-demand
- active_flush(int target_id)
  - Enforces the completion of active accesses issued by the calling process and targeted at target_id
  - Implemented with an active get issued at a special *flushing page*

Process p
Memory
Process q
IOMMU
Memory

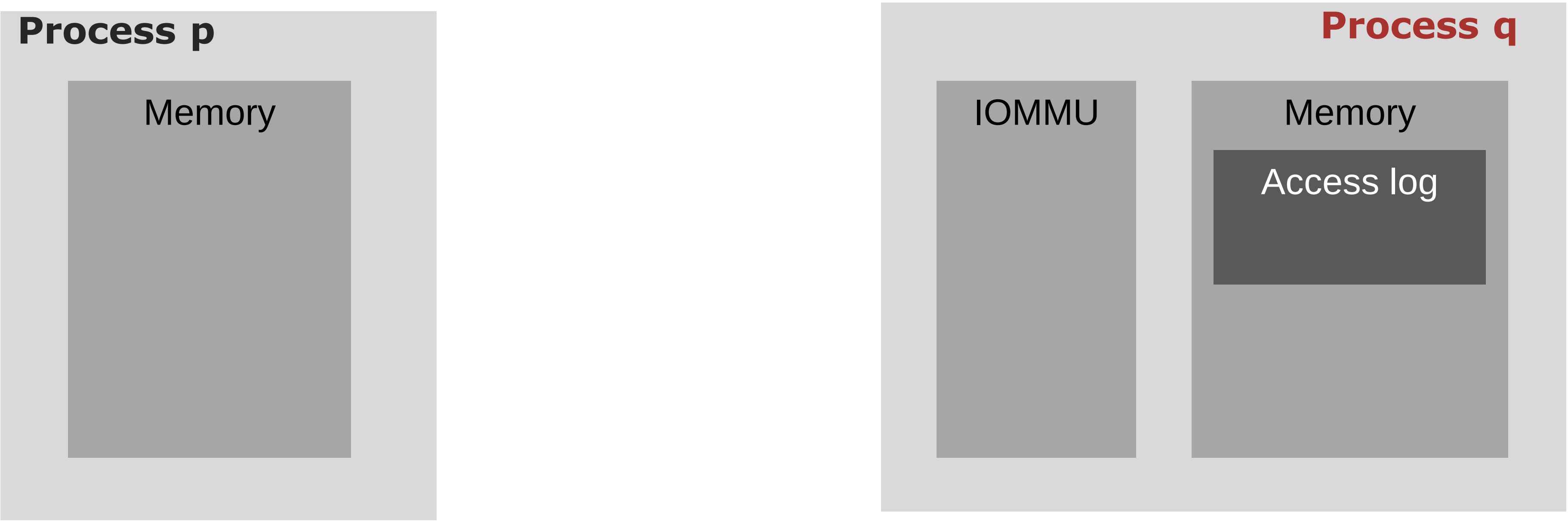
[1] K. Gharachorloo et al. Memory consistency and event ordering in scalable shared-memory multiprocessors. ISCA '90

Consistency



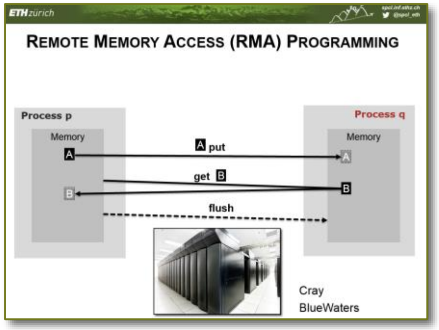
Remote Memory Access (RMA) Programming

- A weak consistency model [1]
  - Consistency on-demand
- active_flush(int target_id)
  - Enforces the completion of active accesses issued by the calling process and targeted at target_id
  - Implemented with an active get issued at a special *flushing page*



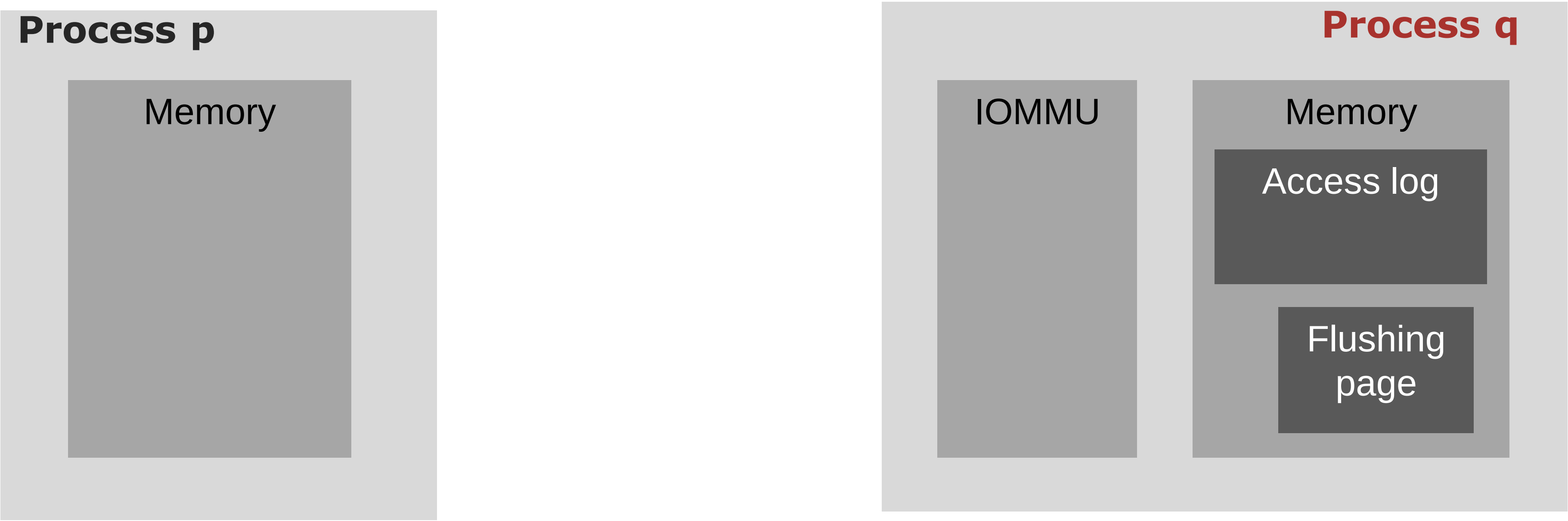
[1] K. Gharachorloo et al. Memory consistency and event ordering in scalable shared-memory multiprocessors. ISCA '90

CONSISTENCY



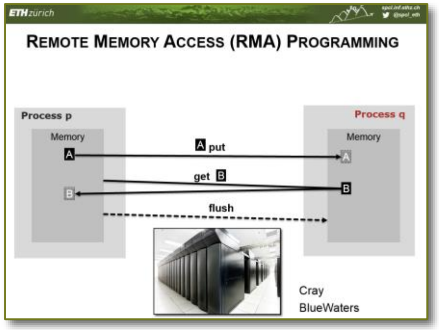
REMOTE MEMORY ACCESS (RMA) PROGRAMMING

- A weak consistency model [1]
  - Consistency on-demand
- active_flush(int target_id)
  - Enforces the completion of active accesses issued by the calling process and targeted at target_id
  - Implemented with an active get issued at a special *flushing page*



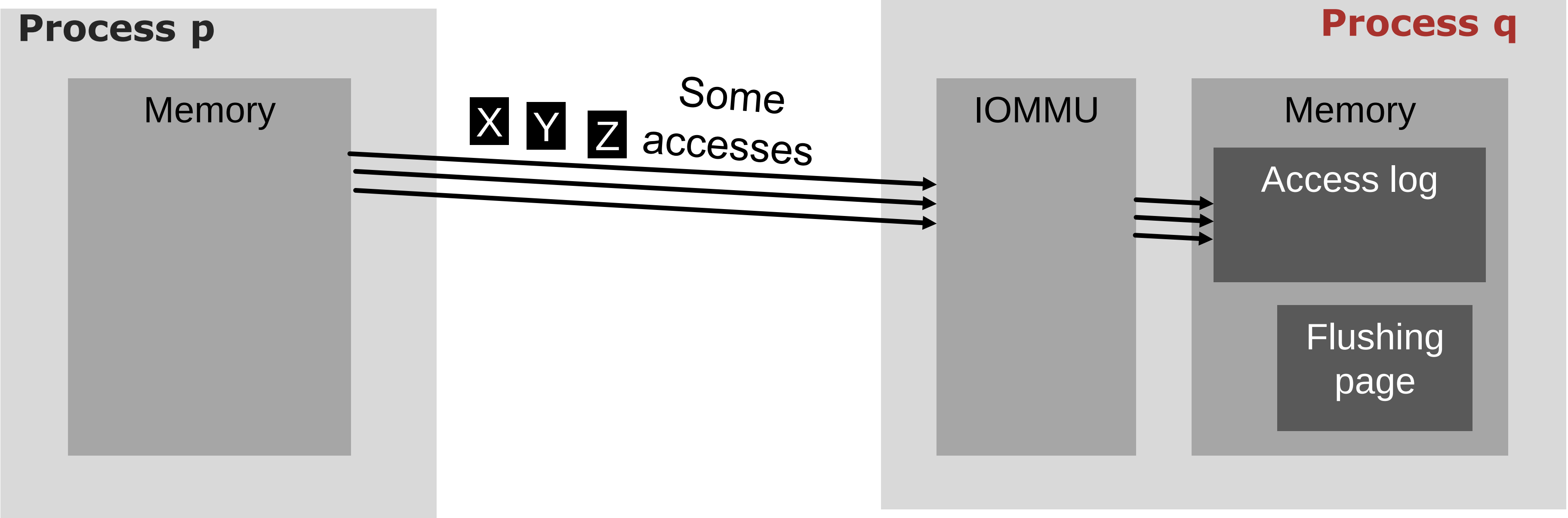
[1] K. Gharachorloo et al. Memory consistency and event ordering in scalable shared-memory multiprocessors. ISCA '90

CONSISTENCY



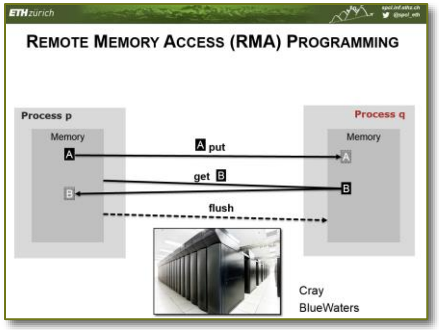
REMOTE MEMORY ACCESS (RMA) PROGRAMMING

- A weak consistency model [1]
  - Consistency on-demand
- active_flush(int target_id)
  - Enforces the completion of active accesses issued by the calling process and targeted at target_id
  - Implemented with an active get issued at a special *flushing page*

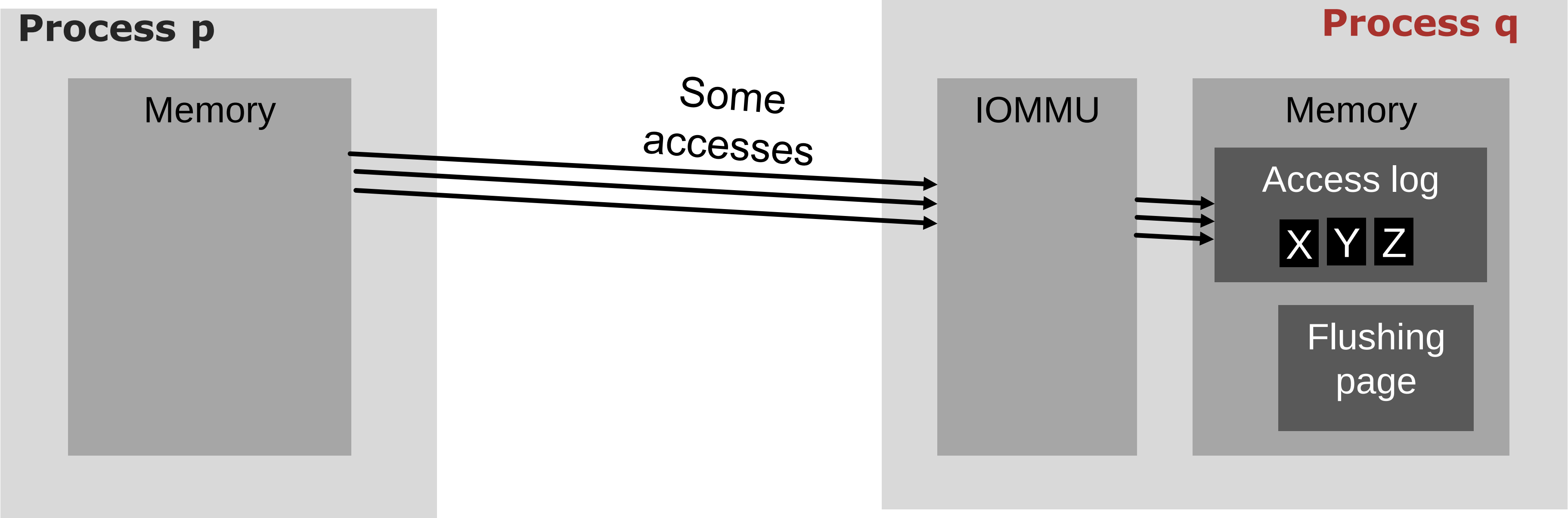


[1] K. Gharachorloo et al. Memory consistency and event ordering in scalable shared-memory multiprocessors. ISCA '90

Consistency

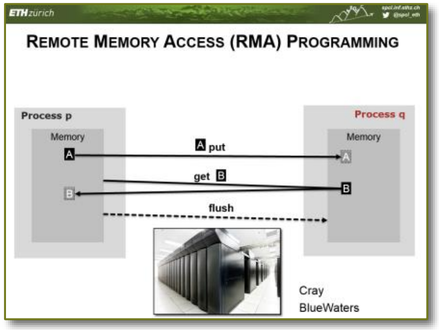

REMOTE MEMORY ACCESS (RMA) PROGRAMMING

- A weak consistency model [1]
  - Consistency on-demand
- active_flush(int target_id)
  - Enforces the completion of active accesses issued by the calling process and targeted at target_id
  - Implemented with an active get issued at a special *flushing page*



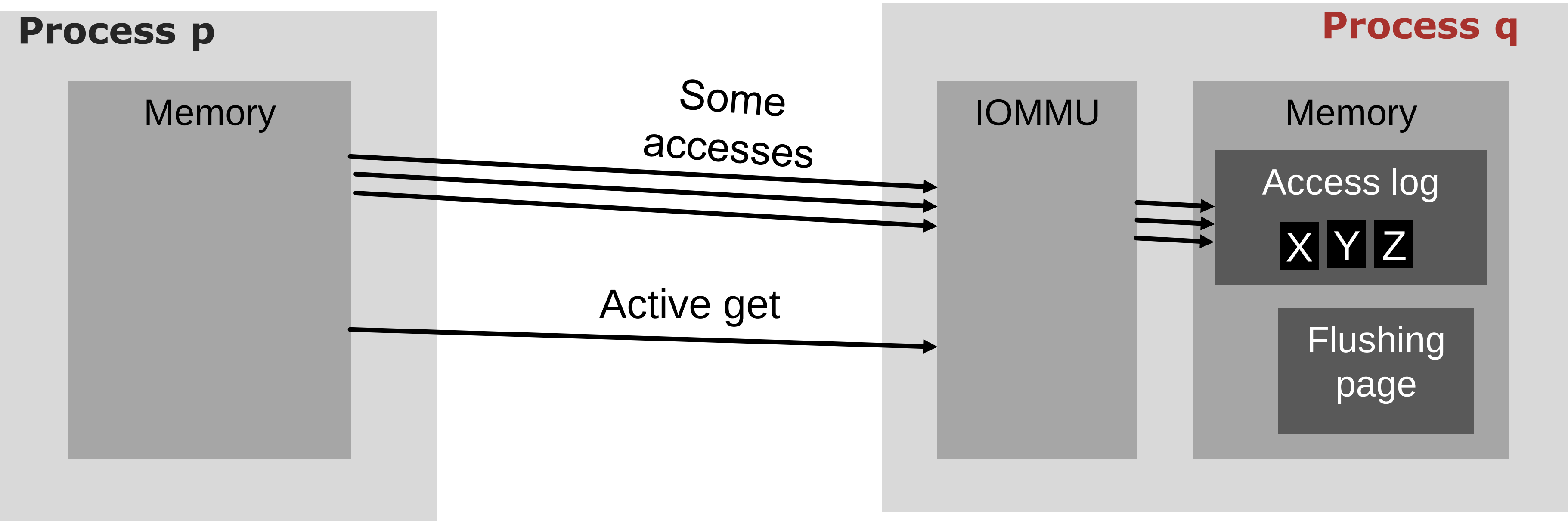
[1] K. Gharachorloo et al. Memory consistency and event ordering in scalable shared-memory multiprocessors. ISCA '90

Consistency



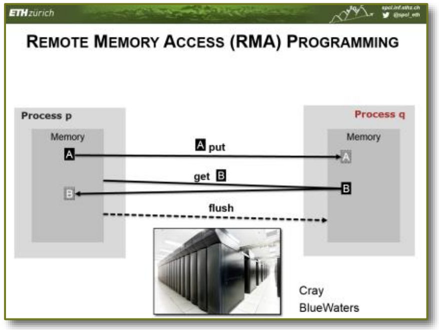
Remote Memory Access (RMA) Programming

- A weak consistency model [1]
  - Consistency on-demand
- active_flush(int target_id)
  - Enforces the completion of active accesses issued by the calling process and targeted at target_id
  - Implemented with an active get issued at a special *flushing page*



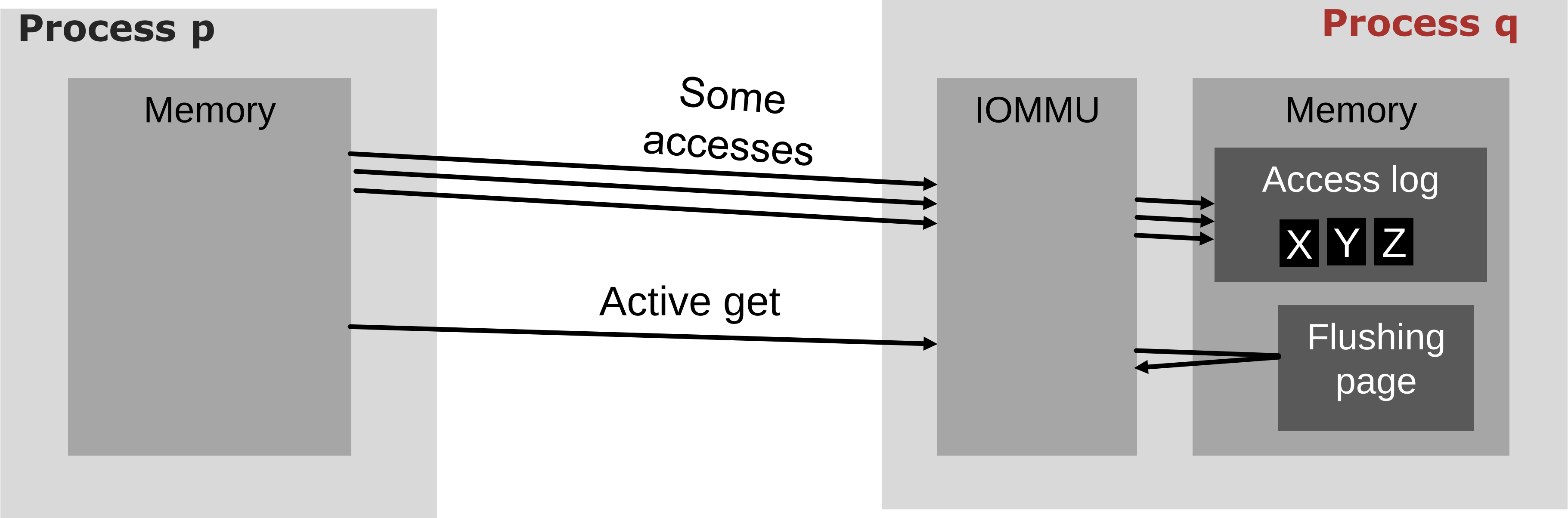
[1] K. Gharachorloo et al. Memory consistency and event ordering in scalable shared-memory multiprocessors. ISCA '90

Consistency



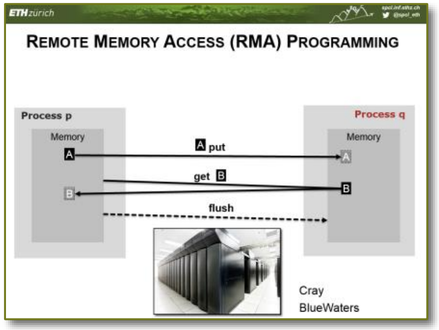
Remote Memory Access (RMA) Programming

- A weak consistency model [1]
  - Consistency on-demand
- active_flush(int target_id)
  - Enforces the completion of active accesses issued by the calling process and targeted at target_id
  - Implemented with an active get issued at a special *flushing page*

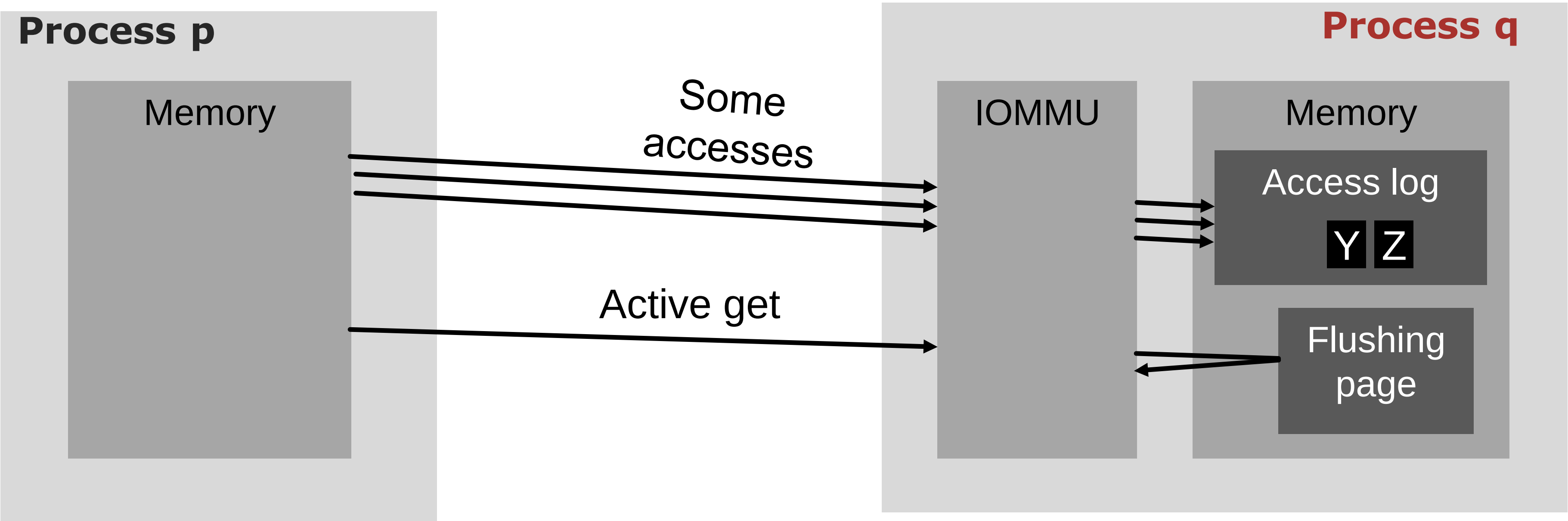


[1] K. Gharachorloo et al. Memory consistency and event ordering in scalable shared-memory multiprocessors. ISCA '90

Consistency

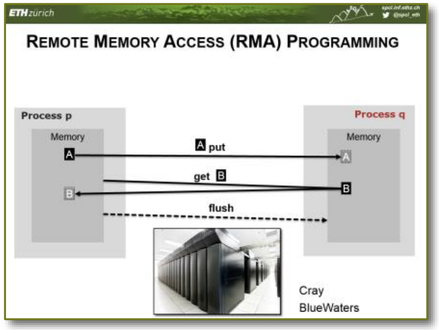

Remote Memory Access (RMA) Programming

- A weak consistency model [1]
  - Consistency on-demand
- active_flush(int target_id)
  - Enforces the completion of active accesses issued by the calling process and targeted at target_id
  - Implemented with an active get issued at a special *flushing page*



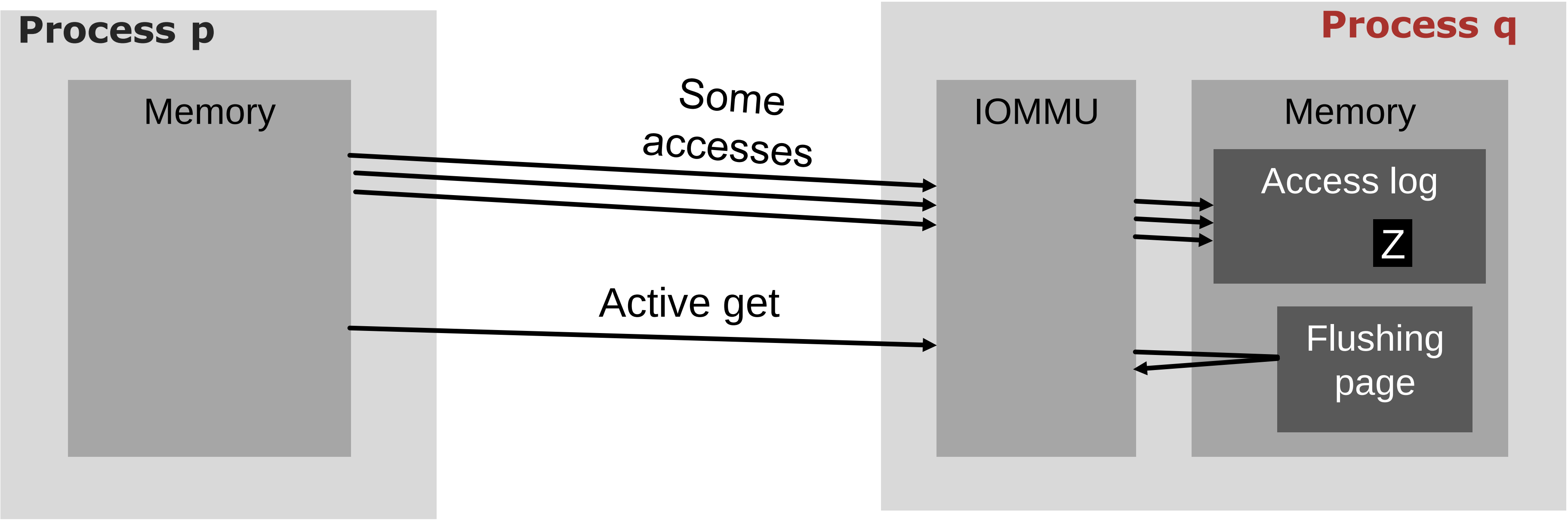
[1] K. Gharachorloo et al. Memory consistency and event ordering in scalable shared-memory multiprocessors. ISCA '90

CONSISTENCY



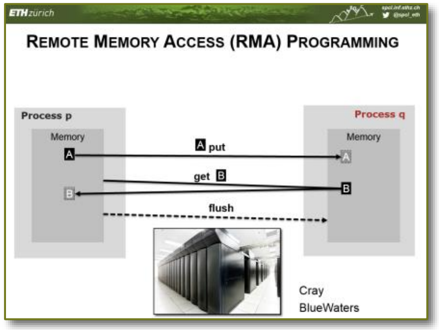
REMOTE MEMORY ACCESS (RMA) PROGRAMMING

- A weak consistency model [1]
  - Consistency on-demand
- active_flush(int target_id)
  - Enforces the completion of active accesses issued by the calling process and targeted at target_id
  - Implemented with an active get issued at a special *flushing page*



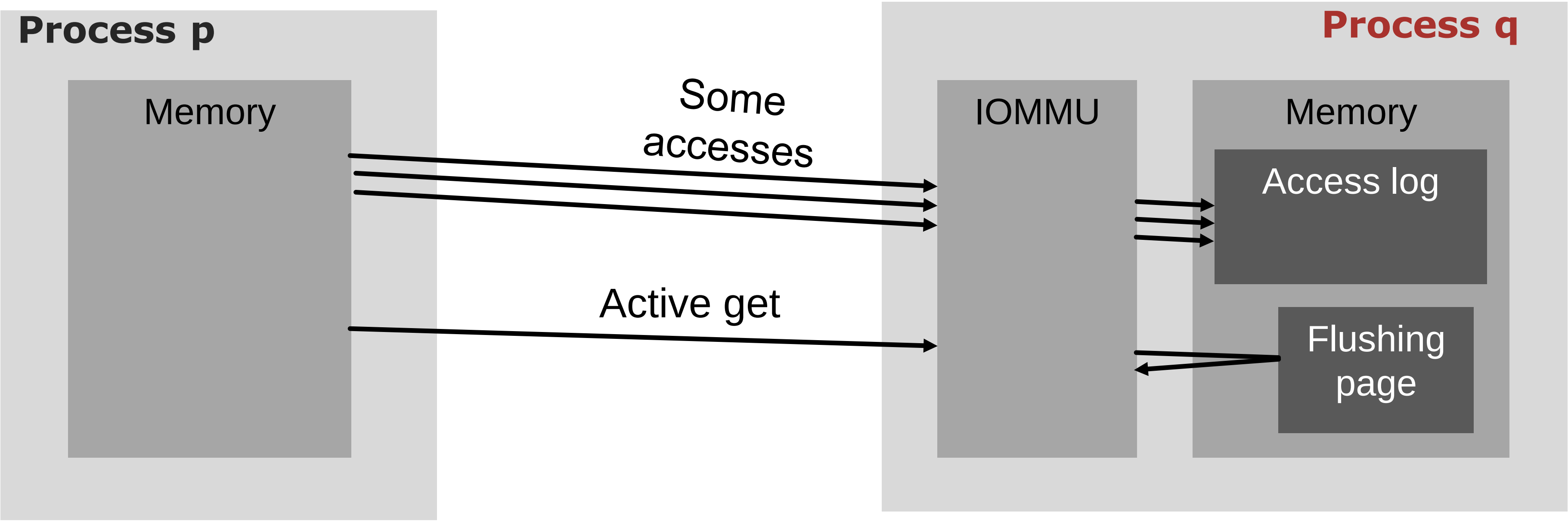
[1] K. Gharachorloo et al. Memory consistency and event ordering in scalable shared-memory multiprocessors. ISCA '90

Consistency



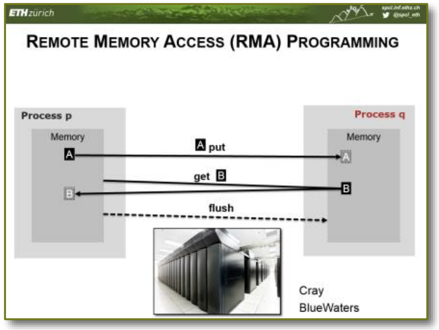
Remote Memory Access (RMA) Programming

- A weak consistency model [1]
  - Consistency on-demand
- active_flush(int target_id)
  - Enforces the completion of active accesses issued by the calling process and targeted at target_id
  - Implemented with an active get issued at a special *flushing page*



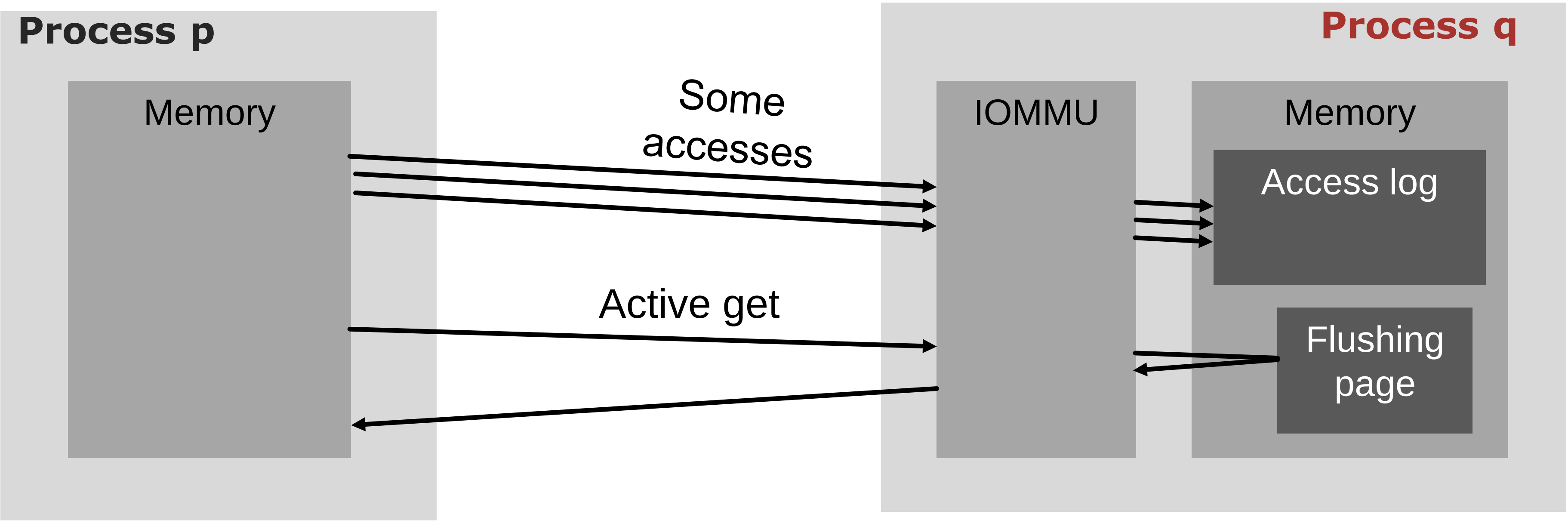
[1] K. Gharachorloo et al. Memory consistency and event ordering in scalable shared-memory multiprocessors. ISCA '90

Consistency



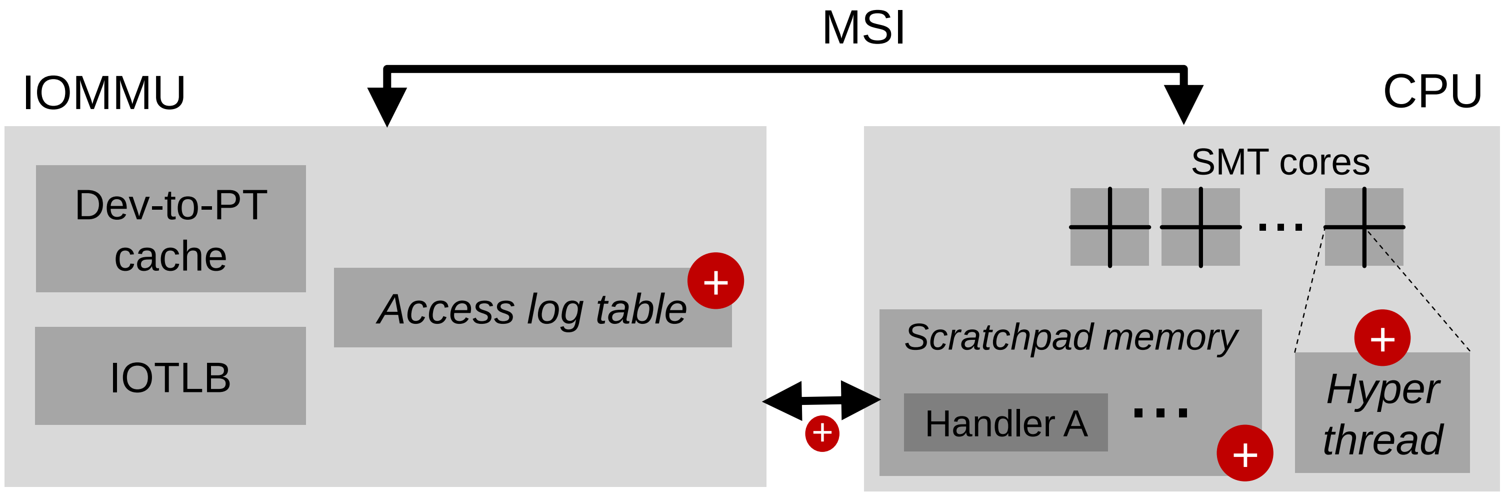
REMOTE MEMORY ACCESS (RMA) PROGRAMMING

- A weak consistency model [1]
  - Consistency on-demand
- active_flush(int target_id)
  - Enforces the completion of active accesses issued by the calling process and targeted at target_id
  - Implemented with an active get issued at a special *flushing page*

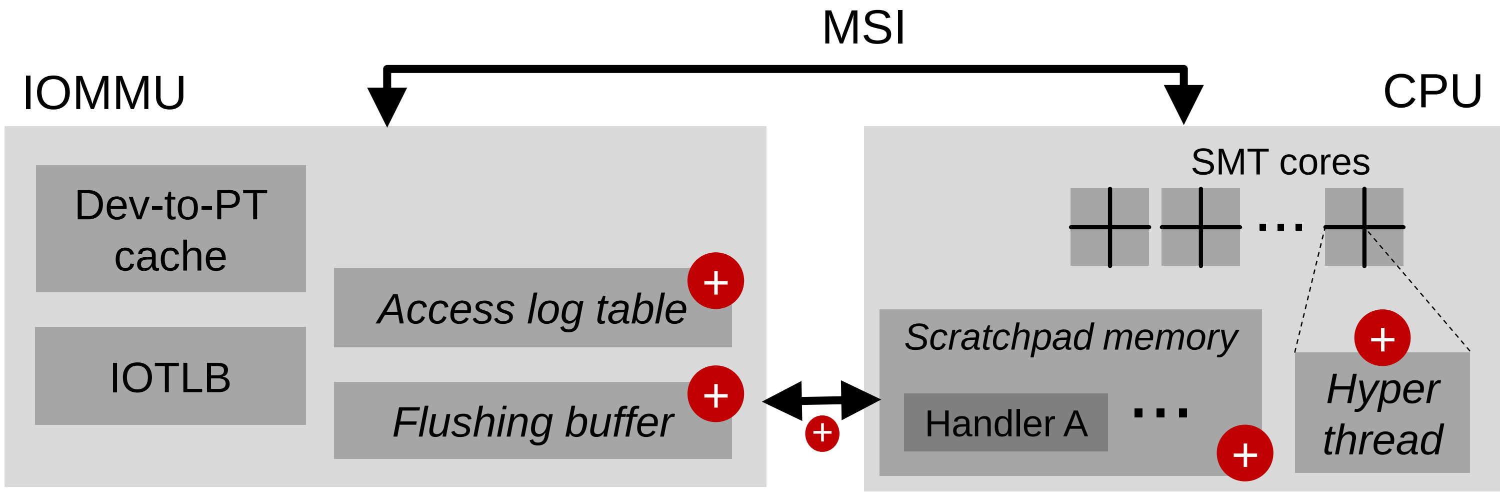


[1] K. Gharachorloo et al. Memory consistency and event ordering in scalable shared-memory multiprocessors. ISCA '90

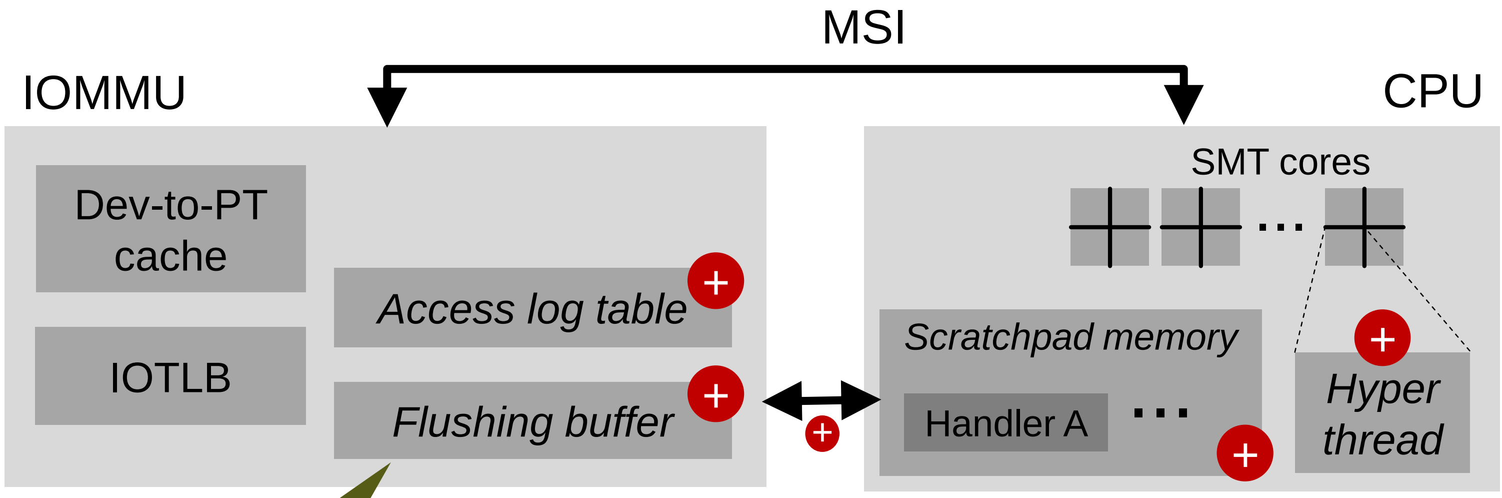
Consistency



Consistency

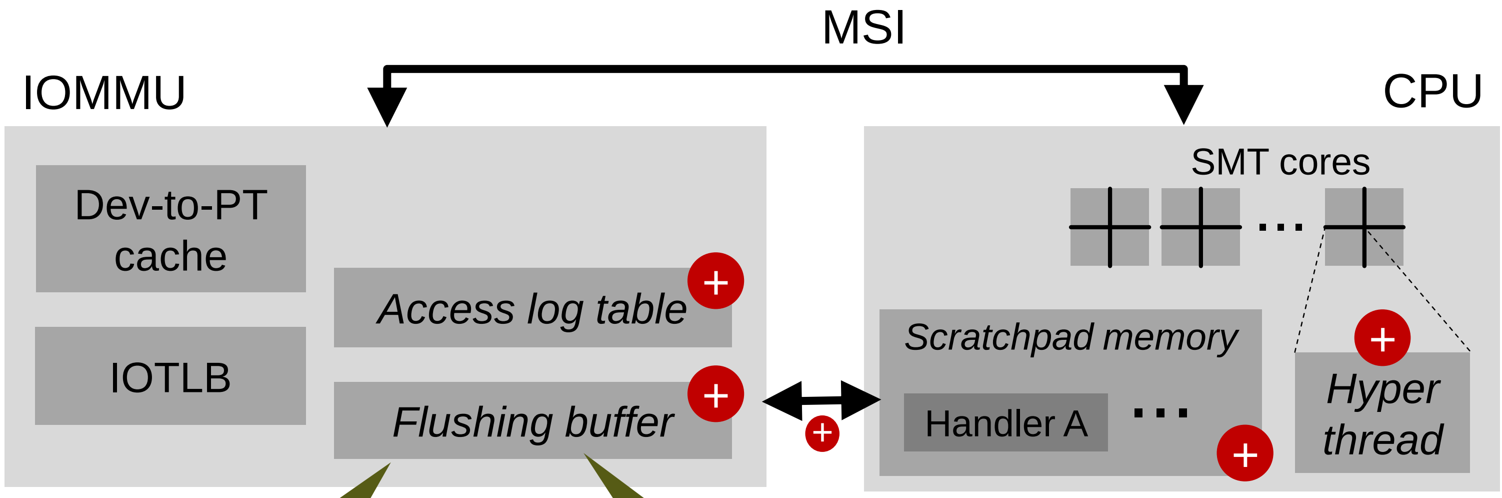


Consistency



Contains the addresses of flushing pages

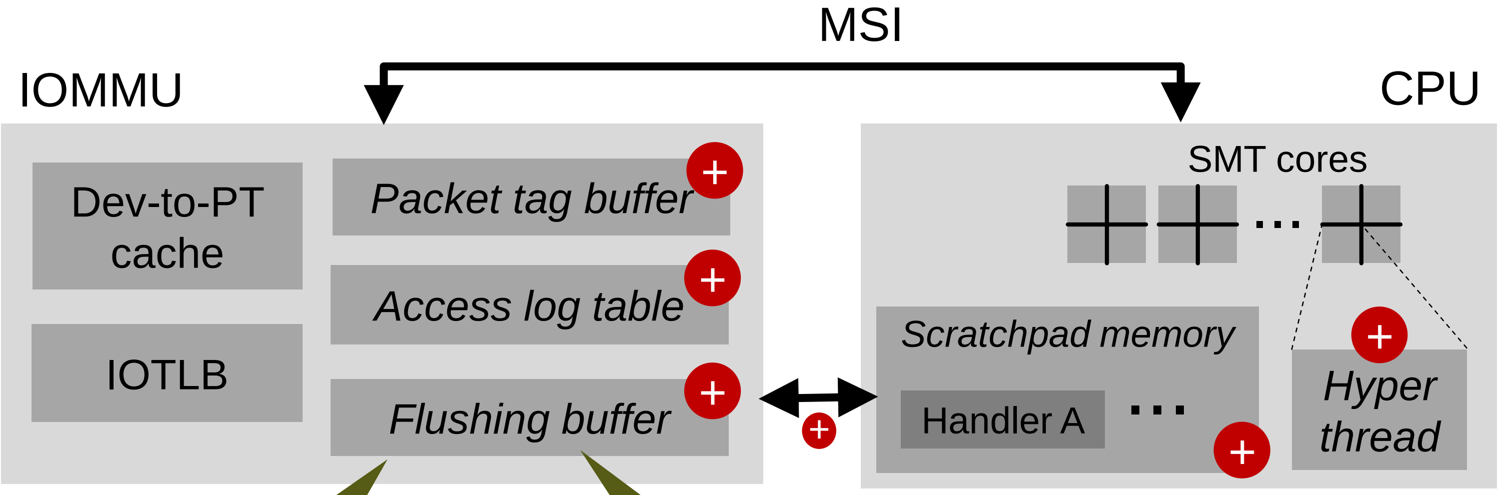
Consistency



Contains the addresses of flushing pages

Maps flushing pages to IUIDs and access logs

Consistency

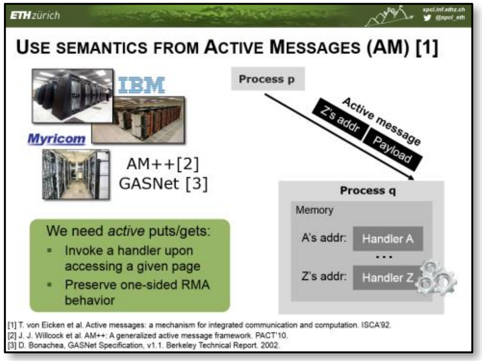


Contains the addresses of flushing pages

Maps flushing pages to IUIDs and access logs

Let’s summarize…

Let's summarize…



USE SEMANTICS FROM ACTIVE MESSAGES (AM) [1]

AM++[2]
GASNet [3]

We need *active* puts/gets:

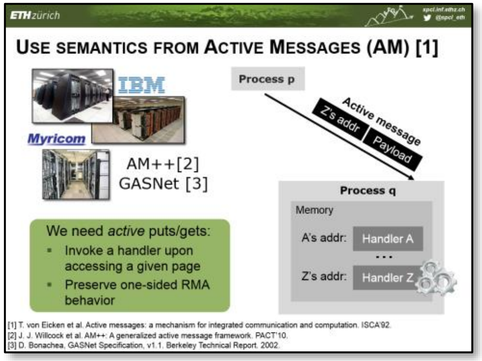
- Invoke a handler upon accessing a given page
- Preserve one-sided RMA behavior

[1] T. von Eicken et al. Active messages: a mechanism for integrated communication and computation. ISCA'92.
[2] J. J. Willcock et al. AM++: A generalized active message framework. PACT'10.
[3] D. Bonachea, GASNet Specification, v1.1. Berkeley Technical Report. 2002.

Active Messages

Let’s summarize…

IOMMUs



Use semantics from Active Messages (AM) [1]

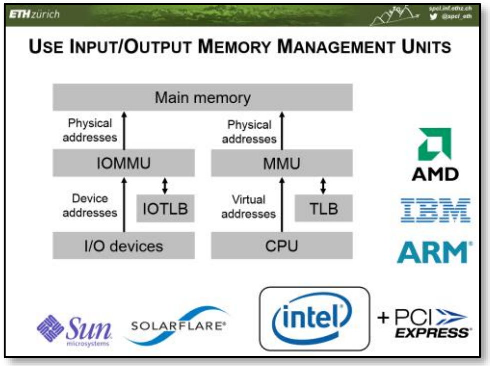
AM++[2]
GASNet [3]

We need *active* puts/gets:

- Invoke a handler upon accessing a given page
- Preserve one-sided RMA behavior

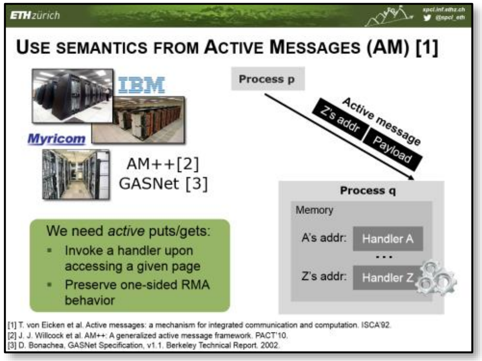
[1] T. von Eicken et al. Active messages: a mechanism for integrated communication and computation. ISCA'92.
[2] J. J. Willcock et al. AM++: A generalized active message framework. PACT'10.
[3] D. Bonachea, GASNet Specification, v1.1. Berkeley Technical Report. 2002.

Active Messages



Use Input/Output Memory Management Units

Let’s summarize…



Use semantics from Active Messages (AM) [1]

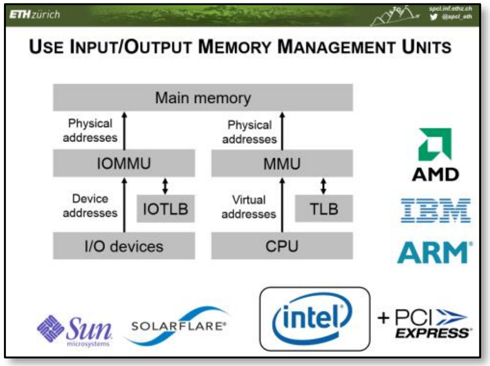
We need *active* puts/gets:

- Invoke a handler upon accessing a given page
- Preserve one-sided RMA behavior

[1] T. von Eicken et al. Active messages: a mechanism for integrated communication and computation. ISCA'92.
[2] J. J. Willcock et al. AM++: A generalized active message framework. PACT'10.
[3] D. Bonachea, GASNet Specification, v1.1. Berkeley Technical Report. 2002.

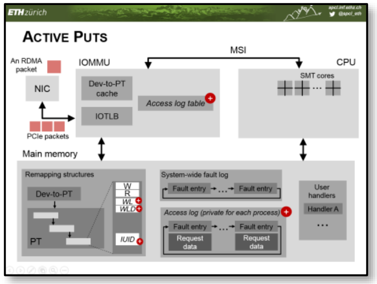
Active Messages

IOMMUs

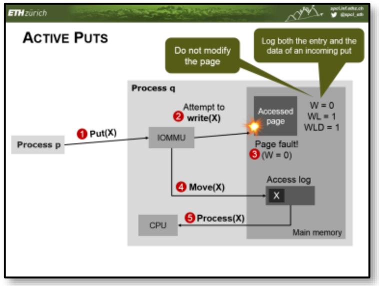


Use Input/Output Memory Management Units

Active Puts/Gets

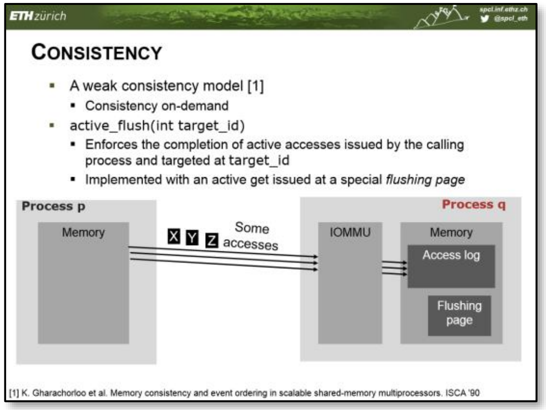


Active Puts



Active Puts

Let’s summarize…



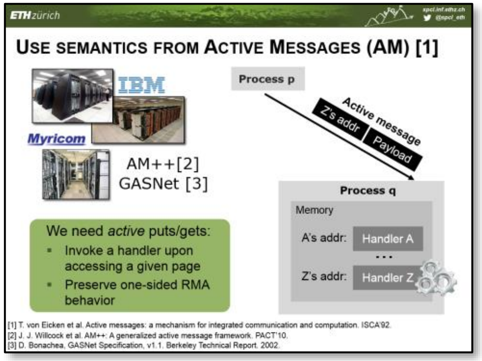
ETH zürich spcl.inf.ethz.ch @spcl_eth

Consistency

- A weak consistency model [1]
  - Consistency on-demand
- active_flush(int target_id)
  - Enforces the completion of active accesses issued by the calling process and targeted at target_id
  - Implemented with an active get issued at a special *flushing page*

[1] K. Gharachorloo et al. Memory consistency and event ordering in scalable shared-memory multiprocessors. ISCA '90

Consistency



ETH zürich spcl.inf.ethz.ch @spcl_eth

Use semantics from Active Messages (AM) [1]

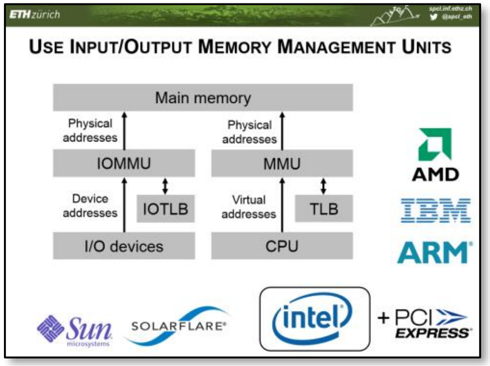
We need *active* puts/gets:

- Invoke a handler upon accessing a given page
- Preserve one-sided RMA behavior

[1] T. von Eicken et al. Active messages: a mechanism for integrated communication and computation. ISCA'92.
[2] J. J. Willcock et al. AM++: A generalized active message framework. PACT'10.
[3] D. Bonachea, GASNet Specification, v1.1. Berkeley Technical Report. 2002.

Active Messages

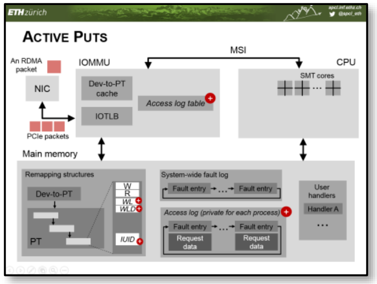
IOMMUs



ETH zürich spcl.inf.ethz.ch @spcl_eth

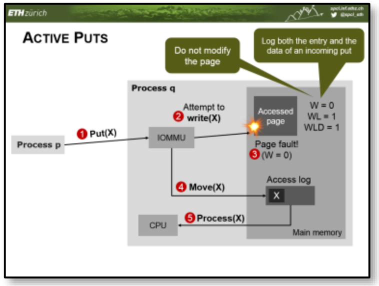
Use Input/Output Memory Management Units

Active Puts/Gets



ETH zürich spcl.inf.ethz.ch @spcl_eth

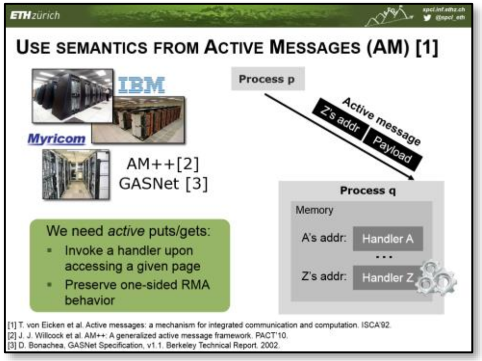
Active Puts



ETH zürich spcl.inf.ethz.ch @spcl_eth

Active Puts

Let's summarize…



Use semantics from Active Messages (AM) [1]

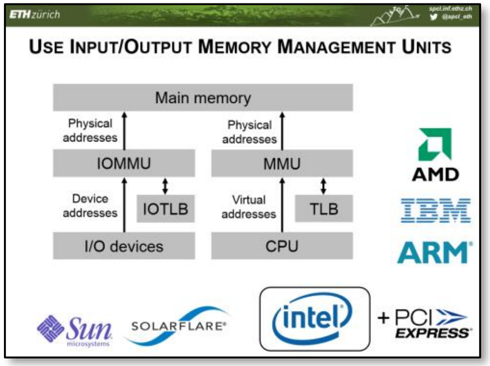
We need *active* puts/gets:

- Invoke a handler upon accessing a given page
- Preserve one-sided RMA behavior

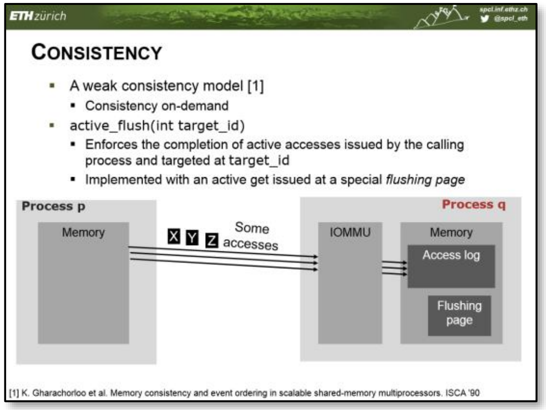
[1] T. von Eicken et al. Active messages: a mechanism for integrated communication and computation. ISCA'92.
[2] J. J. Willcock et al. AM++: A generalized active message framework. PACT'10.
[3] D. Bonachea, GASNet Specification, v1.1. Berkeley Technical Report. 2002.

Active Messages

IOMMUs



Use Input/Output Memory Management Units



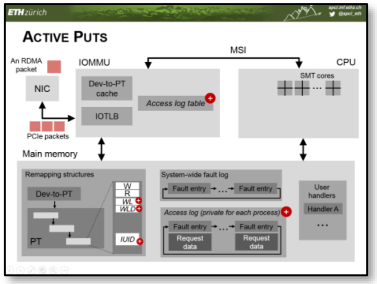
Consistency

- A weak consistency model [1]
  - Consistency on-demand
- active_flush(int target_id)
  - Enforces the completion of active accesses issued by the calling process and targeted at target_id
  - Implemented with an active get issued at a special *flushing page*

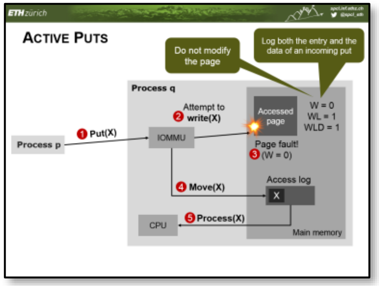
[1] K. Gharachorloo et al. Memory consistency and event ordering in scalable shared-memory multiprocessors. ISCA '90

Consistency

Active Puts/Gets



Active Puts



Active Puts

How can we use it?

Active Access Use-Cases

Distributed Hashtable

Active Access Use-Cases

Distributed Hashtable

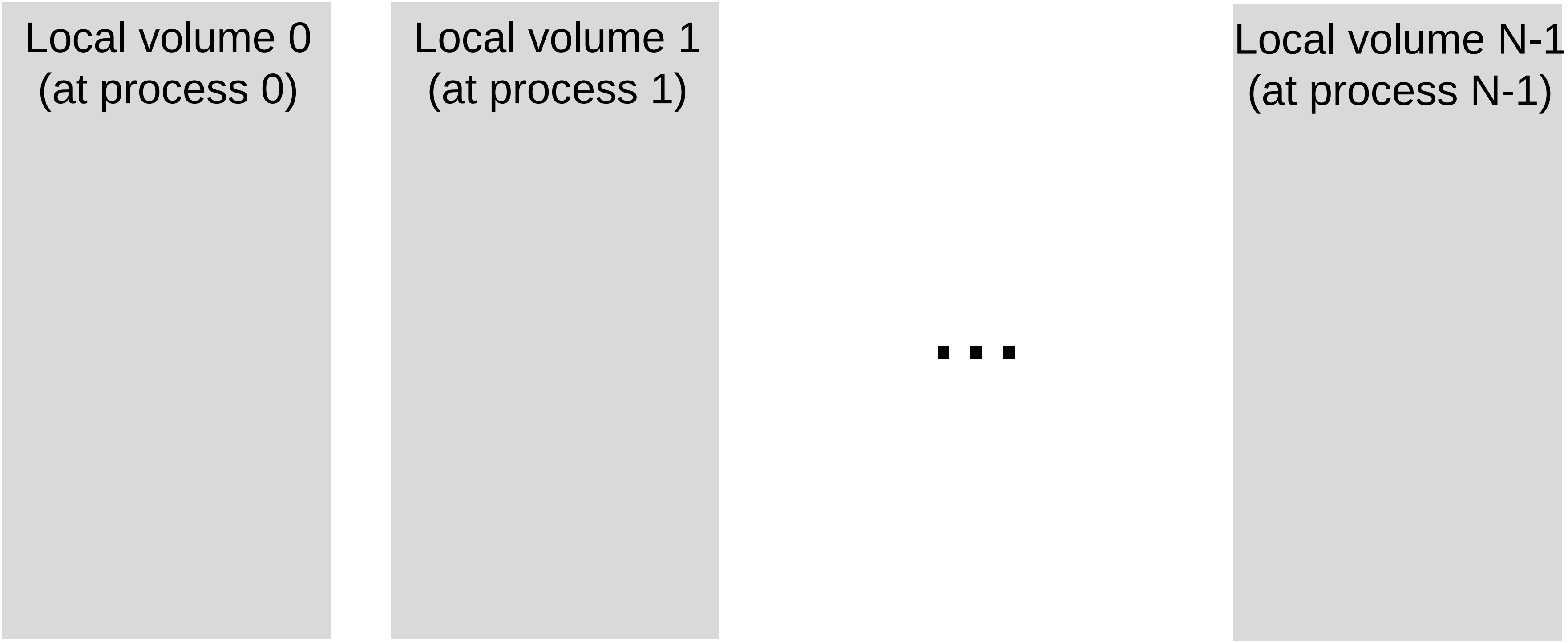
- Used to construct key-value stores (e.g., Memcached [1])

[1] B. Fitzpatrick. Distributed caching with memcached. Linux journal, 2004.

Active Access Use-Cases

Distributed Hashtable

- Used to construct key-value stores (e.g., Memcached [1])

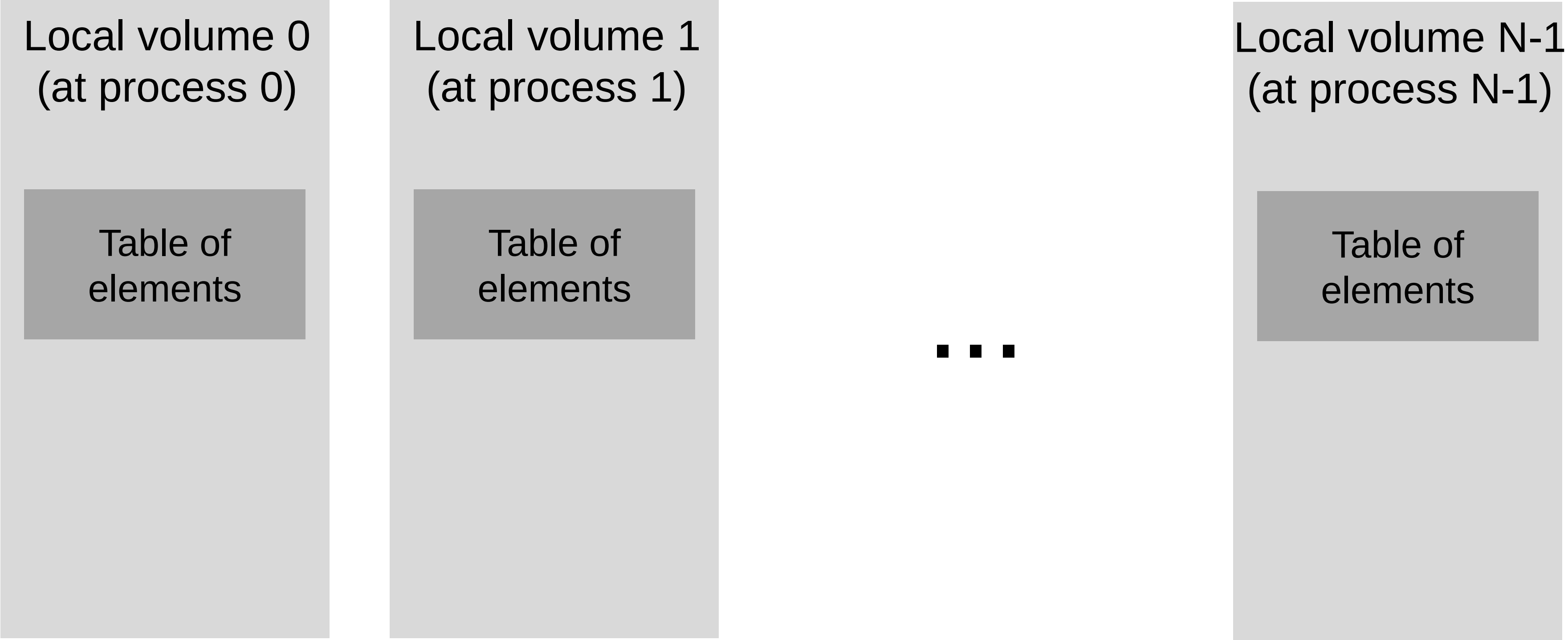


[1] B. Fitzpatrick. Distributed caching with memcached. Linux journal, 2004.

ACTIVE ACCESS USE-CASES

DISTRIBUTED HASHTABLE

- Used to construct key-value stores (e.g., Memcached [1])



Local volume 0 (at process 0)

Table of elements

Local volume 1 (at process 1)

Table of elements

...

Local volume N-1 (at process N-1)

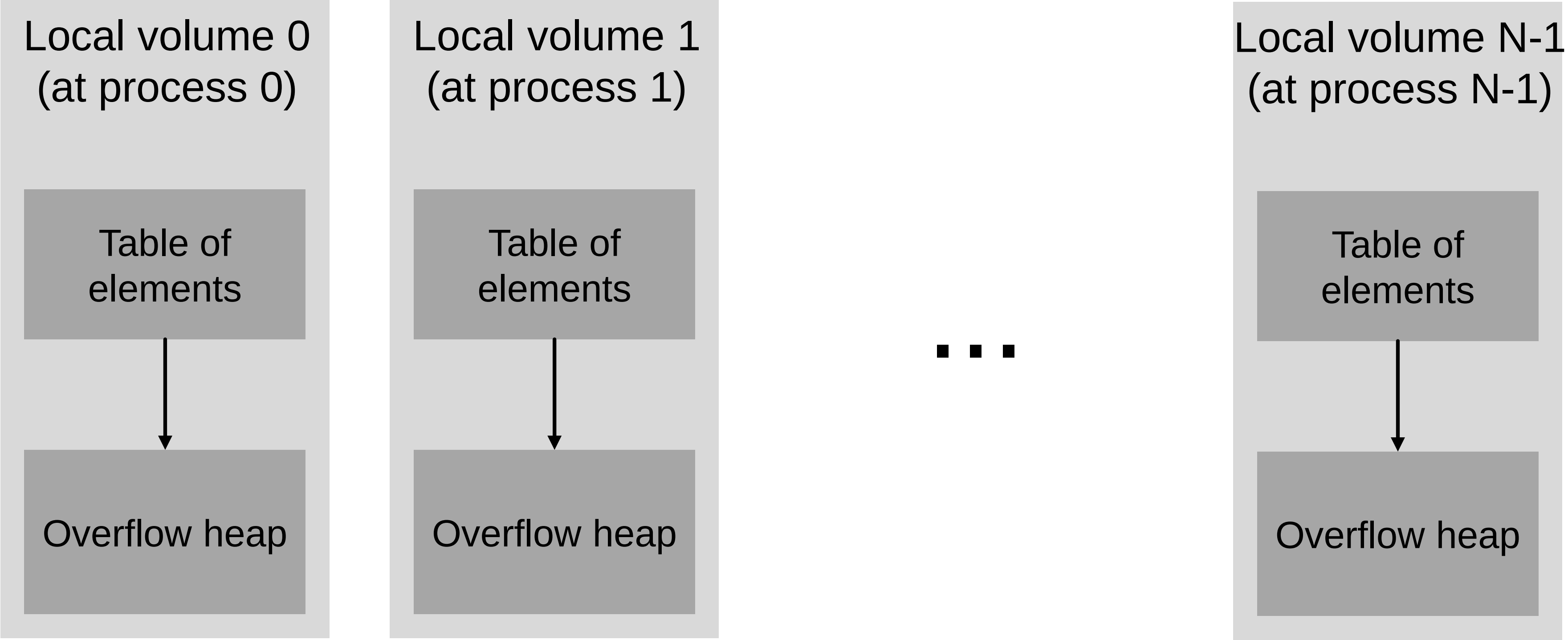
Table of elements

[1] B. Fitzpatrick. Distributed caching with memcached. Linux journal, 2004.

Active Access Use-Cases

Distributed Hashtable

- Used to construct key-value stores (e.g., Memcached [1])



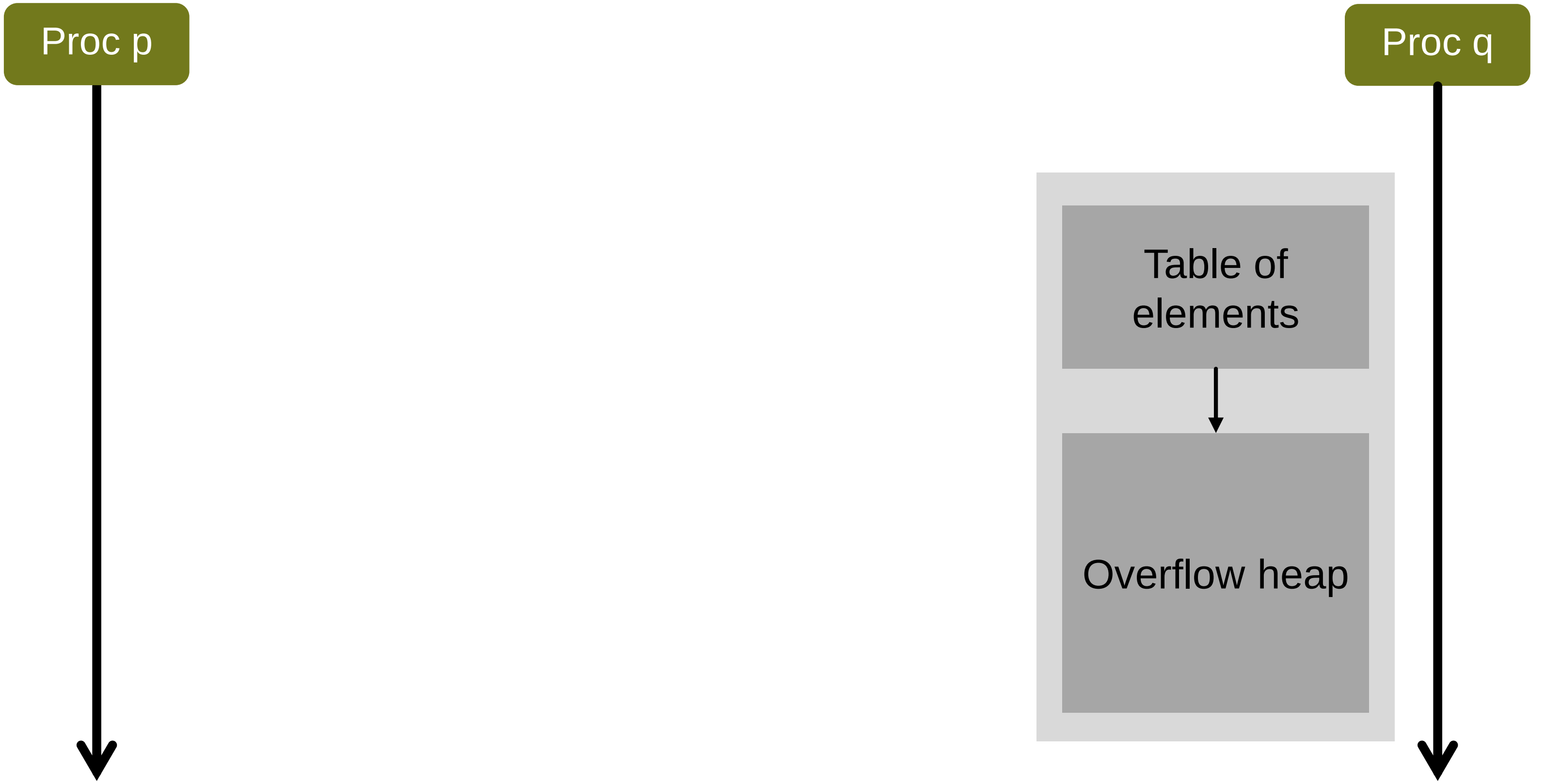
[1] B. Fitzpatrick. Distributed caching with memcached. Linux journal, 2004.

ACTIVE ACCESS USE-CASES

DISTRIBUTED HASHTABLE: INSERTS (RMA)

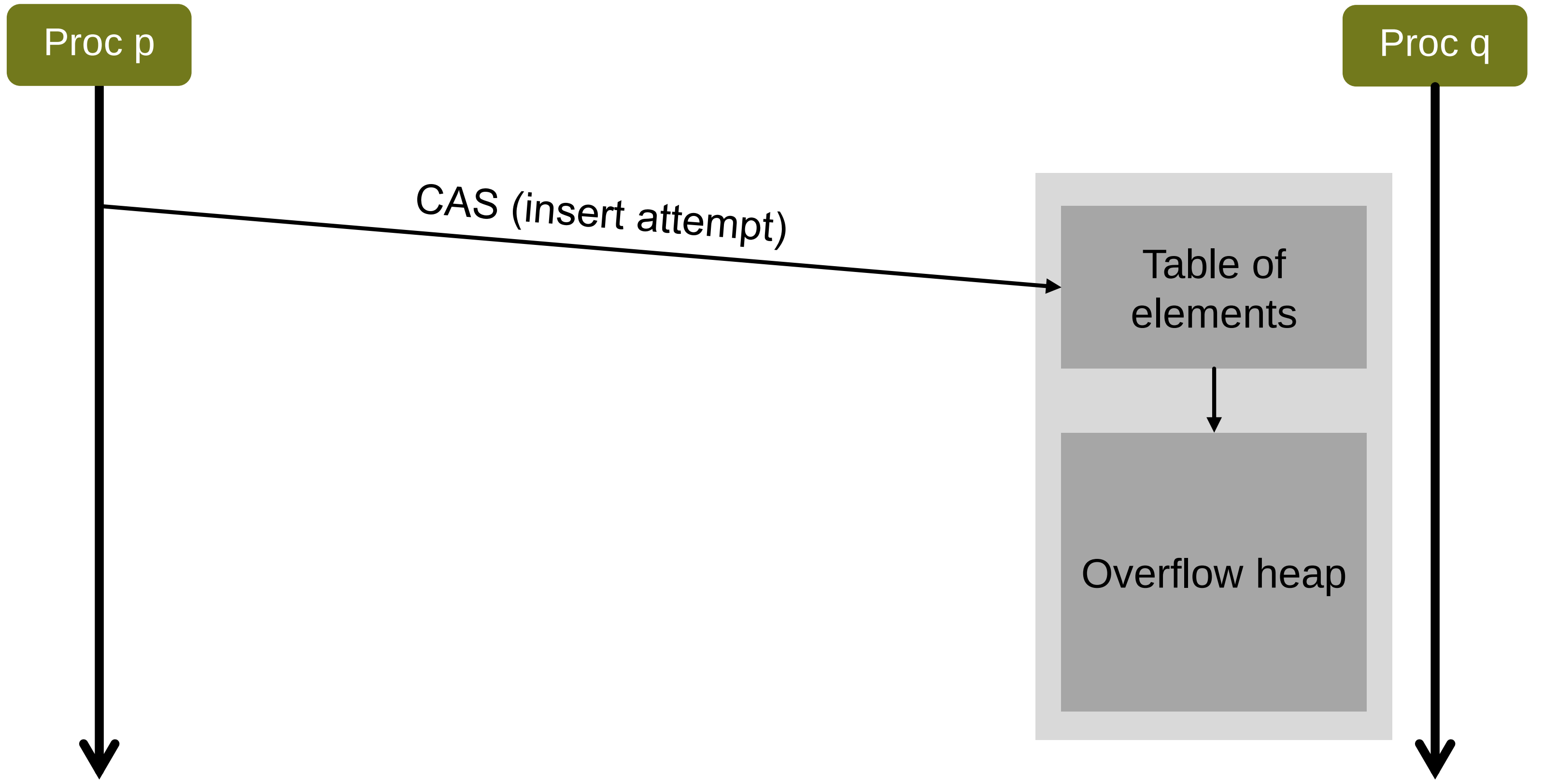
Active Access Use-Cases

Distributed Hashtable: Inserts (RMA)



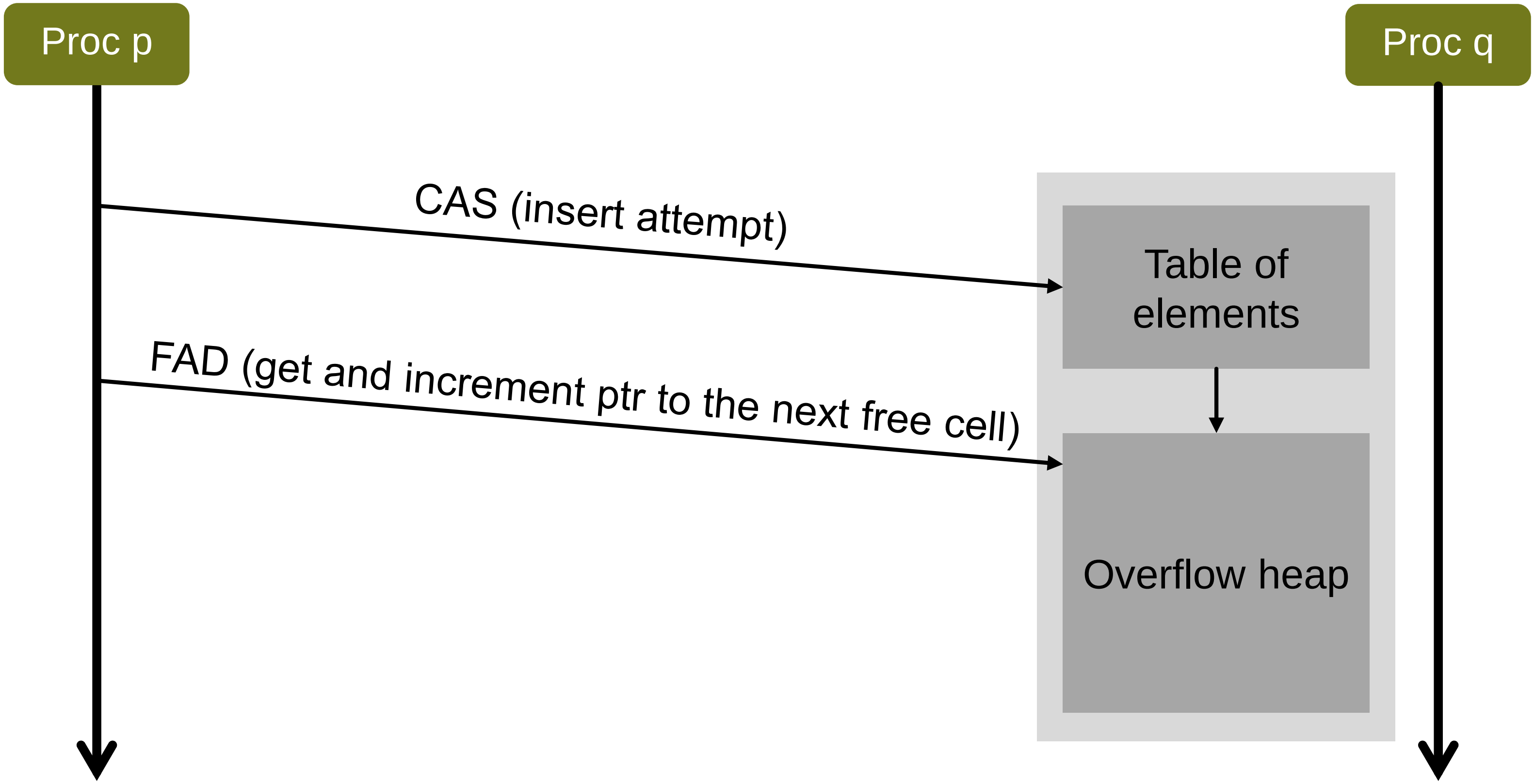
Active Access Use-Cases

Distributed Hashtable: Inserts (RMA)



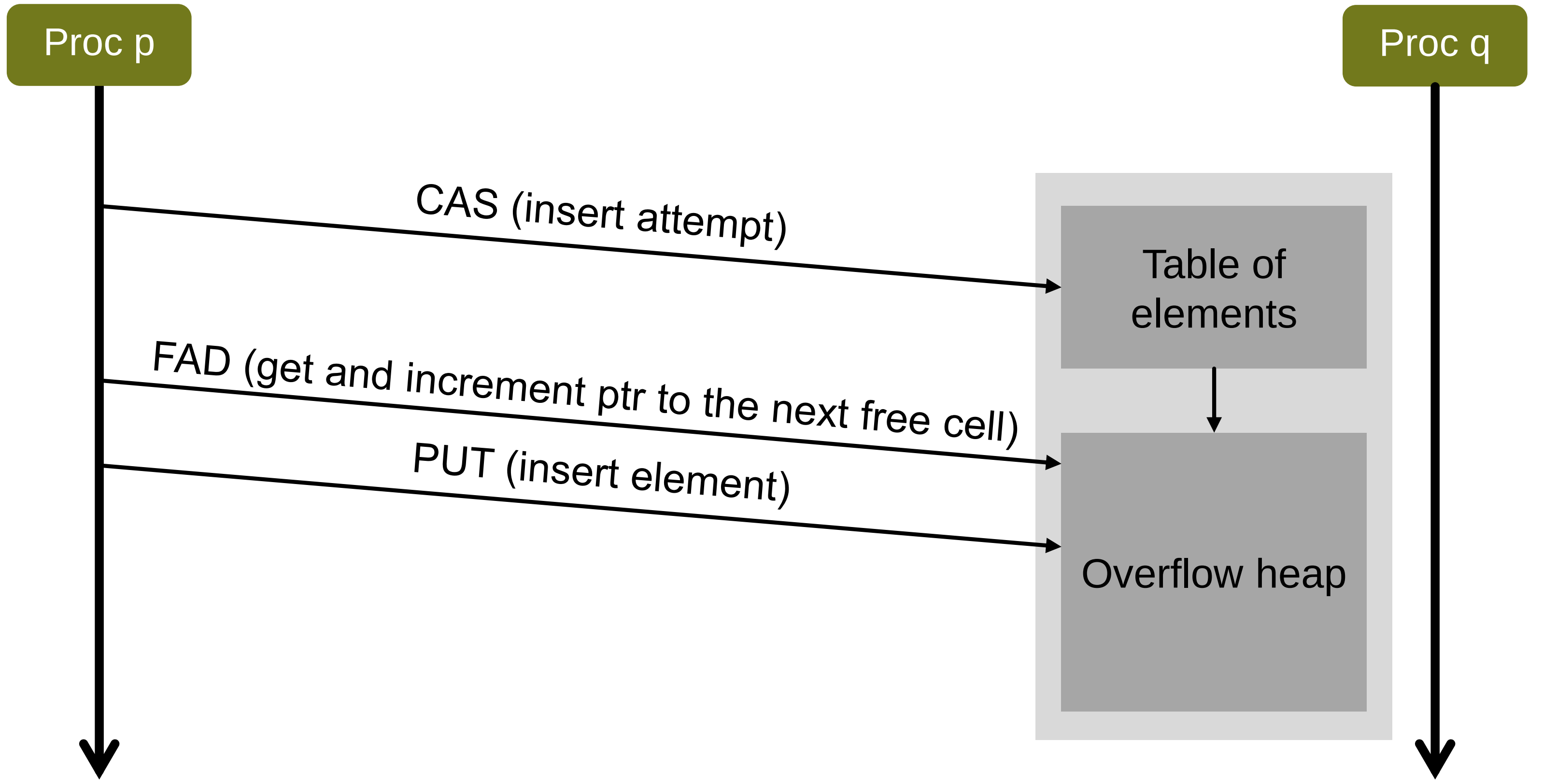
Active Access Use-Cases

Distributed Hashtable: Inserts (RMA)



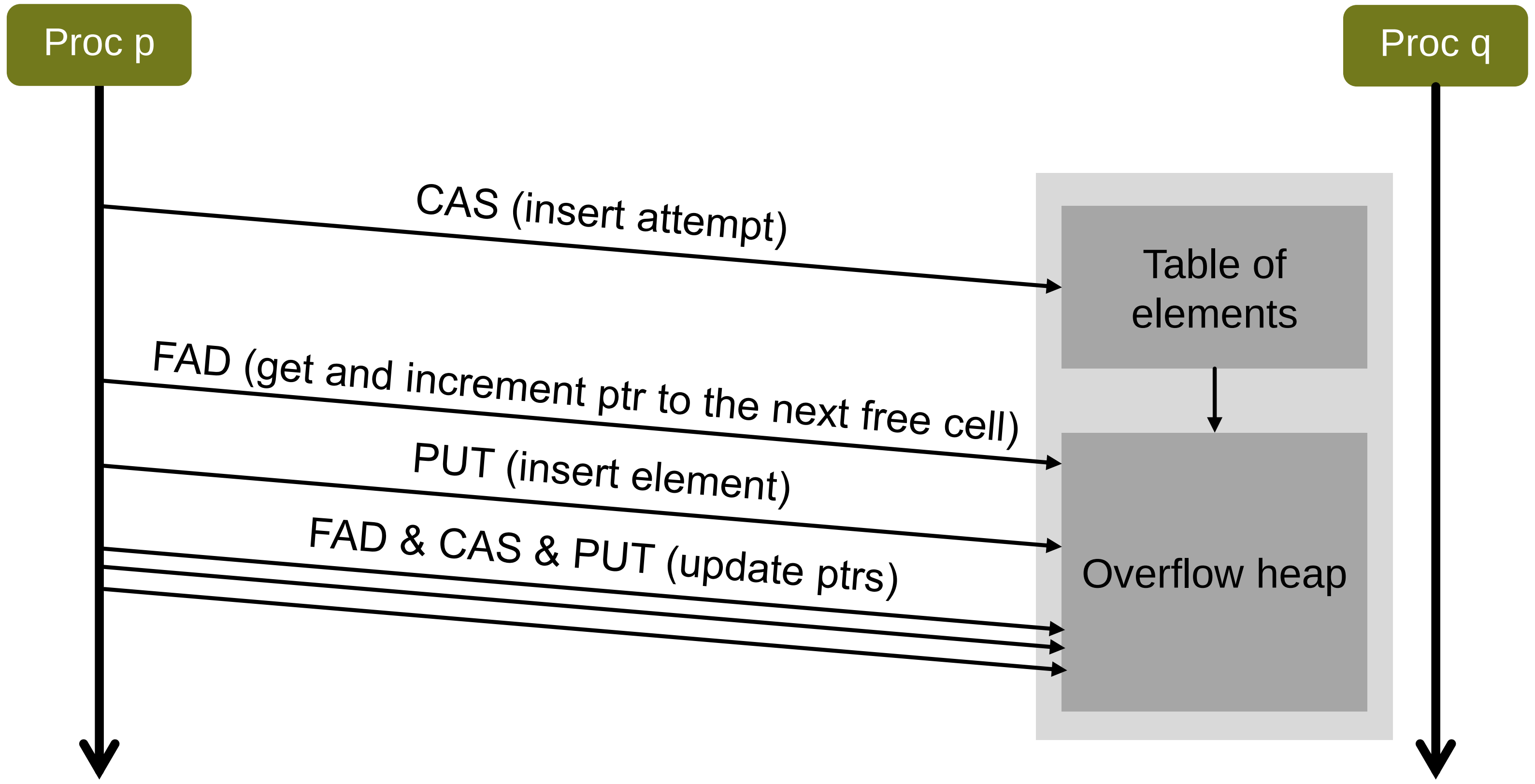
Active Access Use-Cases

Distributed Hashtable: Inserts (RMA)



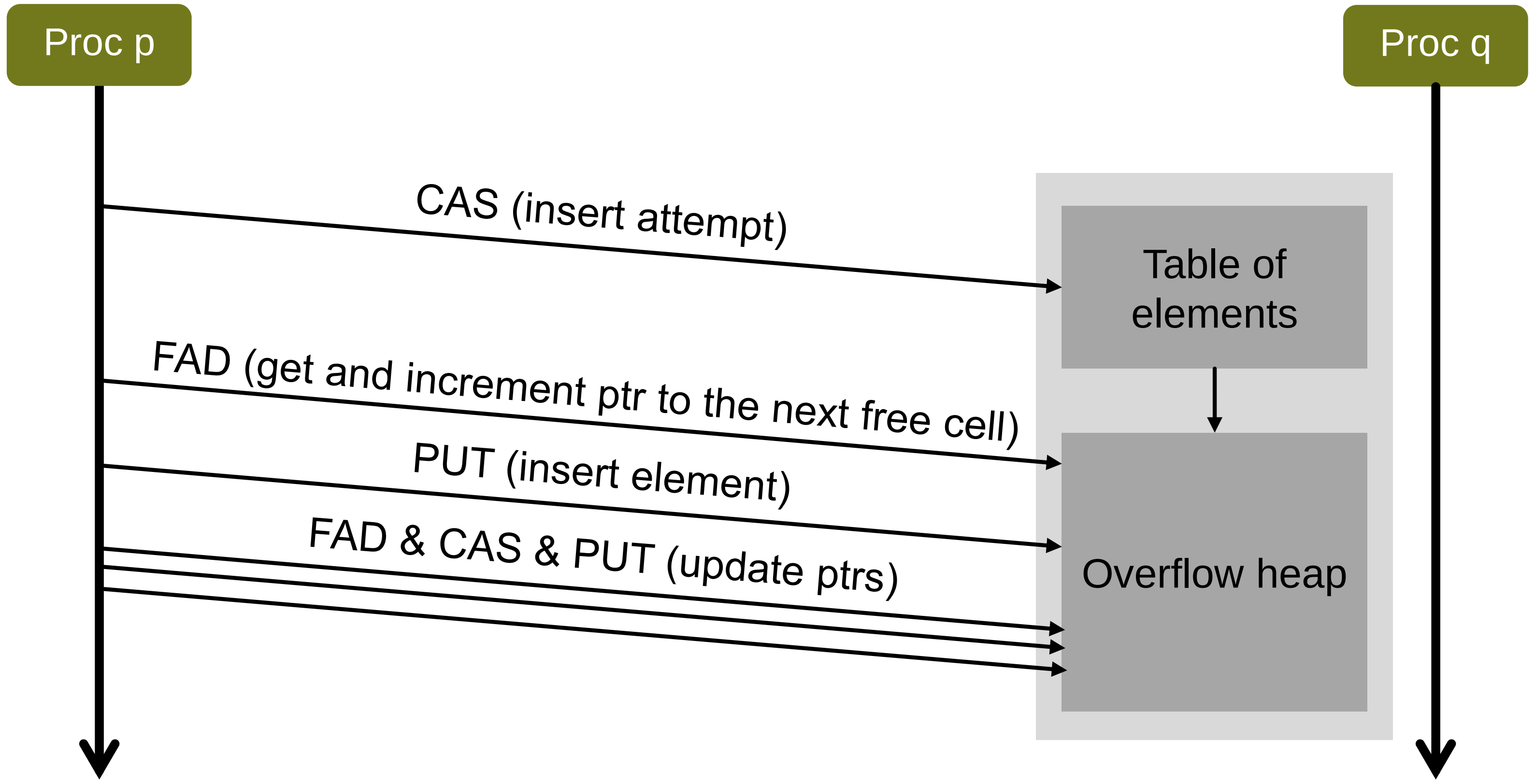
Active Access Use-Cases

Distributed Hashtable: Inserts (RMA)



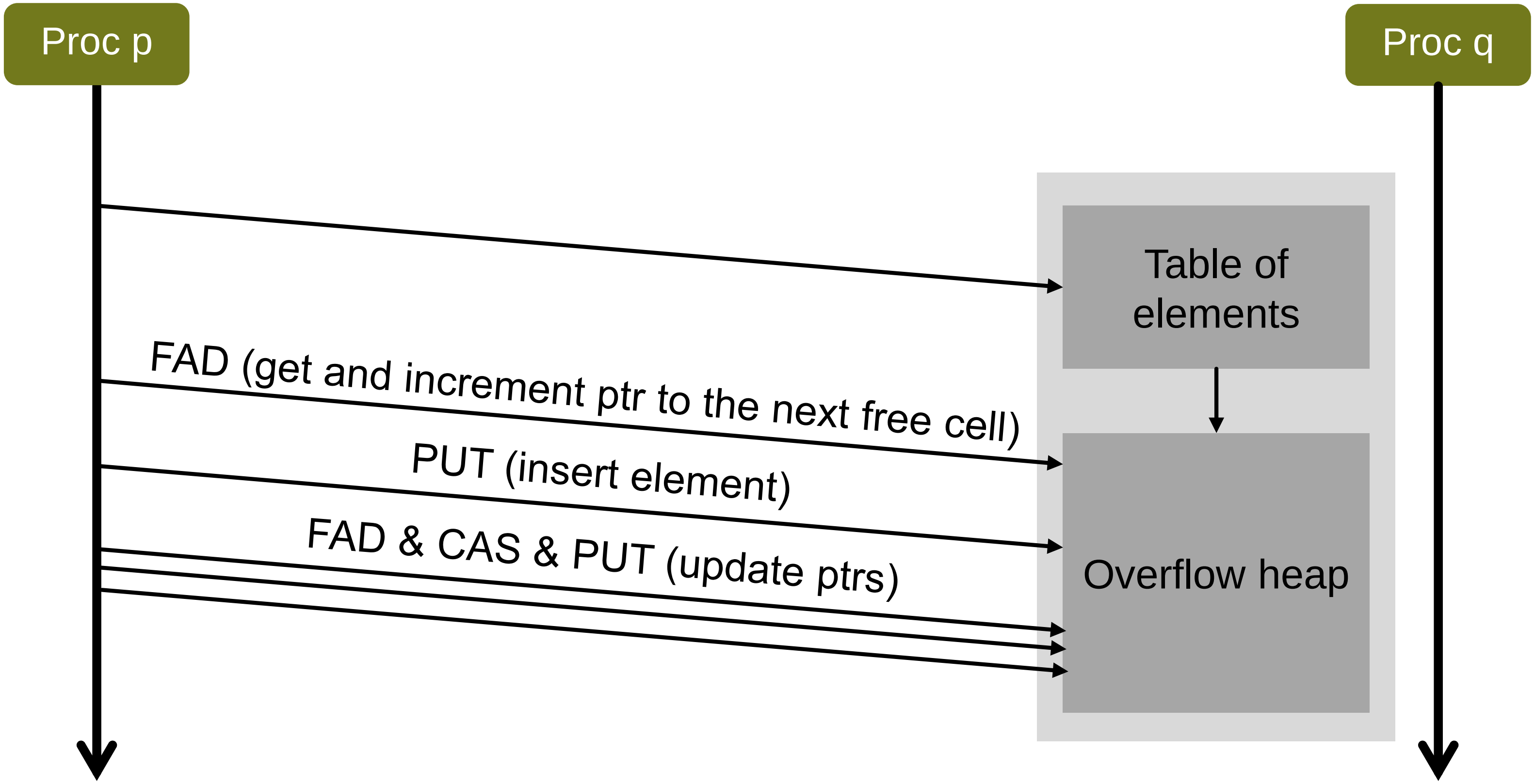
Active Access Use-Cases

Distributed Hashtable: Inserts (AA)



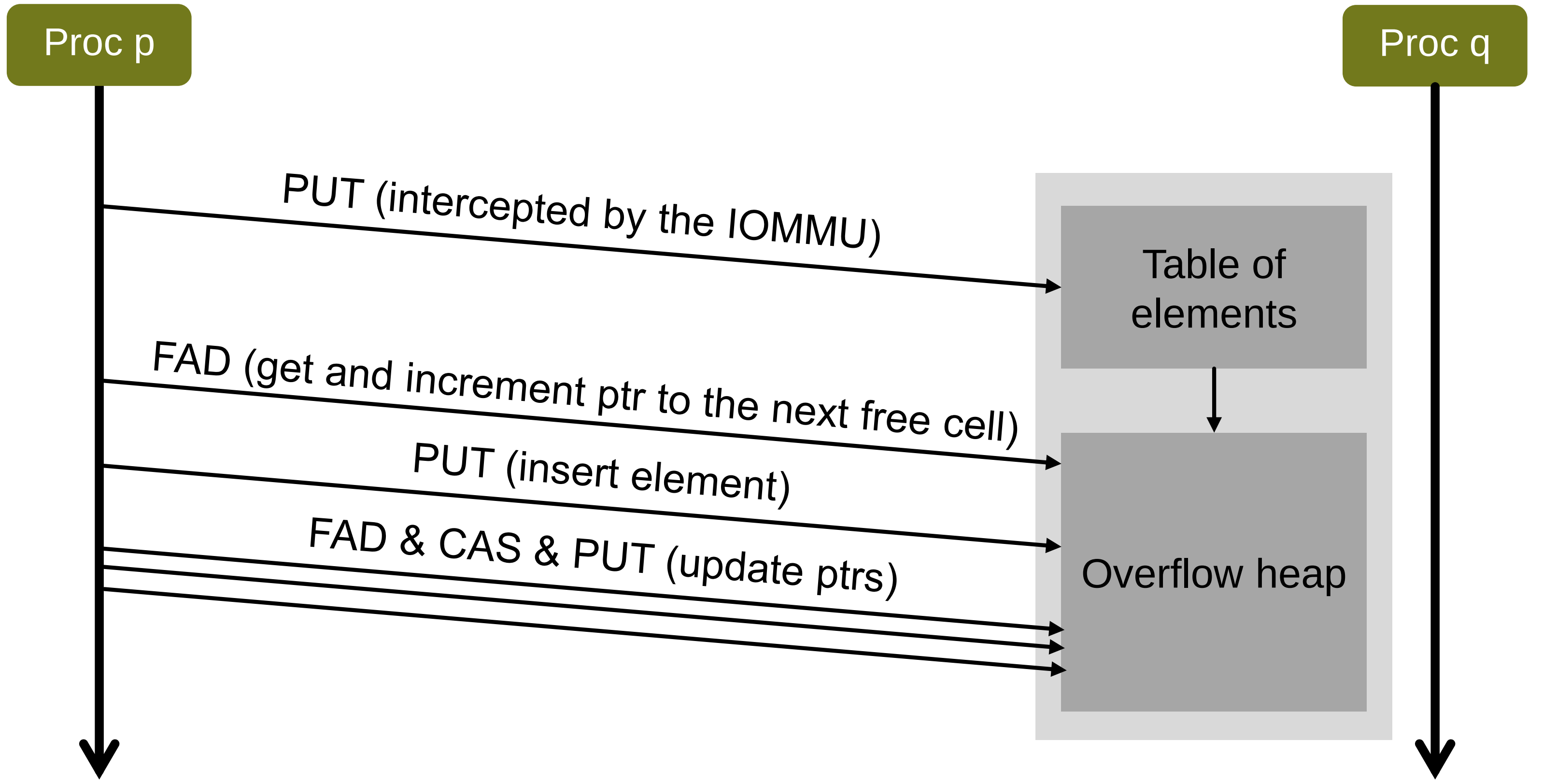
ACTIVE ACCESS USE-CASES

DISTRIBUTED HASHTABLE: INSERTS (AA)


Proc p
Proc q
Table of elements
Overflow heap
FAD (get and increment ptr to the next free cell)
PUT (insert element)
FAD & CAS & PUT (update ptrs)

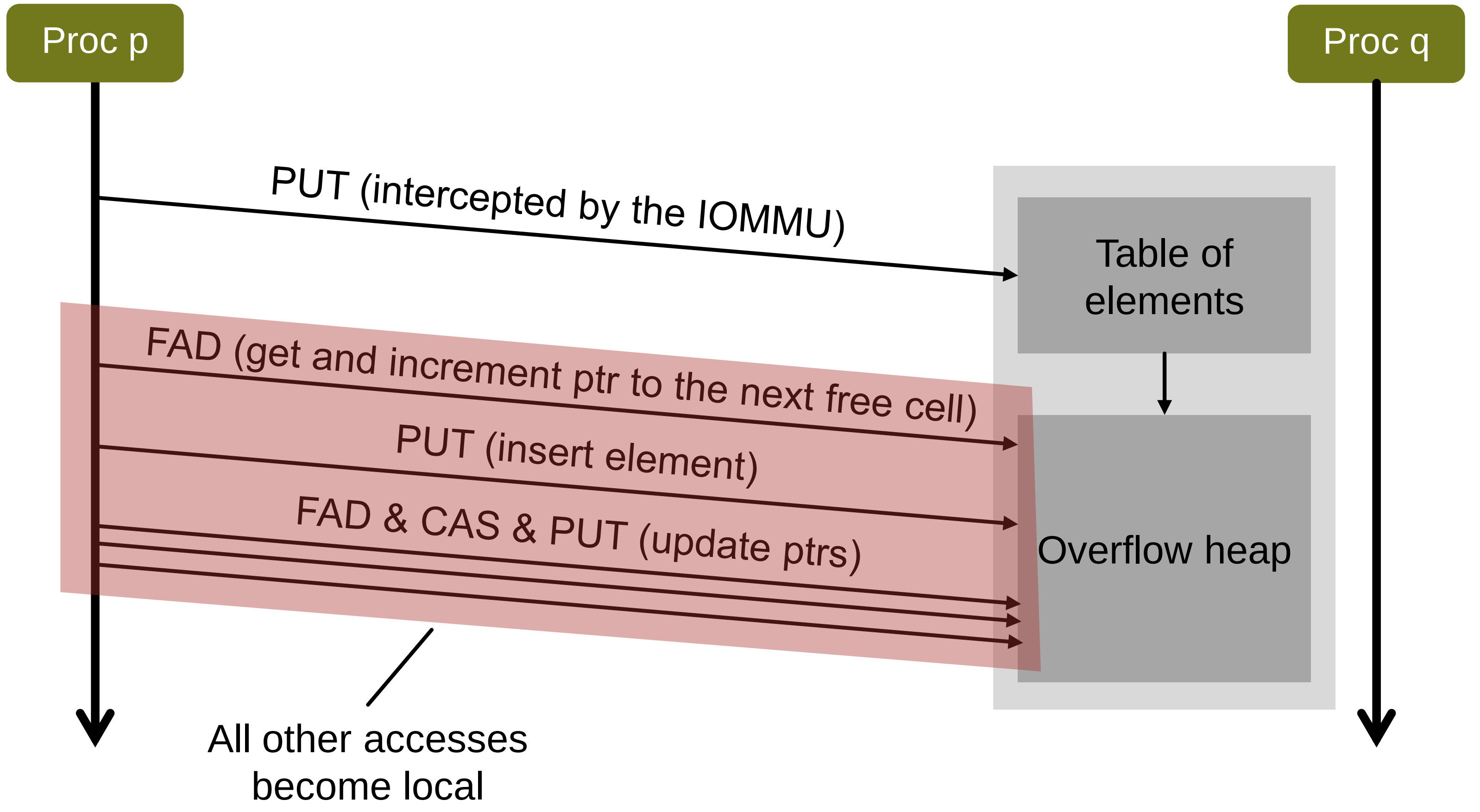
ACTIVE ACCESS USE-CASES

DISTRIBUTED HASHTABLE: INSERTS (AA)


Proc p
Proc q
PUT (intercepted by the IOMMU)
FAD (get and increment ptr to the next free cell)
PUT (insert element)
FAD & CAS & PUT (update ptrs)
Table of elements
Overflow heap

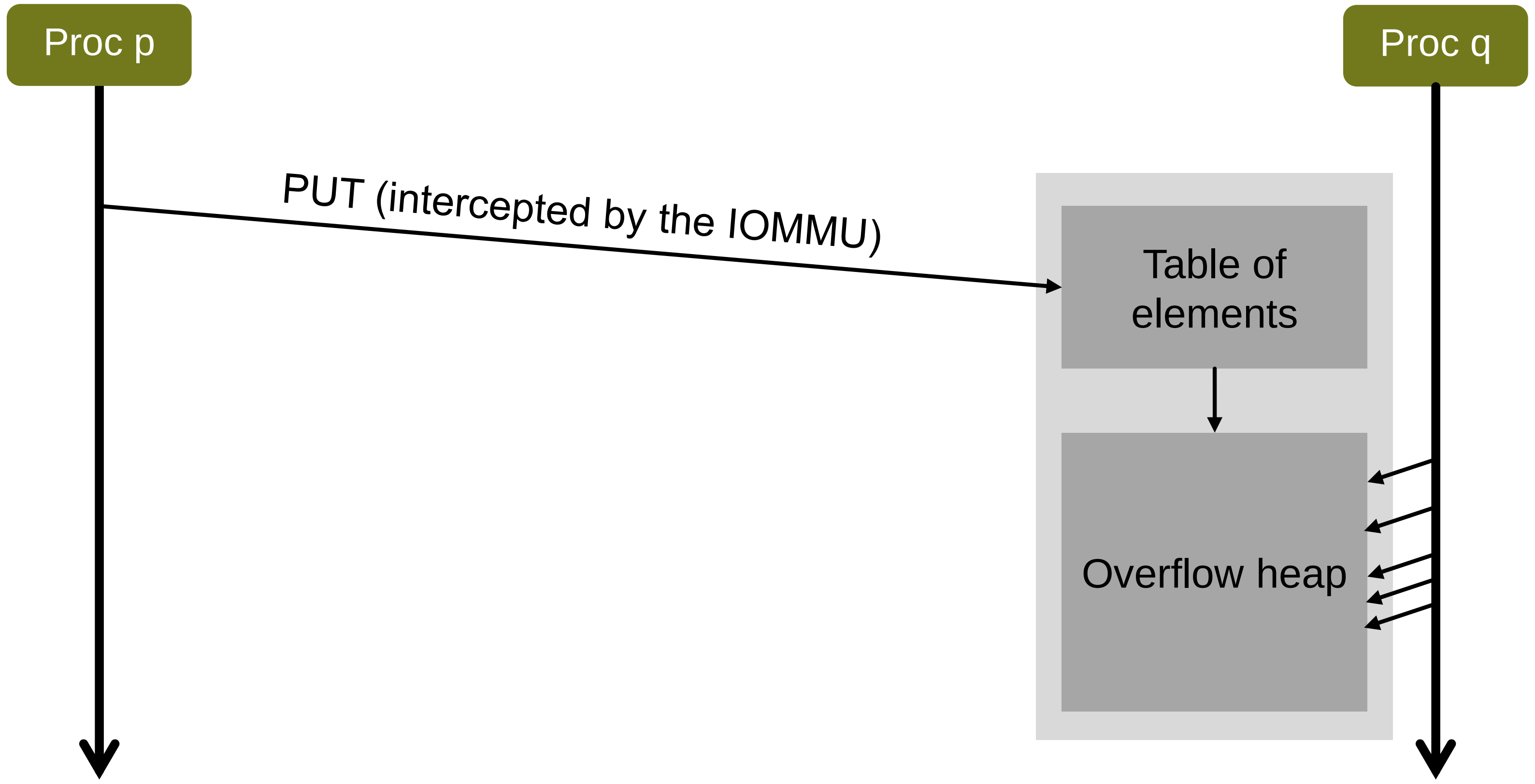
Active Access Use-Cases

Distributed Hashtable: Inserts (AA)


Proc p
Proc q
PUT (intercepted by the IOMMU)
Table of elements
FAD (get and increment ptr to the next free cell)
PUT (insert element)
FAD & CAS & PUT (update ptrs)
Overflow heap
All other accesses become local

Active Access Use-Cases

Distributed Hashtable: Inserts (AA)

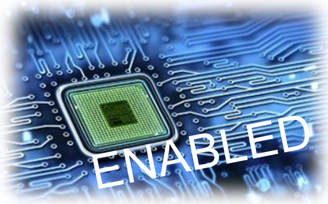


Active Access Use-Cases

Virtual Global Address Space (V-GAS)

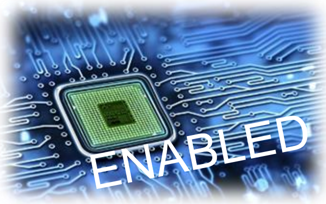
Active Access Use-Cases

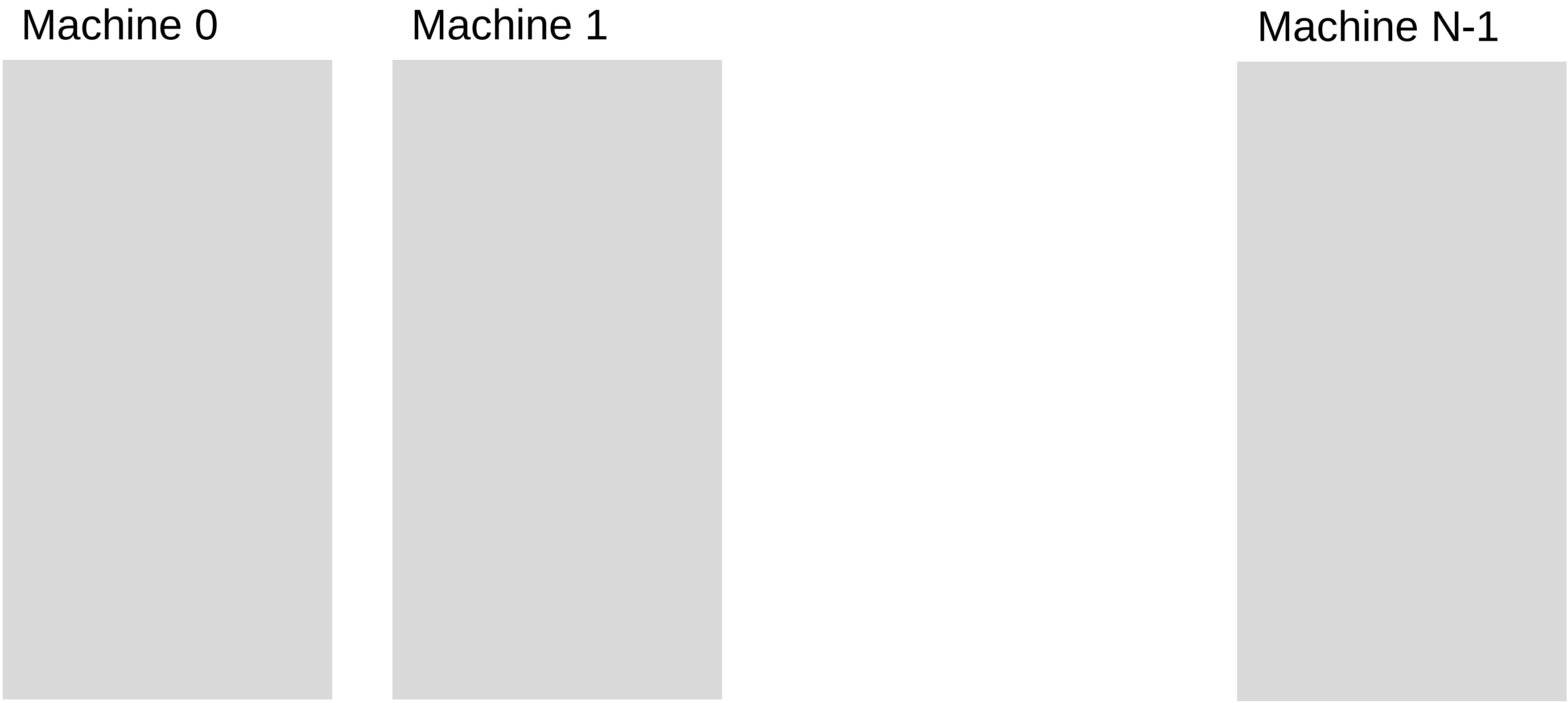
Virtual Global Address Space (V-GAS)



Active Access Use-Cases

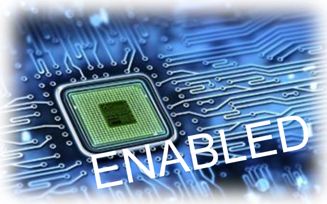
Virtual Global Address Space (V-GAS)

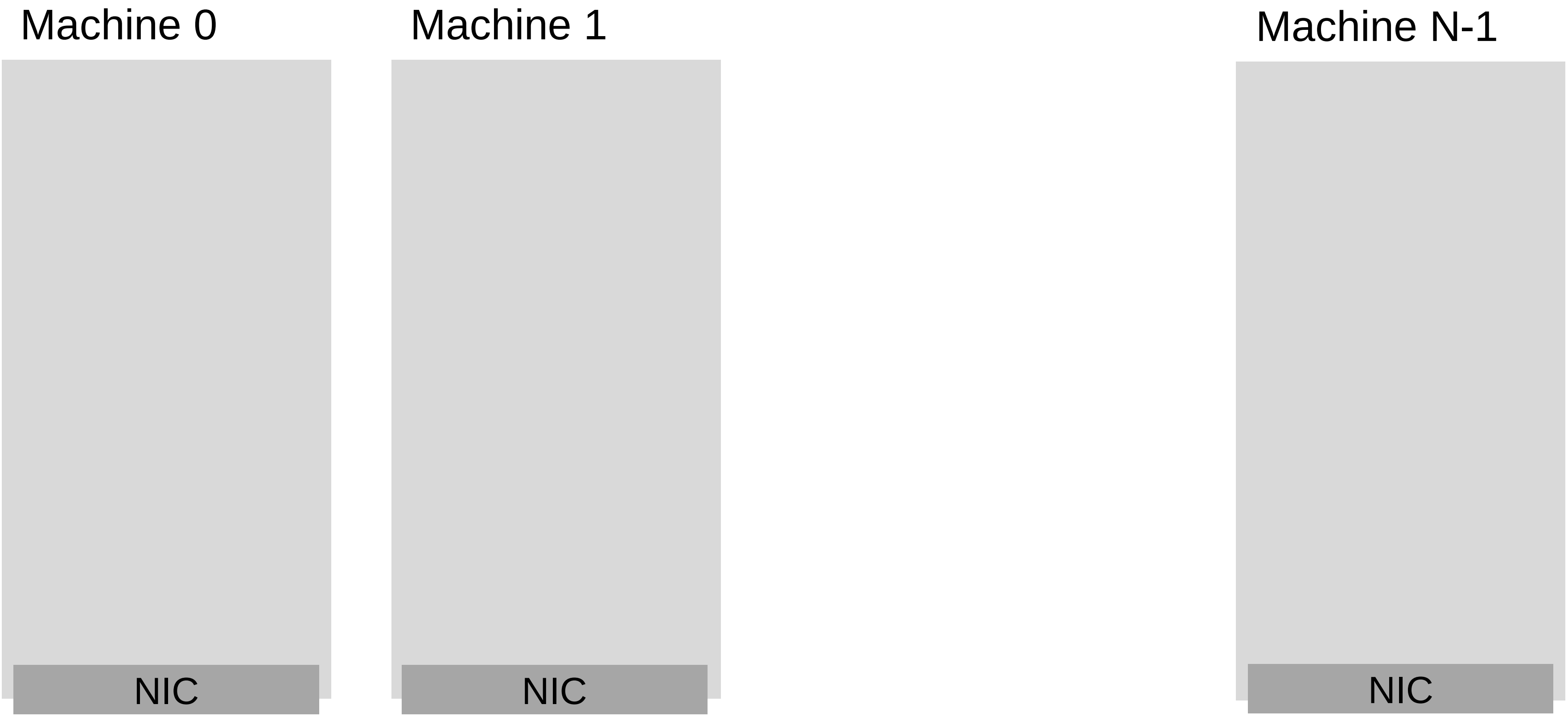




ACTIVE ACCESS USE-CASES

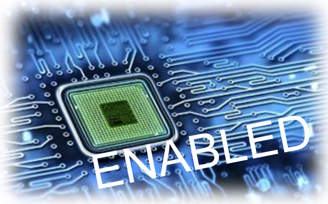
VIRTUAL GLOBAL ADDRESS SPACE (V-GAS)

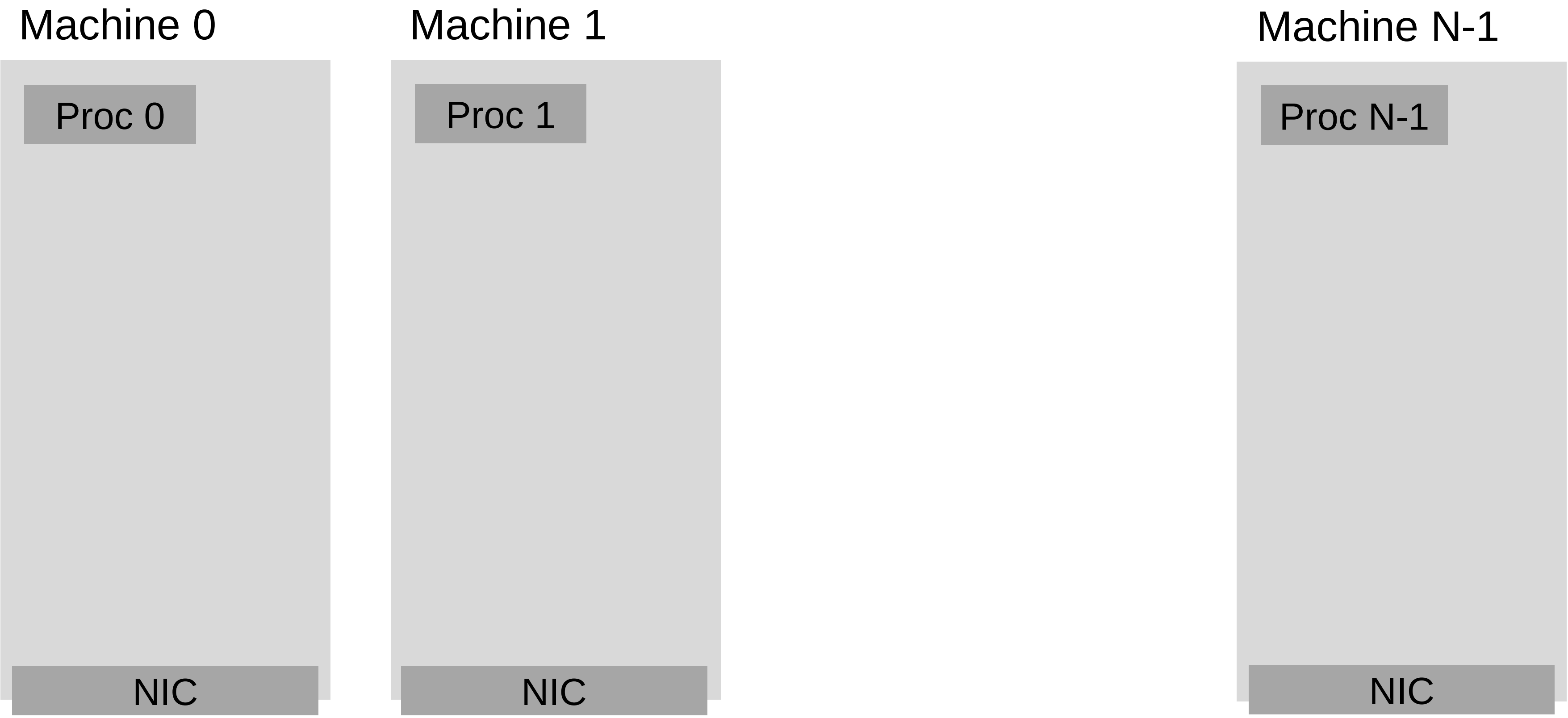




Active Access Use-Cases

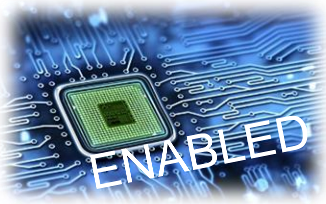
Virtual Global Address Space (V-GAS)

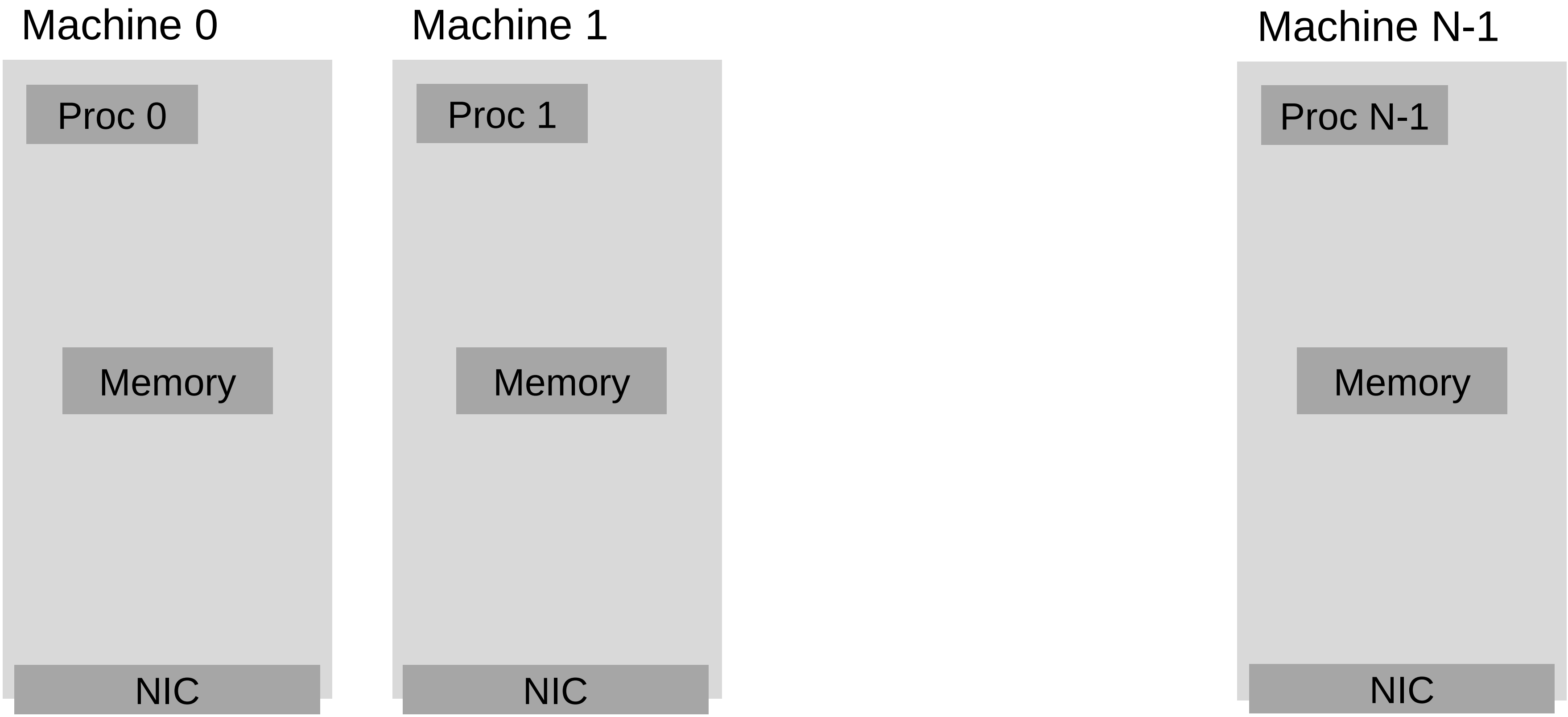




Active Access Use-Cases

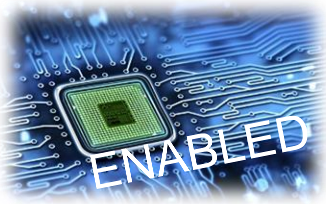
Virtual Global Address Space (V-GAS)

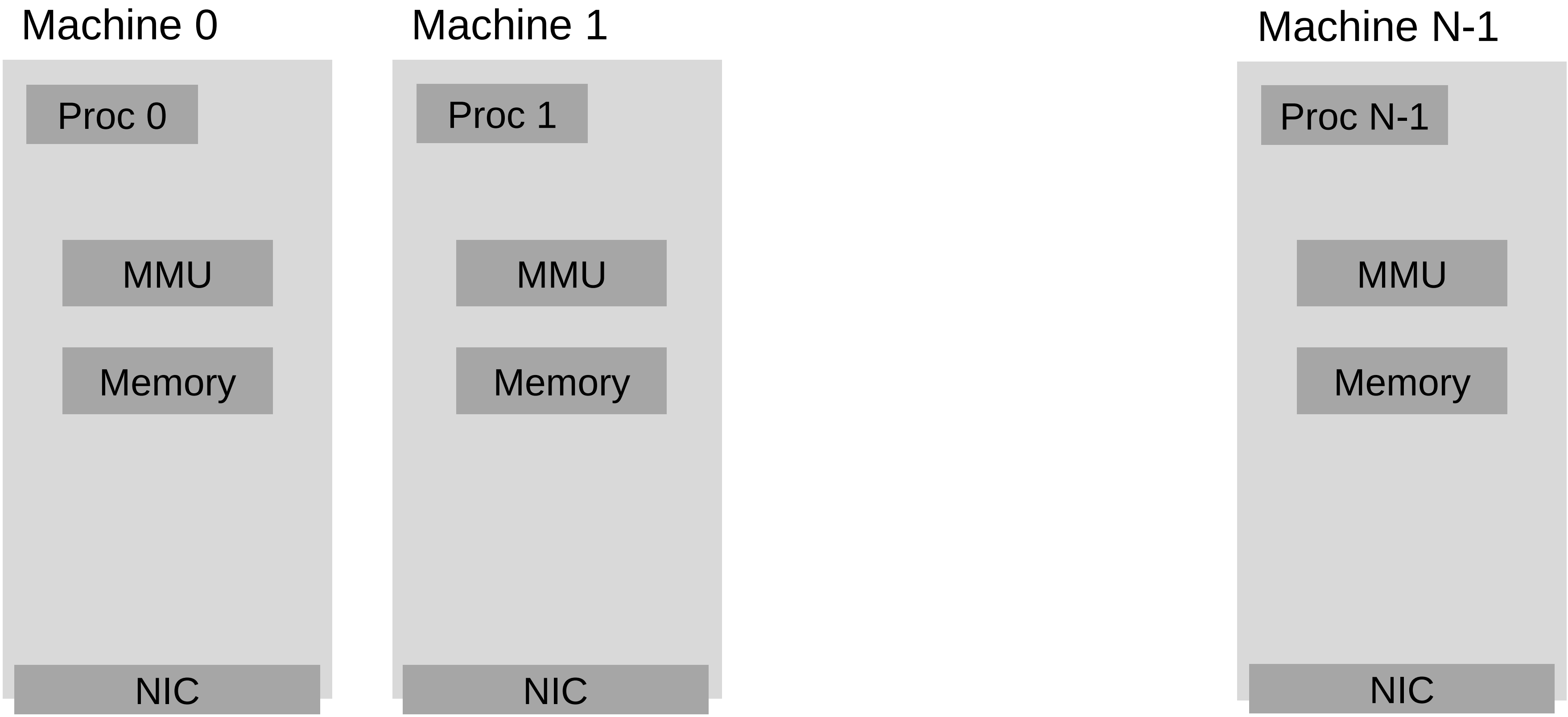




Active Access Use-Cases

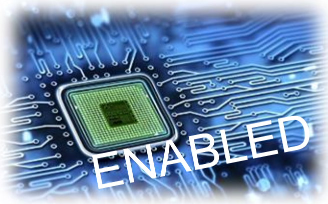
Virtual Global Address Space (V-GAS)

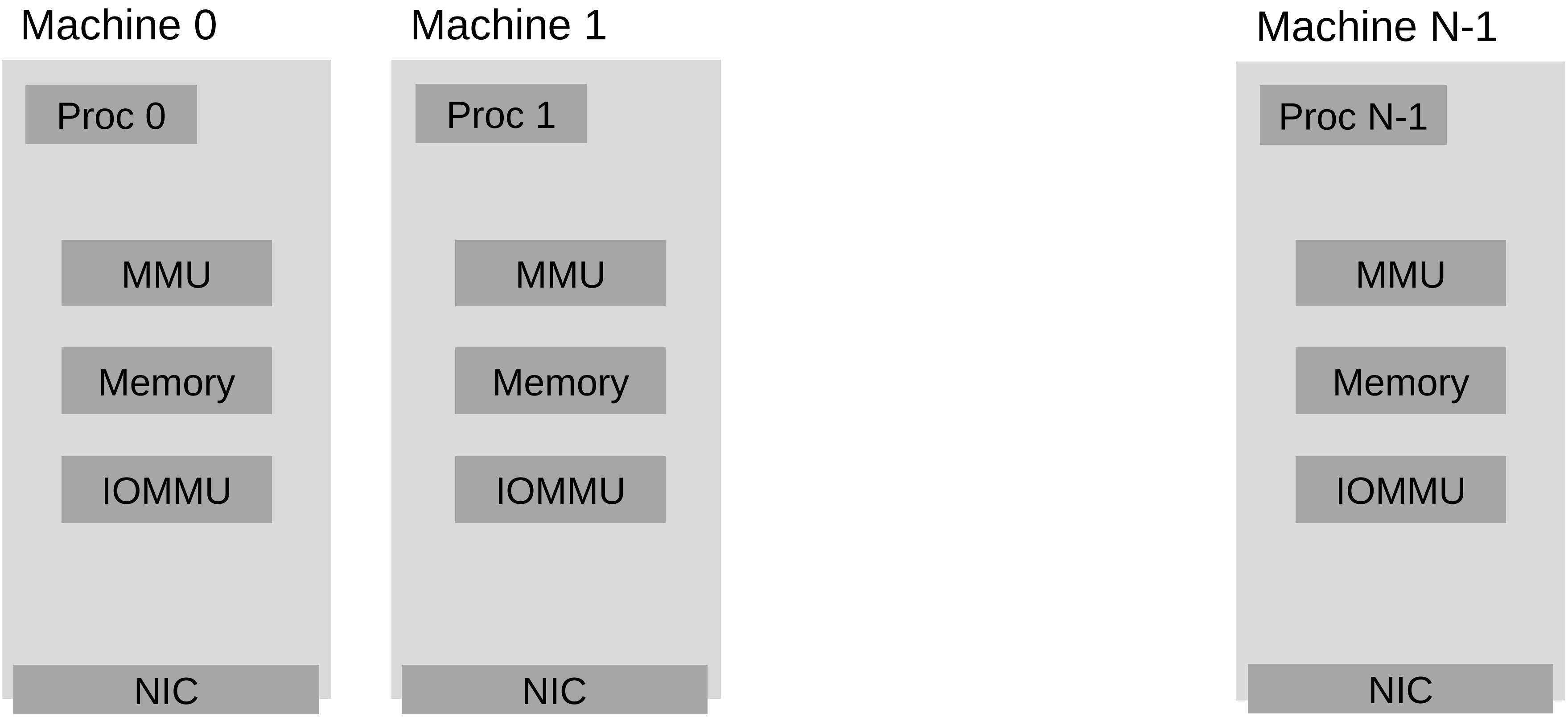




Active Access Use-Cases

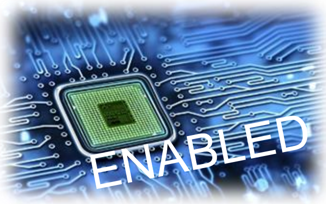
Virtual Global Address Space (V-GAS)

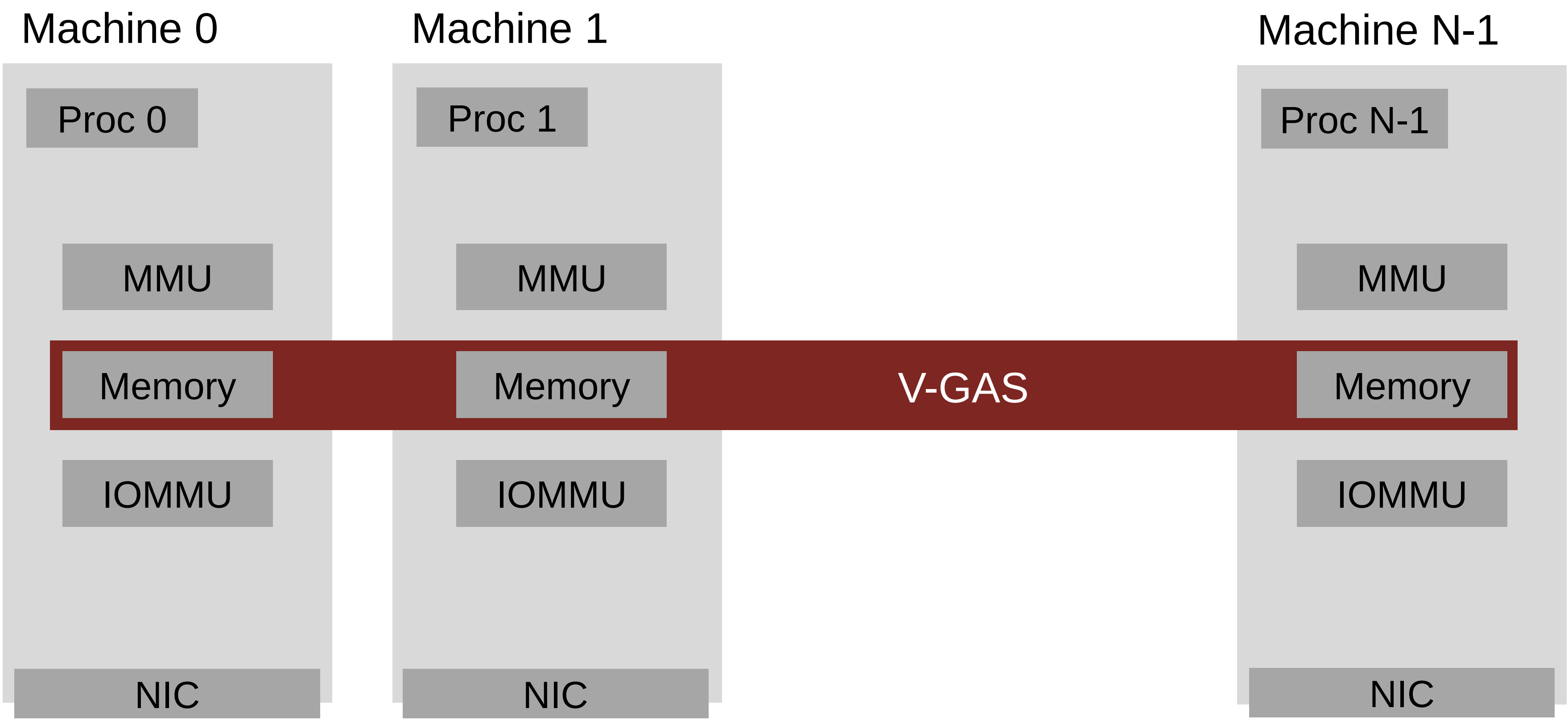




Active Access Use-Cases

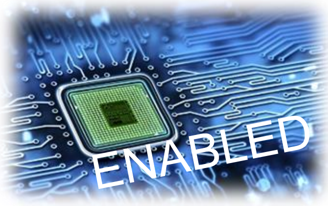
Virtual Global Address Space (V-GAS)

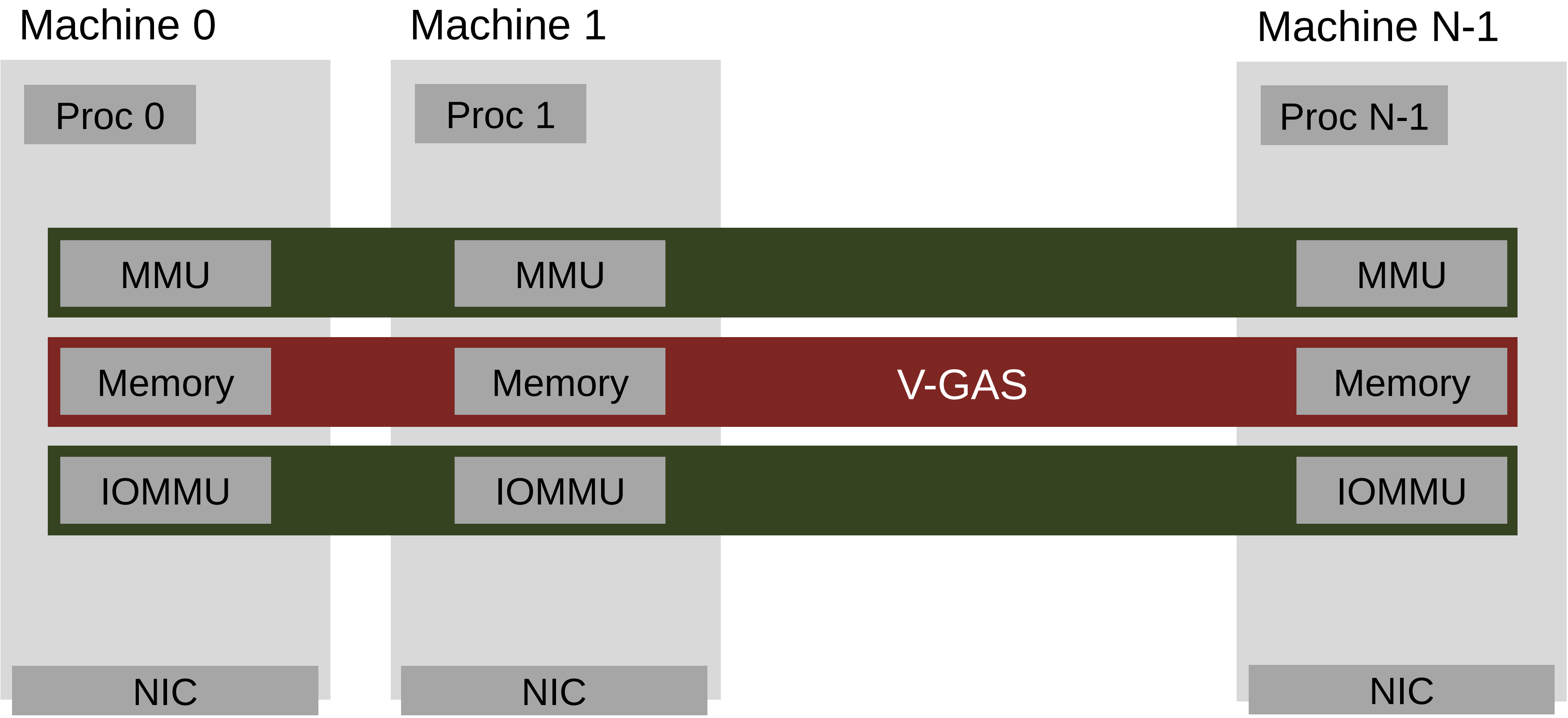




Active Access Use-Cases

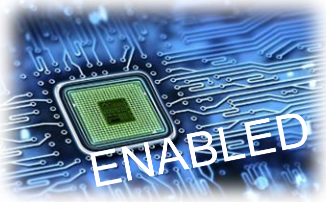
Virtual Global Address Space (V-GAS)

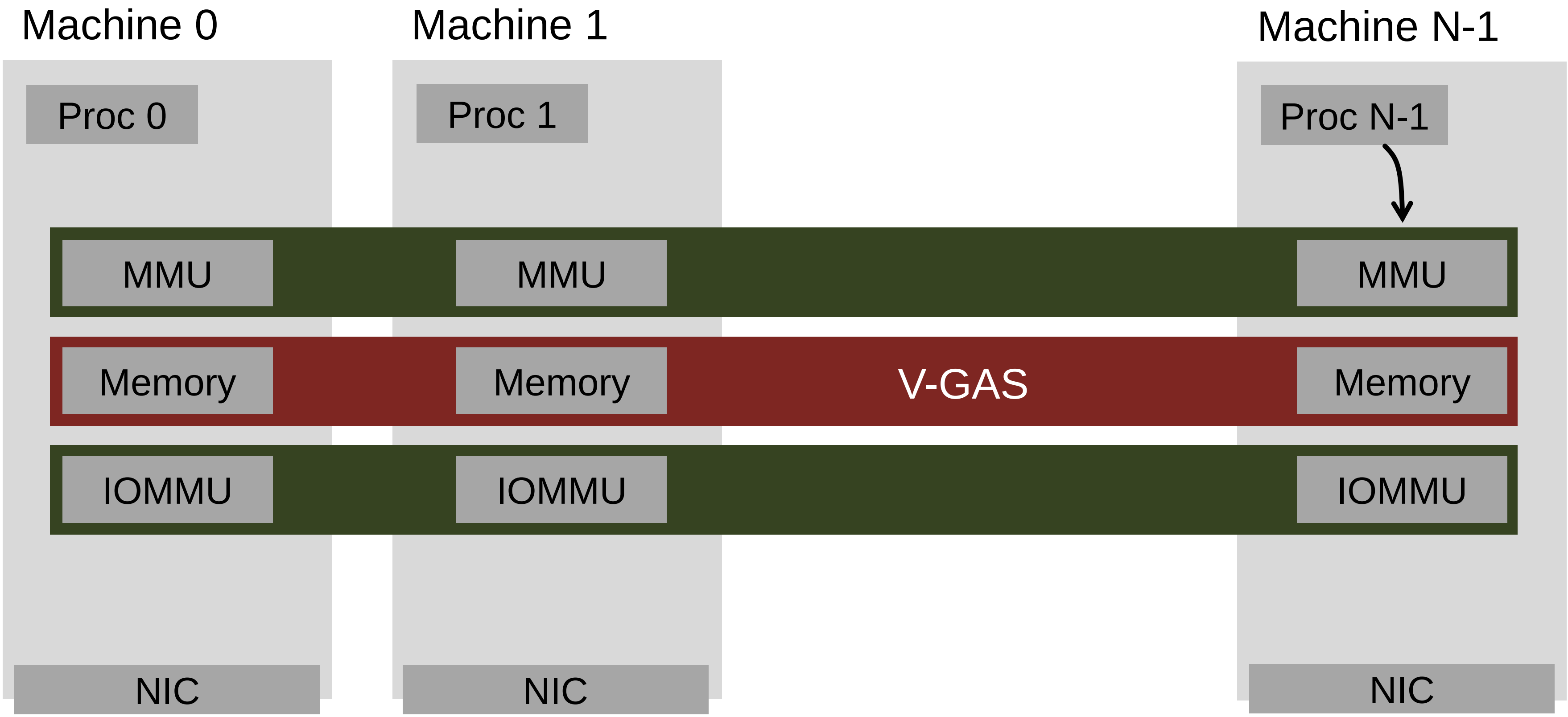




Active Access Use-Cases

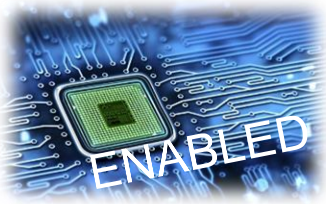
Virtual Global Address Space (V-GAS)

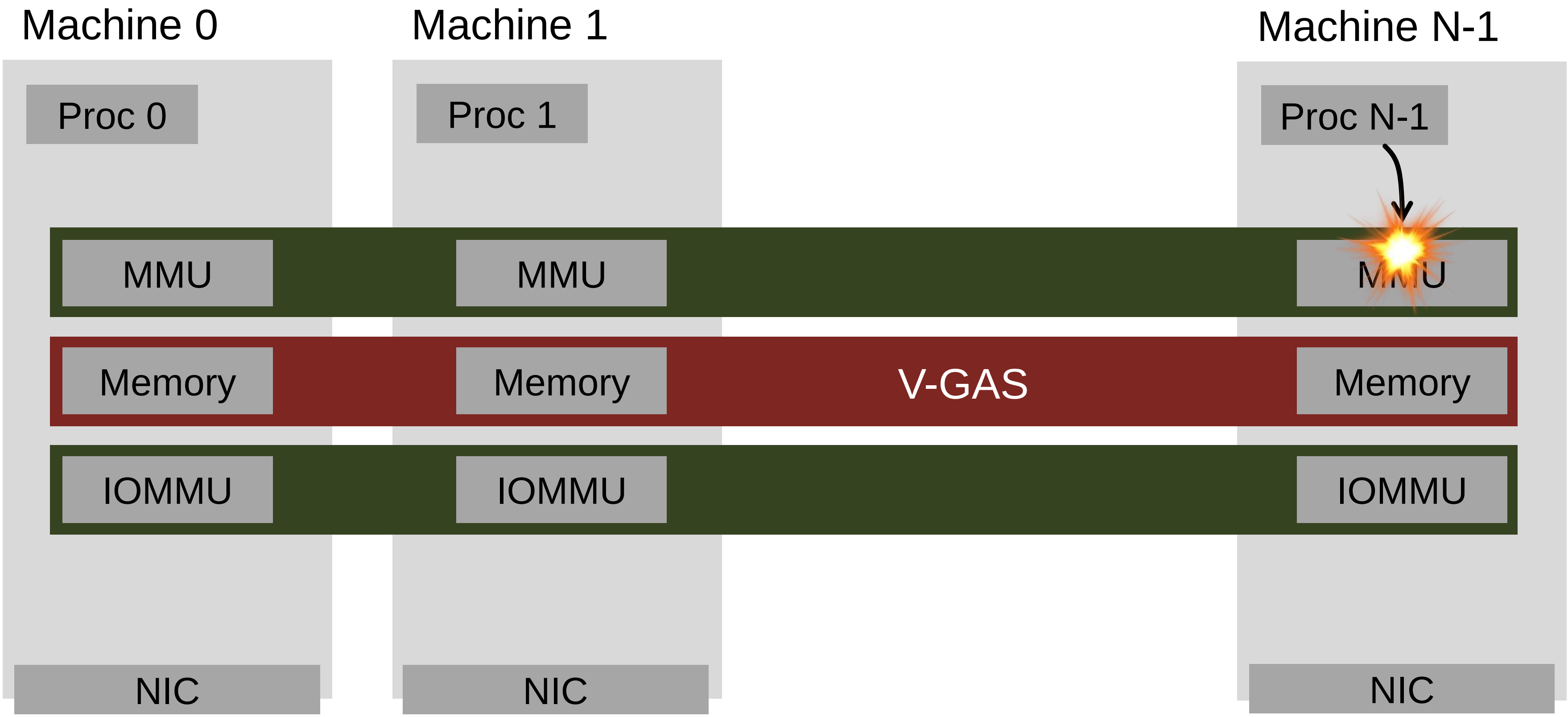




Active Access Use-Cases

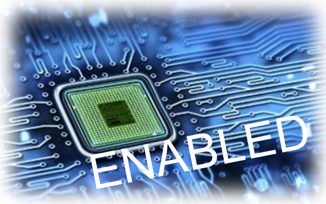
Virtual Global Address Space (V-GAS)

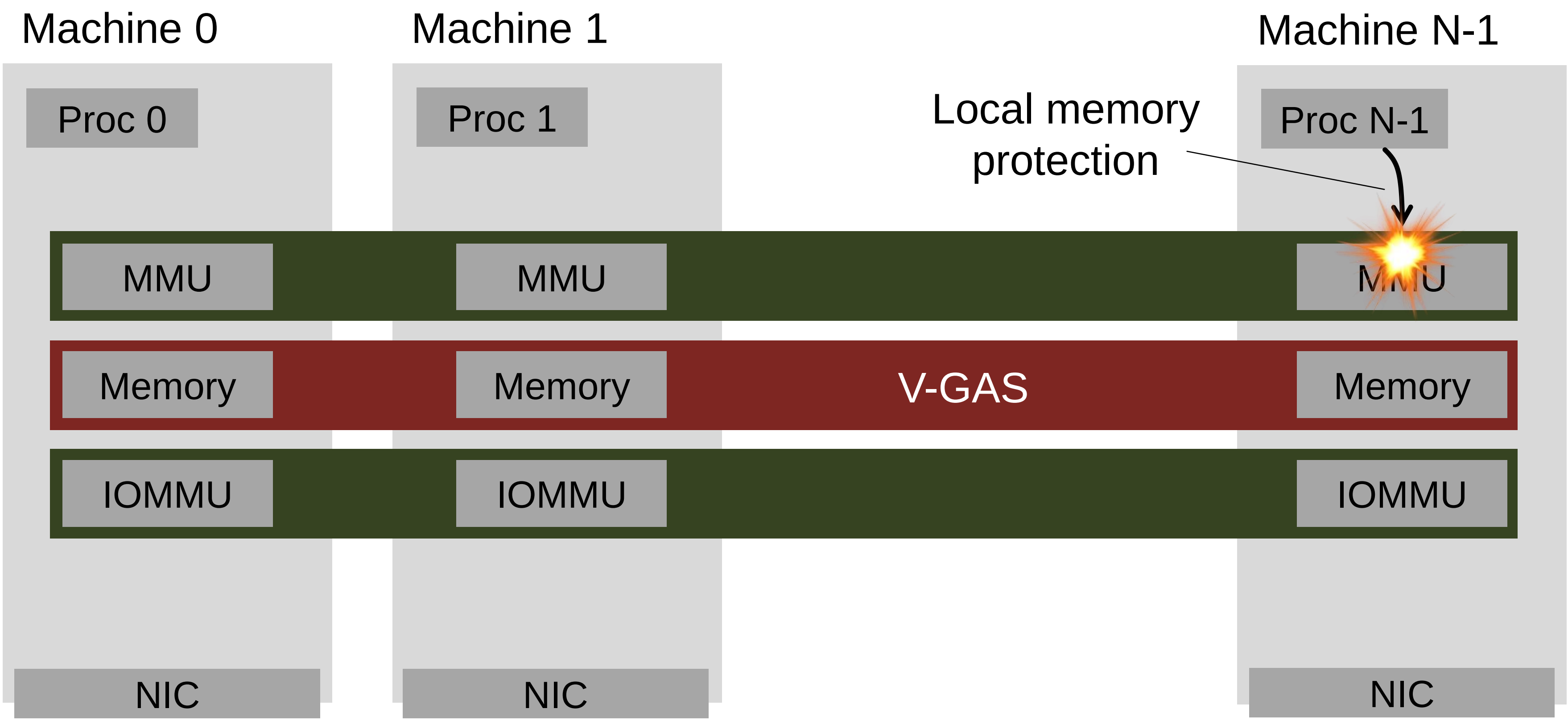




Active Access Use-Cases

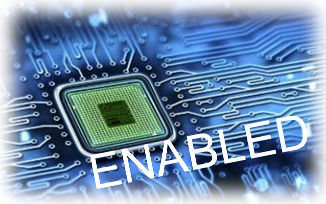
Virtual Global Address Space (V-GAS)

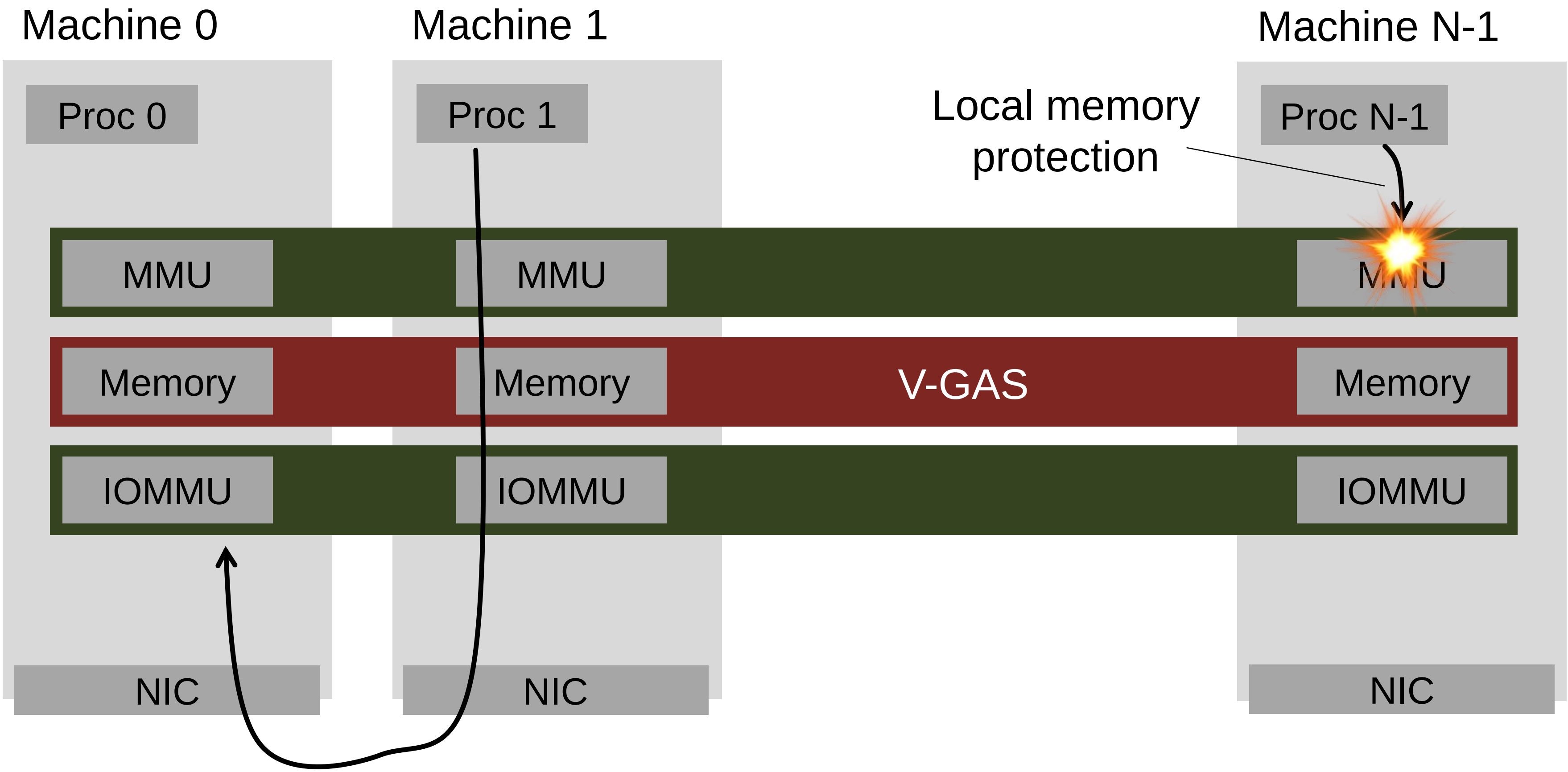




Active Access Use-Cases

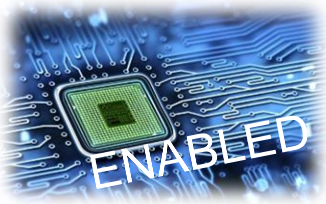
Virtual Global Address Space (V-GAS)

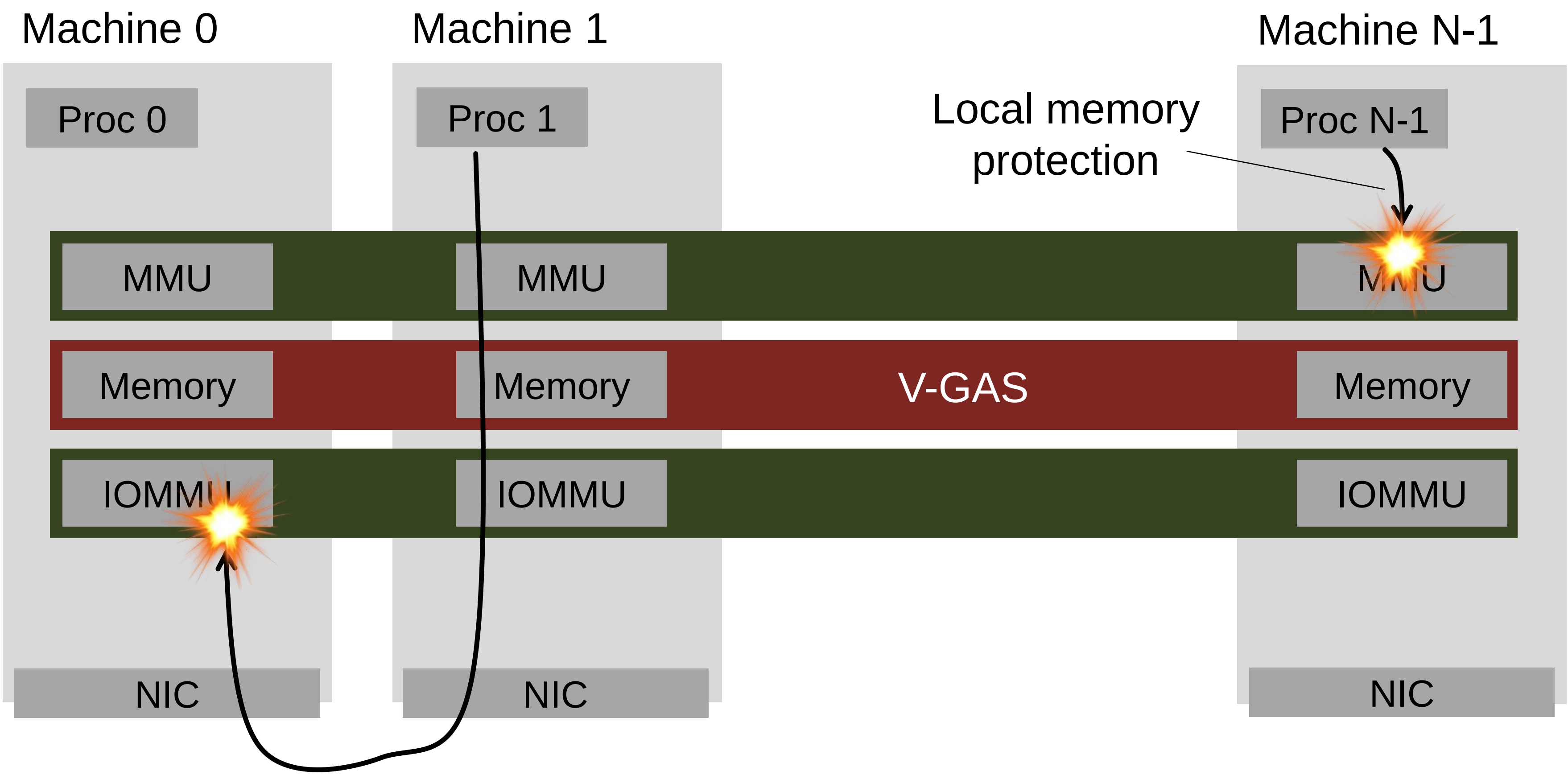




Active Access Use-Cases

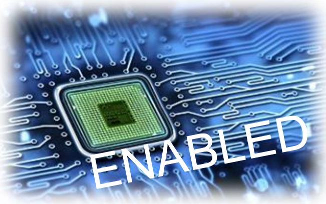
Virtual Global Address Space (V-GAS)

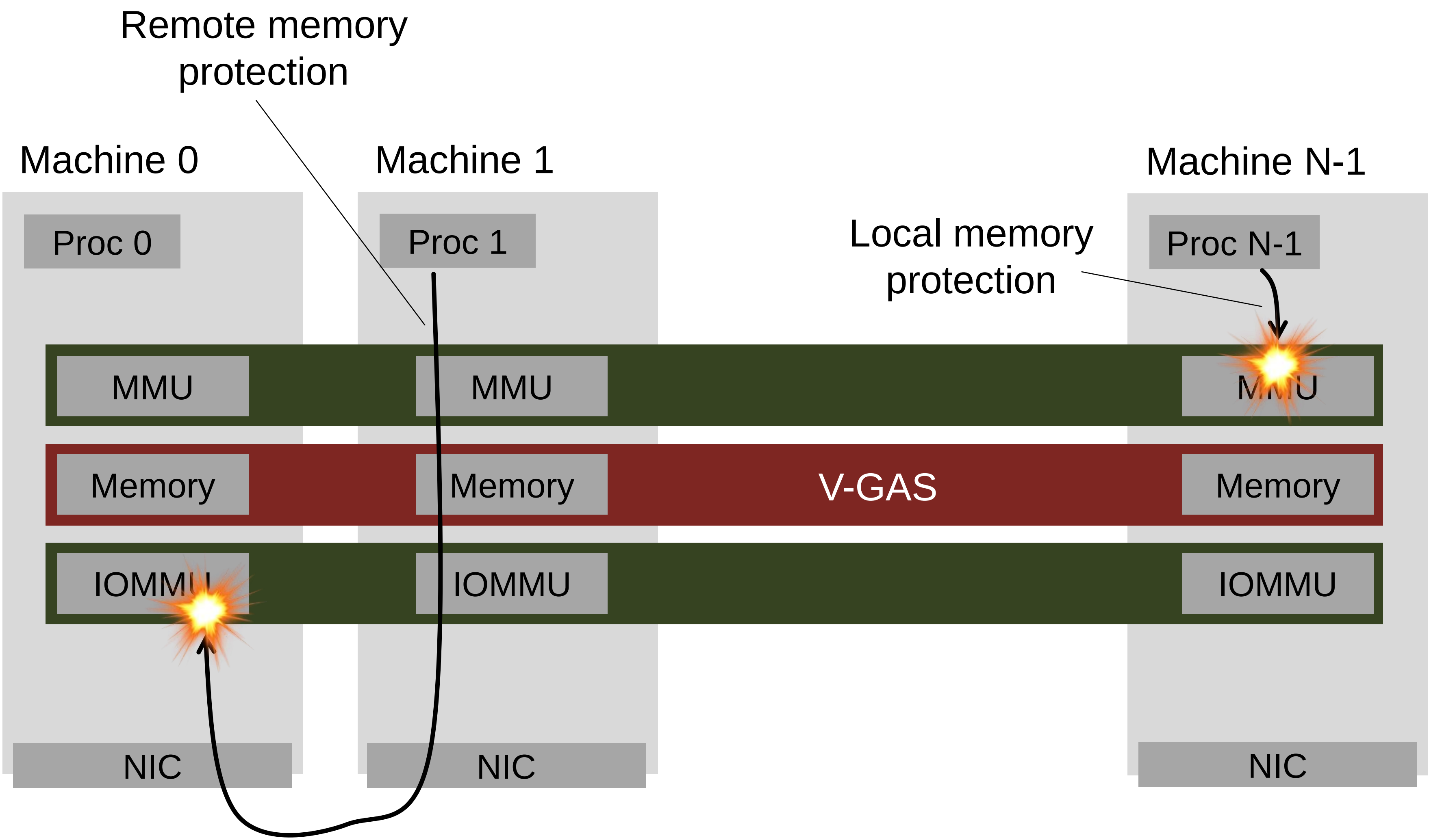




ACTIVE ACCESS USE-CASES

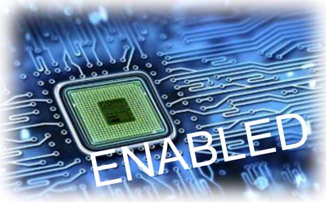
VIRTUAL GLOBAL ADDRESS SPACE (V-GAS)

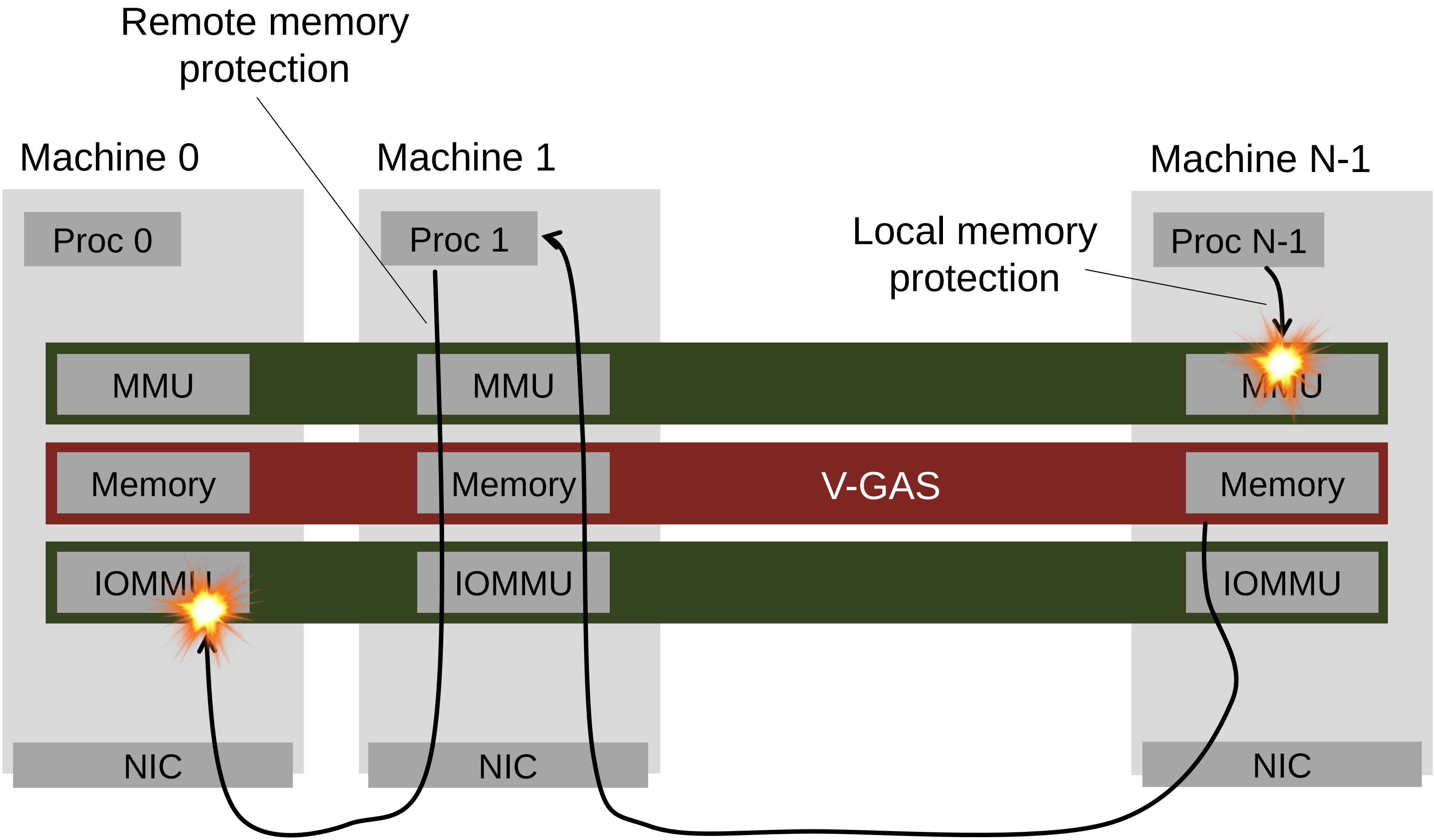




ACTIVE ACCESS USE-CASES

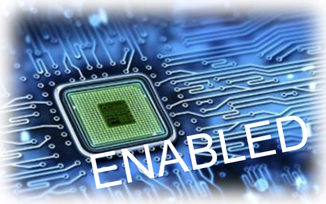
VIRTUAL GLOBAL ADDRESS SPACE (V-GAS)

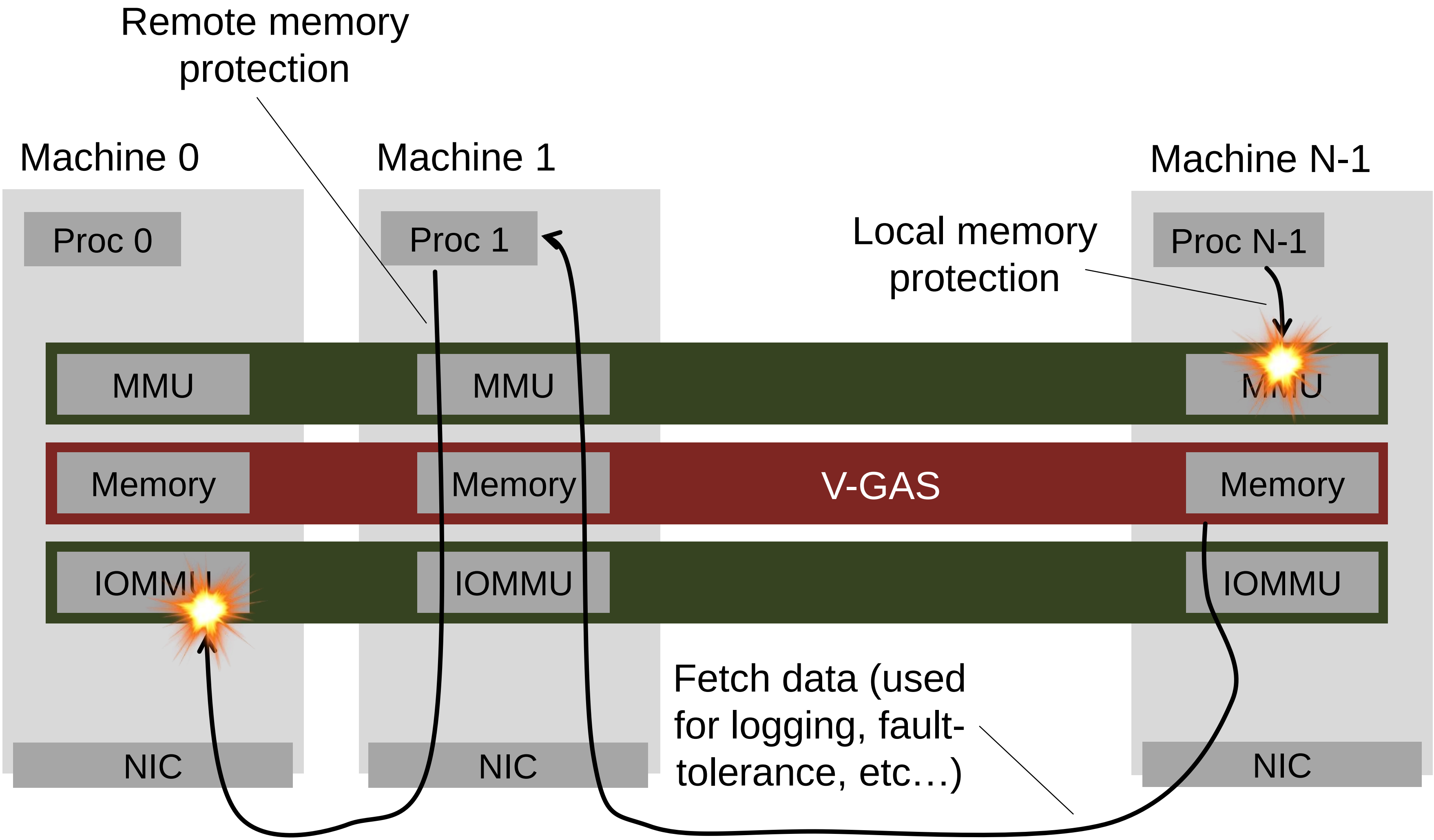




Active Access Use-Cases

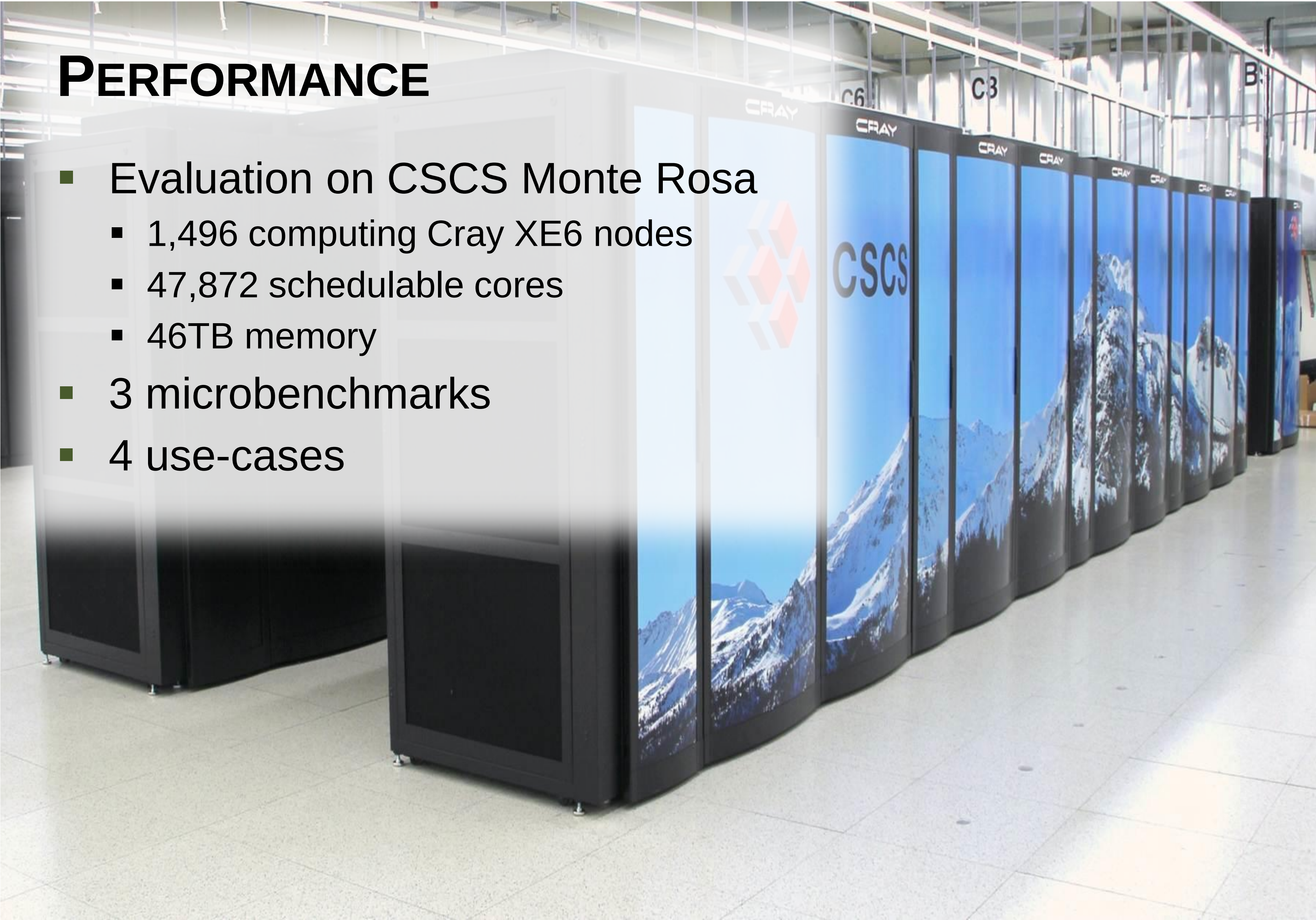
Virtual Global Address Space (V-GAS)





PERFORMANCE

- Evaluation on CSCS Monte Rosa
  - 1,496 computing Cray XE6 nodes
  - 47,872 schedulable cores
  - 46TB memory
- 3 microbenchmarks
- 4 use-cases



PERFORMANCE: MICROBENCHMARKS

RAW DATA TRANSFER

PERFORMANCE: MICROBENCHMARKS

RAW DATA TRANSFER

- Workload simulated with [1]:

PERFORMANCE: MICROBENCHMARKS
RAW DATA TRANSFER

- Workload simulated with [1]:

[1] N. Binkert et al. The gem5 simulator. SIGARCH Comput. Archit. News. 2011

PERFORMANCE: MICROBENCHMARKS

RAW DATA TRANSFER

- Workload simulated with [1]:

gem5

- Data generated with:

[1] N. Binkert et al. The gem5 simulator. SIGARCH Comput. Archit. News. 2011

PERFORMANCE: MICROBENCHMARKS

RAW DATA TRANSFER

- Workload simulated with [1]:

  gem5

- Data generated with:
  - PktGen [2]

[1] N. Binkert et al. The gem5 simulator. SIGARCH Comput. Archit. News. 2011

[2] R. Olsson. PktGen the linux packet generator. Linux Symposium. 2005

Performance: Microbenchmarks

Raw Data Transfer

- Workload simulated with [1]:

gem5

- Data generated with:
  - PktGen [2]
  - Netmap [3]

[1] N. Binkert et al. The gem5 simulator. SIGARCH Comput. Archit. News. 2011

[2] R. Olsson. PktGen the linux packet generator. Linux Symposium. 2005

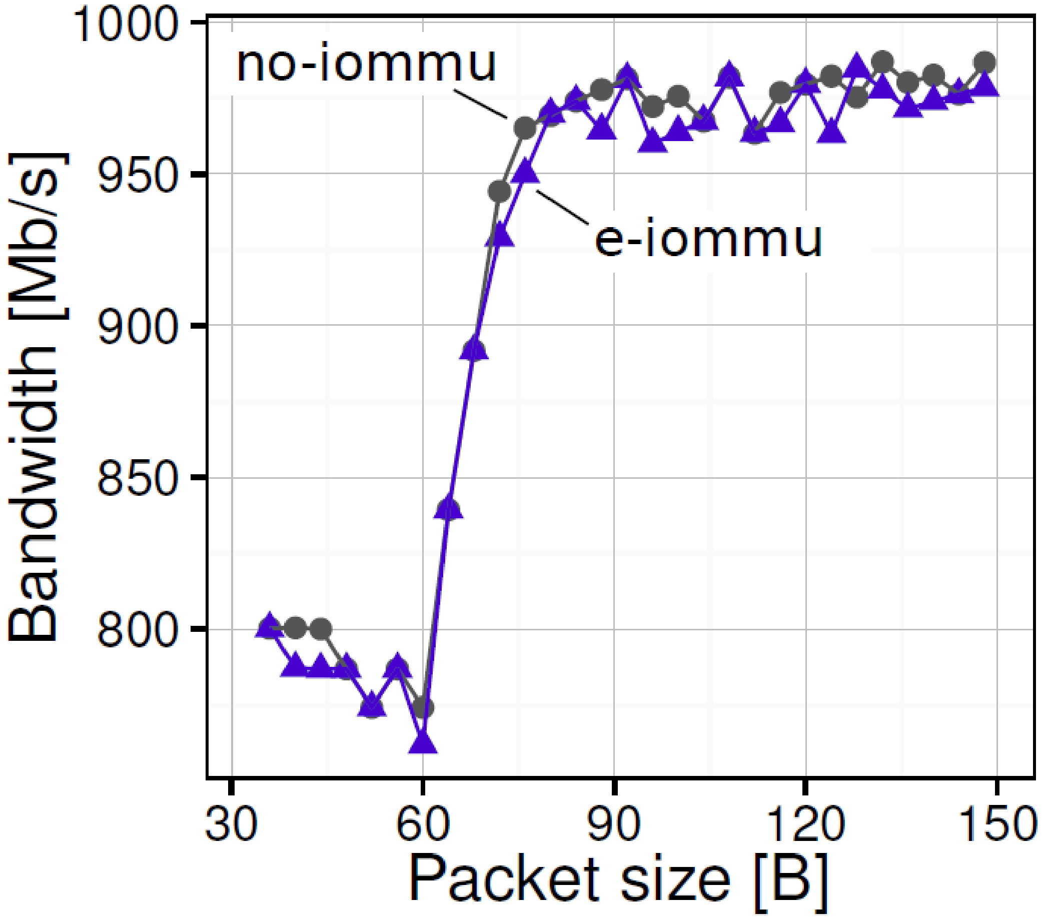
[3] L. Rizzo. netmap: A novel framework for fast packet i/o. USENIX Annual Technical Conference. 2012

PERFORMANCE: MICROBENCHMARKS

RAW DATA TRANSFER

- Workload simulated with [1]:

- Data generated with:
  - PktGen [2]
  - Netmap [3]



[1] N. Binkert et al. The gem5 simulator. SIGARCH Comput. Archit. News. 2011

[2] R. Olsson. PktGen the linux packet generator. Linux Symposium. 2005

[3] L. Rizzo. netmap: A novel framework for fast packet i/o. USENIX Annual Technical Conference. 2012

Performance: Large-Scale Codes

Comparison Targets

PERFORMANCE: LARGE-SCALE CODES

COMPARISON TARGETS

Active Access

PERFORMANCE: LARGE-SCALE CODES

COMPARISON TARGETS

Active Access

AA-Int

Performance: Large-Scale Codes

Comparison Targets

Active Access

AA-Poll

AA-Int

Performance: Large-Scale Codes

Comparison Targets

Active Access

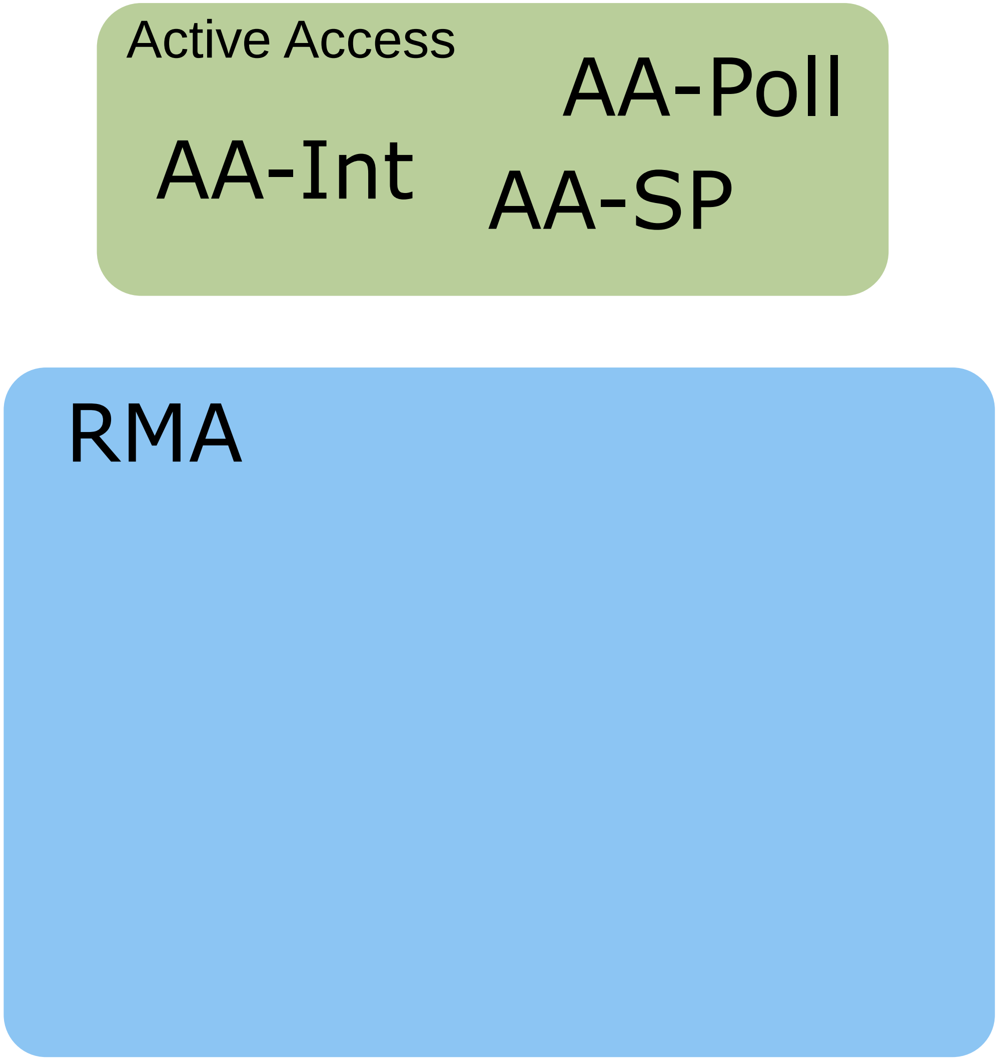
AA-Poll

AA-Int

AA-SP

Performance: Large-Scale Codes

Comparison Targets



Active Access

AA-Poll

AA-Int

AA-SP

RMA

Performance: Large-Scale Codes

Comparison Targets

Active Access

AA-Poll

AA-Int

AA-SP

RMA

CRAY

DMAPP

Performance: Large-Scale Codes

Comparison Targets

Active Access

AA-Poll

AA-Int

AA-SP

RMA

CRAY

DMAPP

INFINIBAND

Performance: Large-Scale Codes

Comparison Targets

Active Access

AA-Poll

AA-Int

AA-SP

RMA

IBM

Cell

CRAY

DMAPP

INFINIBAND

Performance: Large-Scale Codes

Comparison Targets

Active Access

AA-Poll

AA-Int

AA-SP

RMA

IBM

Cell

CRAY

DMAPP

InfiniBand

Mellanox Technologies

RoCE

Performance: Large-Scale Codes

Comparison Targets

Active Access

AA-Poll

AA-Int

AA-SP

Active Messages

RMA

IBM

Cell

CRAY

DMAPP

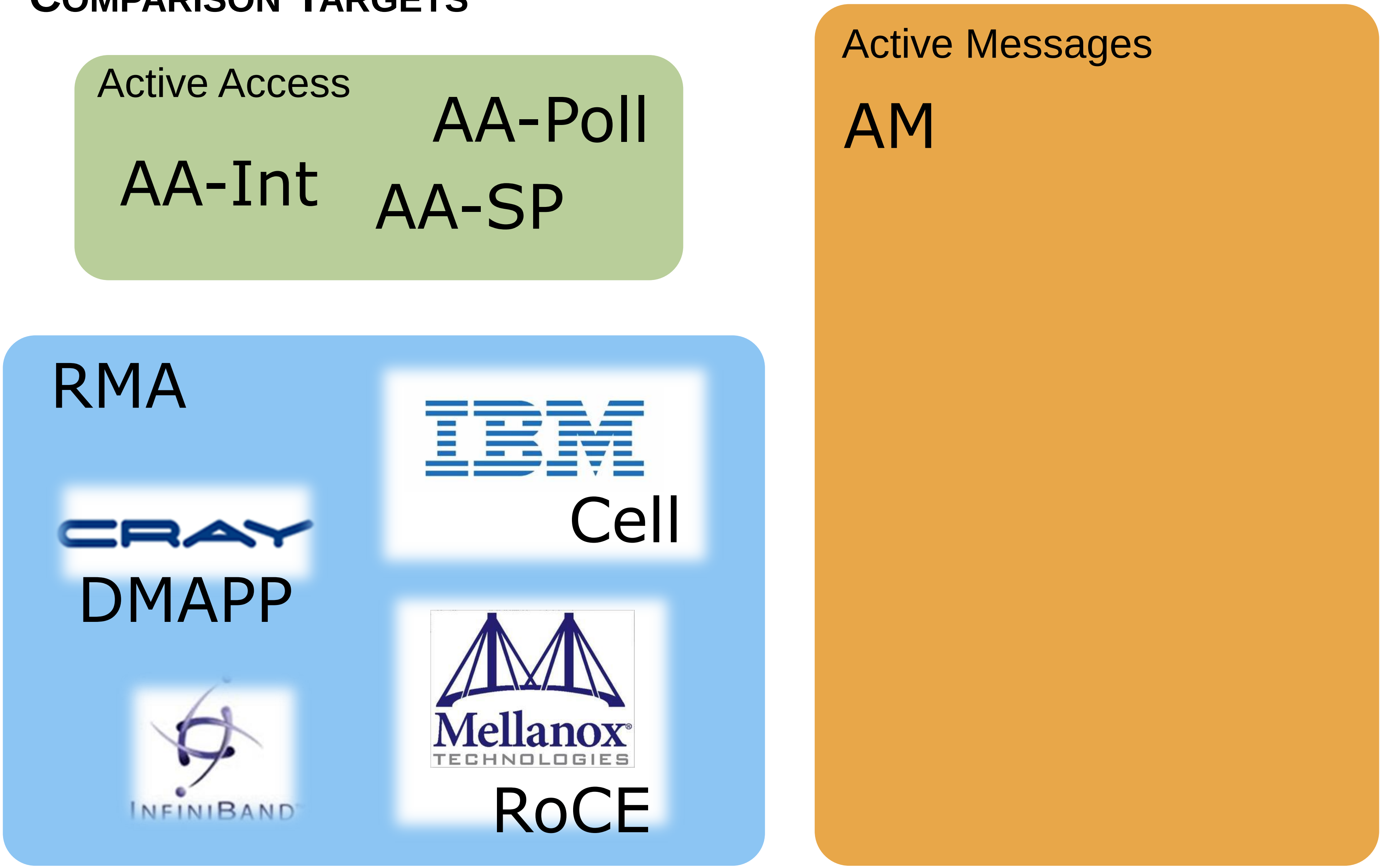
Mellanox TECHNOLOGIES

RoCE

INFINIBAND

Performance: Large-Scale Codes

Comparison Targets



Active Access

AA-Poll

AA-Int

AA-SP

RMA

Cell

DMAPP

RoCE

Active Messages

AM

Performance: Large-Scale Codes

Comparison Targets

Active Access

AA-Poll

AA-Int

AA-SP

RMA

IBM

Cell

CRAY

DMAPP

INFINIBAND

Mellanox TECHNOLOGIES

RoCE

Active Messages

AM

AM-Exp

Performance: Large-Scale Codes

Comparison Targets

Active Access

AA-Poll

AA-Int

AA-SP

RMA

IBM

Cell

CRAY

DMAPP

INFINIBAND

Mellanox TECHNOLOGIES

RoCE

Active Messages

AM AM-Onload

AM-Exp

Performance: Large-Scale Codes

Comparison Targets

Active Access

AA-Poll

AA-Int

AA-SP

Active Messages

AM AM-Onload

AM-Exp AM-Ints

RMA

IBM

Cell

CRAY

DMAPP

INFINIBAND

Mellanox TECHNOLOGIES

RoCE

Performance: Large-Scale Codes

Comparison Targets

Active Access

AA-Poll

AA-Int

AA-SP

RMA

IBM

Cell

CRAY

DMAPP

INFINIBAND

Mellanox TECHNOLOGIES

RoCE

Active Messages

AM AM-Onload

AM-Exp AM-Ints

IBM

DCMF LAPI

PAMI

Performance: Large-Scale Codes

Comparison Targets

Active Access

AA-Poll

AA-Int

AA-SP

RMA

Cell

DMAPP

RoCE

Active Messages

AM AM-Onload

AM-Exp AM-Ints

DCMF LAPI

PAMI

MX

Performance: Large-Scale Codes

Comparison Targets

Active Access

AA-Poll

AA-Int

AA-SP

RMA

IBM

Cell

CRAY

DMAPP

INFINIBAND

Mellanox TECHNOLOGIES

RoCE

Active Messages

AM AM-Onload

AM-Exp AM-Ints

IBM

DCMF LAPI

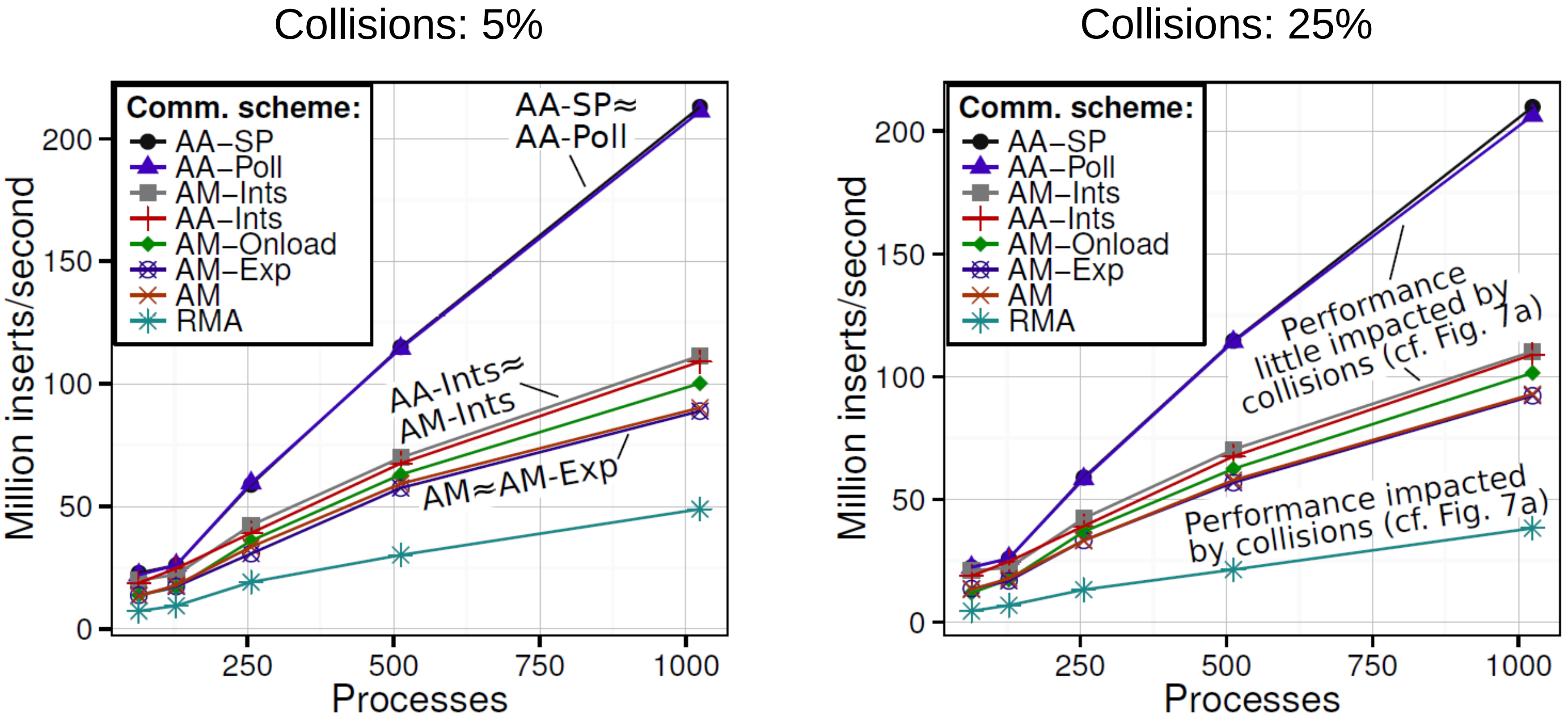
PAMI

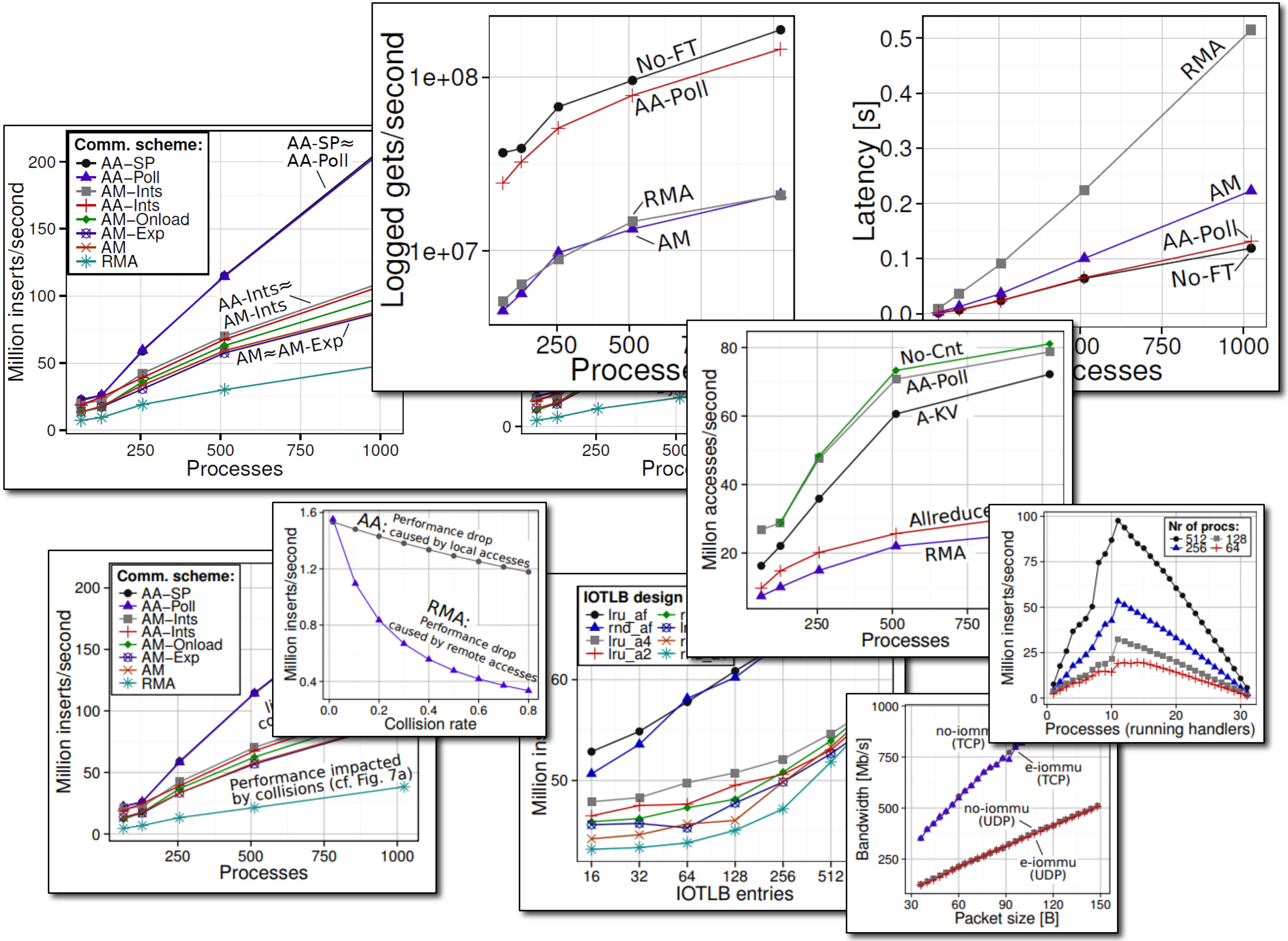
Myricom MX

AM++ GASNet

Performance: Large-Scale Codes

Distributed Hashtable




Comm. scheme:
AA–SP
AA–Poll
AM–Ints
AA–Ints
AM–Onload
AM–Exp
AM
RMA
AA-SP≈
AA-Poll
AA-Ints≈
AM-Ints
AM≈AM-Exp
Million inserts/second
200
150
100
50
0
250
500
750
1000
Processes
Logged gets/second
1e+08
1e+07
No-FT
AA-Poll
RMA
AM
Processes
Latency [s]
0.5
0.4
0.3
0.2
0.1
0.0
RMA
AM
AA-Poll
No-FT
750
1000
Millon accesses/second
80
60
40
20
No-Cnt
AA-Poll
A-KV
Allreduce
RMA
250
500
750
Processes
Million inserts/second
1.6
1.2
0.8
0.4
0.0
0.2
0.4
0.6
0.8
AA: Performance drop caused by local accesses
RMA: Performance drop caused by remote accesses
Collision rate
Performance impacted by collisions (cf. Fig. 7a)
IOTLB design
lru_af
rnd_af
lru_a4
lru_a2
60
50
16
32
64
128
256
512
IOTLB entries
Million inserts/second
100
75
50
25
0
Nr of procs:
512
128
256
64
10
20
30
Processes (running handlers)
Bandwidth [Mb/s]
1000
750
500
250
no-iommu (TCP)
e-iommu (TCP)
no-iommu (UDP)
e-iommu (UDP)
30
60
90
120
150
Packet size [B]

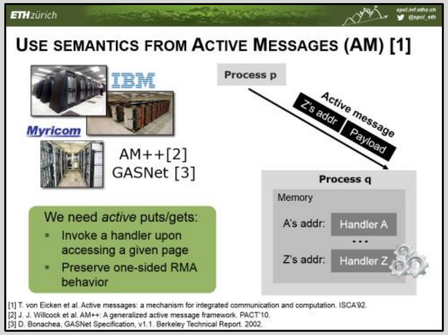
CONCLUSIONS

CONCLUSIONS

Active Access

CONCLUSIONS

Active Access



USE SEMANTICS FROM ACTIVE MESSAGES (AM) [1]

AM++[2]
GASNet [3]

We need *active* puts/gets:

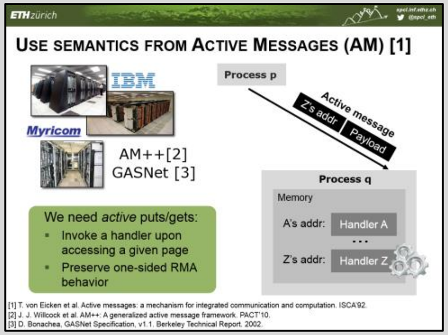
- Invoke a handler upon accessing a given page
- Preserve one-sided RMA behavior

[1] T. von Eicken et al. Active messages: a mechanism for integrated communication and computation. ISCA'92.
[2] J. J. Willcock et al. AM++: A generalized active message framework. PACT'10.
[3] D. Bonachea, GASNet Specification, v1.1. Berkeley Technical Report. 2002.

Alleviates RMA's problems with AMs while preserving one-sided semantics

CONCLUSIONS

Active Access



ETH zürich spcl.inf.ethz.ch @spcl_eth

USE SEMANTICS FROM ACTIVE MESSAGES (AM) [1]

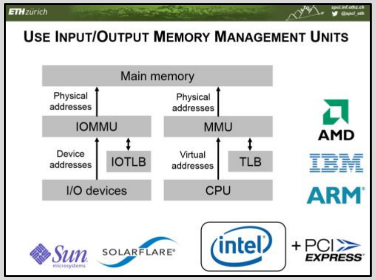
We need *active* puts/gets:

- Invoke a handler upon accessing a given page
- Preserve one-sided RMA behavior

[1] T. von Eicken et al. Active messages: a mechanism for integrated communication and computation. ISCA'92.
[2] J. J. Willcock et al. AM++: A generalized active message framework. PACT'10.
[3] D. Bonachea, GASNet Specification, v1.1. Berkeley Technical Report. 2002.

Alleviates RMA's problems with AMs
while preserving one-sided semantics

Uses commodity
& common IOMMUs

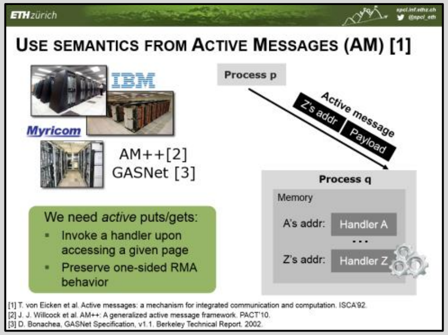


ETH zürich spcl.inf.ethz.ch @spcl_eth

USE INPUT/OUTPUT MEMORY MANAGEMENT UNITS

Conclusions

Active Access



Use semantics from Active Messages (AM) [1]

AM++[2]
GASNet [3]

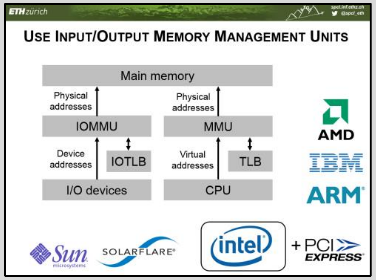
We need *active* puts/gets:

- Invoke a handler upon accessing a given page
- Preserve one-sided RMA behavior

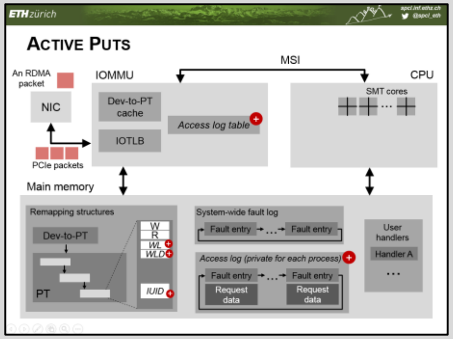
[1] T. von Eicken et al. Active messages: a mechanism for integrated communication and computation. ISCA'92.
[2] J. J. Willcock et al. AM++: A generalized active message framework. PACT'10.
[3] D. Bonachea, GASNet Specification, v1.1. Berkeley Technical Report. 2002.

Alleviates RMA's problems with AMs while preserving one-sided semantics

Uses commodity & common IOMMUs



Use Input/Output Memory Management Units

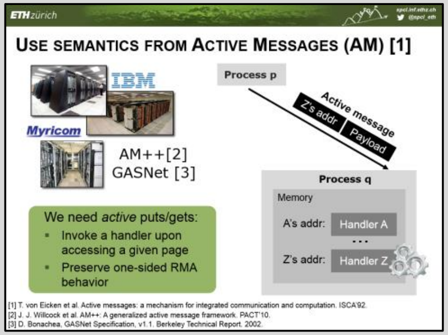


Active Puts

Extends paging capabilities in a distributed environment

CONCLUSIONS

Active Access



USE SEMANTICS FROM ACTIVE MESSAGES (AM) [1]

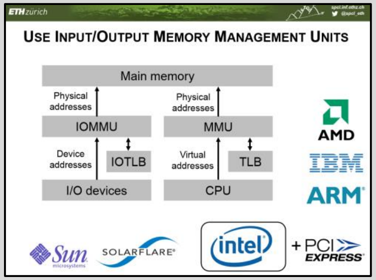
We need *active* puts/gets:

- Invoke a handler upon accessing a given page
- Preserve one-sided RMA behavior

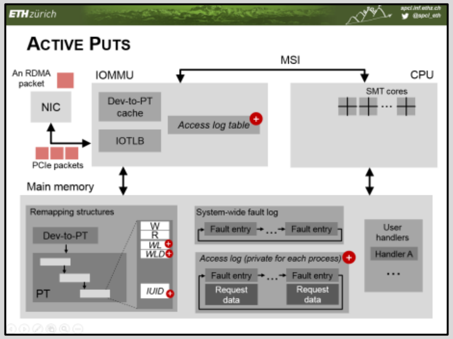
[1] T. von Eicken et al. Active messages: a mechanism for integrated communication and computation. ISCA'92.
[2] J. J. Willcock et al. AM++: A generalized active message framework. PACT'10.
[3] D. Bonachea, GASNet Specification, v1.1. Berkeley Technical Report. 2002.

Alleviates RMA's problems with AMs while preserving one-sided semantics

Uses commodity & common IOMMUs



USE INPUT/OUTPUT MEMORY MANAGEMENT UNITS



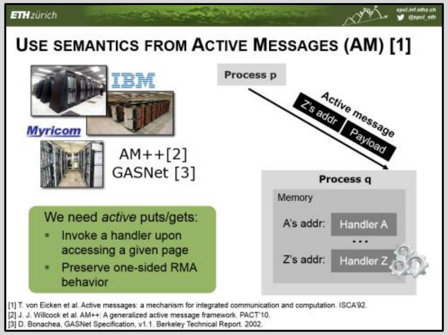
ACTIVE PUTS

Extends paging capabilities in a distributed environment

Data-centric programming

Conclusions

Active Access



Use semantics from Active Messages (AM) [1]

AM++[2]
GASNet [3]

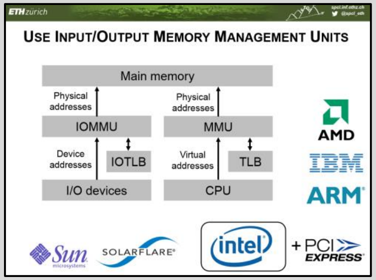
We need *active* puts/gets:

- Invoke a handler upon accessing a given page
- Preserve one-sided RMA behavior

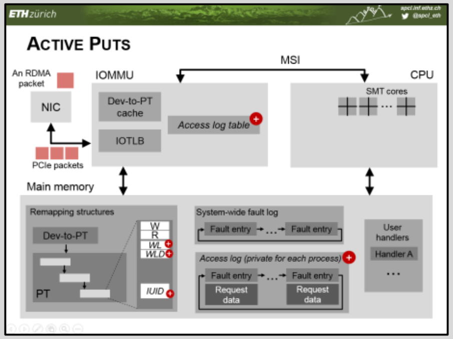
[1] T. von Eicken et al. Active messages: a mechanism for integrated communication and computation. ISCA'92.
[2] J. J. Willcock et al. AM++: A generalized active message framework. PACT'10.
[3] D. Bonachea, GASNet Specification, v1.1. Berkeley Technical Report. 2002.

Alleviates RMA's problems with AMs while preserving one-sided semantics

Uses commodity & common IOMMUs



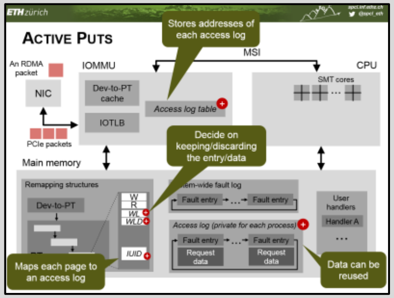
Use Input/Output Memory Management Units



Active Puts

Extends paging capabilities in a distributed environment

Data-centric programming



Active Puts

Stores addresses of each access log

Decide on keeping/discarding the entry/data

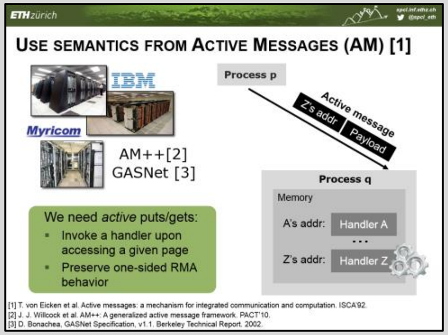
Maps each page to an access log

Data can be reused

Addresses of pages guide the execution of handlers

Conclusions

Active Access



USE SEMANTICS FROM ACTIVE MESSAGES (AM) [1]

AM++[2]
GASNet [3]

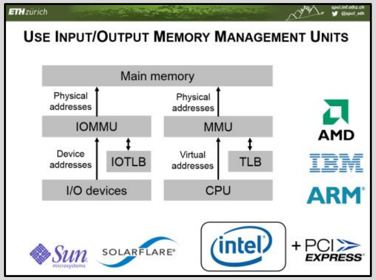
We need *active* puts/gets:

- Invoke a handler upon accessing a given page
- Preserve one-sided RMA behavior

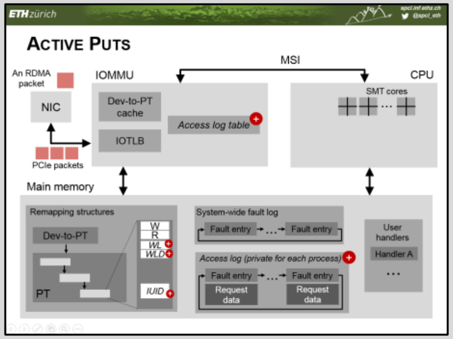
[1] T. von Eicken et al. Active messages: a mechanism for integrated communication and computation. ISCA'92.
[2] J. J. Willcock et al. AM++: A generalized active message framework. PACT'10.
[3] D. Bonachea, GASNet Specification, v1.1. Berkeley Technical Report. 2002.

Alleviates RMA's problems with AMs while preserving one-sided semantics

Uses commodity & common IOMMUs



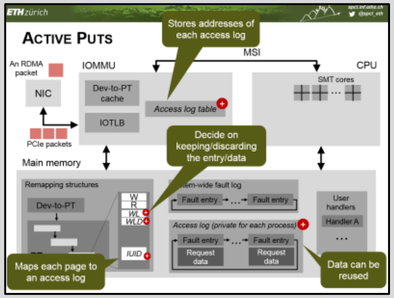
USE INPUT/OUTPUT MEMORY MANAGEMENT UNITS



ACTIVE PUTS

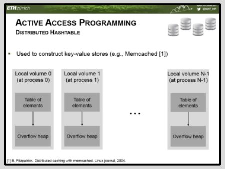
Extends paging capabilities in a distributed environment

Data-centric programming



ACTIVE PUTS

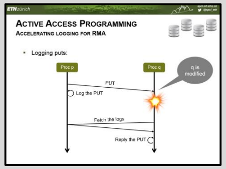
Addresses of pages guide the execution of handlers



ACTIVE ACCESS PROGRAMMING
DISTRIBUTED HASHTABLE

- Used to construct key-value stores (e.g., Memcached [1])

[1] B. Fitzpatrick. Distributed caching with memcached. Linux journal, 2004.



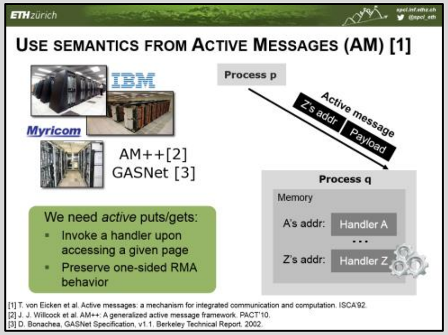
ACTIVE ACCESS PROGRAMMING
ACCELERATING LOGGING FOR RMA

- Logging puts:

Hashtables, logging schemes, counters, V-GAS, checkpointing...

Conclusions

Active Access



Use semantics from Active Messages (AM) [1]

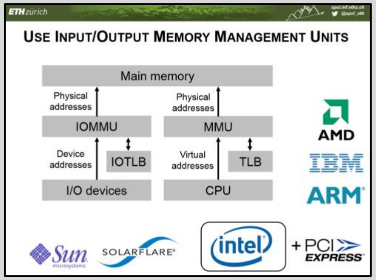
We need *active* puts/gets:

- Invoke a handler upon accessing a given page
- Preserve one-sided RMA behavior

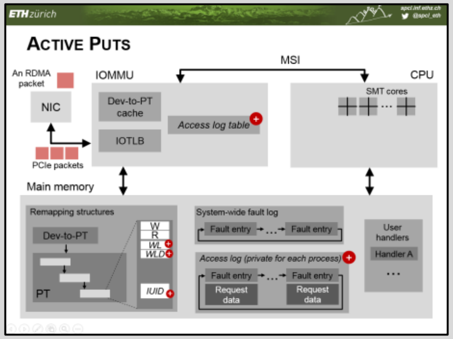
[1] T. von Eicken et al. Active messages: a mechanism for integrated communication and computation. ISCA'92.
[2] J. J. Willcock et al. AM++: A generalized active message framework. PACT'10.
[3] D. Bonachea, GASNet Specification, v1.1. Berkeley Technical Report. 2002.

Alleviates RMA's problems with AMs while preserving one-sided semantics

Uses commodity & common IOMMUs



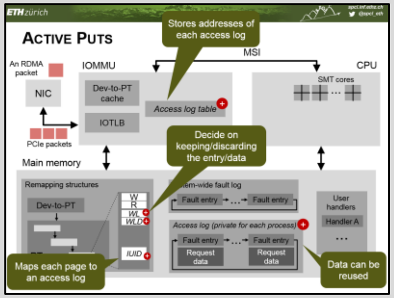
Use Input/Output Memory Management Units



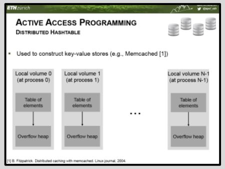
Active Puts

Extends paging capabilities in a distributed environment

Data-centric programming



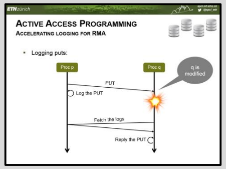
Active Puts



Active Access Programming
Distributed Hashtable

- Used to construct key-value stores (e.g., Memcached [1])

[1] B. Fitzpatrick. Distributed caching with memcached. Linux journal, 2004.



Active Access Programming
Accelerating logging for RMA

- Logging puts:

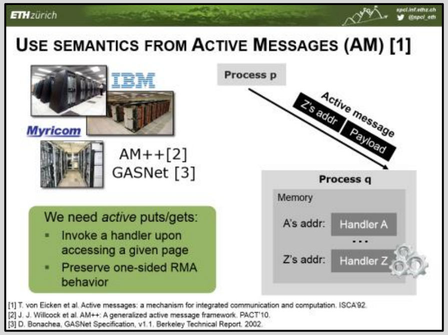
Addresses of pages guide the execution of handlers

Hashtables, logging schemes, counters, V-GAS, checkpointing...

Performance

Conclusions

Active Access



Use semantics from Active Messages (AM) [1]

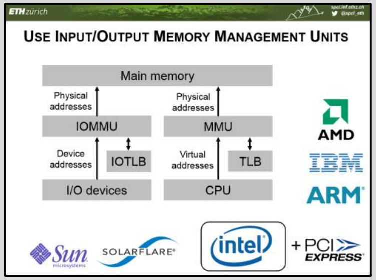
We need *active* puts/gets:

- Invoke a handler upon accessing a given page
- Preserve one-sided RMA behavior

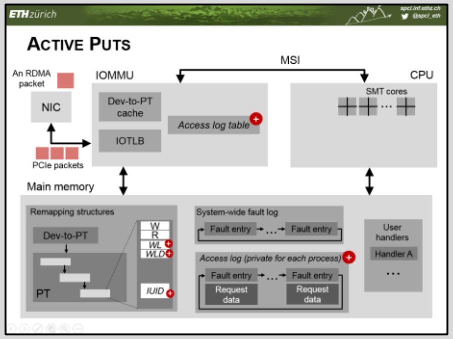
[1] T. von Eicken et al. Active messages: a mechanism for integrated communication and computation. ISCA'92.
[2] J. J. Willcock et al. AM++: A generalized active message framework. PACT'10.
[3] D. Bonachea, GASNet Specification, v1.1. Berkeley Technical Report. 2002.

Alleviates RMA's problems with AMs while preserving one-sided semantics

Uses commodity & common IOMMUs



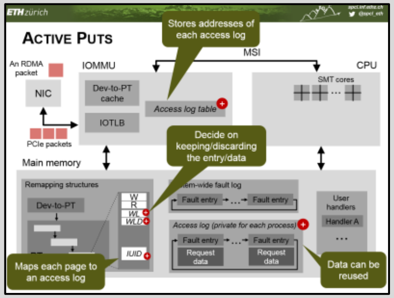
Use Input/Output Memory Management Units



Active Puts

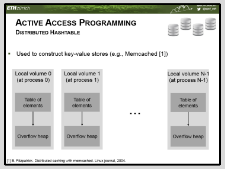
Extends paging capabilities in a distributed environment

Data-centric programming



Active Puts

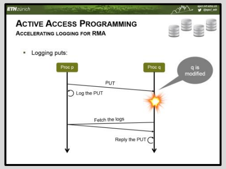
Addresses of pages guide the execution of handlers



Active Access Programming
Distributed Hashtable

- Used to construct key-value stores (e.g., Memcached [1])

[1] B. Fitzpatrick. Distributed caching with memcached. Linux journal, 2004.

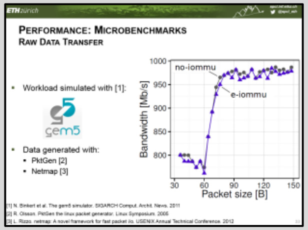


Active Access Programming
Accelerating logging for RMA

- Logging puts:

Hashtables, logging schemes, counters, V-GAS, checkpointing...

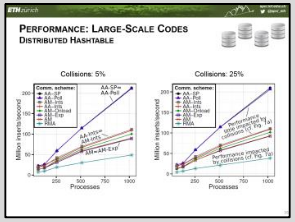
Performance



Performance: Microbenchmarks
Raw Data Transfer

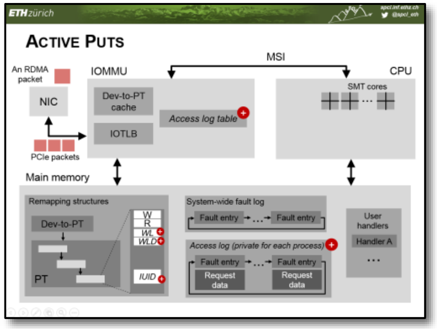
- Workload simulated with [1]:
- Data generated with:
  - PktGen [2]
  - Netmap [3]

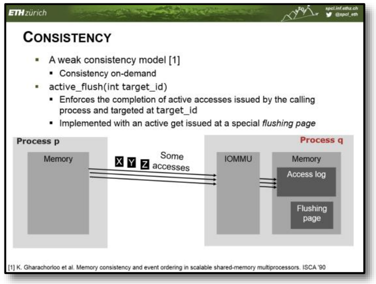
[1] N. Binkert et al. The gem5 simulator. SIGARCH Comput. Archit. News. 2011
[2] R. Olsson. PktGen the linux packet generator. Linux Symposium. 2005
[3] L. Rizzo. netmap: A novel framework for fast packet i/o. USENIX Annual Technical Conference. 2012



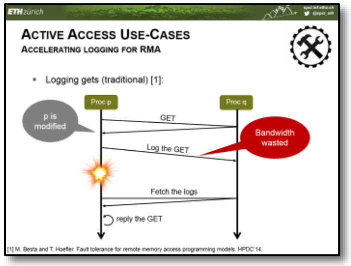
Performance: Large-Scale Codes
Distributed Hashtable

Accelerates various distributed codes


Active Puts

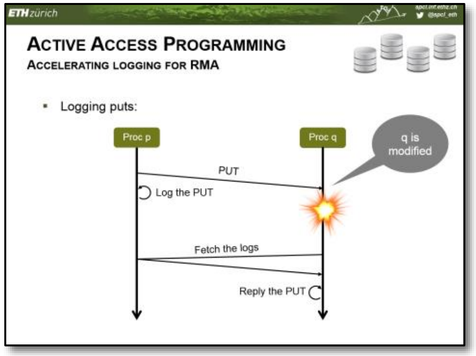

Consistency
- A weak consistency model [1]
  - Consistency on-demand
- active_flush(int target_id)
  - Enforces the completion of active accesses issued by the calling process and targeted at target_id
  - Implemented with an active get issued at a special *flushing page*

[1] K. Gharachorloo et al. Memory consistency and event ordering in scalable shared-memory multiprocessors. ISCA '90

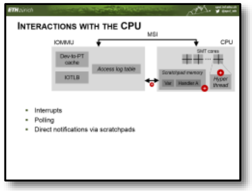

Active Access Use-Cases
Accelerating logging for RMA

- Logging gets (traditional) [1]:

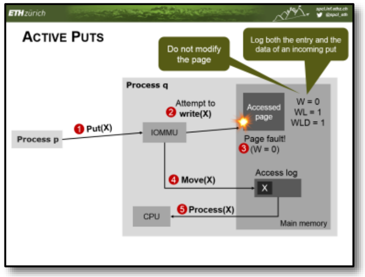
[1] M. Besta and T. Hoefler. Fault tolerance for remote memory access programming models. HPDC'14.

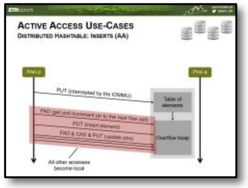

Active Access Programming
Accelerating logging for RMA

- Logging puts:

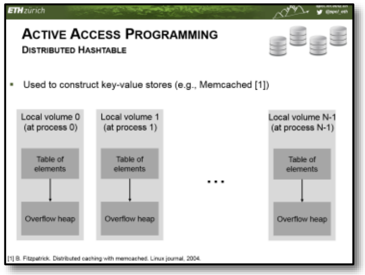

Interactions with the CPU

- Interrupts
- Polling
- Direct notifications via scratchpads


Active Puts


Active Access Use-Cases
Distributed hashtable: inserts (AA)

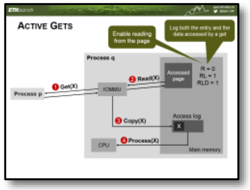
Proc p
Proc q
PUT (intercepted by the IOMMU)
Table of elements
Overflow heap
All other accesses become local

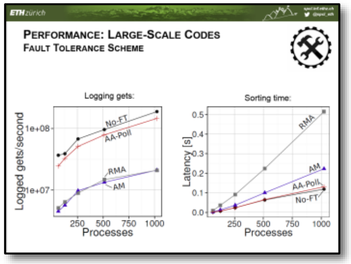

Active Access Programming
Distributed hashtable

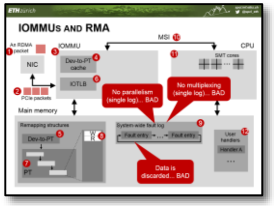
- Used to construct key-value stores (e.g., Memcached [1])

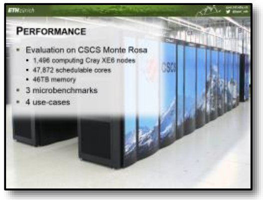
[1] B. Fitzpatrick. Distributed caching with memcached. Linux journal, 2004.

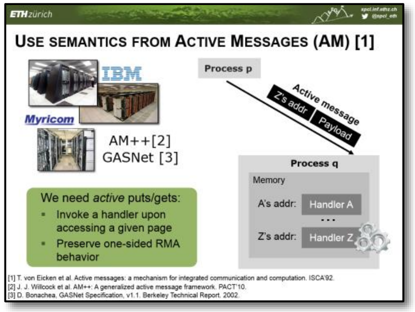
Thank you
for your attention


Active Gets


Performance: Large-Scale Codes
Fault tolerance scheme

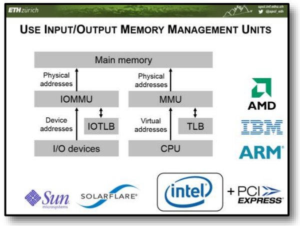

IOMMUs and RMA

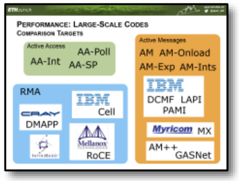

Performance
- Evaluation on CSCS Monte Rosa
  - 1,496 computing Cray XE6 nodes
  - 47,872 schedulable cores
  - 46TB memory
- 3 microbenchmarks
- 4 use-cases

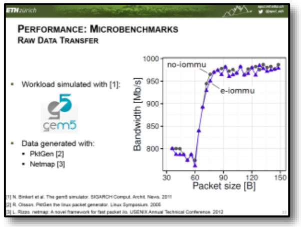

Use semantics from Active Messages (AM) [1]

We need *active* puts/gets:
- Invoke a handler upon accessing a given page
- Preserve one-sided RMA behavior

[1] T. von Eicken et al. Active messages: a mechanism for integrated communication and computation. ISCA'92.
[2] J. J. Willcock et al. AM++: A generalized active message framework. PACT'10.
[3] D. Bonachea. GASNet Specification, v1.1. Berkeley Technical Report. 2002.

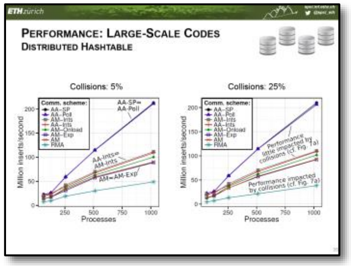

Use Input/Output Memory Management Units


Performance: Large-Scale Codes
Comparison targets


Performance: Microbenchmarks
Raw data transfer

- Workload simulated with [1]:
- Data generated with:
  - PktGen [2]
  - Netmap [3]

[1] N. Binkert et al. The gem5 simulator. SIGARCH Comput. Archit. News. 2011
[2] R. Olsson. PktGen the linux packet generator. Linux Symposium. 2005
[3] L. Rizzo. netmap: A novel framework for fast packet I/O. USENIX Annual Technical Conference. 2012


Performance: Large-Scale Codes
Distributed hashtable

Active Access Use-Cases

Accelerating logging for RMA

ACTIVE ACCESS USE-CASES

ACCELERATING LOGGING FOR RMA

- Logging – a popular mechanism for fault-tolerance.

ACTIVE ACCESS USE-CASES

ACCELERATING LOGGING FOR RMA

- Logging – a popular mechanism for fault-tolerance.
- Remote communication (puts/gets) is logged.

ACTIVE ACCESS USE-CASES

ACCELERATING LOGGING FOR RMA

- Logging – a popular mechanism for fault-tolerance.
- Remote communication (puts/gets) is logged.
- Upon a process crash, it is restored and uses the logs to replay its previous actions.

Active Access Use-Cases

Accelerating logging for RMA

- Logging – a popular mechanism for fault-tolerance.
- Remote communication (puts/gets) is logged.
- Upon a process crash, it is restored and uses the logs to replay its previous actions.
- Logs are stored in volatile memories.

ACTIVE ACCESS USE-CASES

ACCELERATING LOGGING FOR RMA

ACTIVE ACCESS USE-CASES

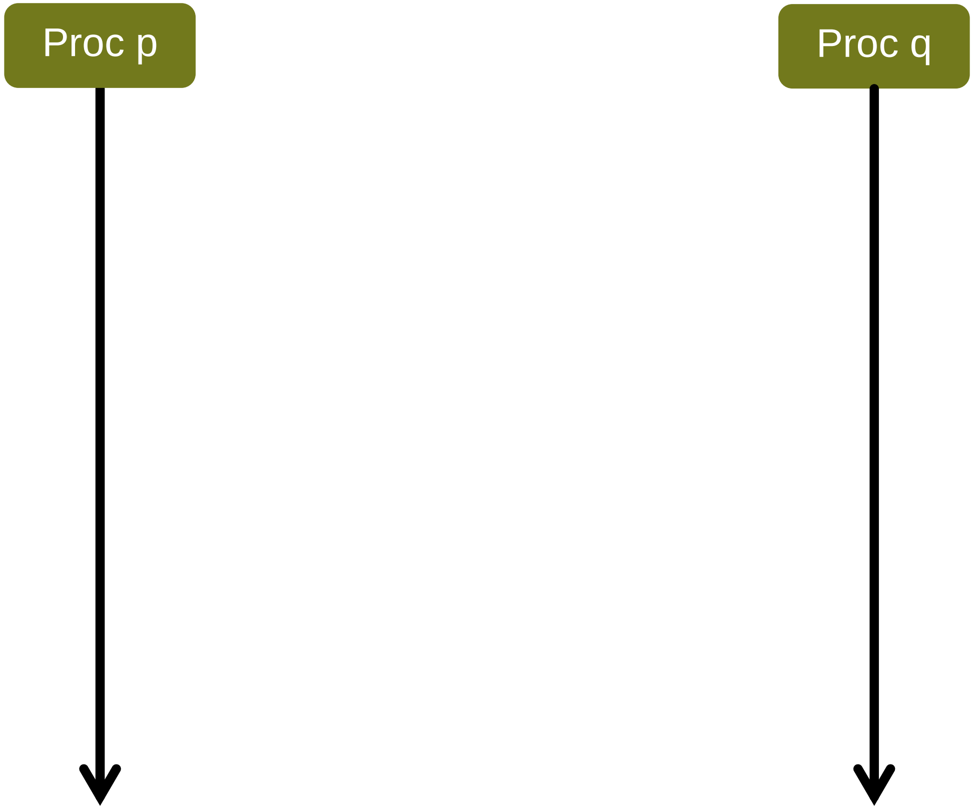
ACCELERATING LOGGING FOR RMA

- Logging puts:

Active Access Use-Cases

Accelerating logging for RMA

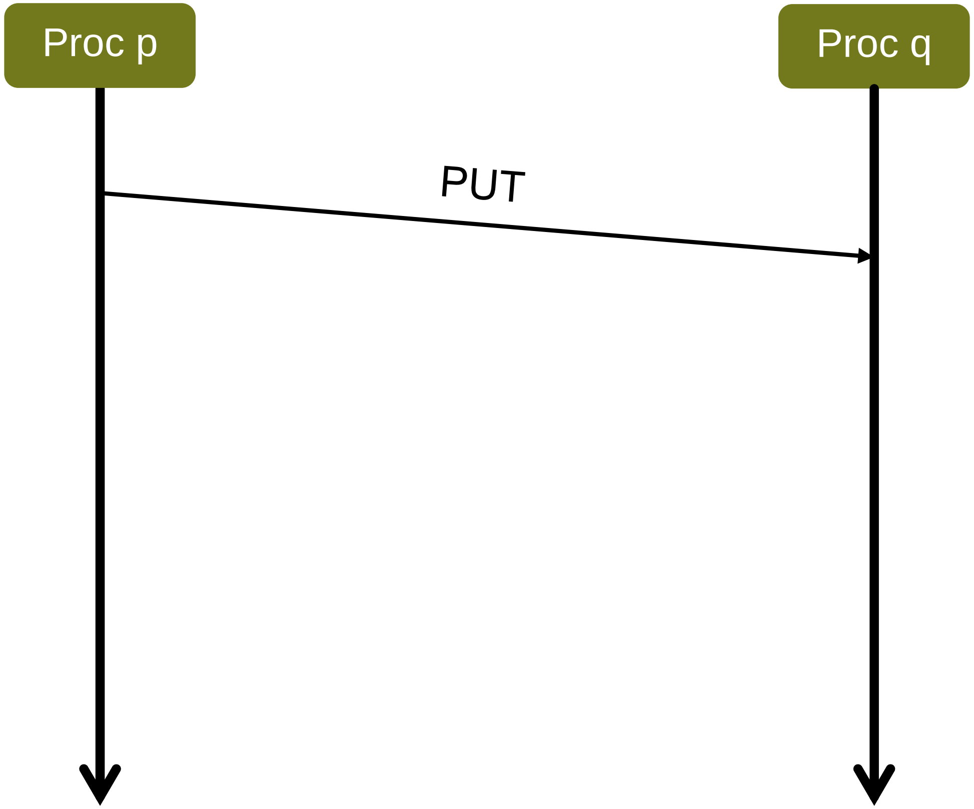
- Logging puts:



Active Access Use-Cases

Accelerating logging for RMA

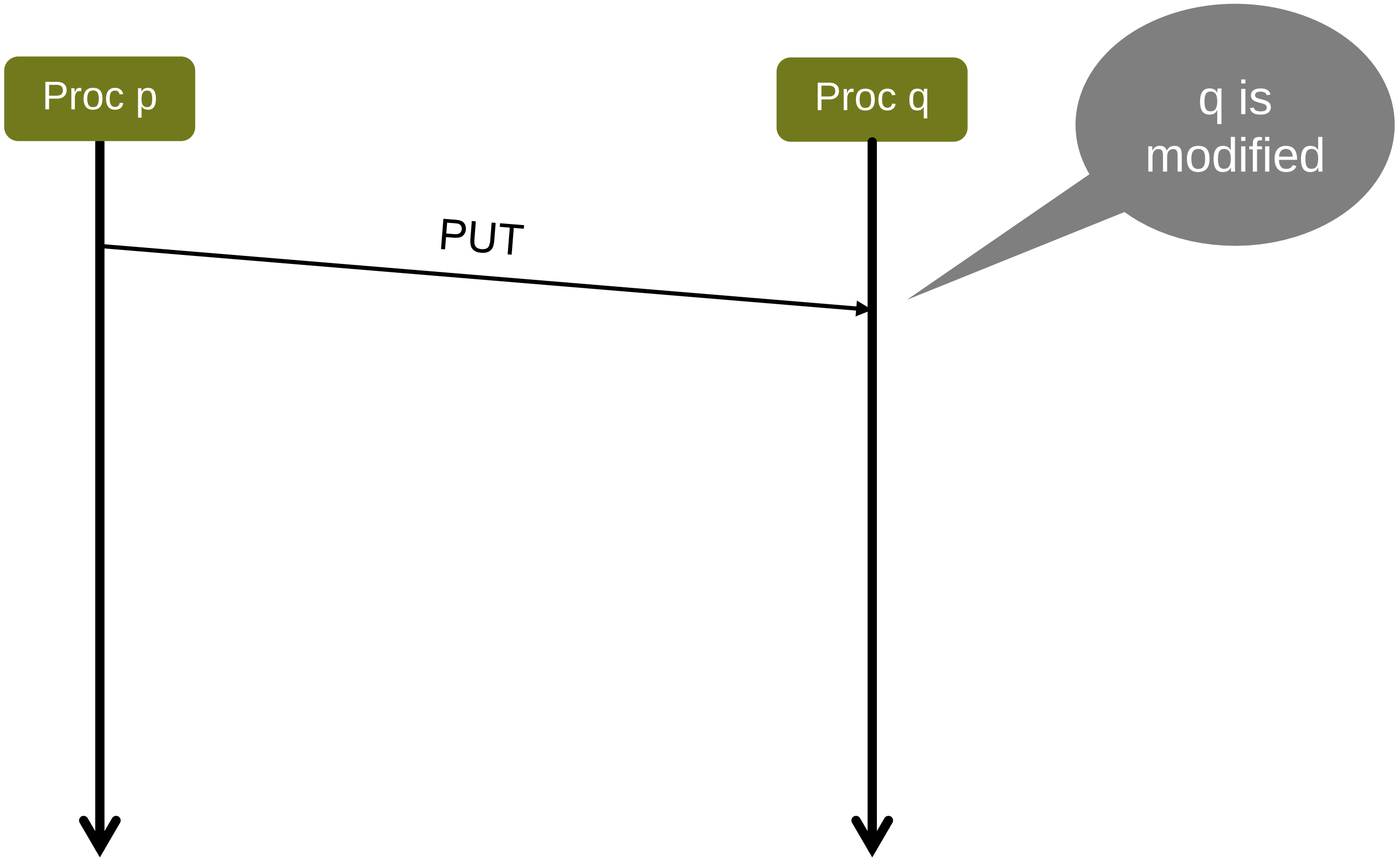
- Logging puts:



Active Access Use-Cases

Accelerating logging for RMA

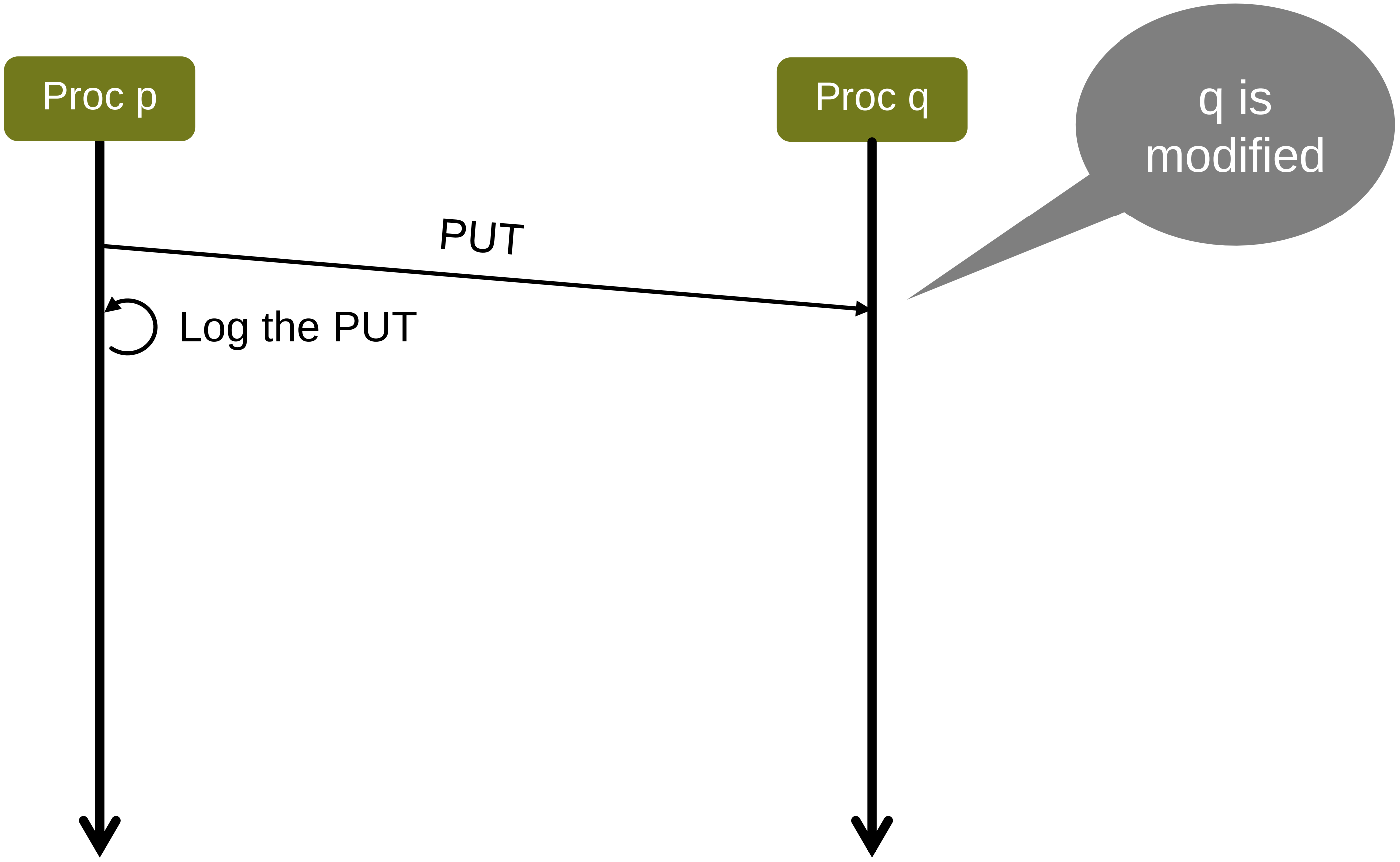
- Logging puts:



ACTIVE ACCESS USE-CASES

ACCELERATING LOGGING FOR RMA

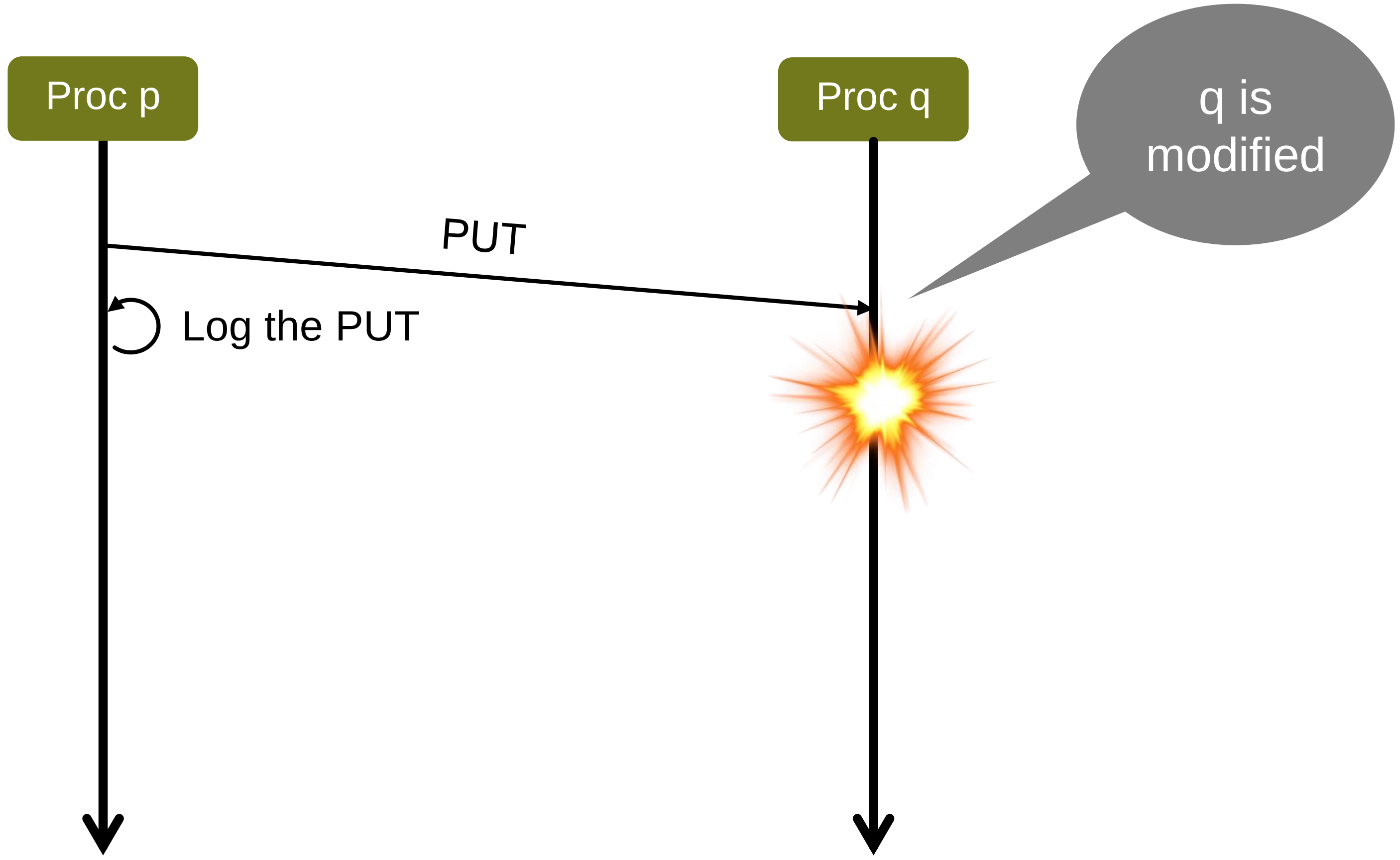
- Logging puts:



Active Access Use-Cases

Accelerating logging for RMA

- Logging puts:



Proc p

Proc q

PUT

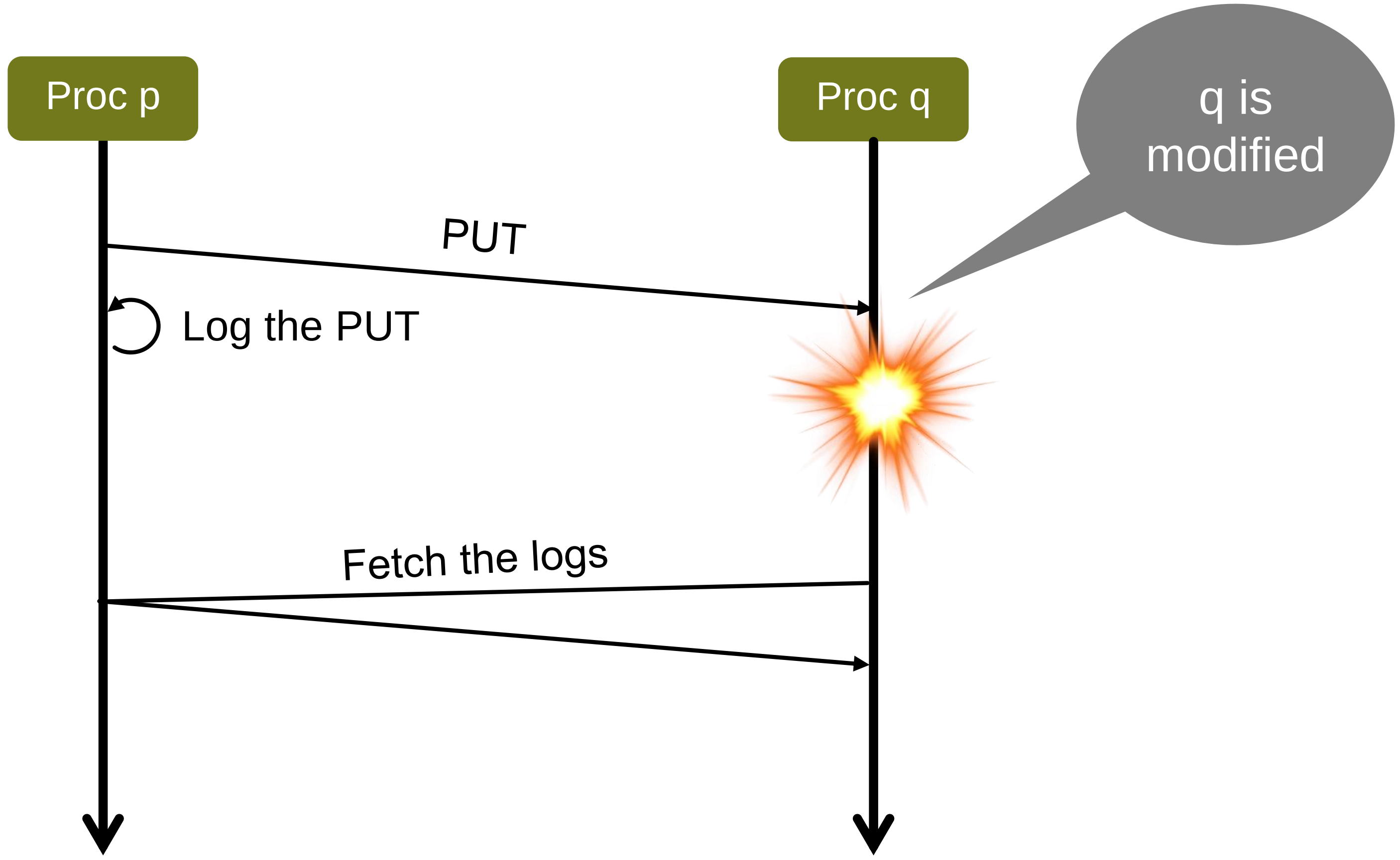
Log the PUT

q is modified

ACTIVE ACCESS USE-CASES

ACCELERATING LOGGING FOR RMA

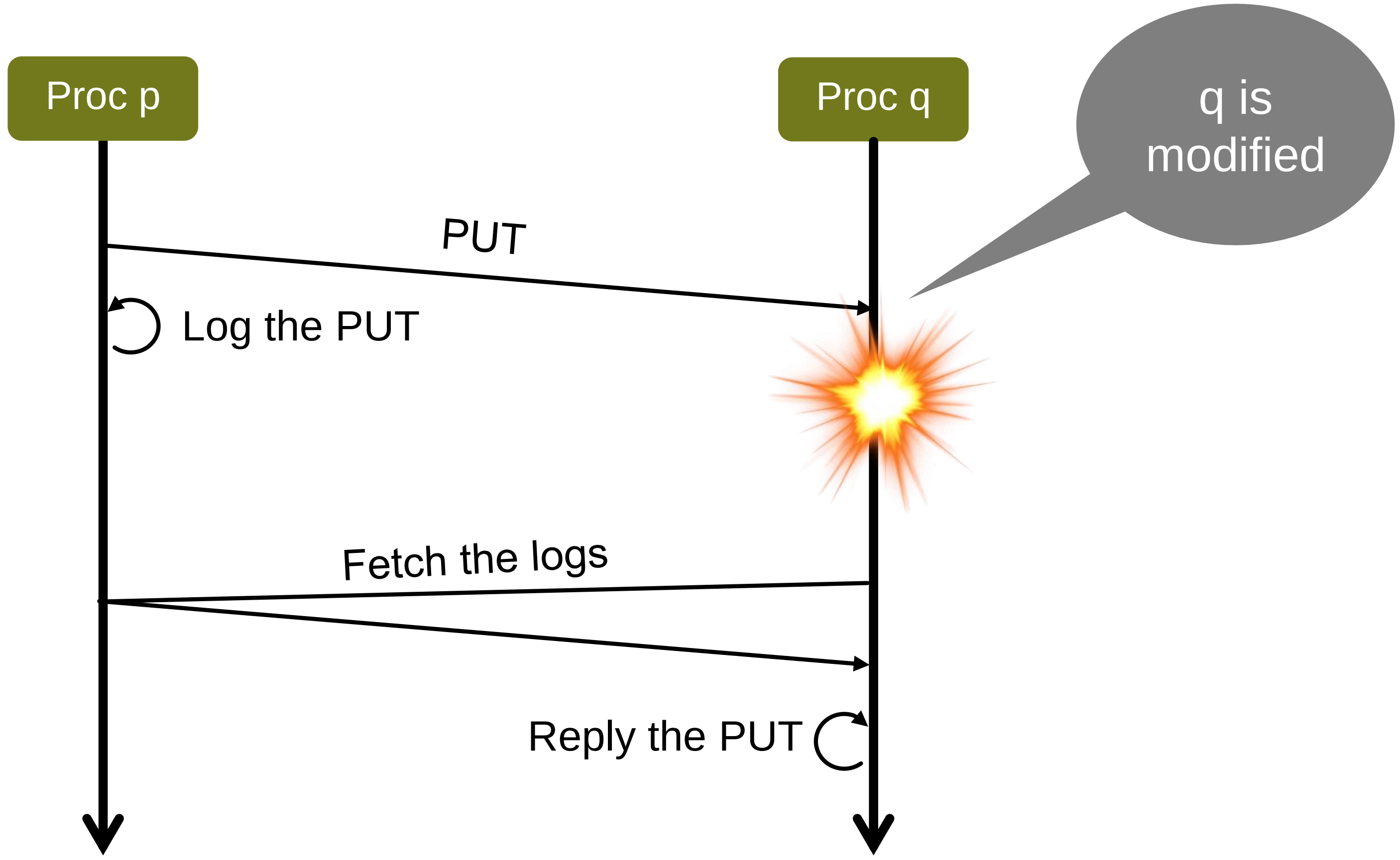
- Logging puts:



ACTIVE ACCESS USE-CASES

ACCELERATING LOGGING FOR RMA

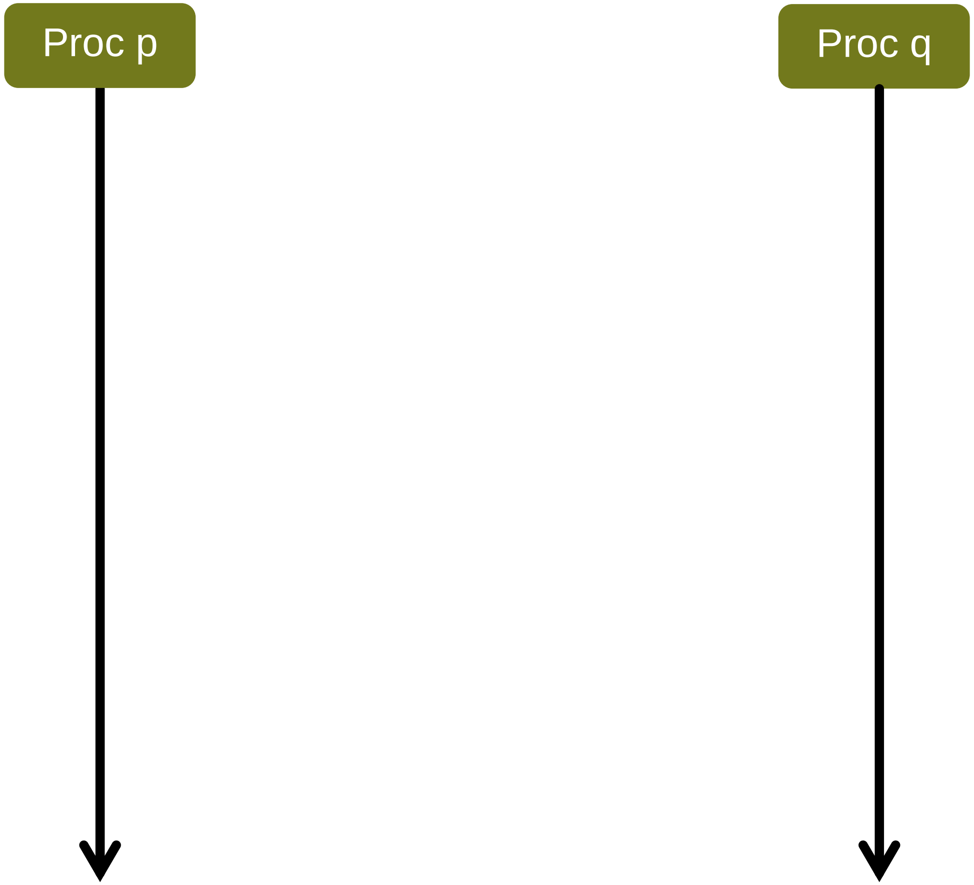
- Logging puts:



Active Access Use-Cases

Accelerating logging for RMA

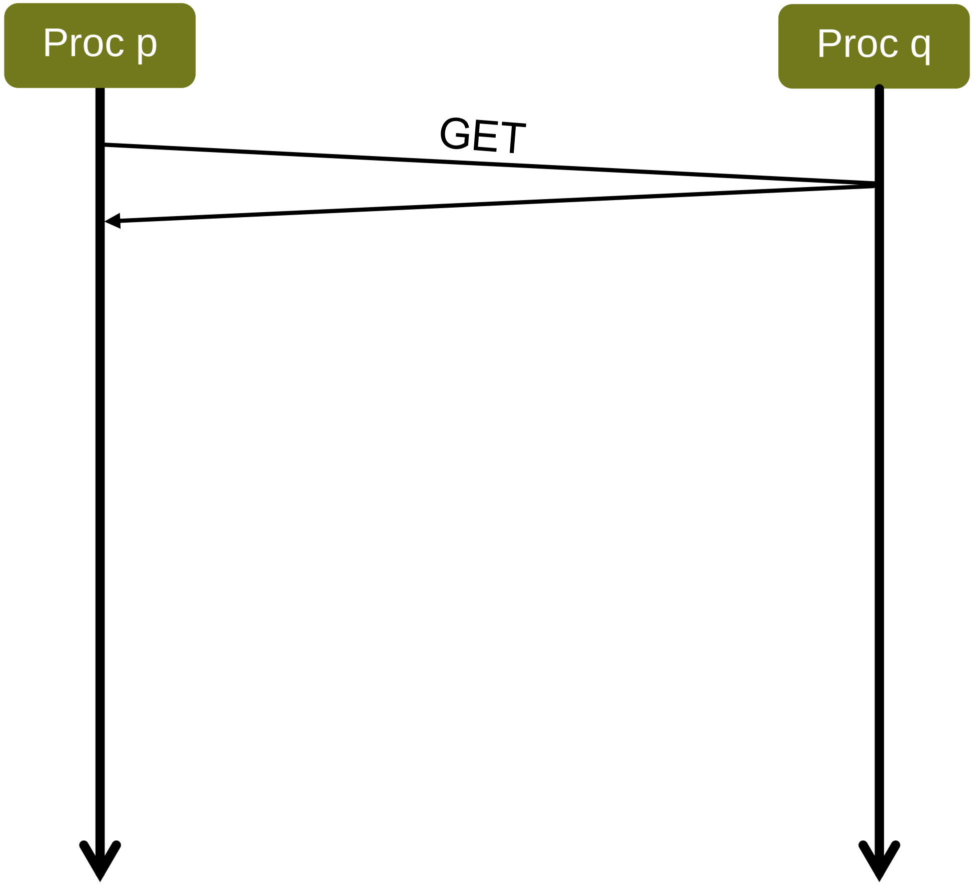
- Logging gets (naive):



ACTIVE ACCESS USE-CASES

ACCELERATING LOGGING FOR RMA

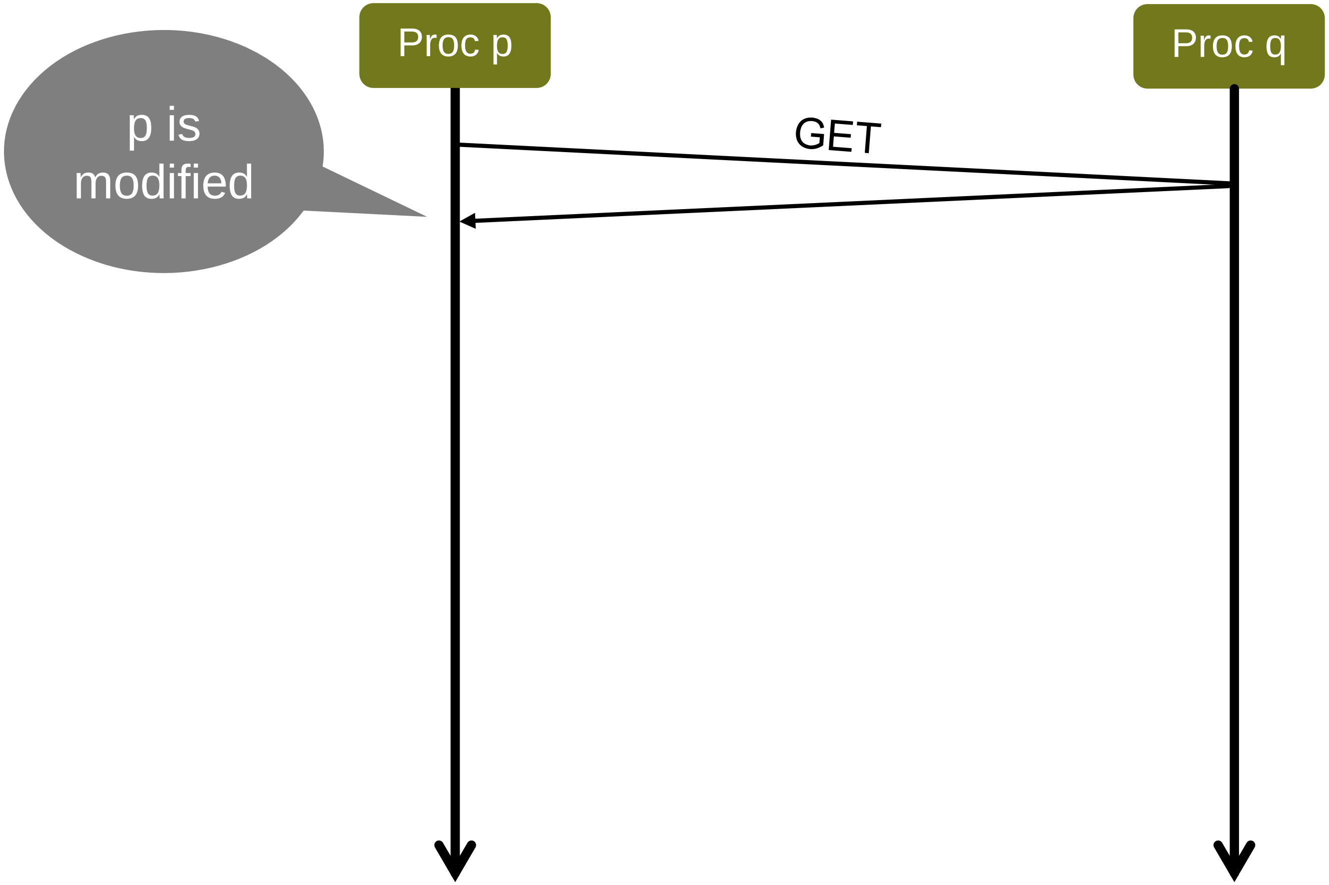
- Logging gets (naive):



Active Access Use-Cases

Accelerating logging for RMA

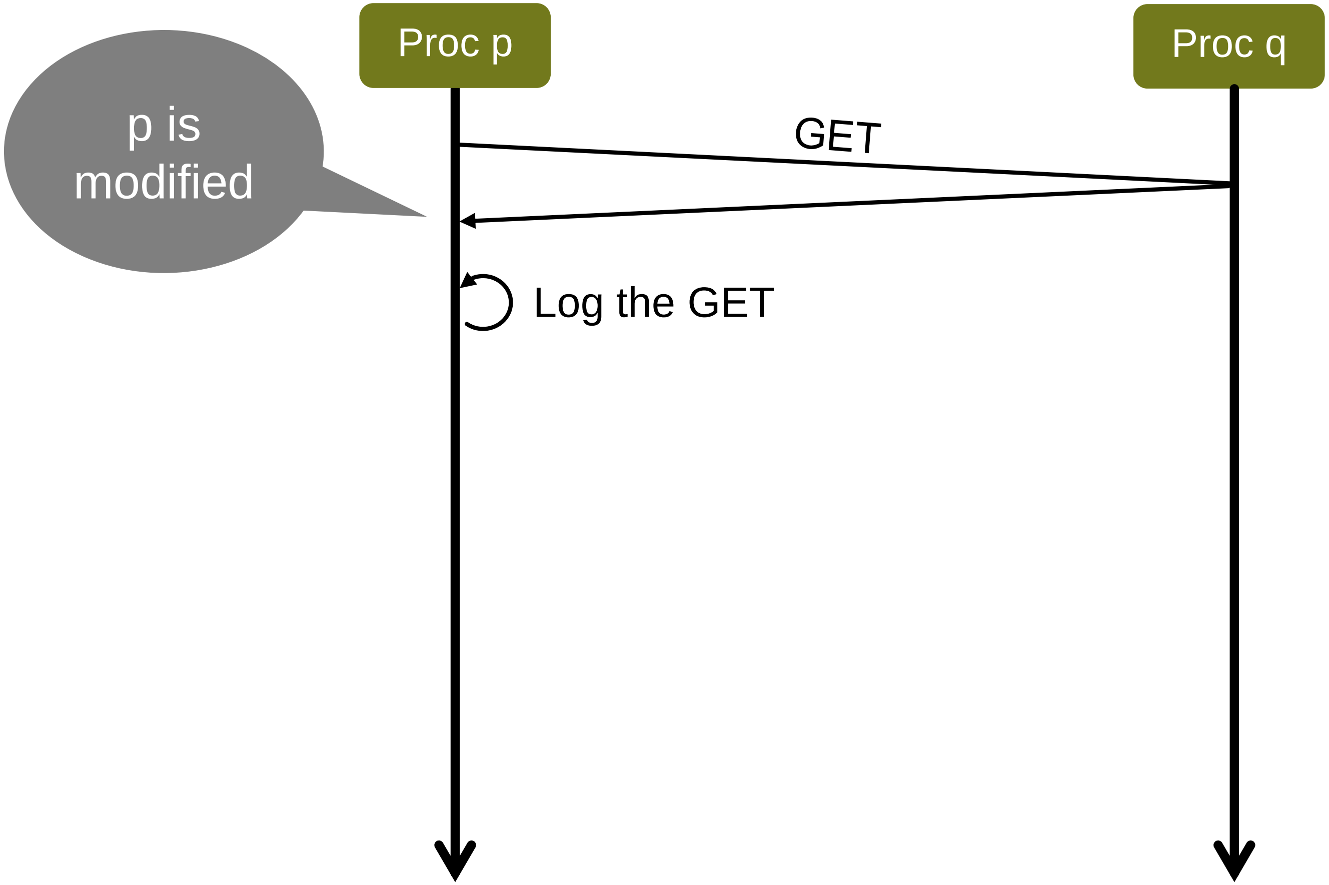
- Logging gets (naive):



Active Access Use-Cases

Accelerating logging for RMA

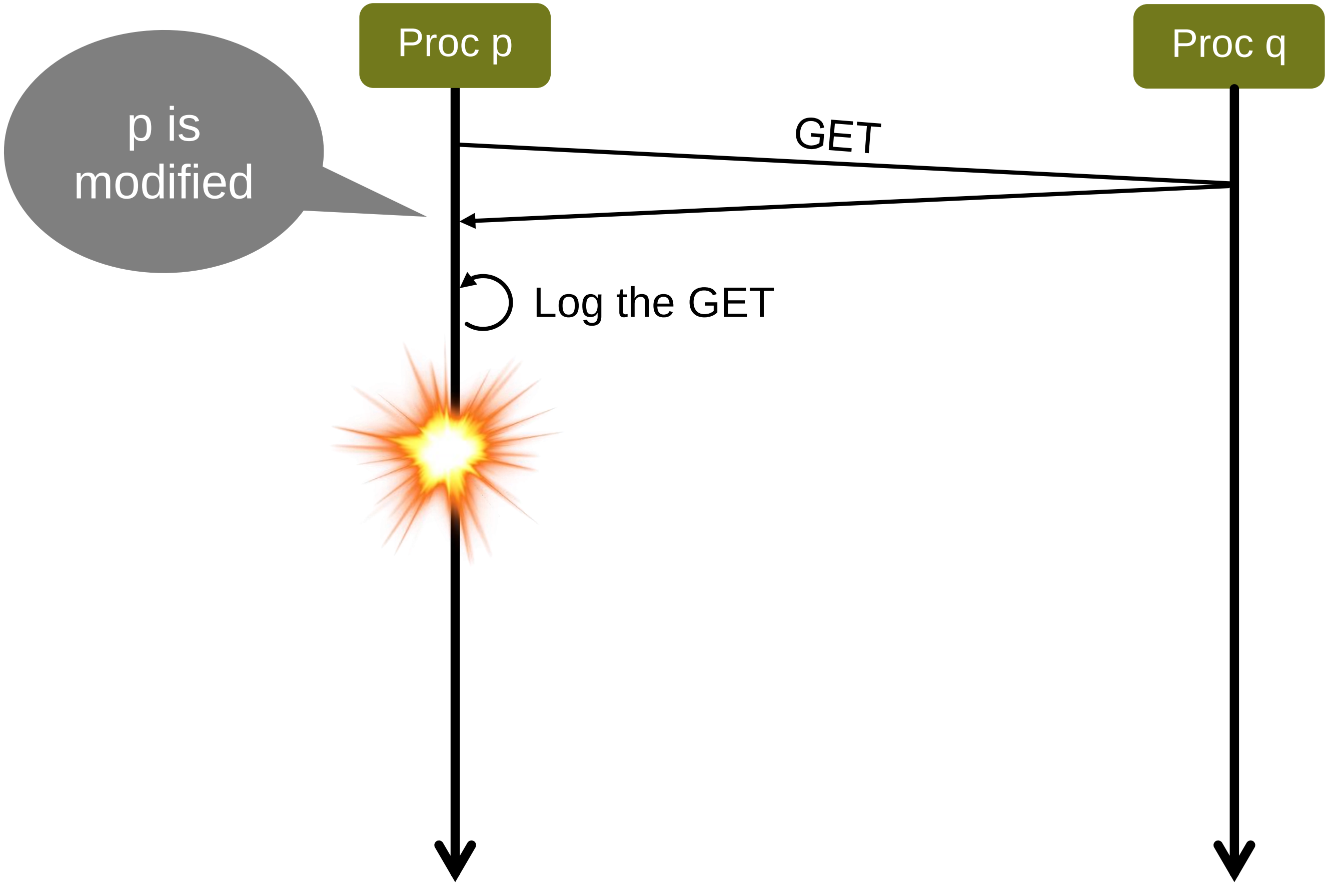
- Logging gets (naive):



ACTIVE ACCESS USE-CASES

ACCELERATING LOGGING FOR RMA

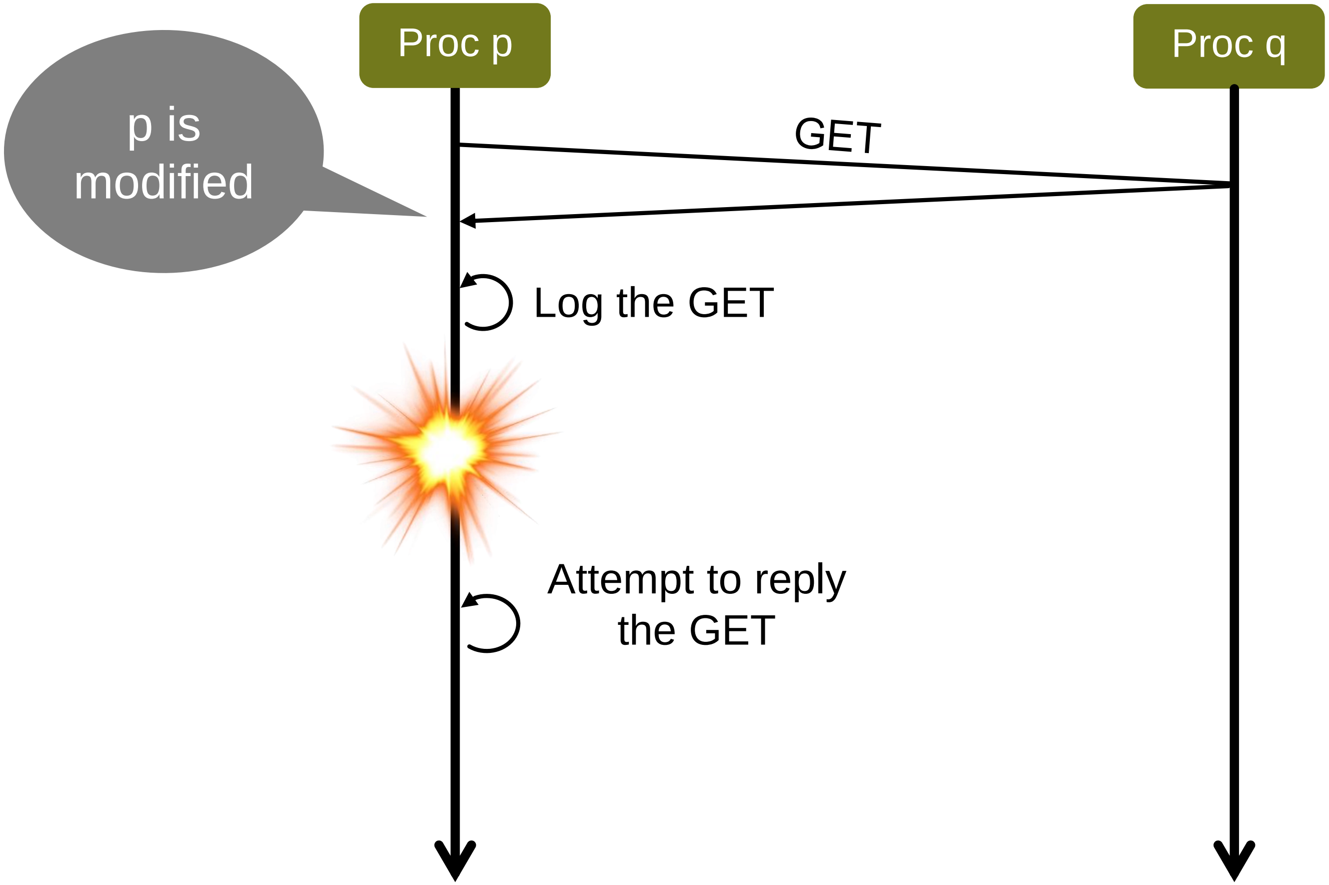
- Logging gets (naive):



Active Access Use-Cases

Accelerating logging for RMA

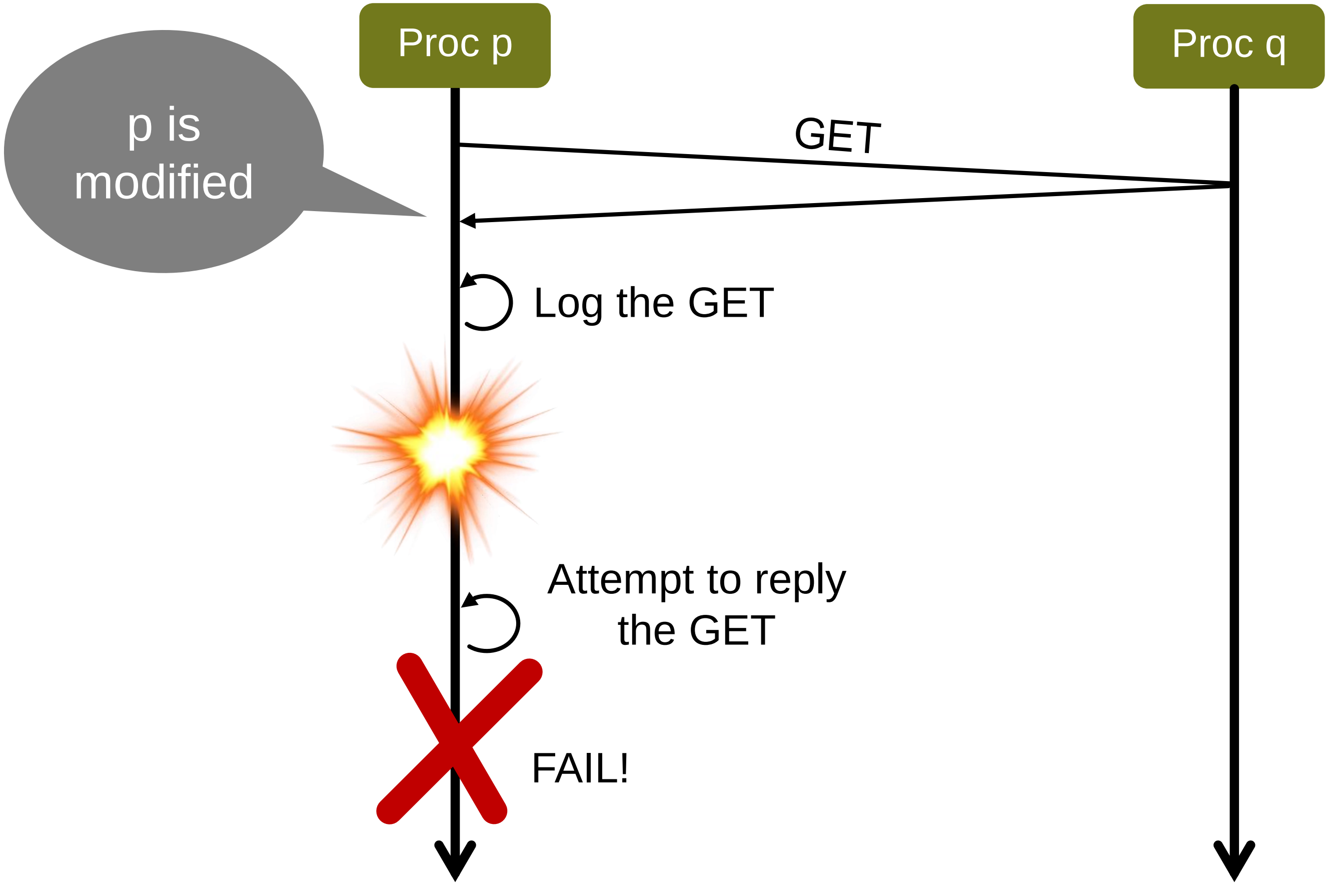
- Logging gets (naive):



ACTIVE ACCESS USE-CASES

ACCELERATING LOGGING FOR RMA

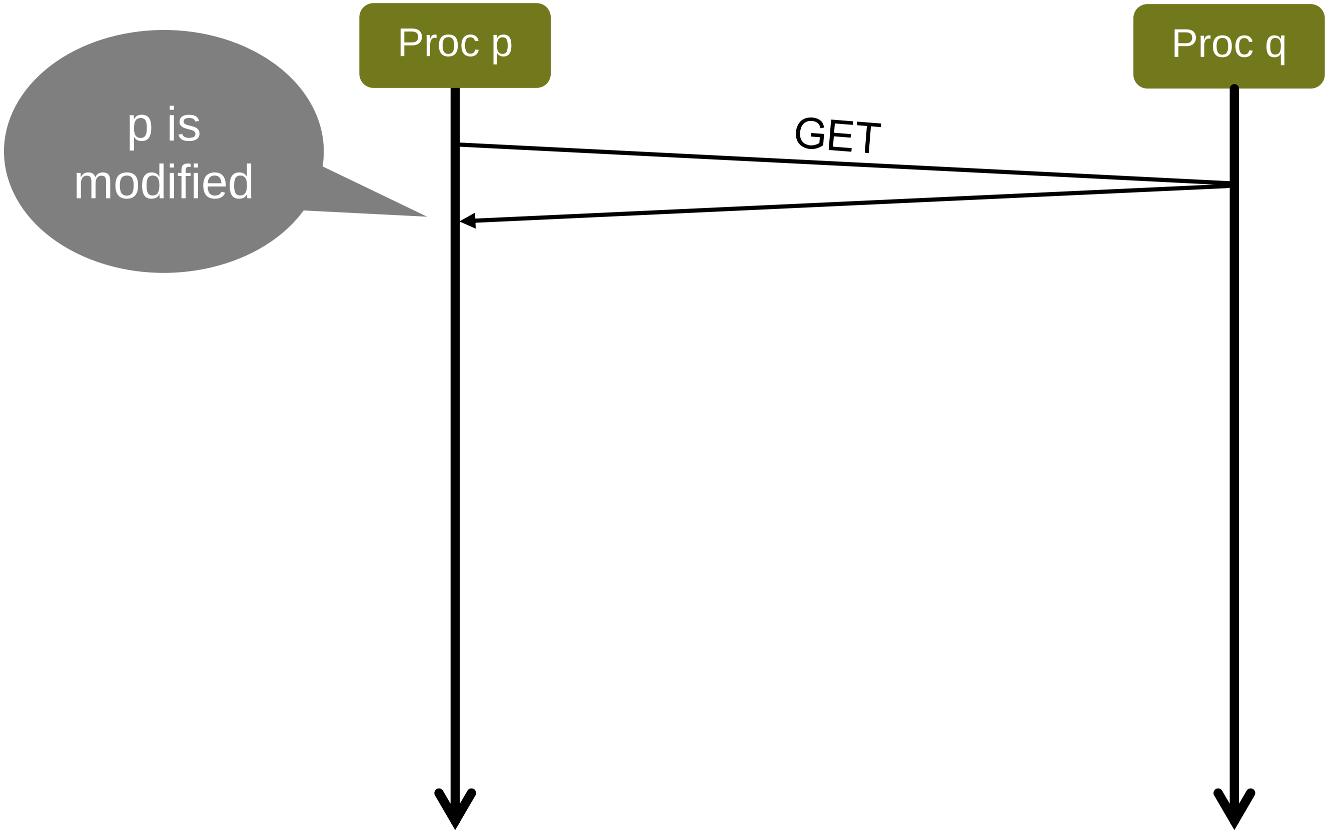
- Logging gets (naive):



ACTIVE ACCESS USE-CASES

ACCELERATING LOGGING FOR RMA

- Logging gets (traditional) [1]:

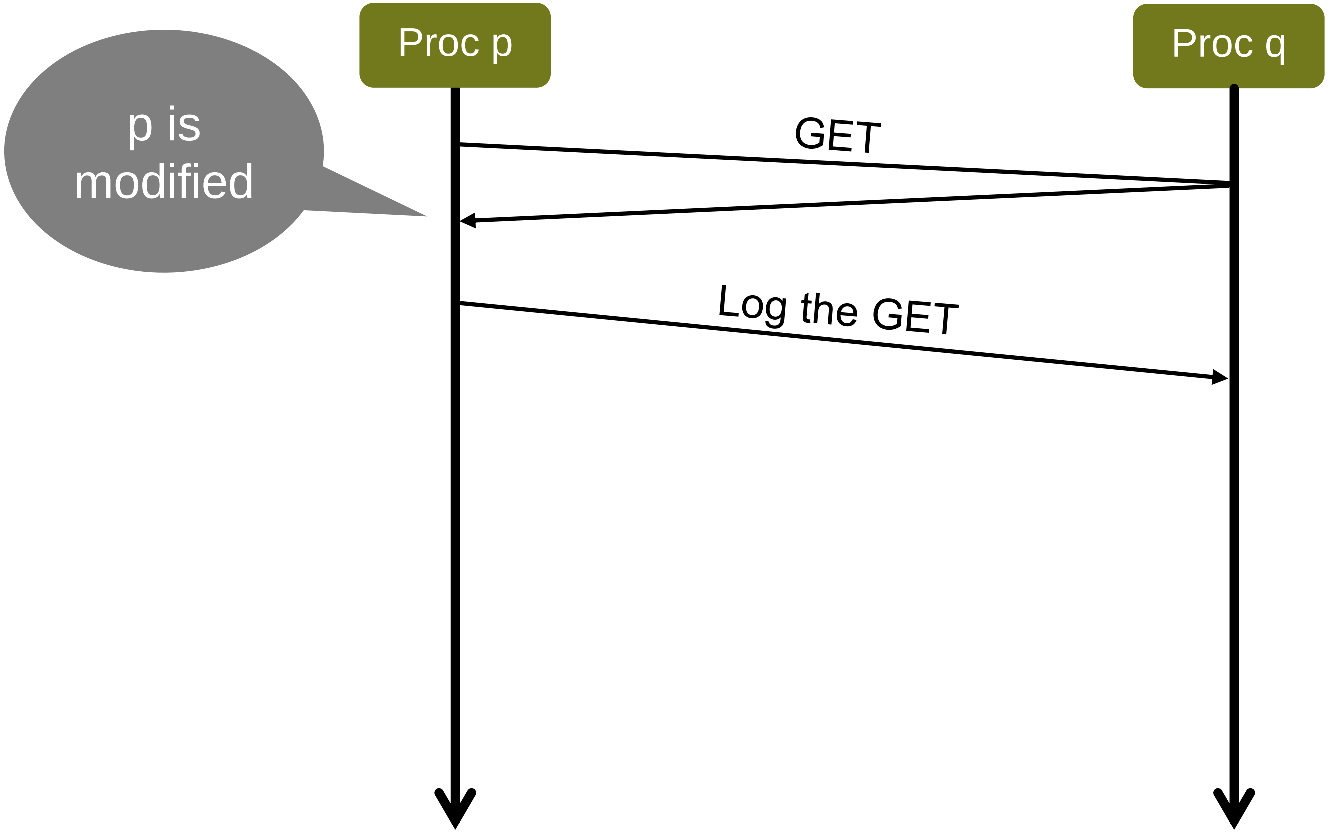


[1] M. Besta and T. Hoefler. Fault tolerance for remote memory access programming models. HPDC'14.

Active Access Use-Cases

Accelerating logging for RMA

- Logging gets (traditional) [1]:

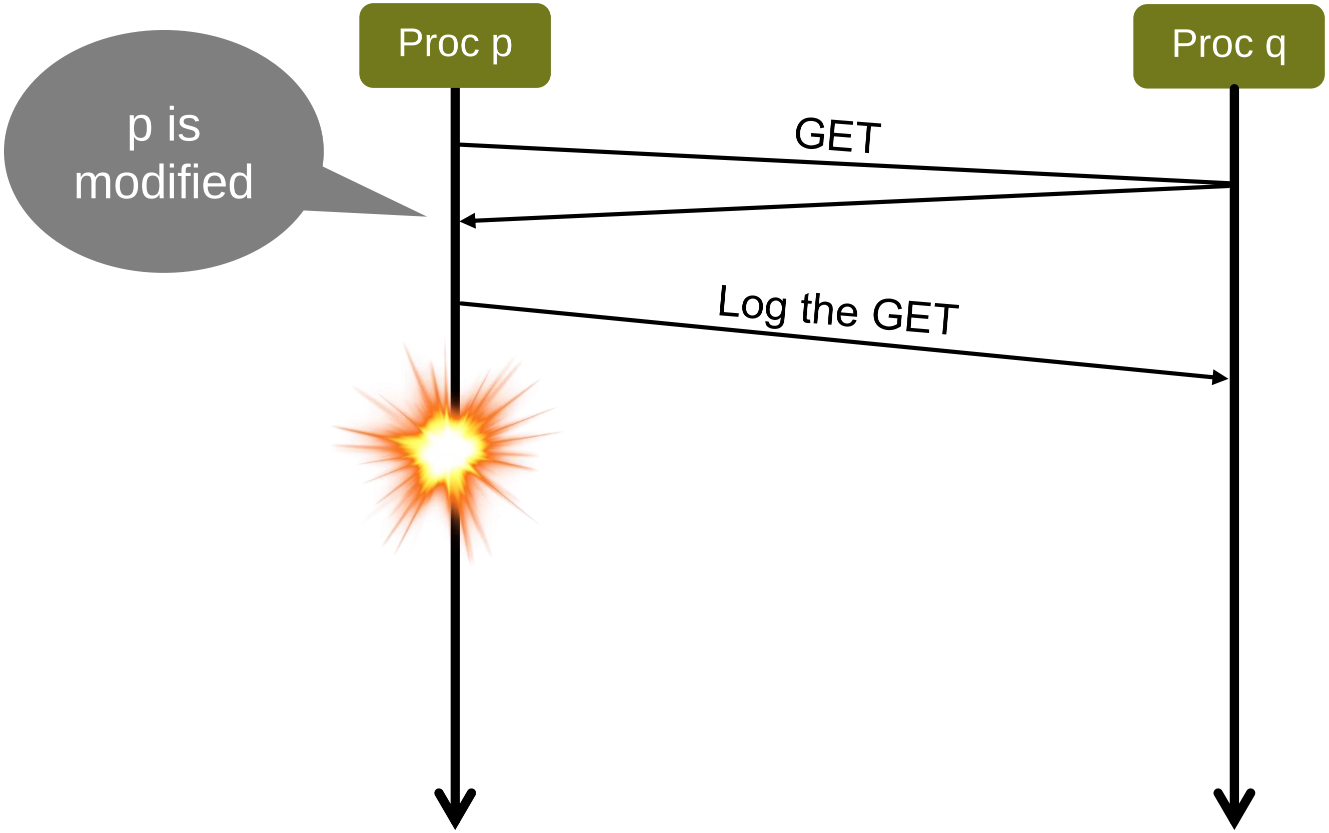


[1] M. Besta and T. Hoefler. Fault tolerance for remote memory access programming models. HPDC'14.

Active Access Use-Cases

Accelerating logging for RMA

- Logging gets (traditional) [1]:

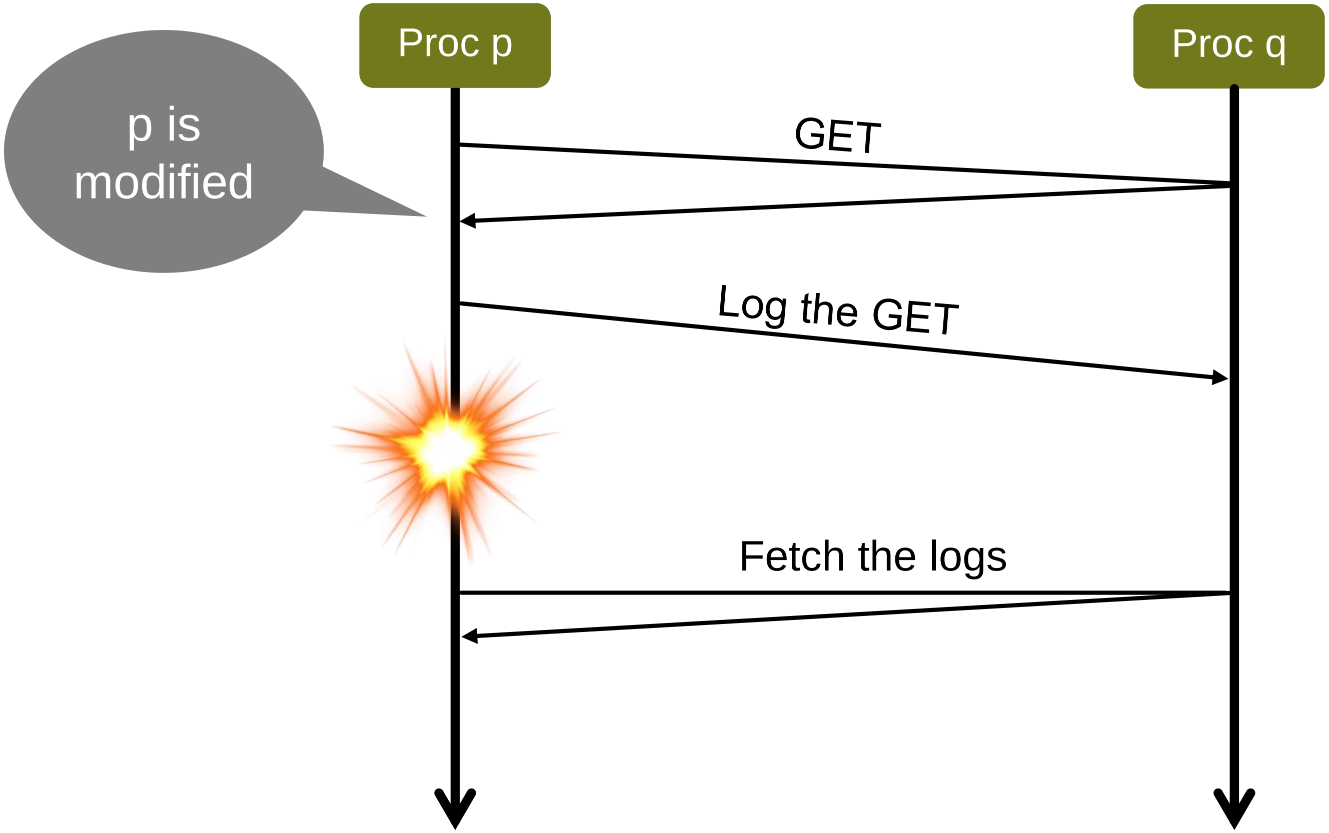


[1] M. Besta and T. Hoefler. Fault tolerance for remote memory access programming models. HPDC'14.

Active Access Use-Cases

Accelerating logging for RMA

- Logging gets (traditional) [1]:

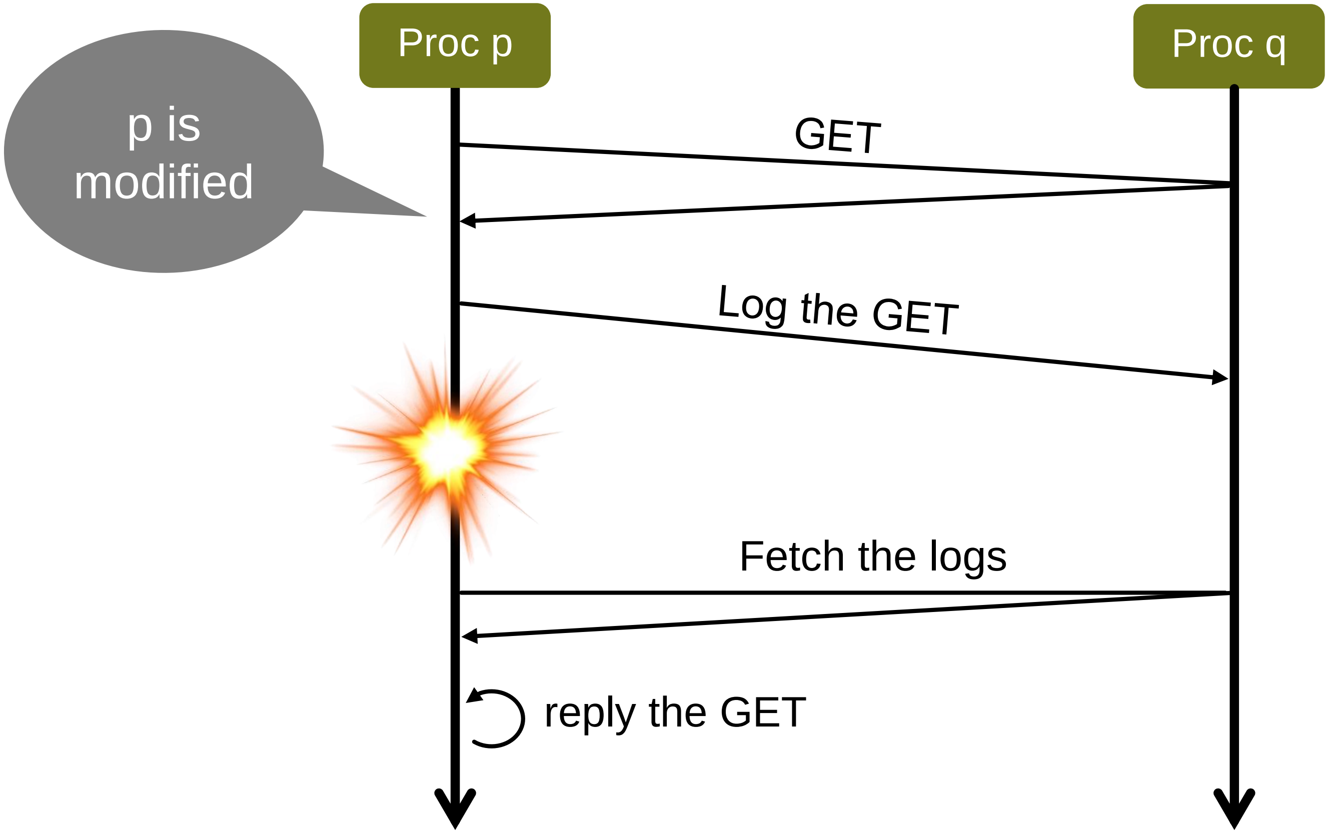


[1] M. Besta and T. Hoefler. Fault tolerance for remote memory access programming models. HPDC'14.

Active Access Use-Cases

Accelerating logging for RMA

- Logging gets (traditional) [1]:

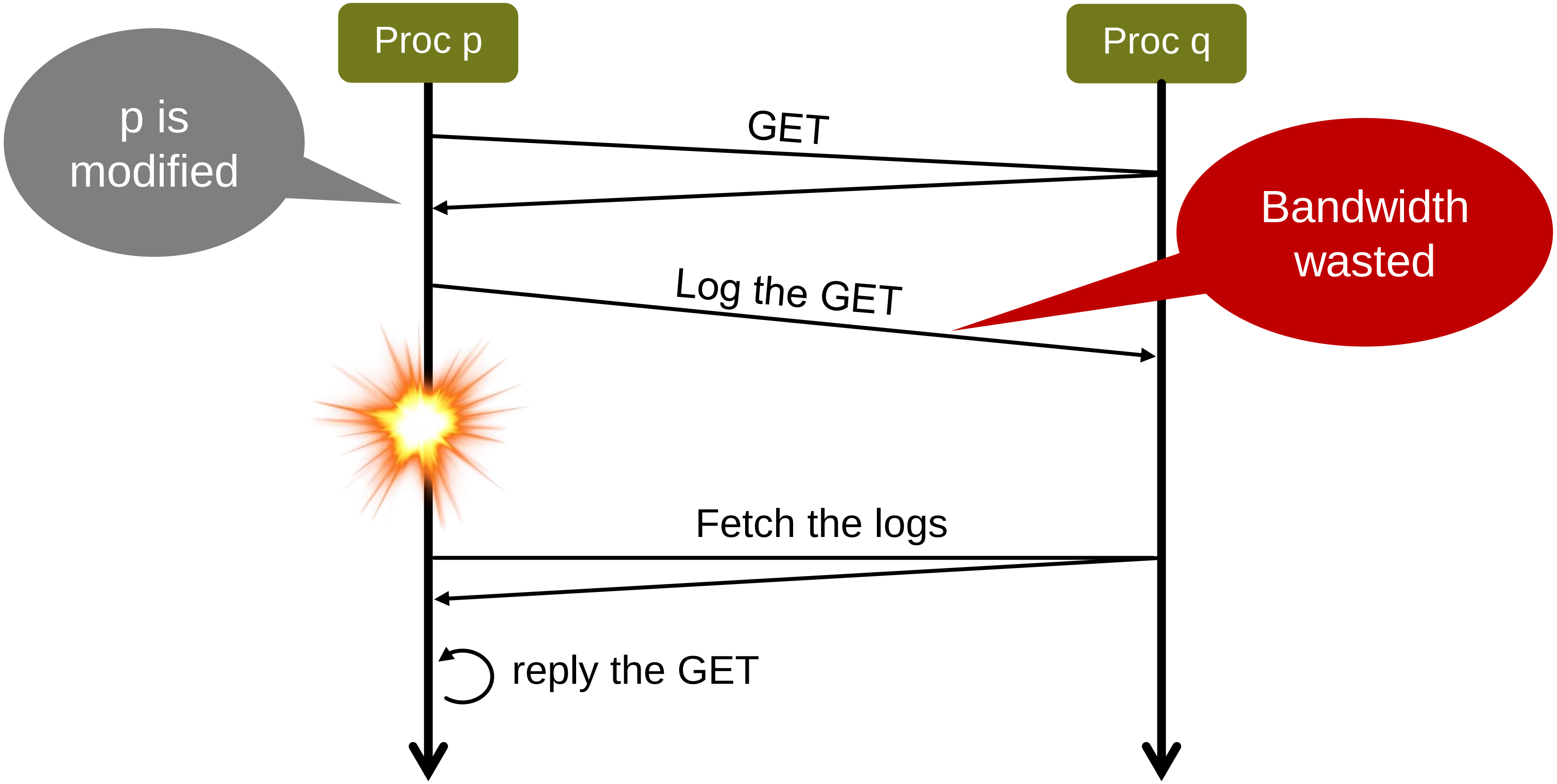


[1] M. Besta and T. Hoefler. Fault tolerance for remote memory access programming models. HPDC'14.

ACTIVE ACCESS USE-CASES

ACCELERATING LOGGING FOR RMA

- Logging gets (traditional) [1]:

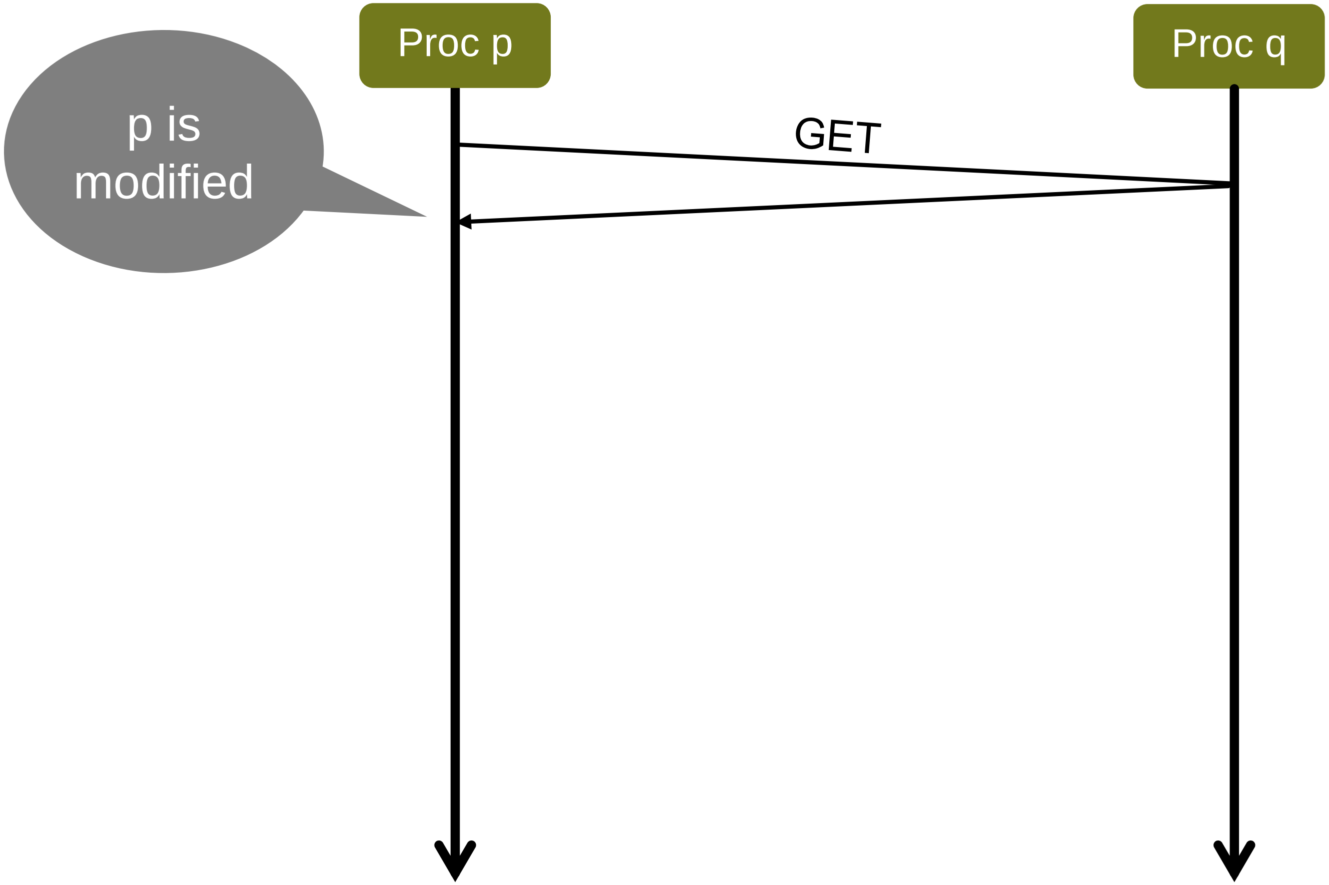


[1] M. Besta and T. Hoefler. Fault tolerance for remote memory access programming models. HPDC'14.

Active Access Use-Cases

Accelerating logging for RMA

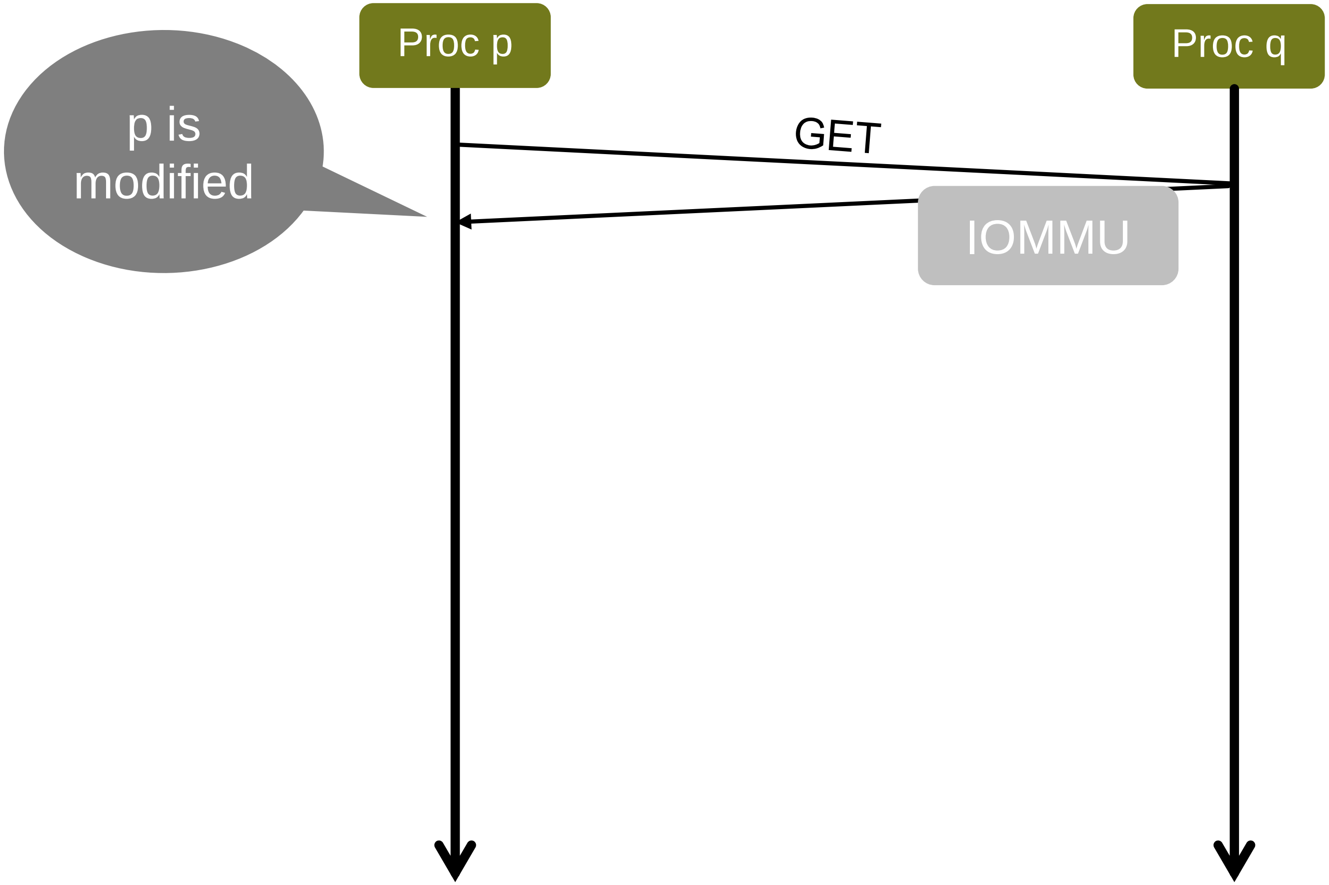
- Logging gets (AA):



Active Access Use-Cases

Accelerating logging for RMA

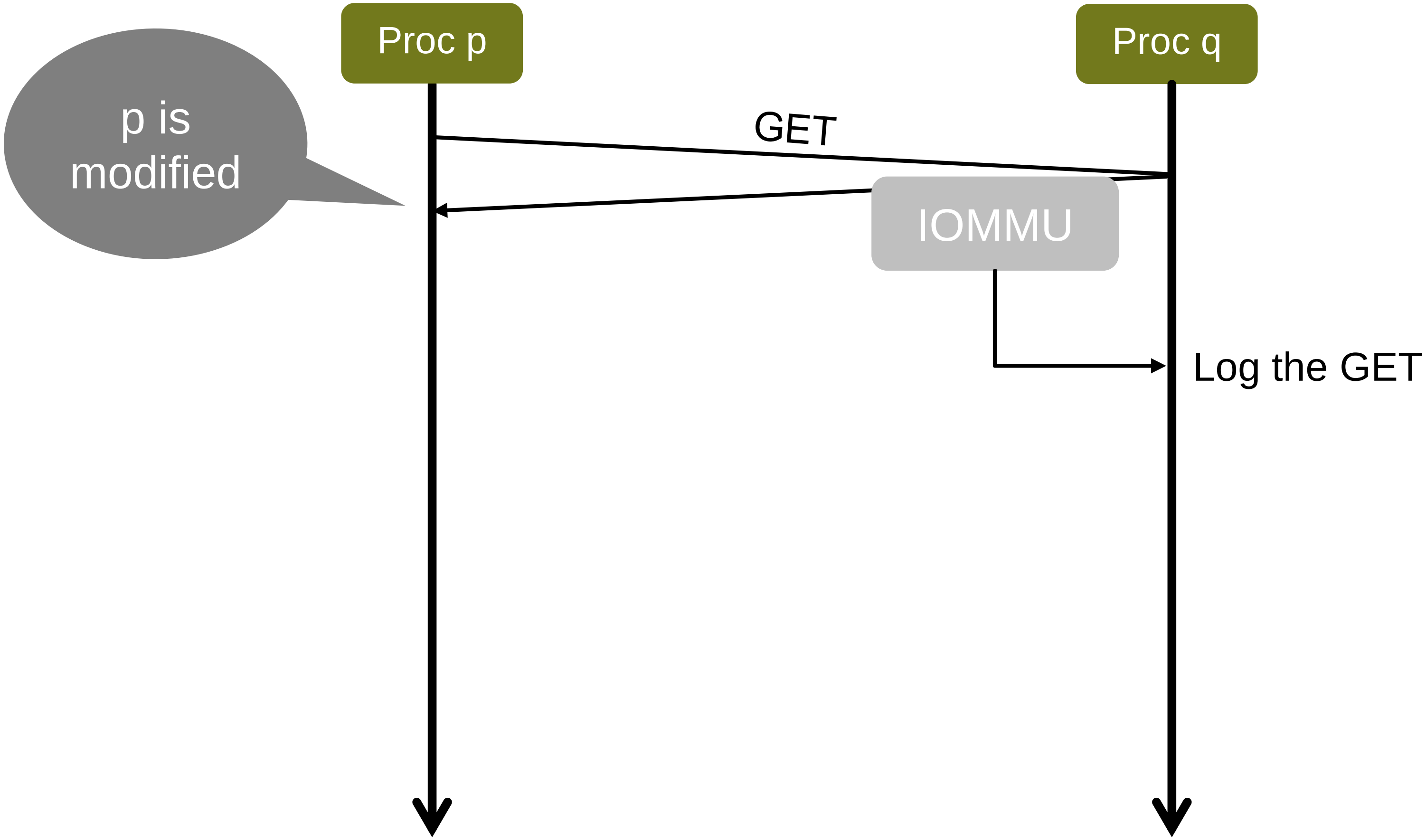
- Logging gets (AA):



Active Access Use-Cases

Accelerating logging for RMA

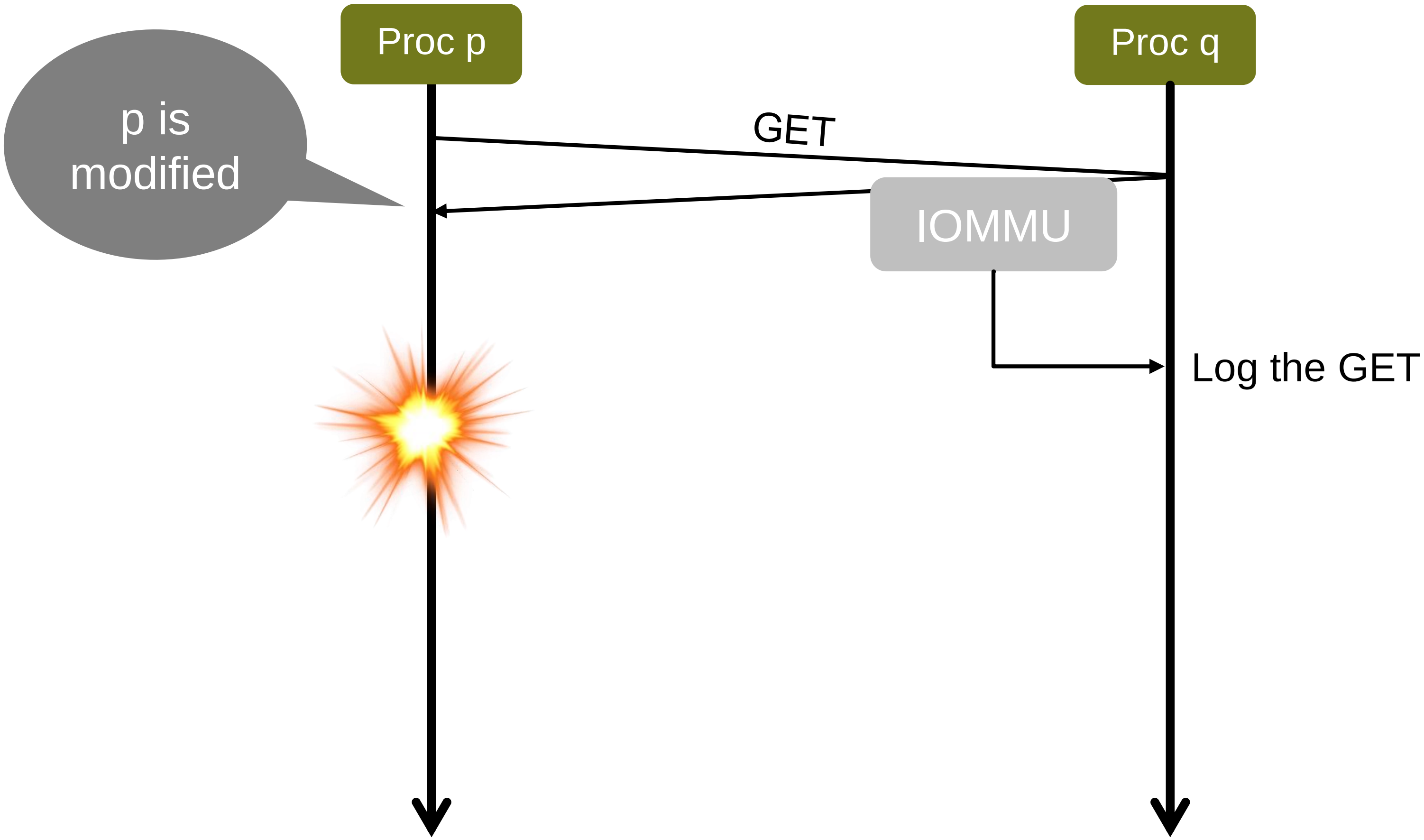
- Logging gets (AA):



ACTIVE ACCESS USE-CASES

ACCELERATING LOGGING FOR RMA

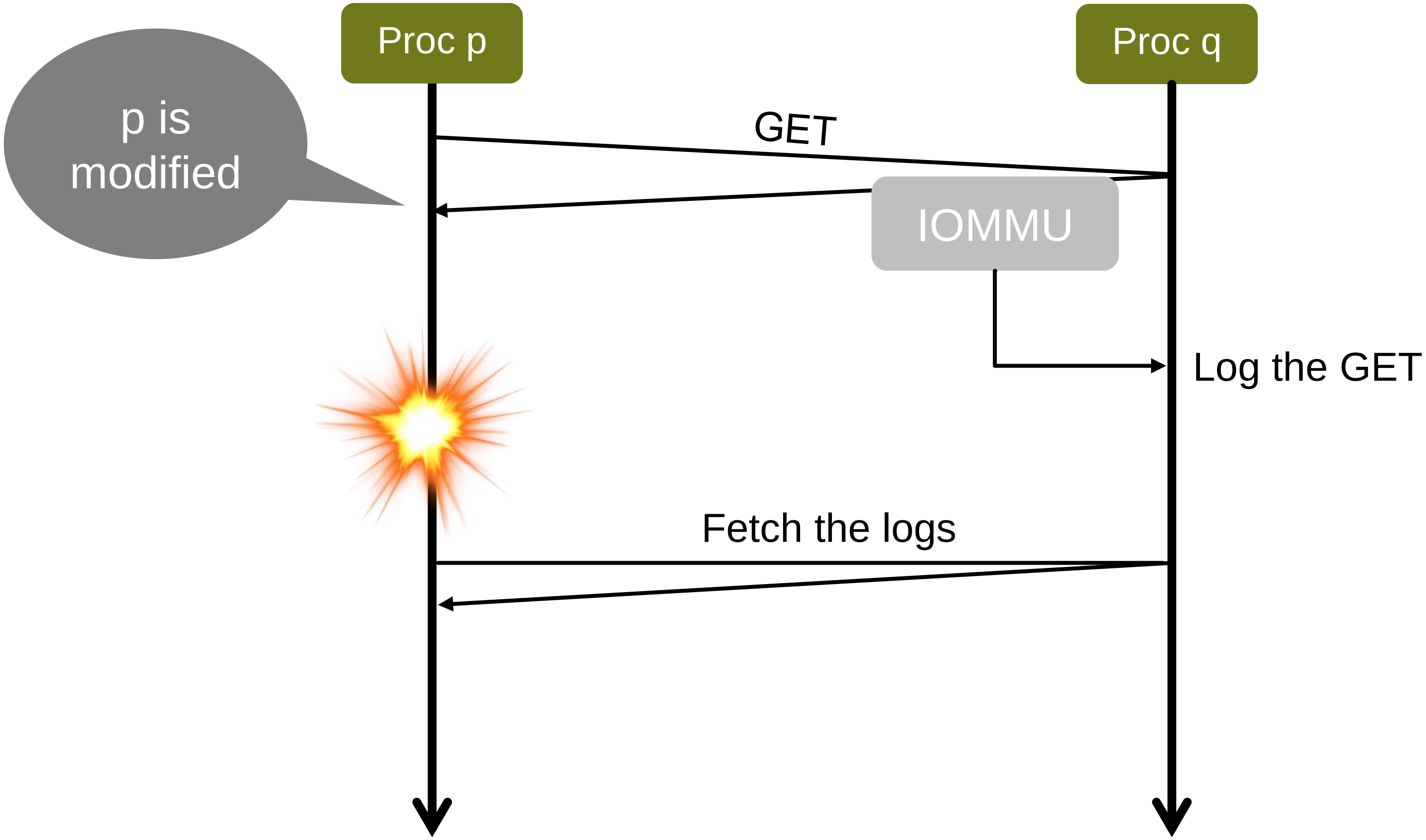
- Logging gets (AA):



Active Access Use-Cases

Accelerating logging for RMA

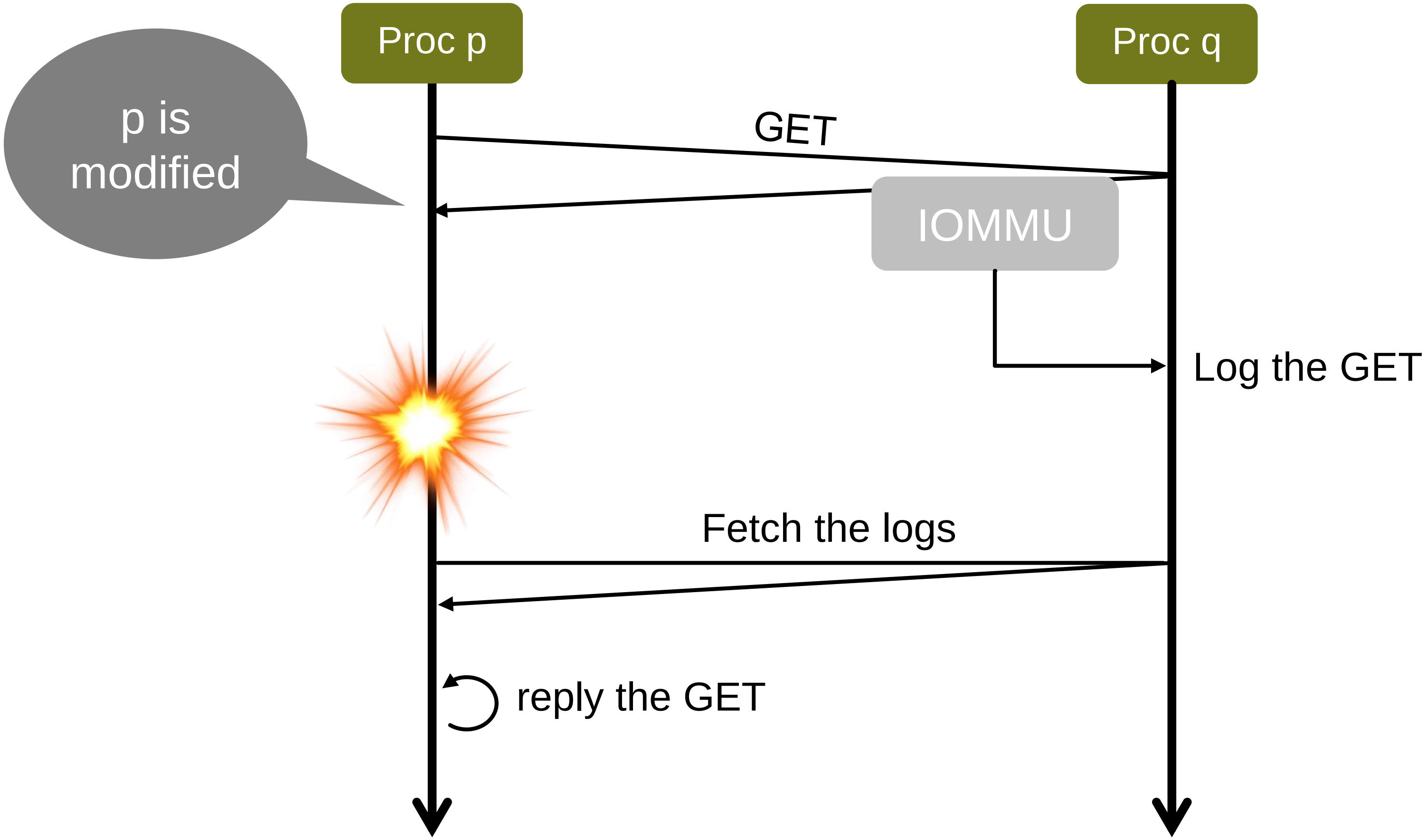
- Logging gets (AA):



Active Access Use-Cases

Accelerating logging for RMA

- Logging gets (AA):

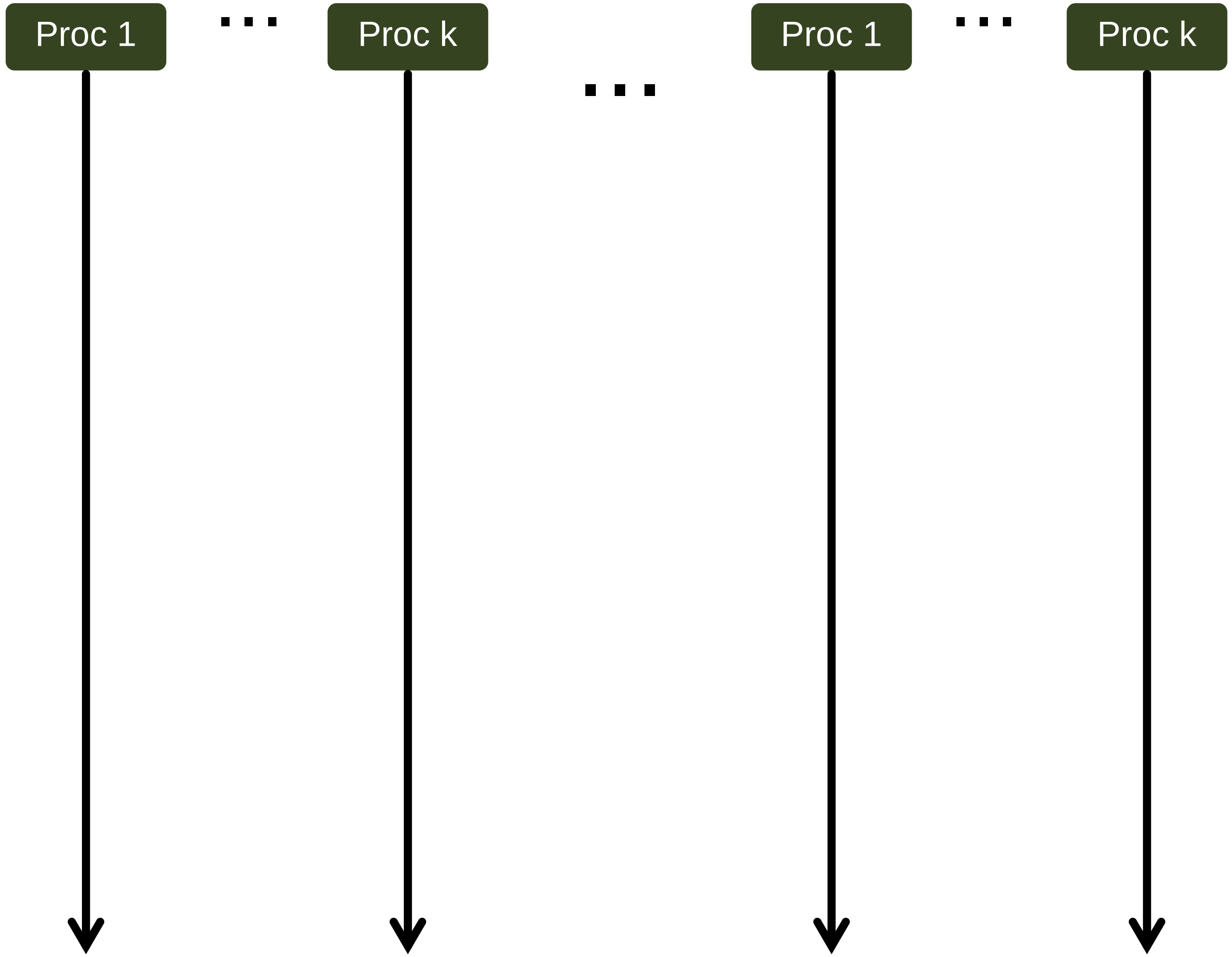


Active Access Use-Cases

Incremental checkpointing for RMA

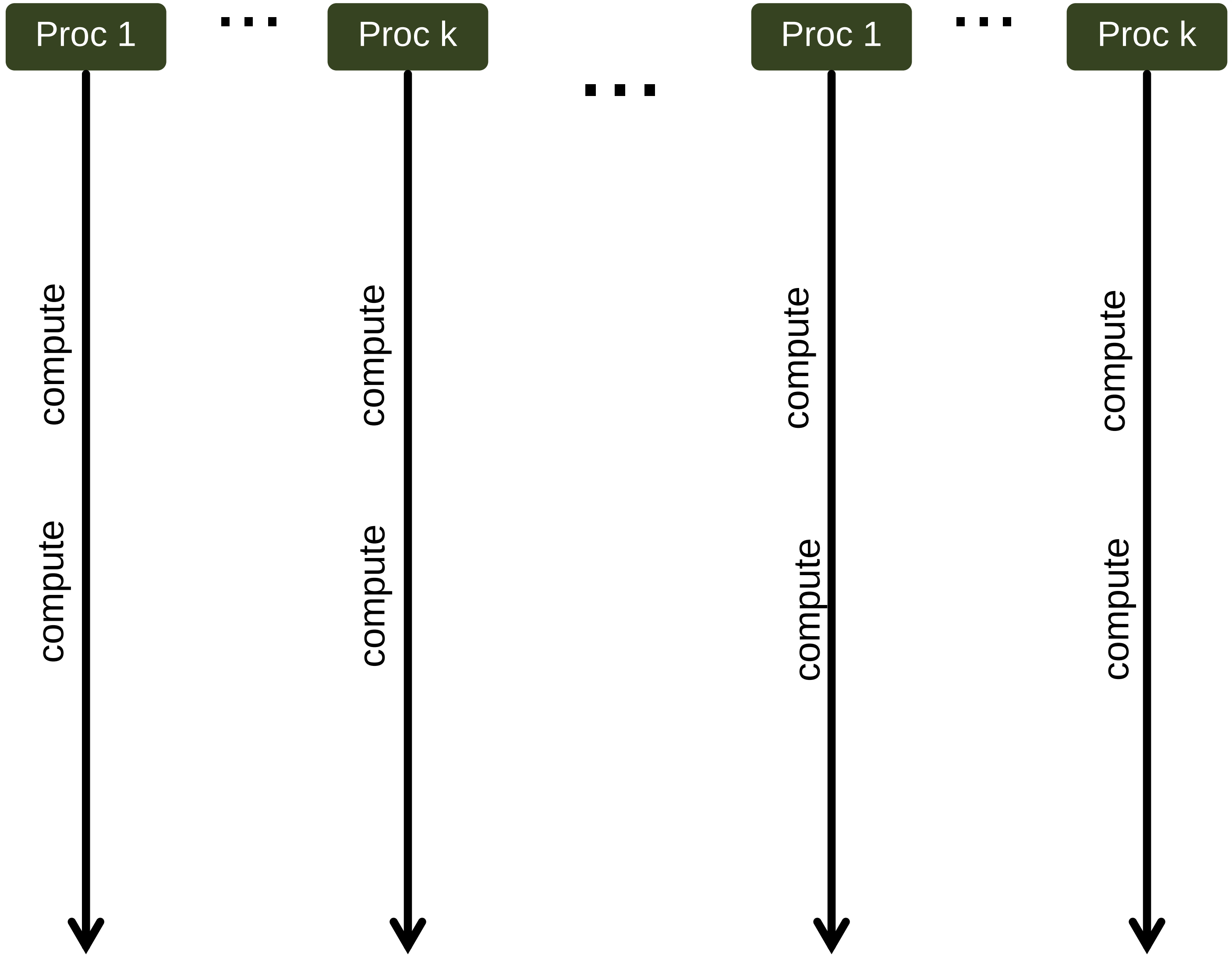
Active Access Use-Cases

Incremental checkpointing for RMA



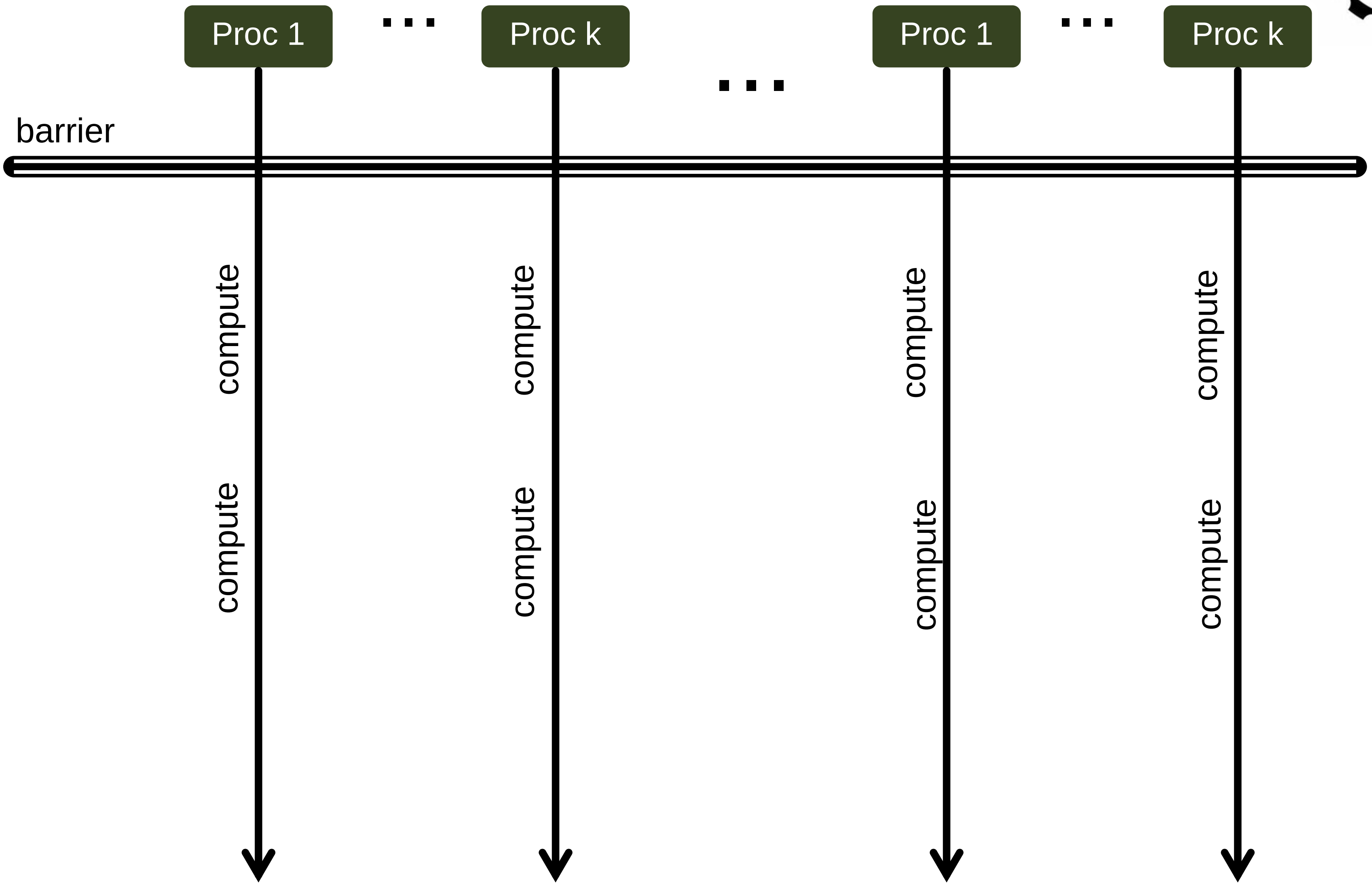
Active Access Use-Cases

Incremental checkpointing for RMA



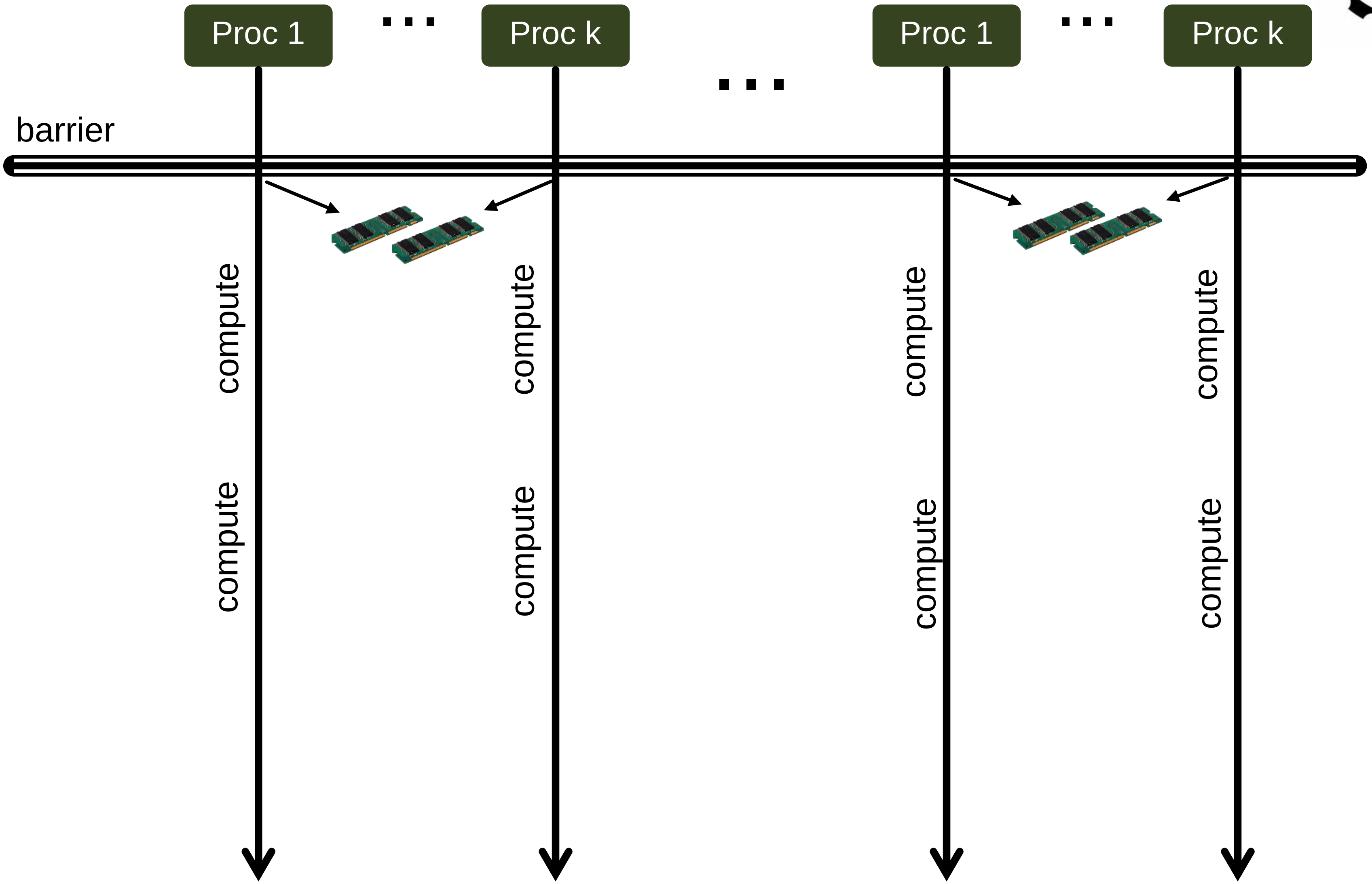
Active Access Use-Cases

Incremental checkpointing for RMA



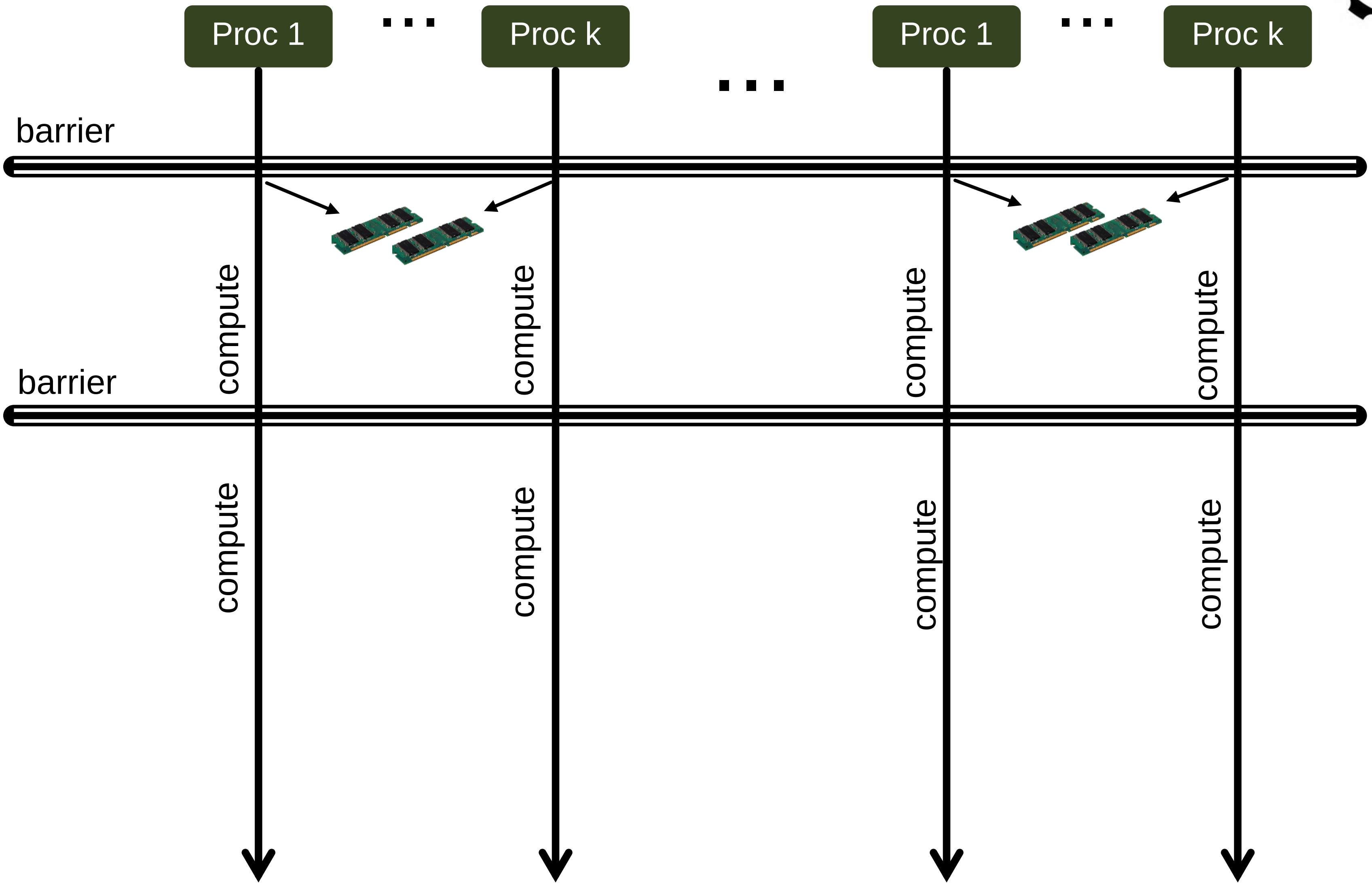
Active Access Use-Cases

Incremental checkpointing for RMA



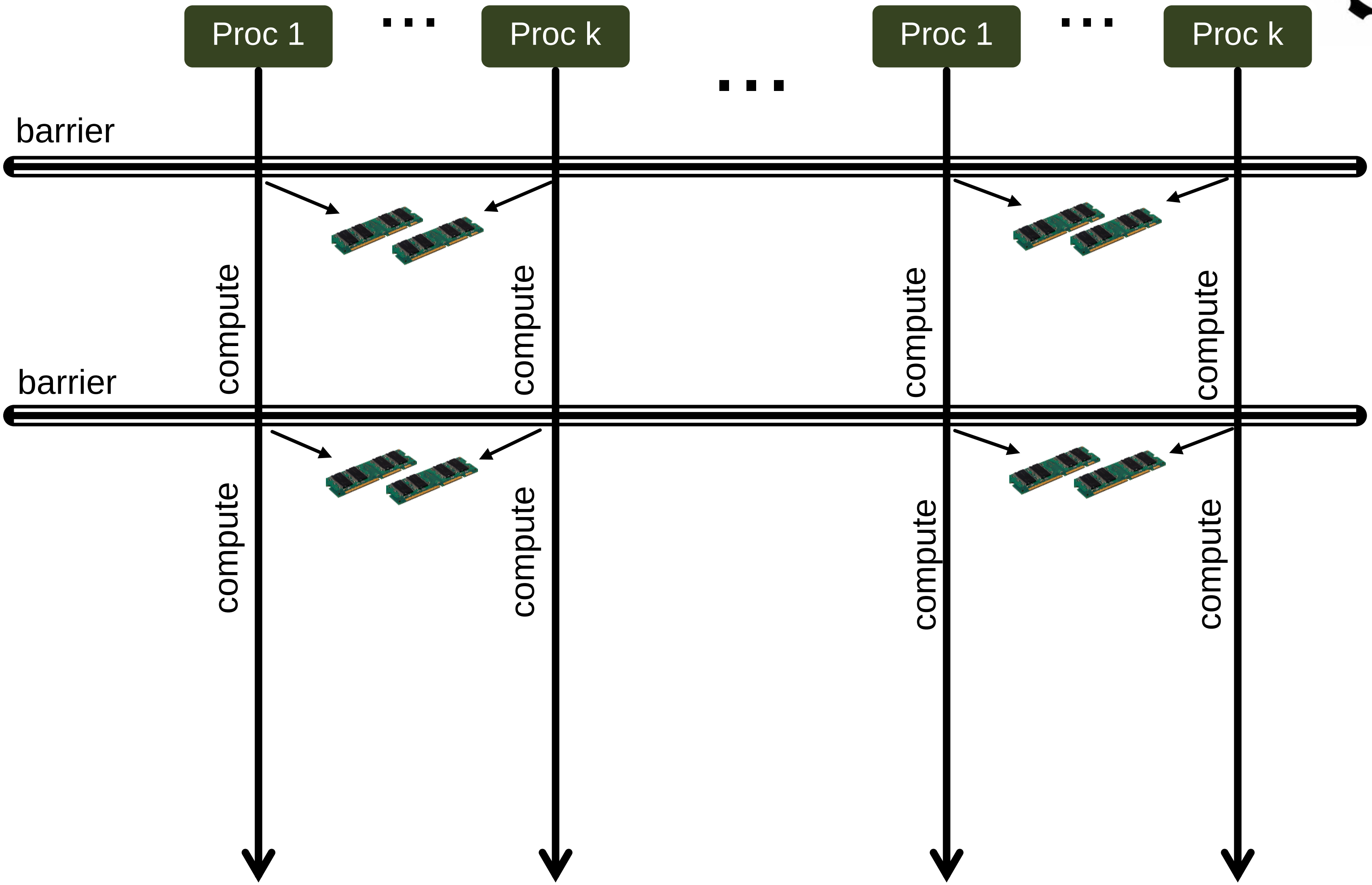
Active Access Use-Cases

Incremental checkpointing for RMA



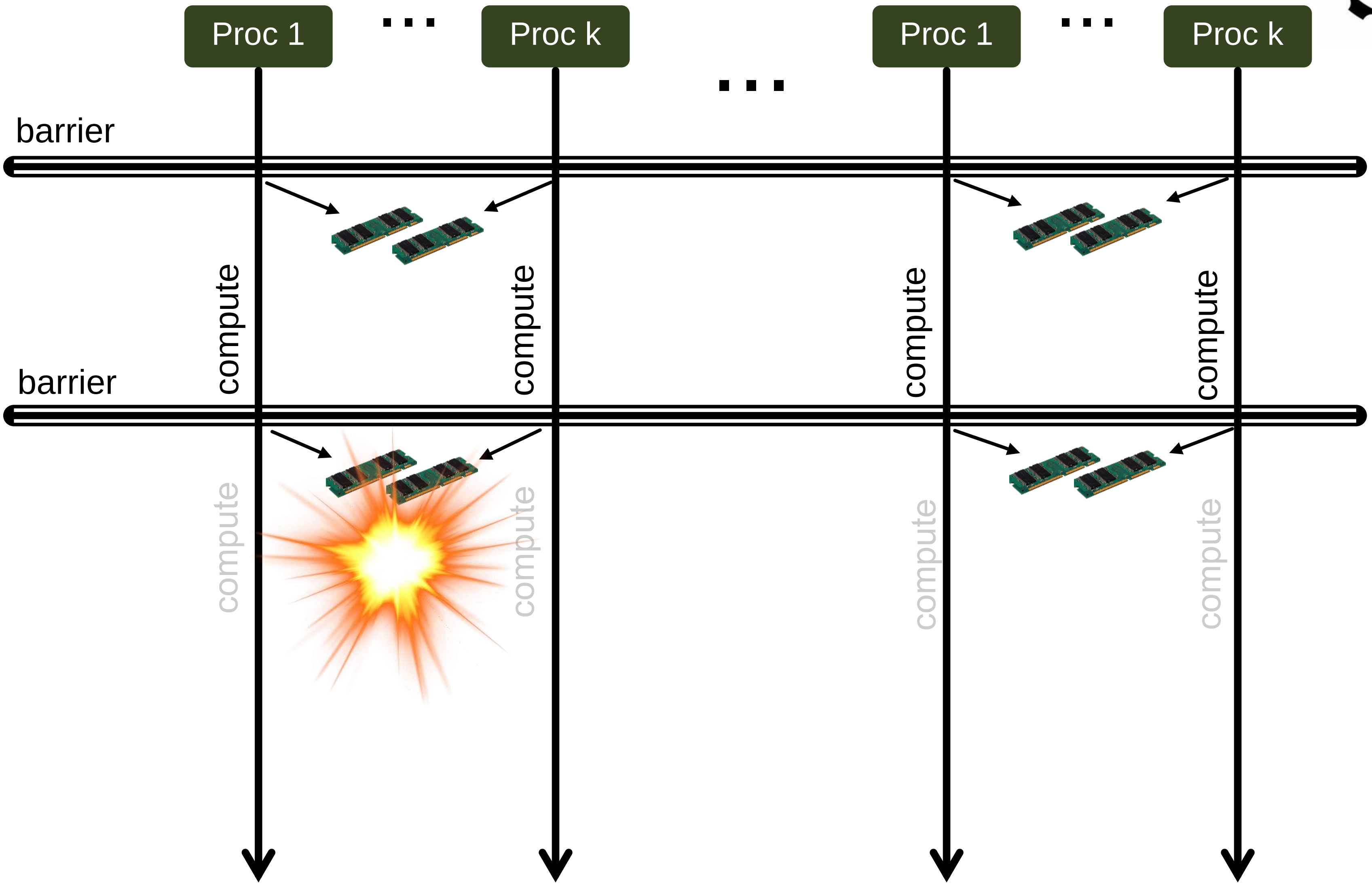
ACTIVE ACCESS USE-CASES

INCREMENTAL CHECKPOINTING FOR RMA



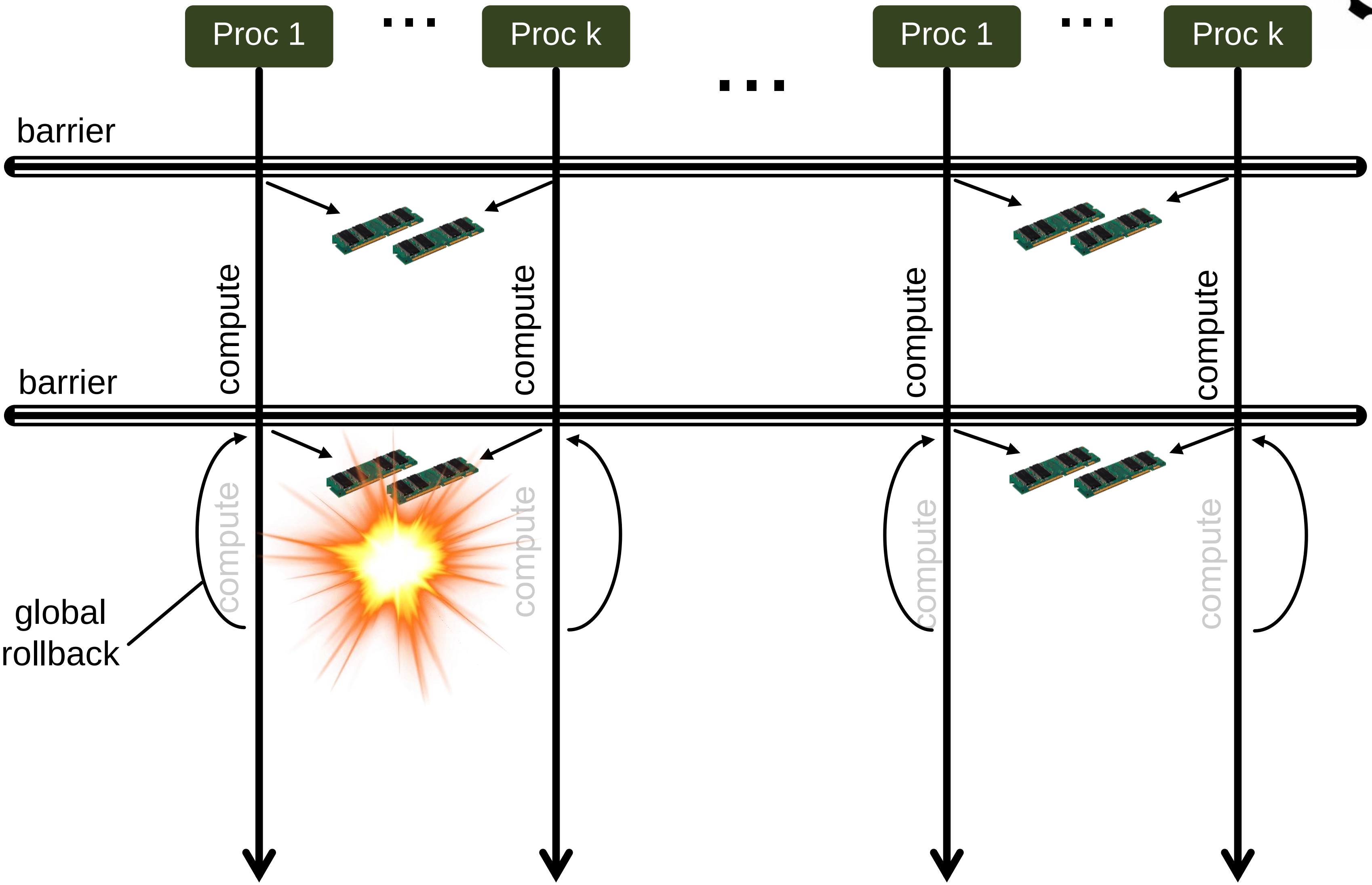
Active Access Use-Cases

Incremental checkpointing for RMA



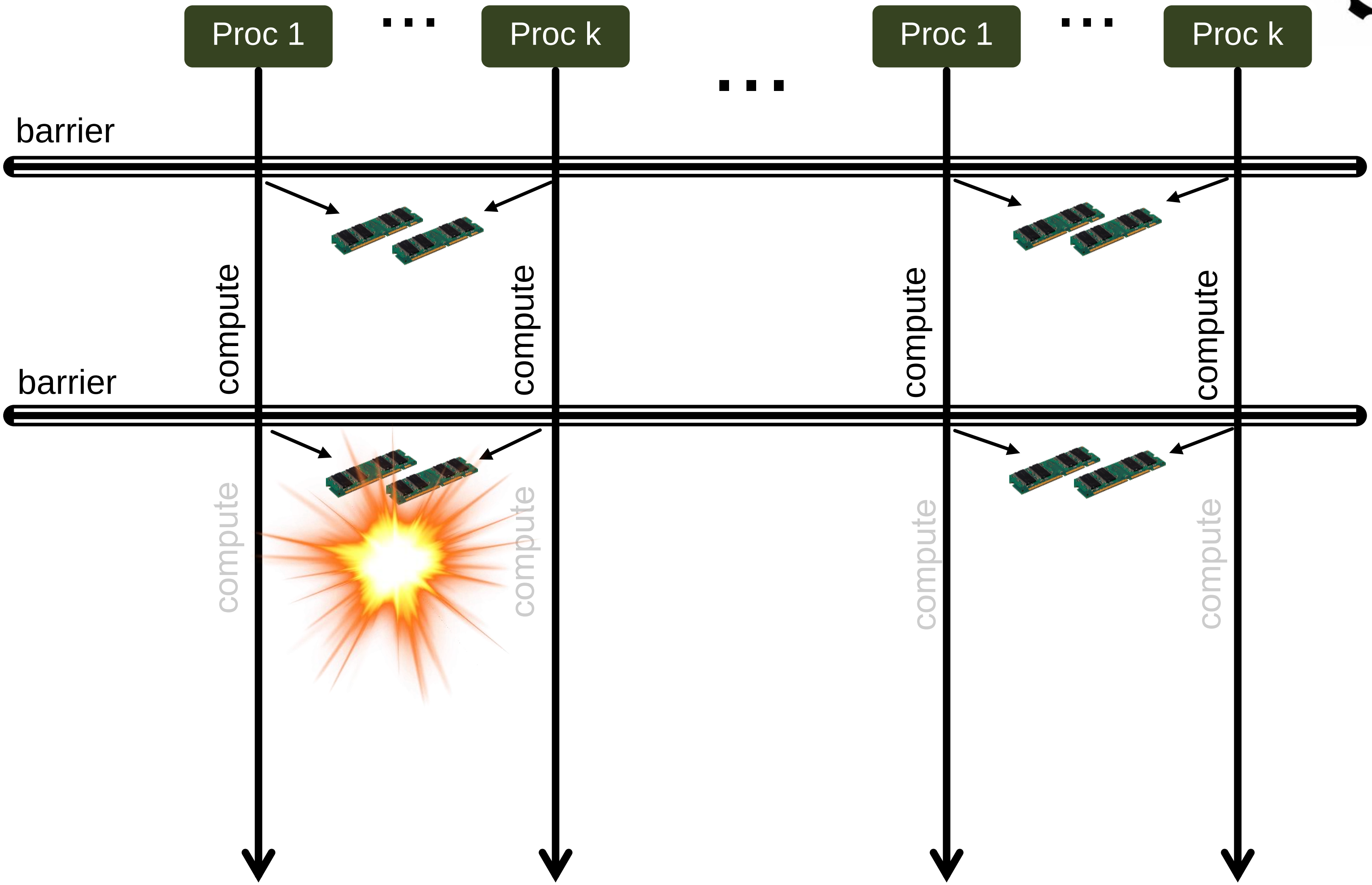
Active Access Use-Cases

Incremental checkpointing for RMA



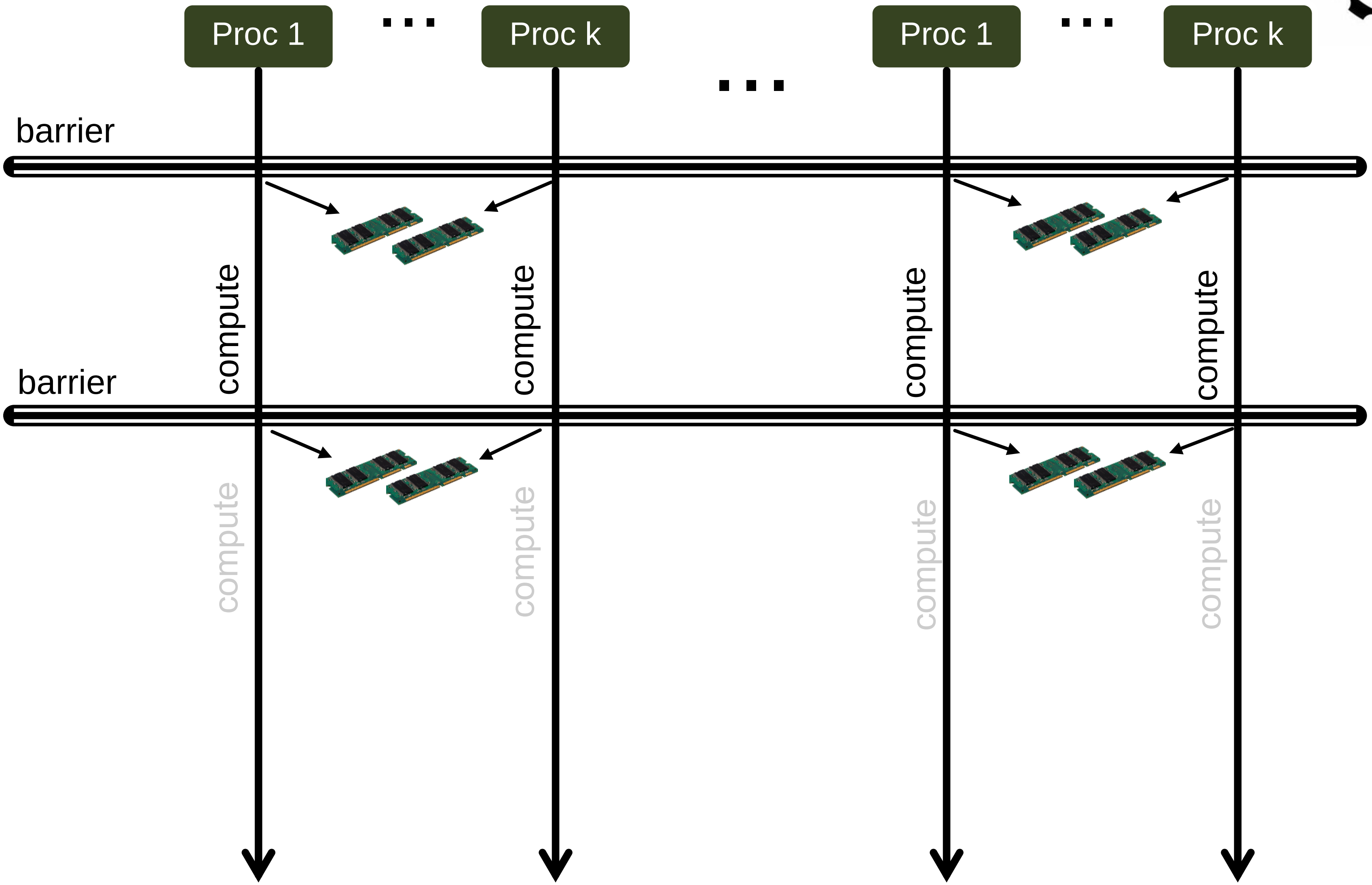
Active Access Use-Cases

Incremental checkpointing for RMA



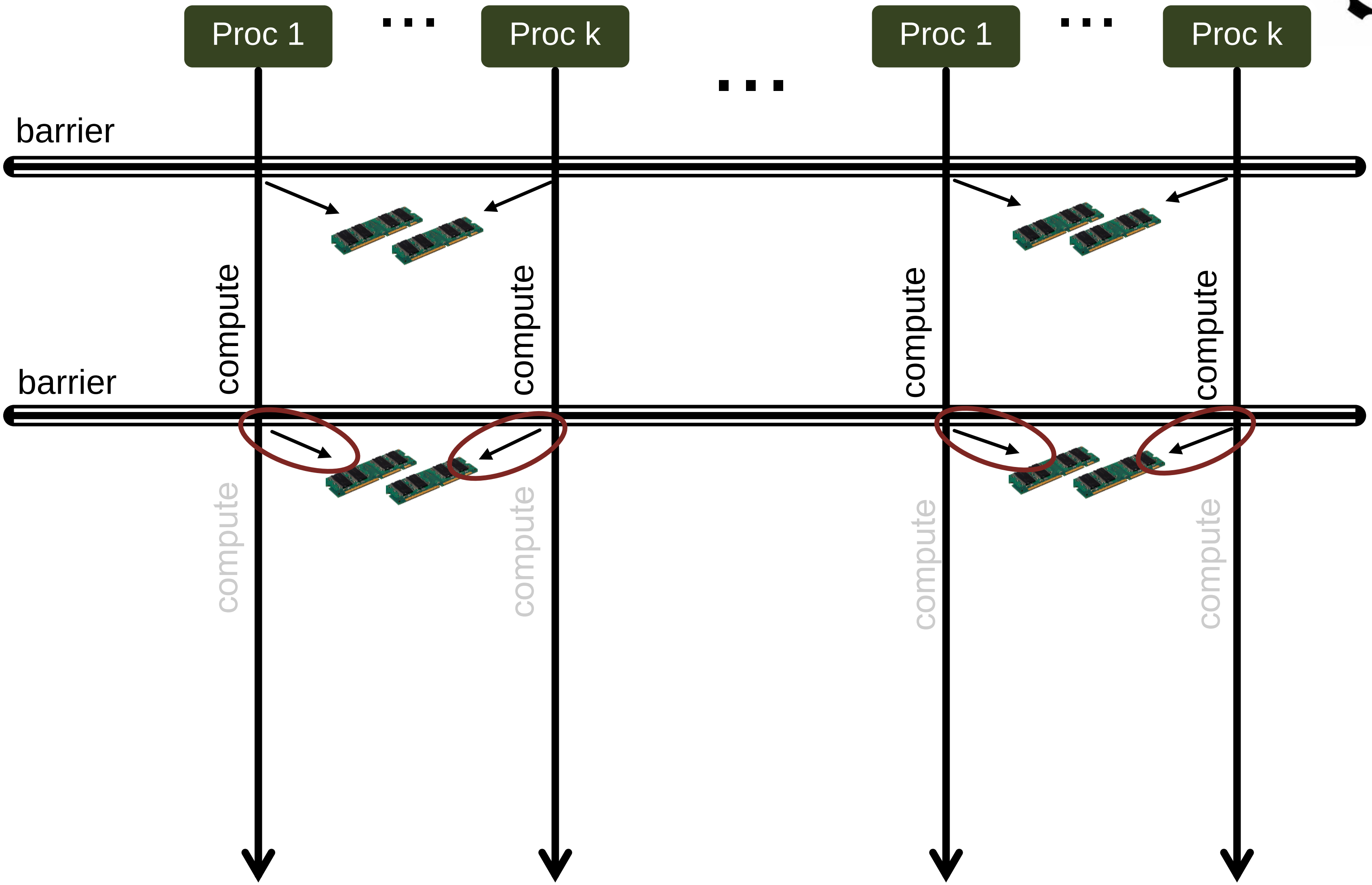
ACTIVE ACCESS USE-CASES

INCREMENTAL CHECKPOINTING FOR RMA



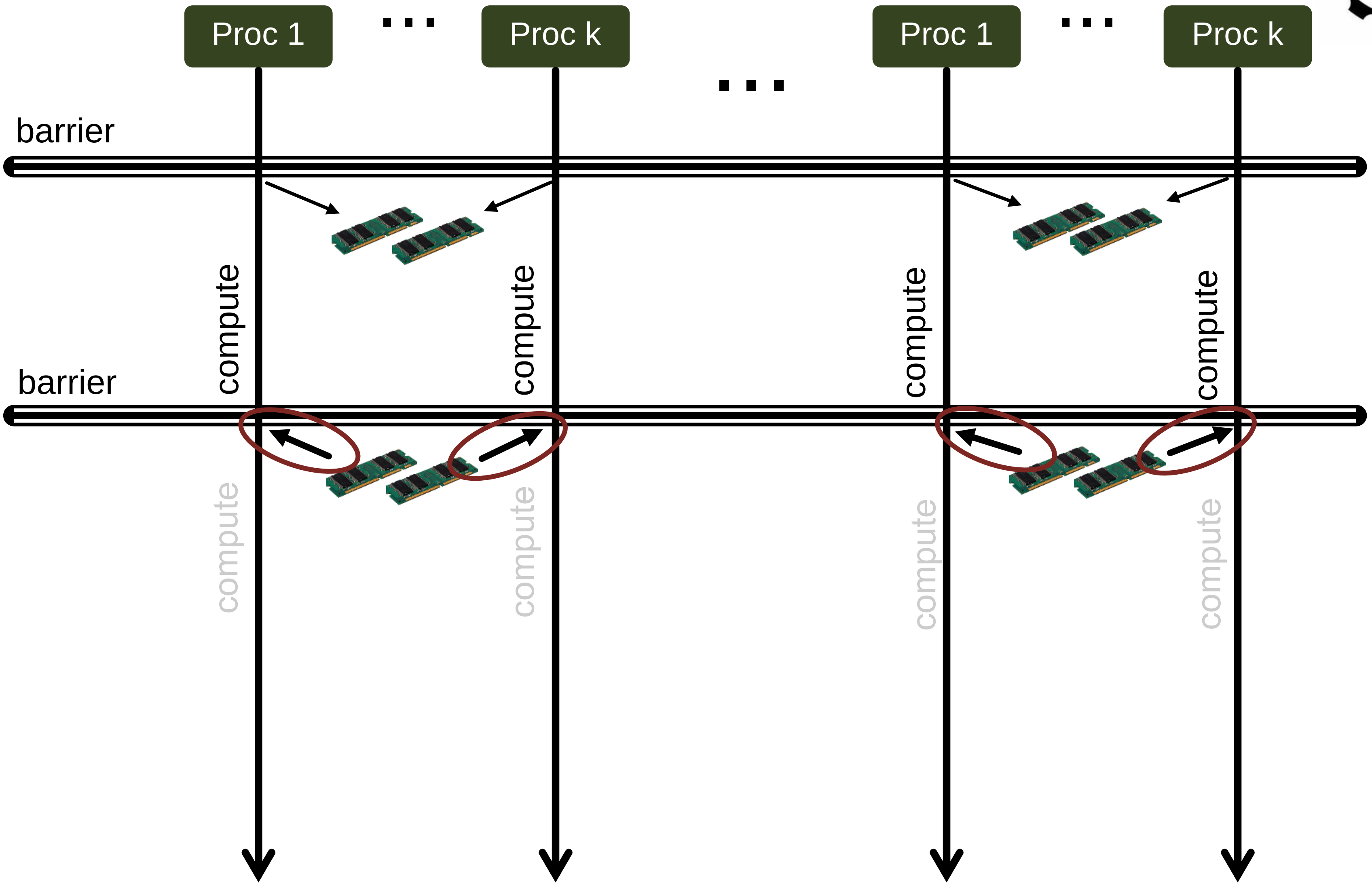
Active Access Use-Cases

Incremental checkpointing for RMA



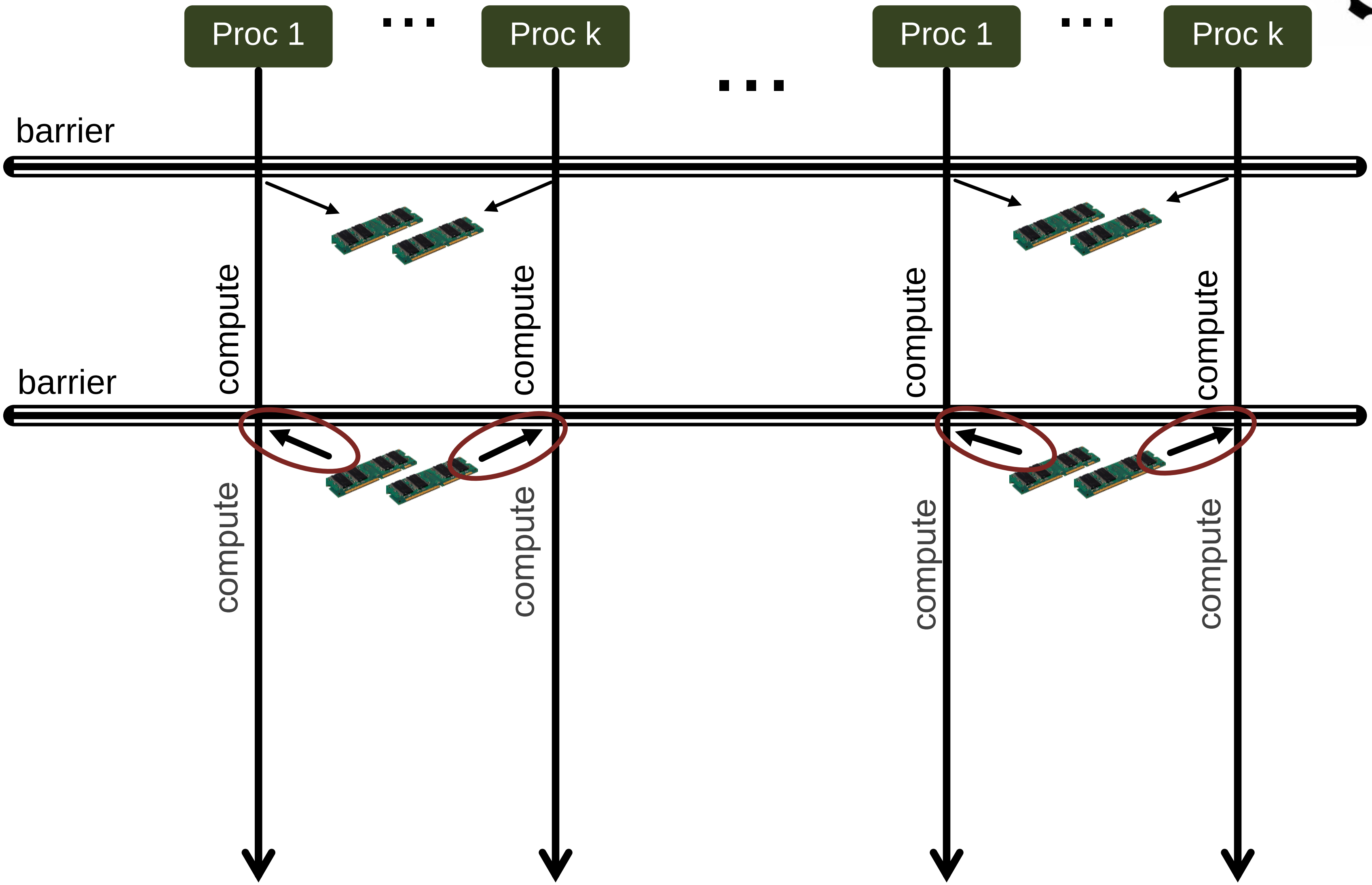
Active Access Use-Cases

Incremental checkpointing for RMA



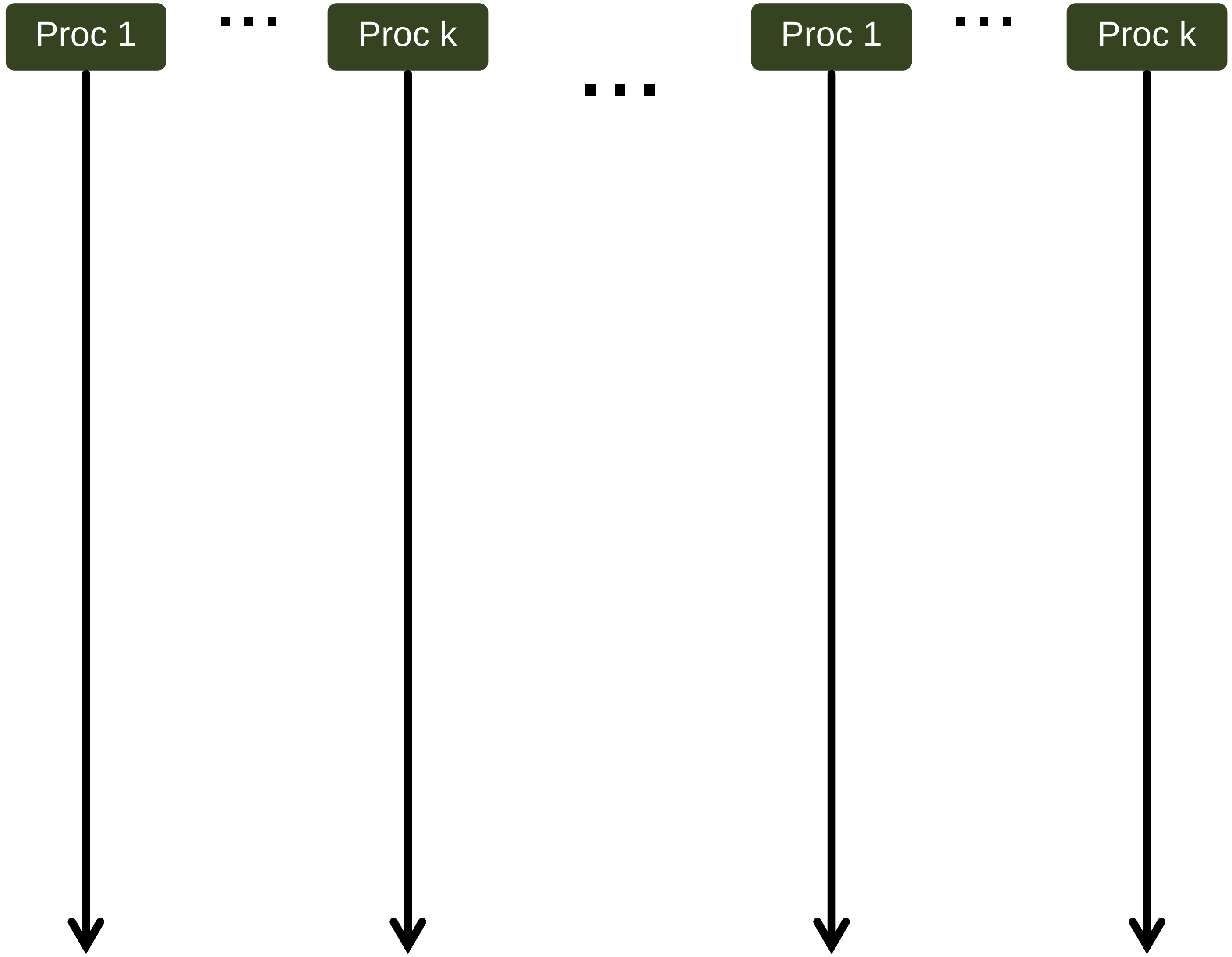
Active Access Use-Cases

Incremental checkpointing for RMA



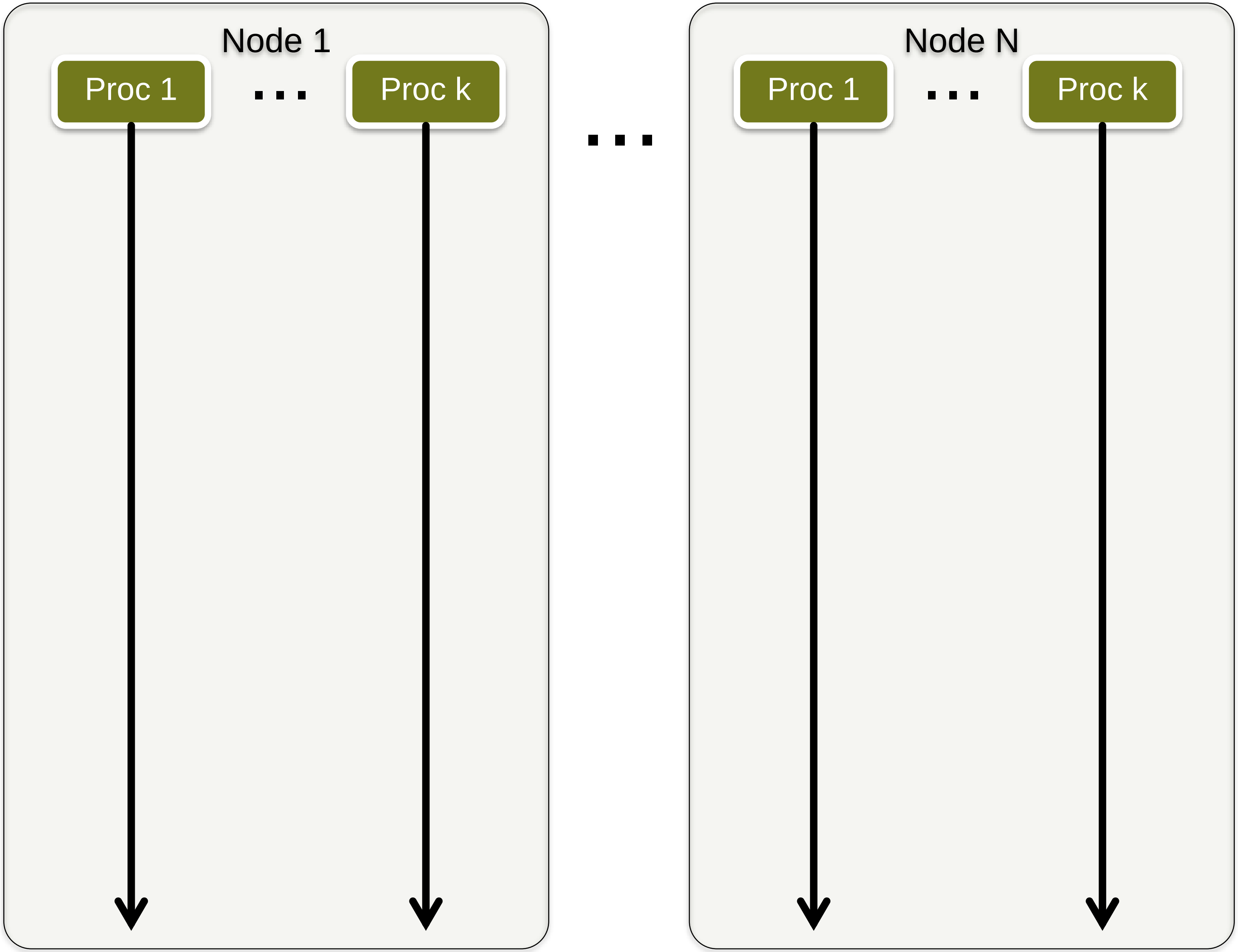
Active Access Use-Cases

Incremental checkpointing for RMA

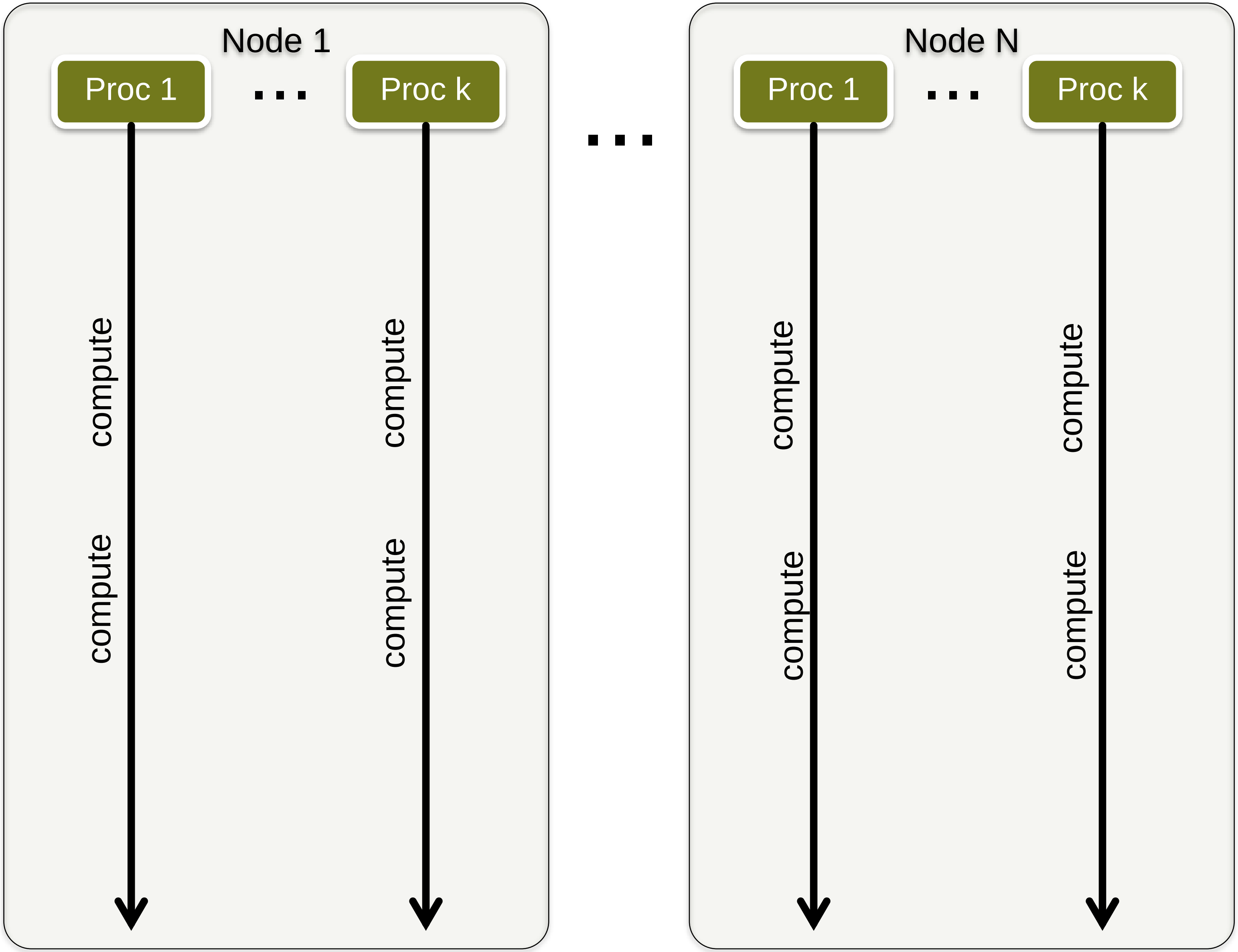


COORDINATED CHECKPOINTING (MP)

COORDINATED CHECKPOINTING (MP)



Coordinated Checkpointing (MP)



Node 1

Proc 1 … Proc k

compute

compute

compute

compute

…

Node N

Proc 1 … Proc k

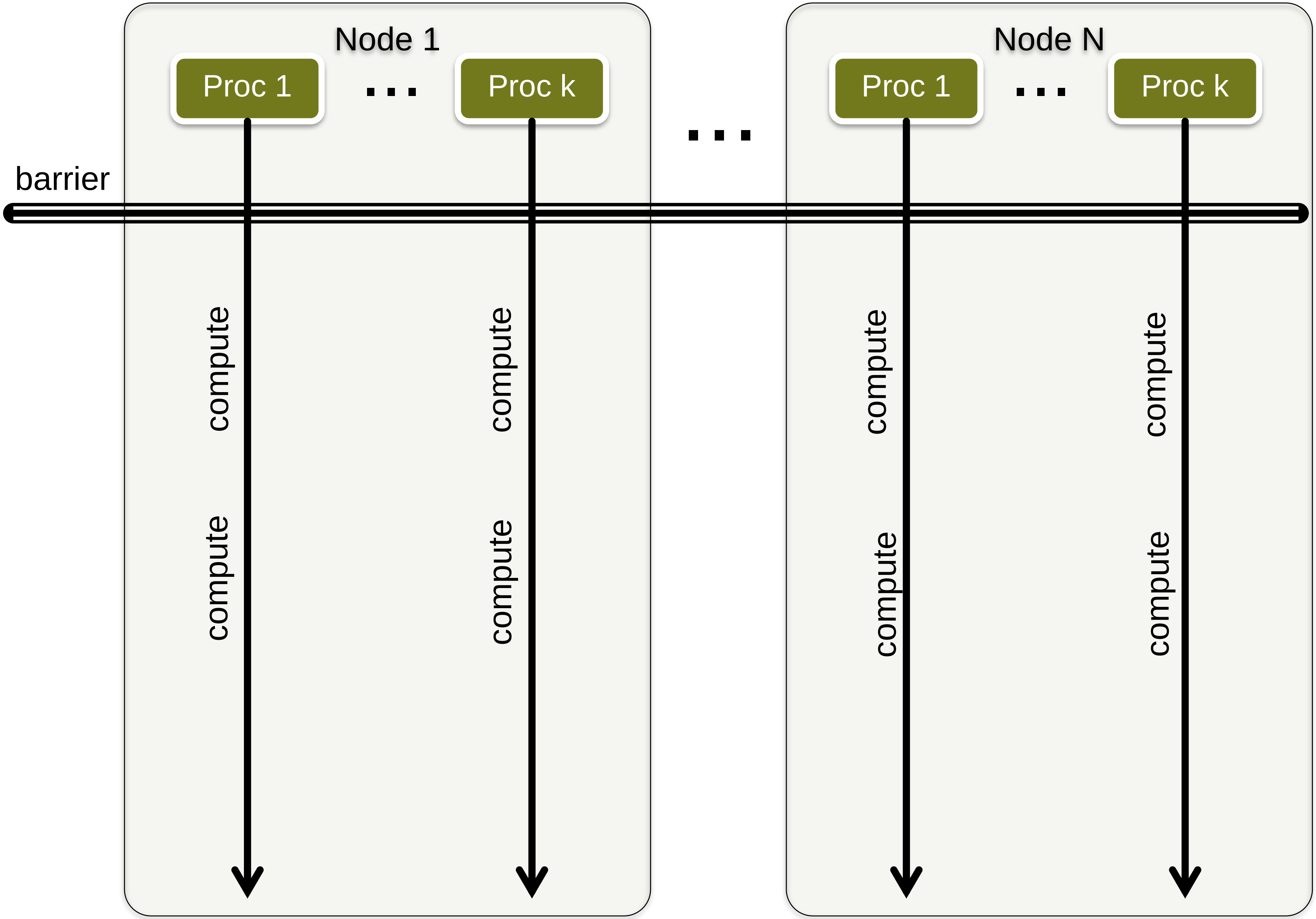
compute

compute

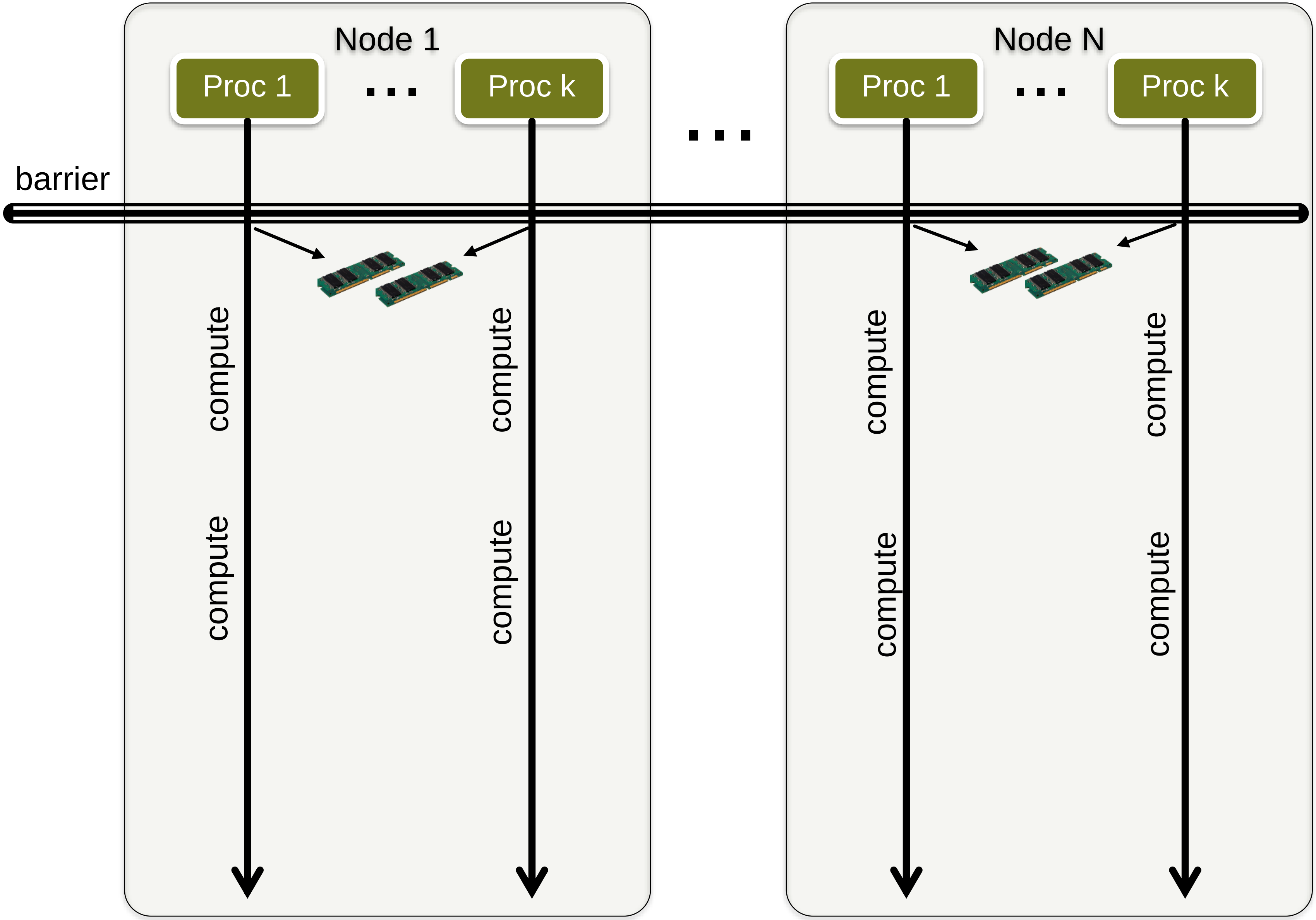
compute

compute

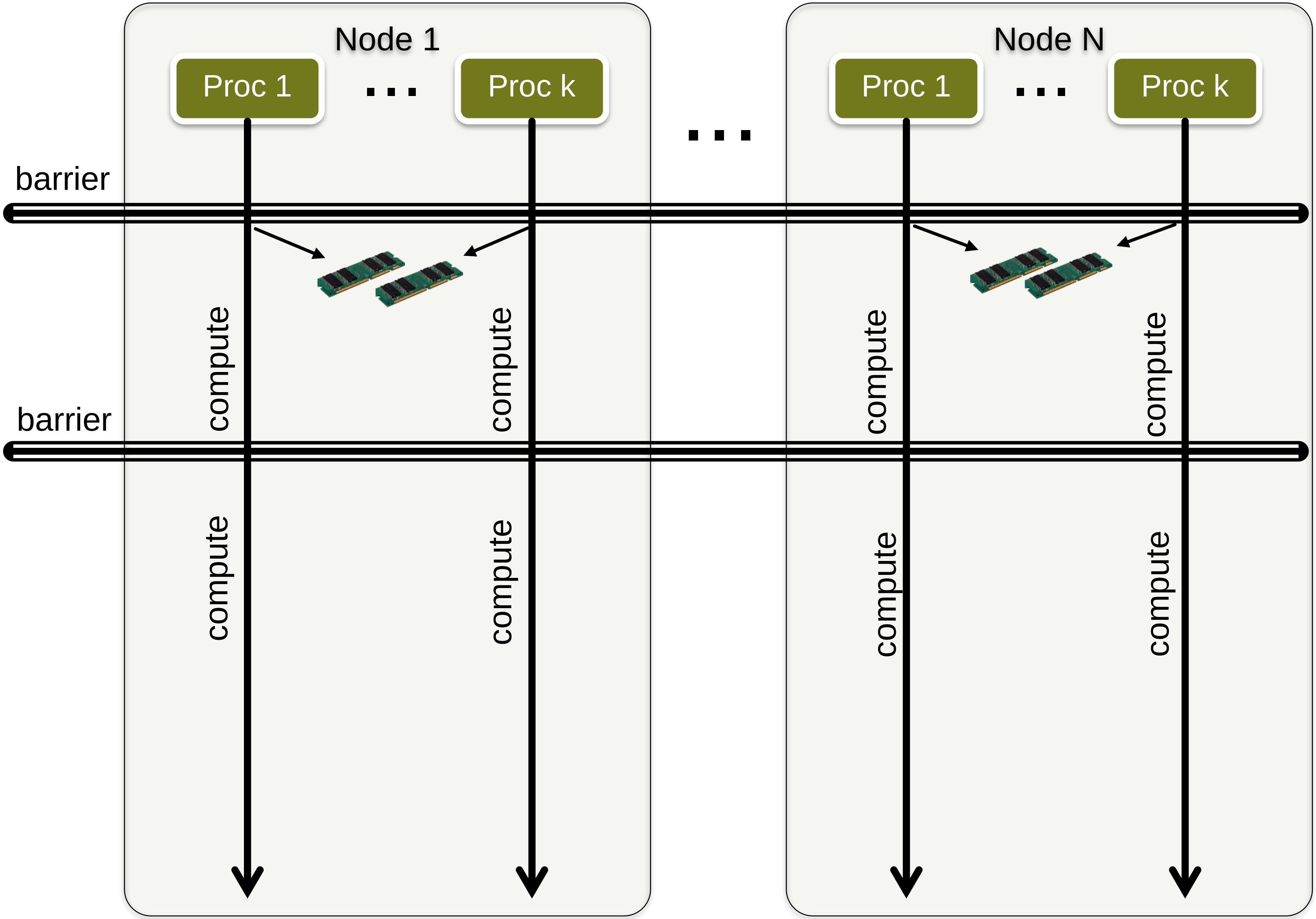
Coordinated Checkpointing (MP)



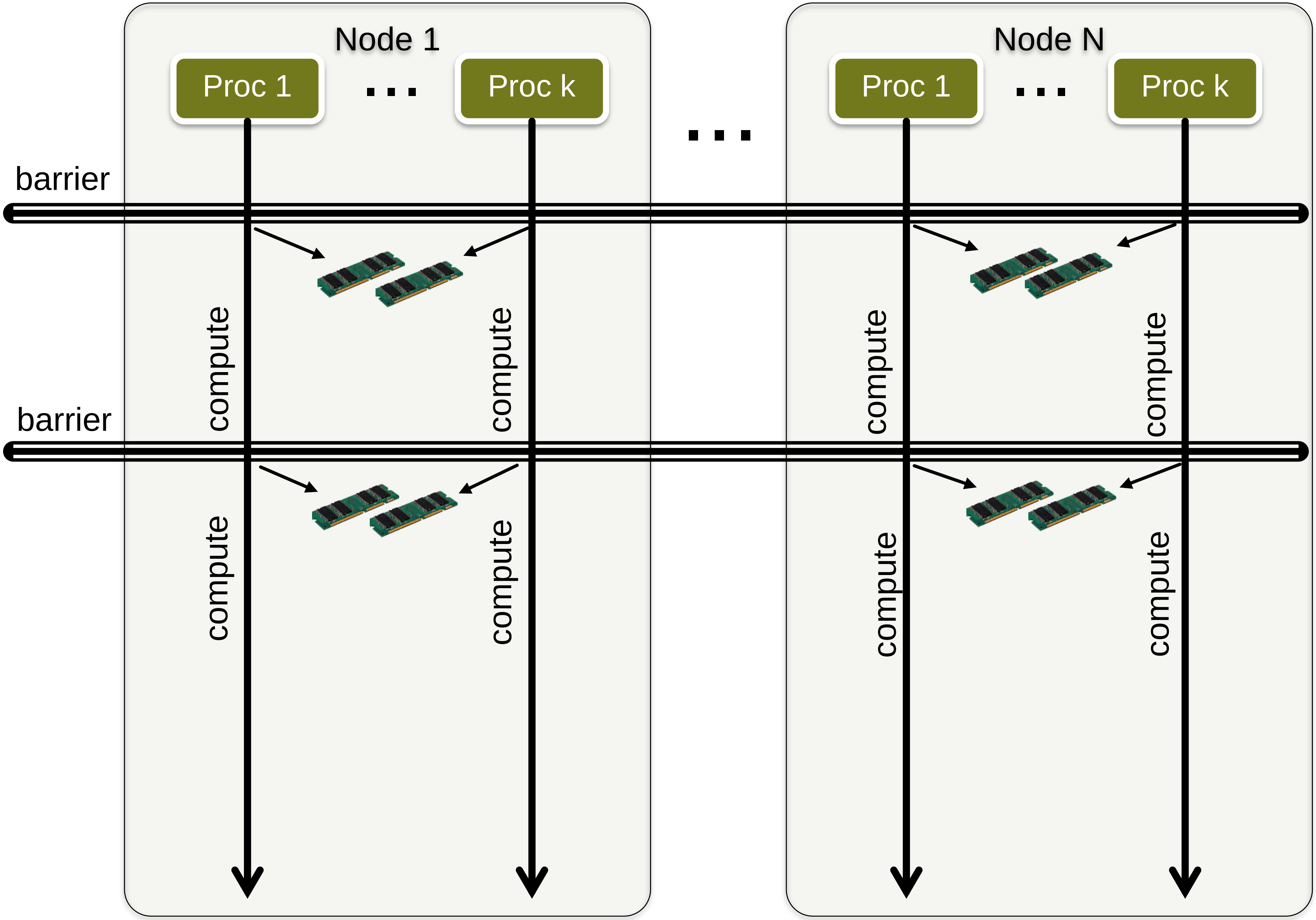
Coordinated Checkpointing (MP)



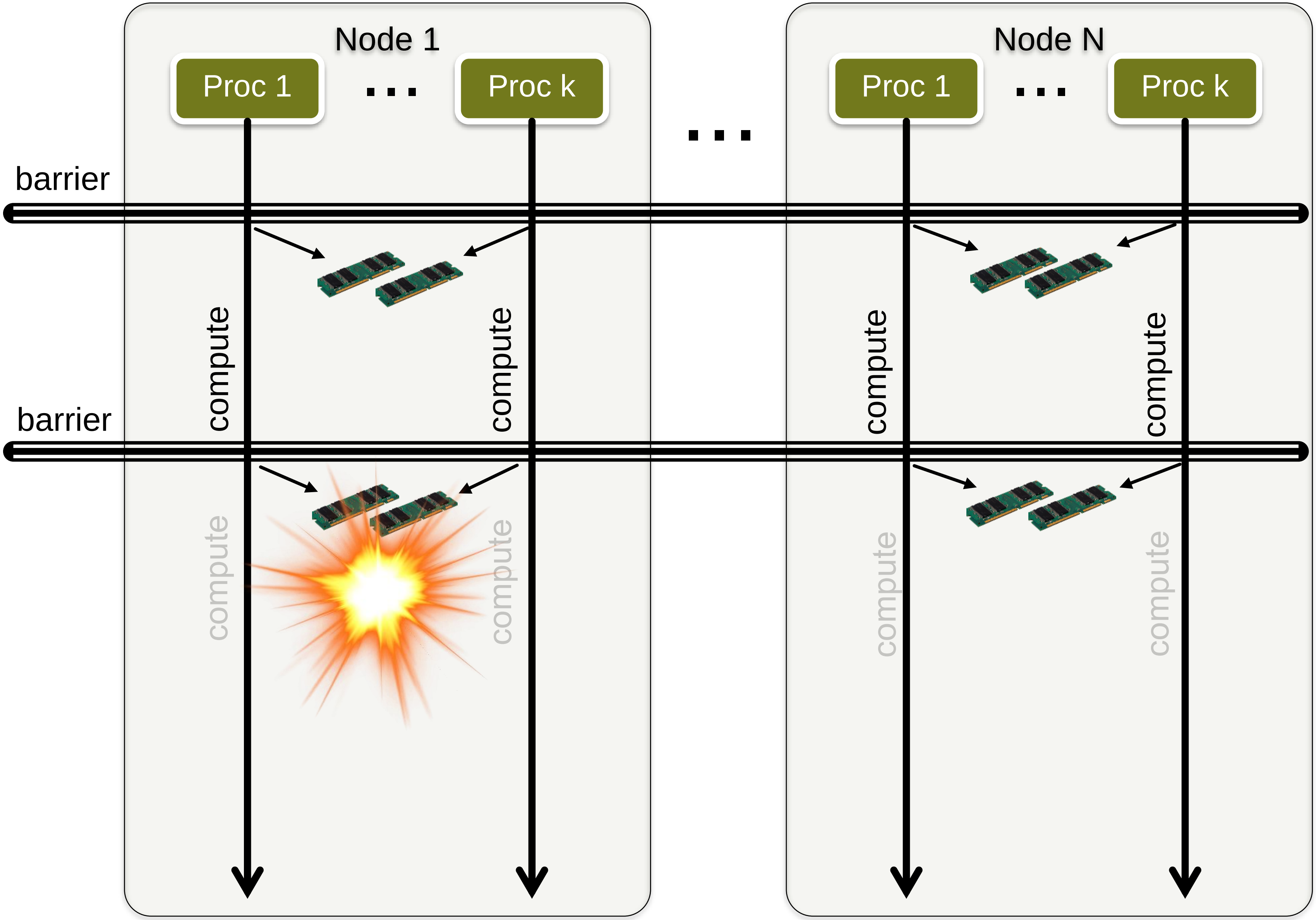
Coordinated Checkpointing (MP)



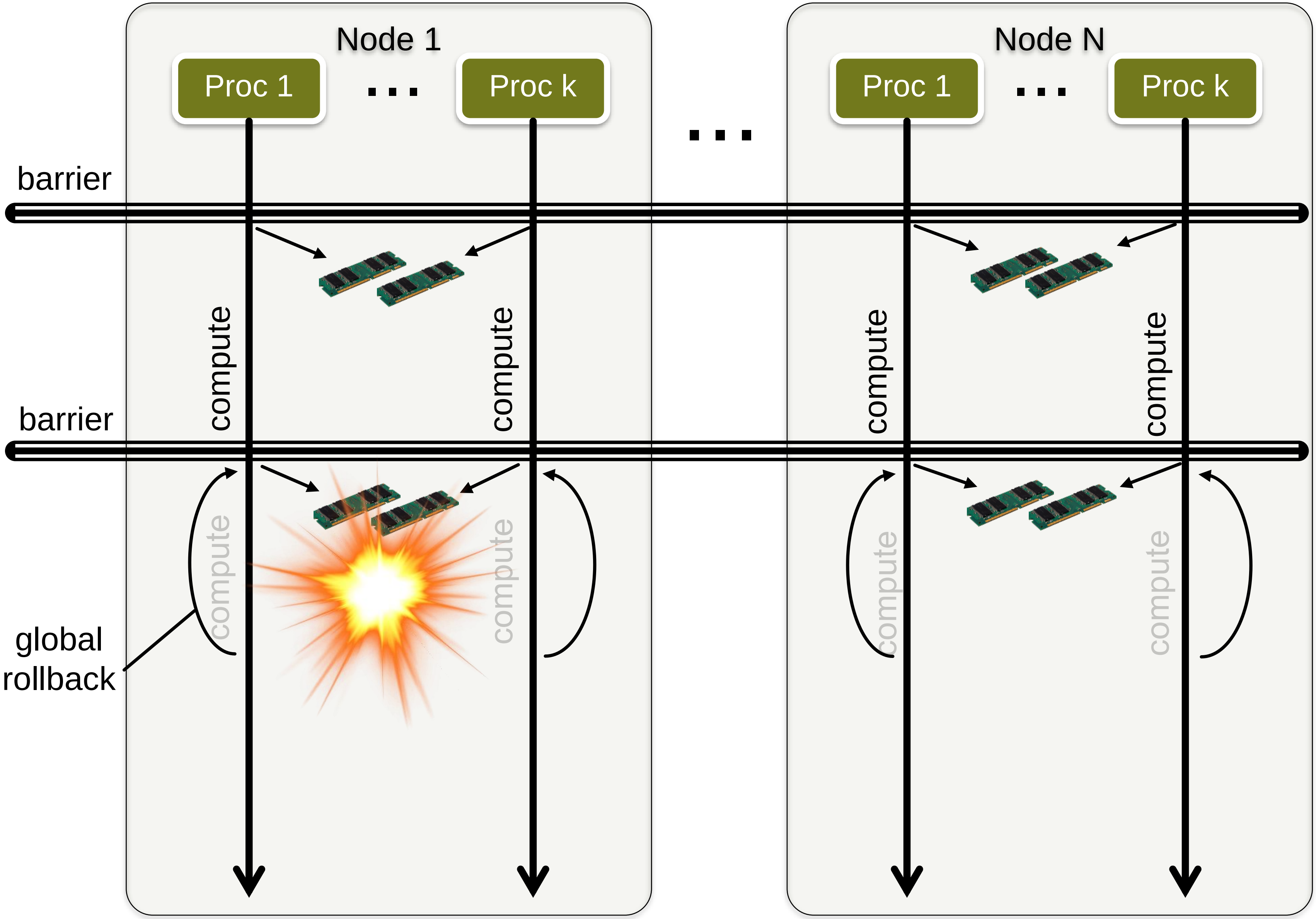
COORDINATED CHECKPOINTING (MP)



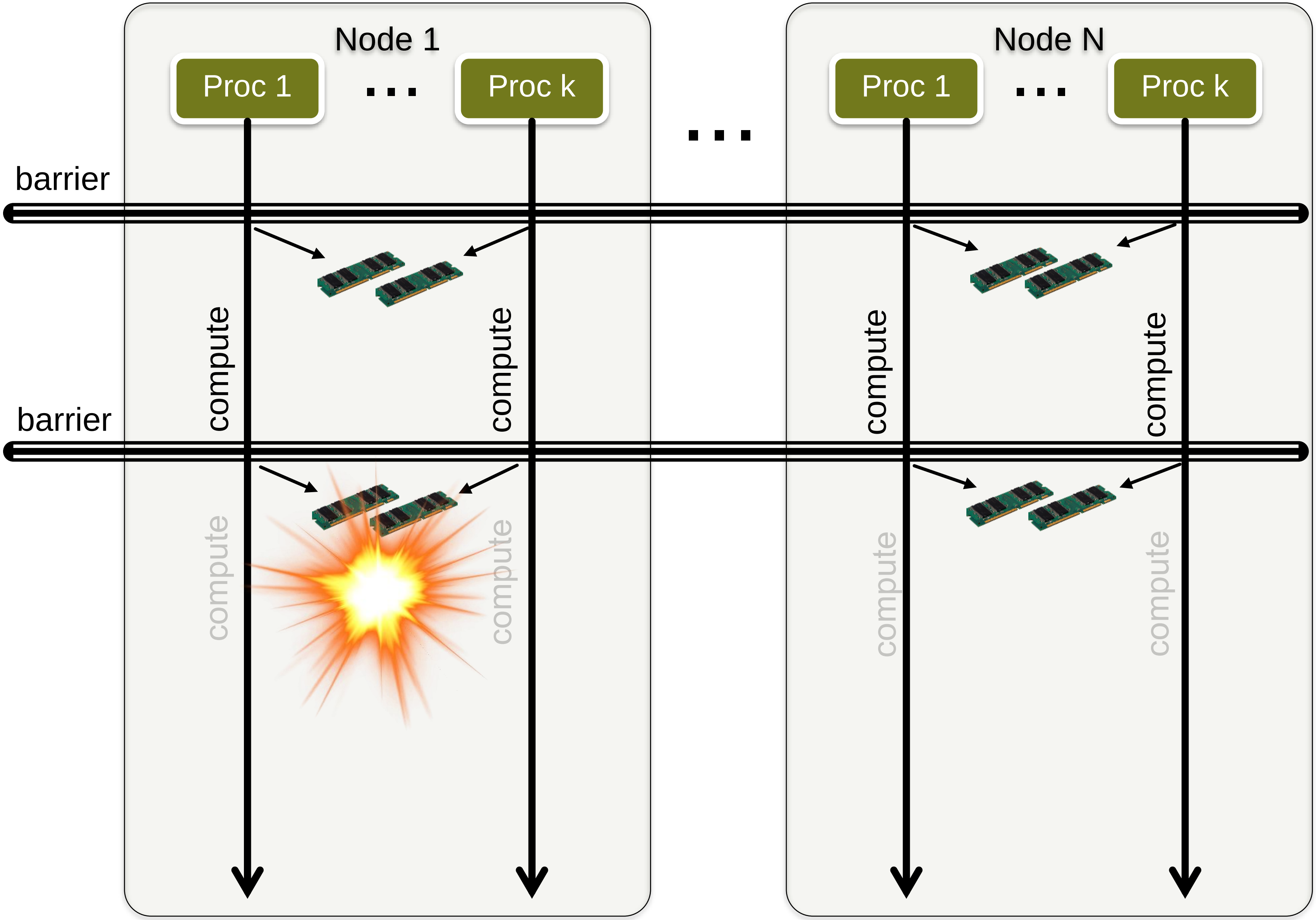
Coordinated Checkpointing (MP)



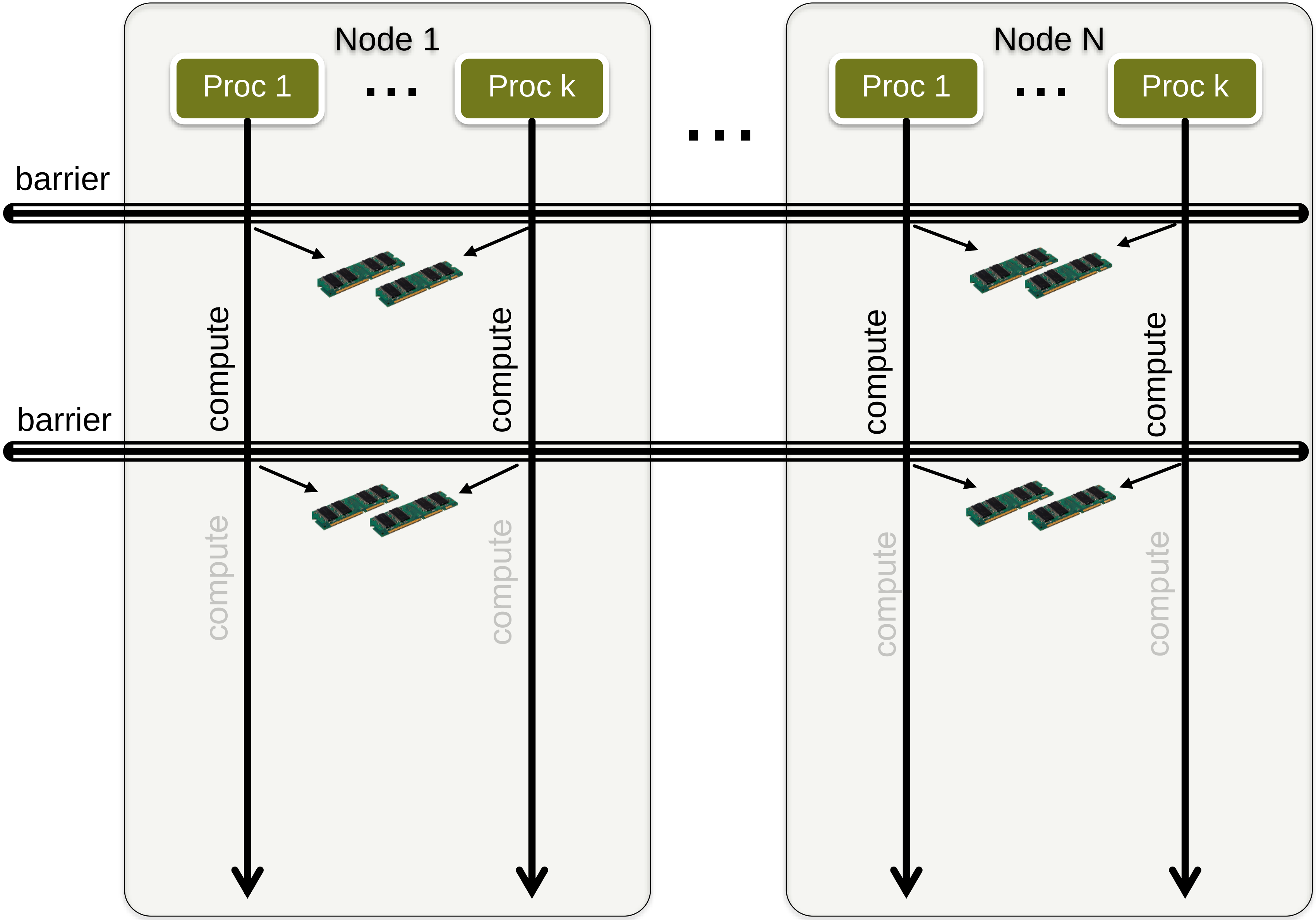
Coordinated Checkpointing (MP)



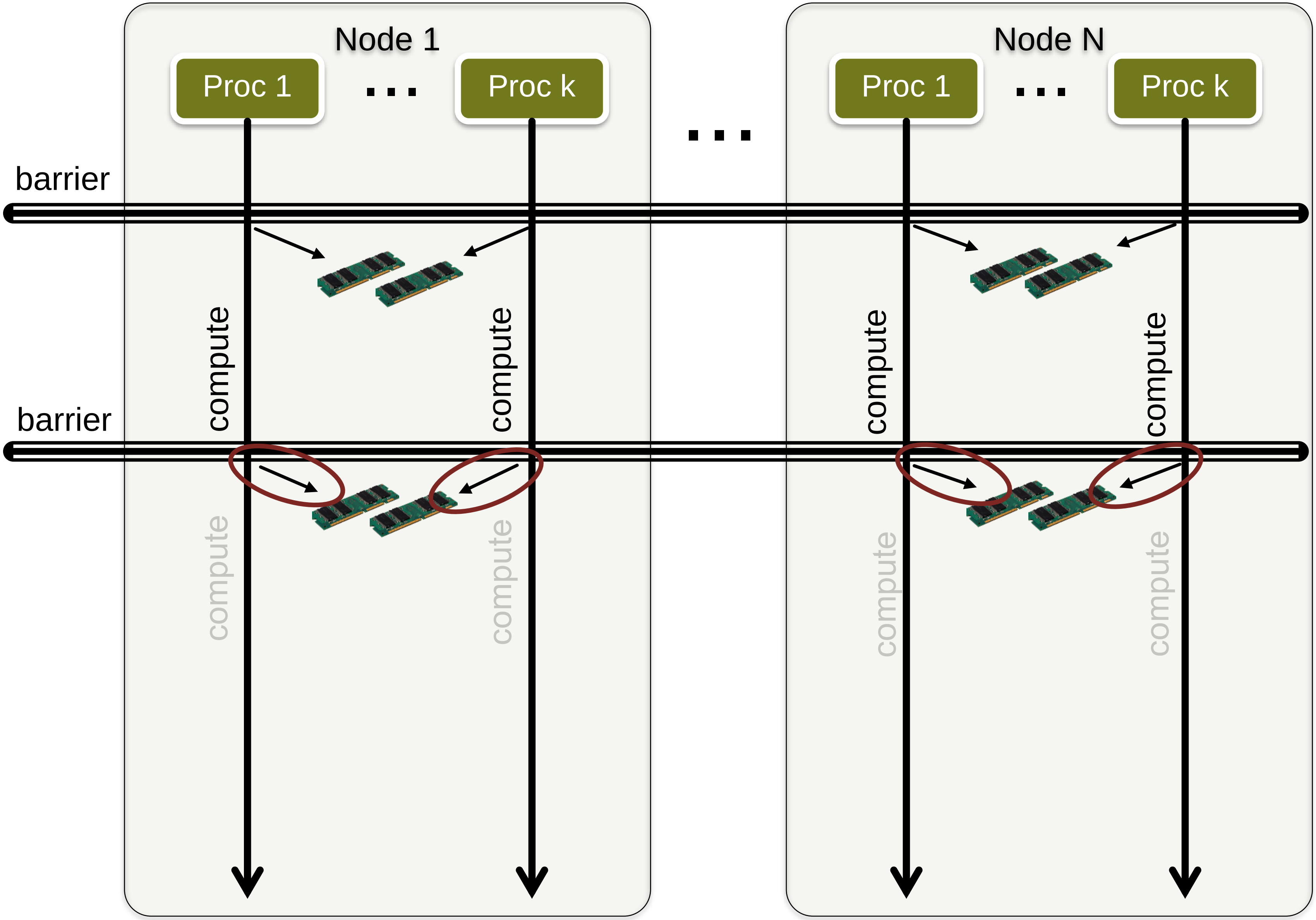
Coordinated Checkpointing (MP)



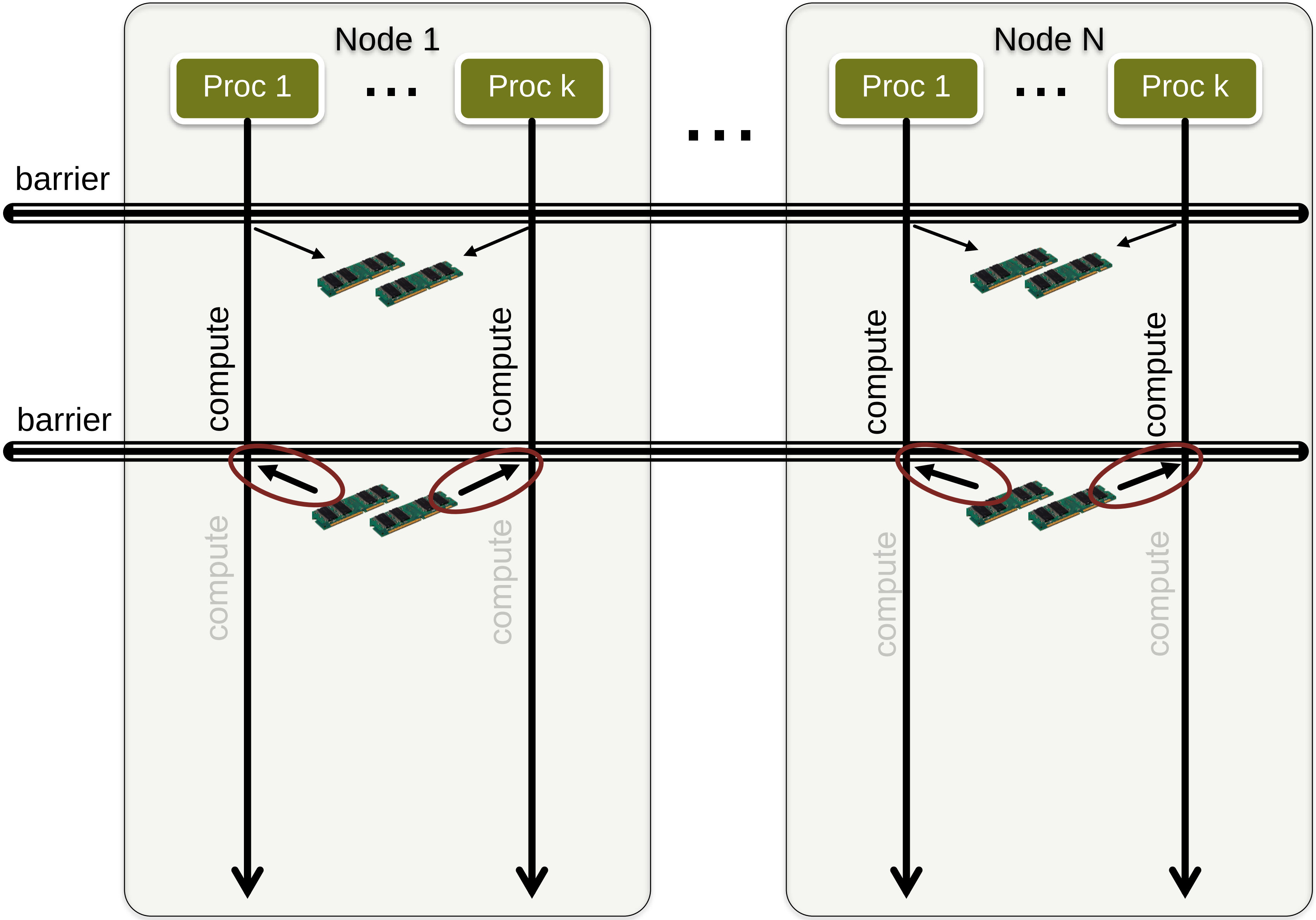
COORDINATED CHECKPOINTING (MP)



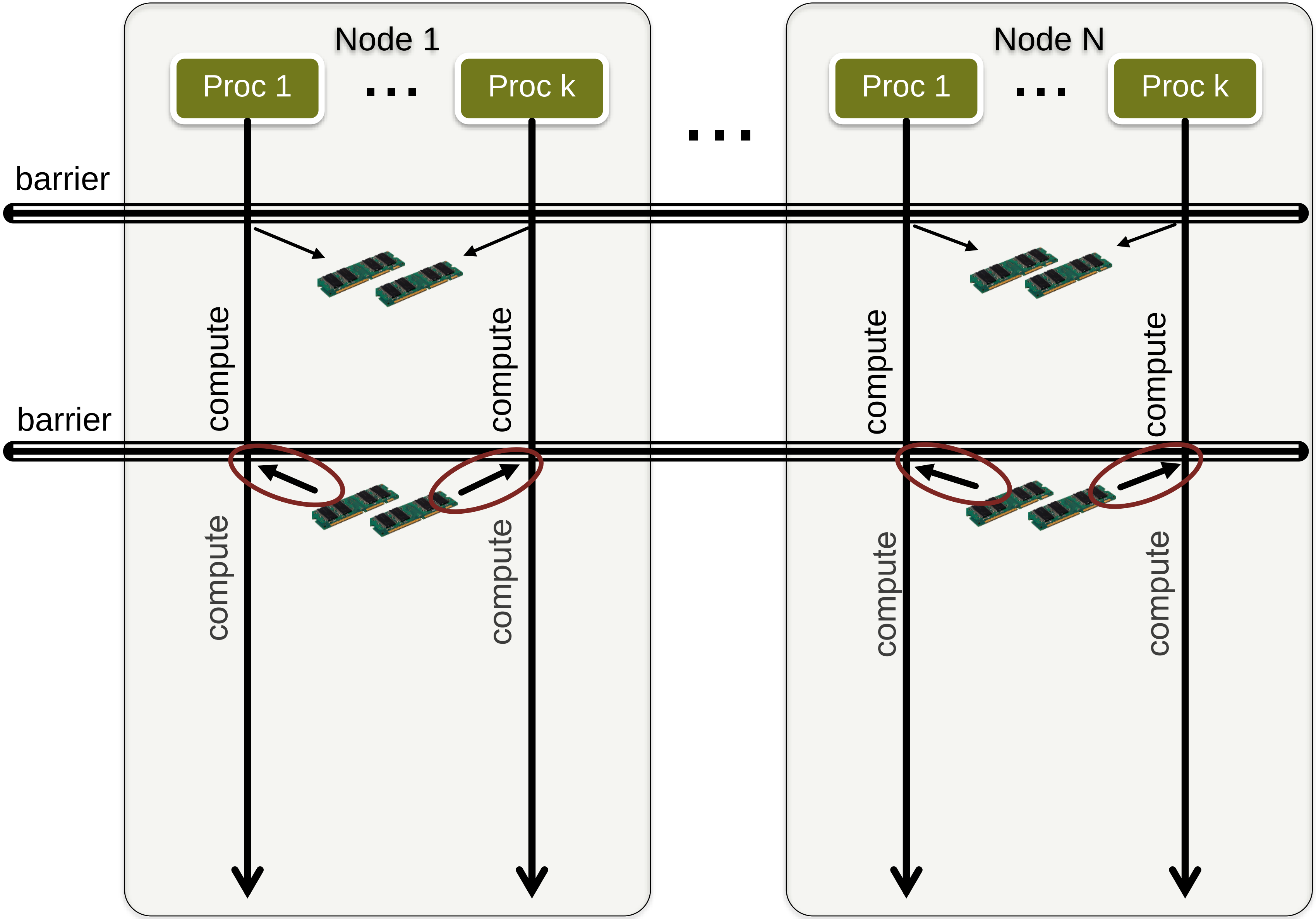
Coordinated Checkpointing (MP)



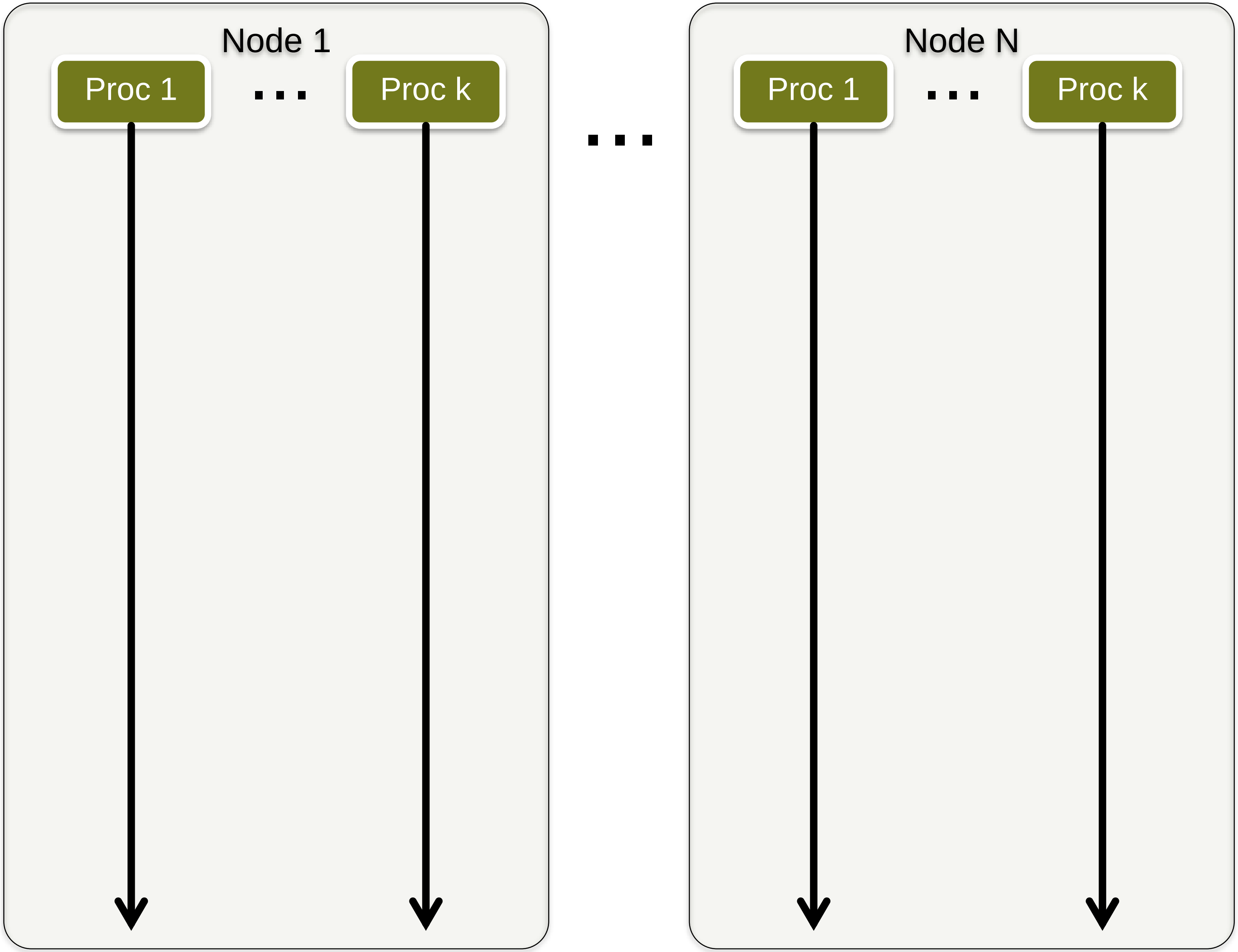
Coordinated Checkpointing (MP)



Coordinated Checkpointing (MP)

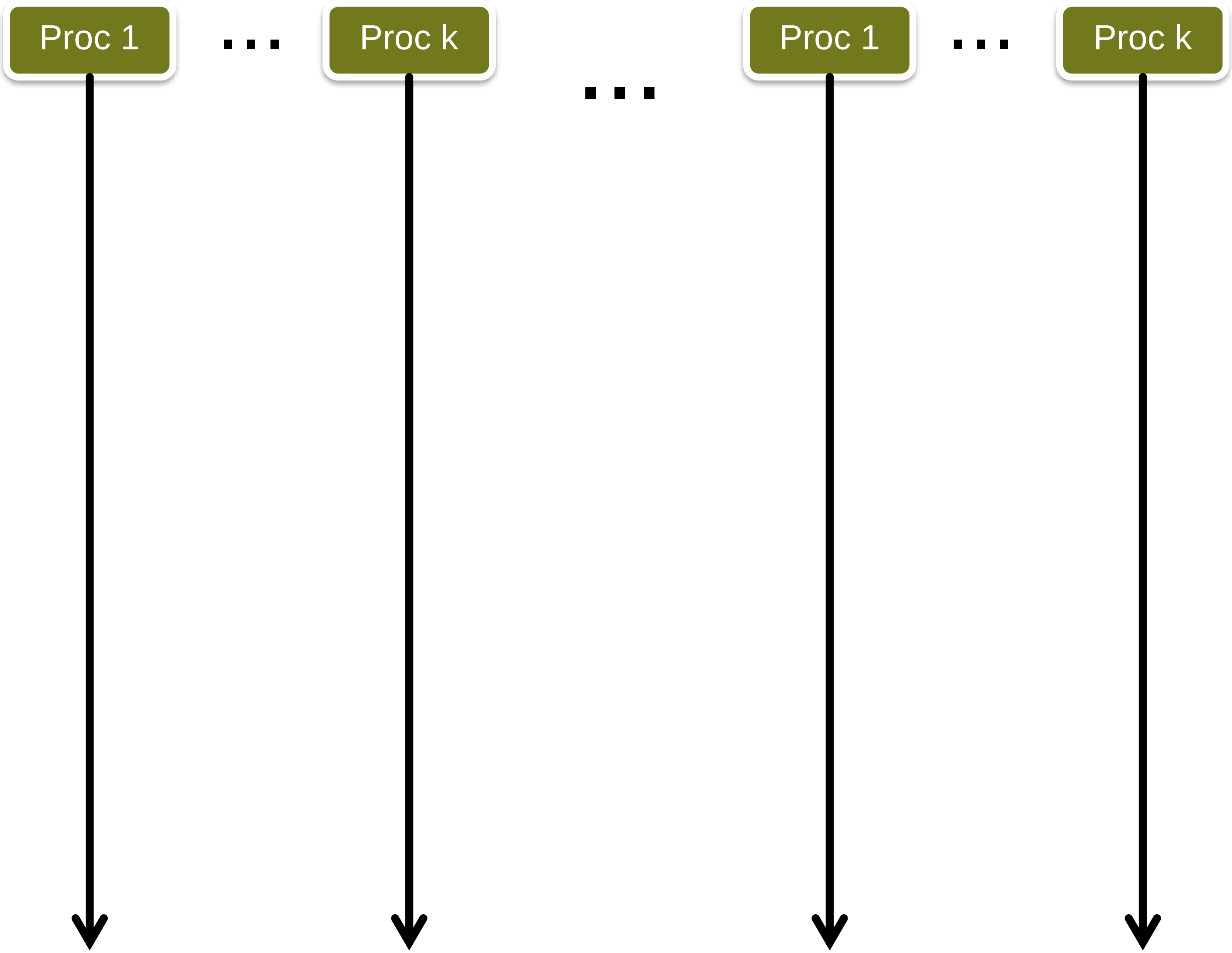


COORDINATED CHECKPOINTING (MP)

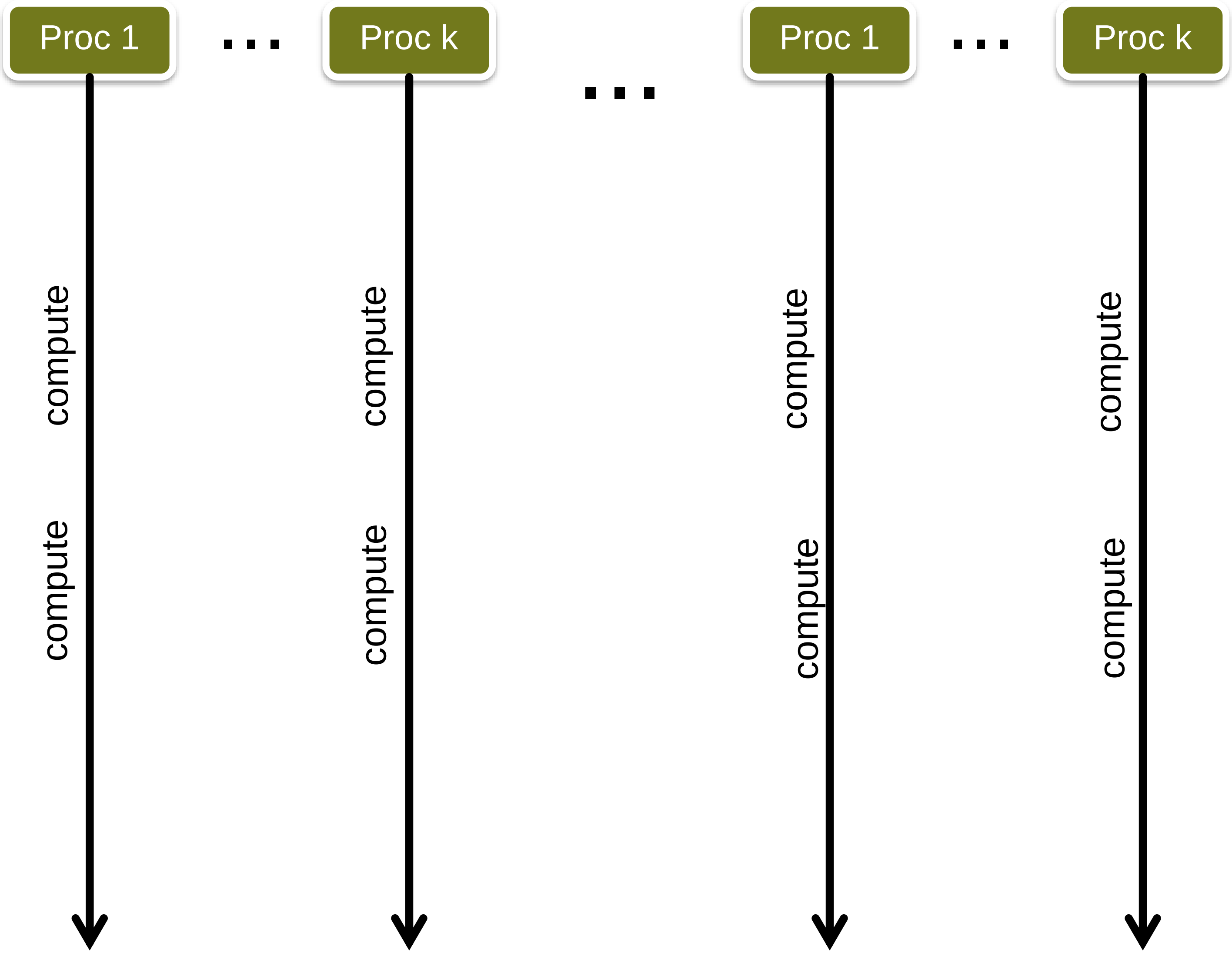


Coordinated Checkpointing (MP)

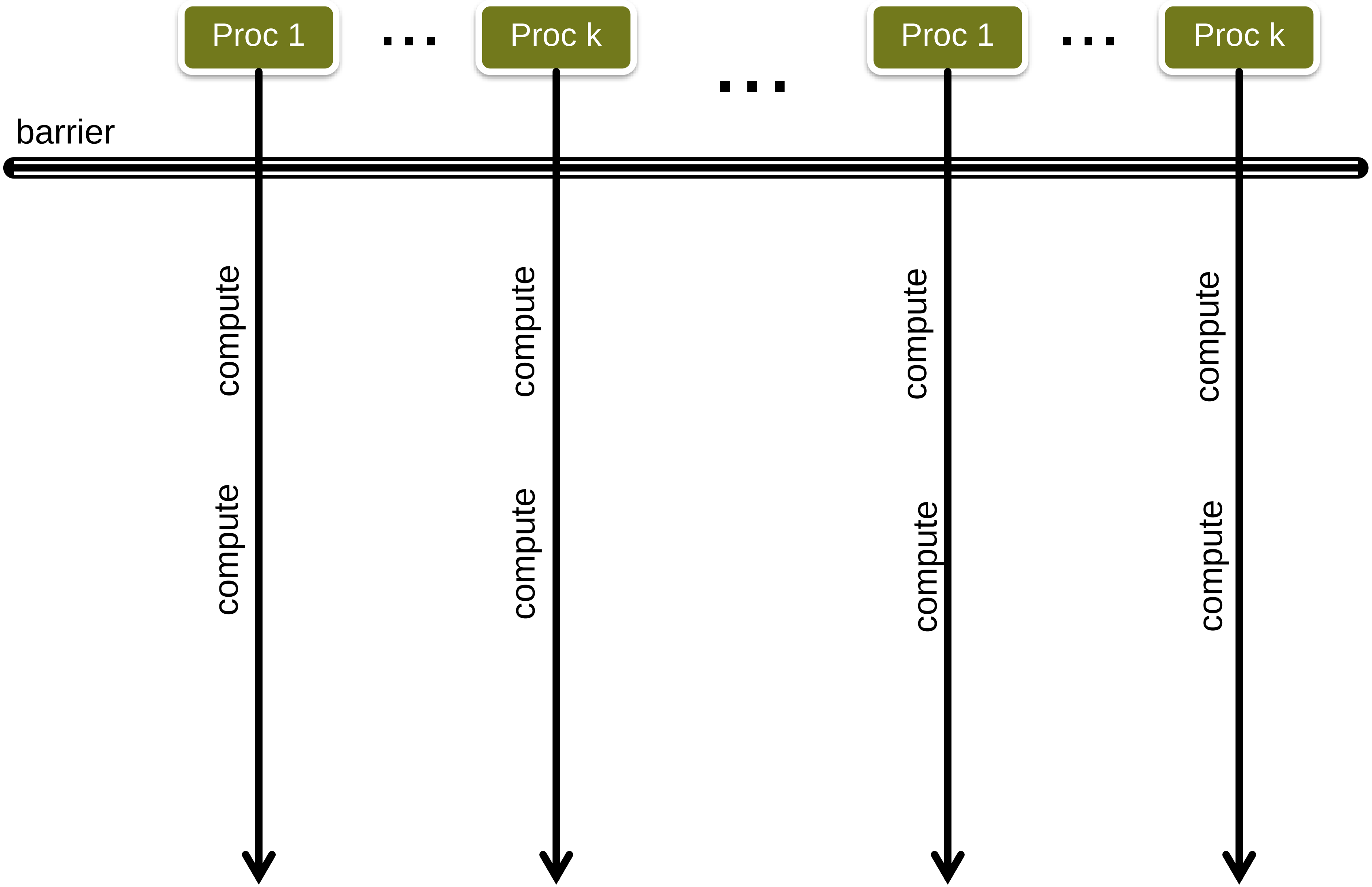
COORDINATED CHECKPOINTING (MP)



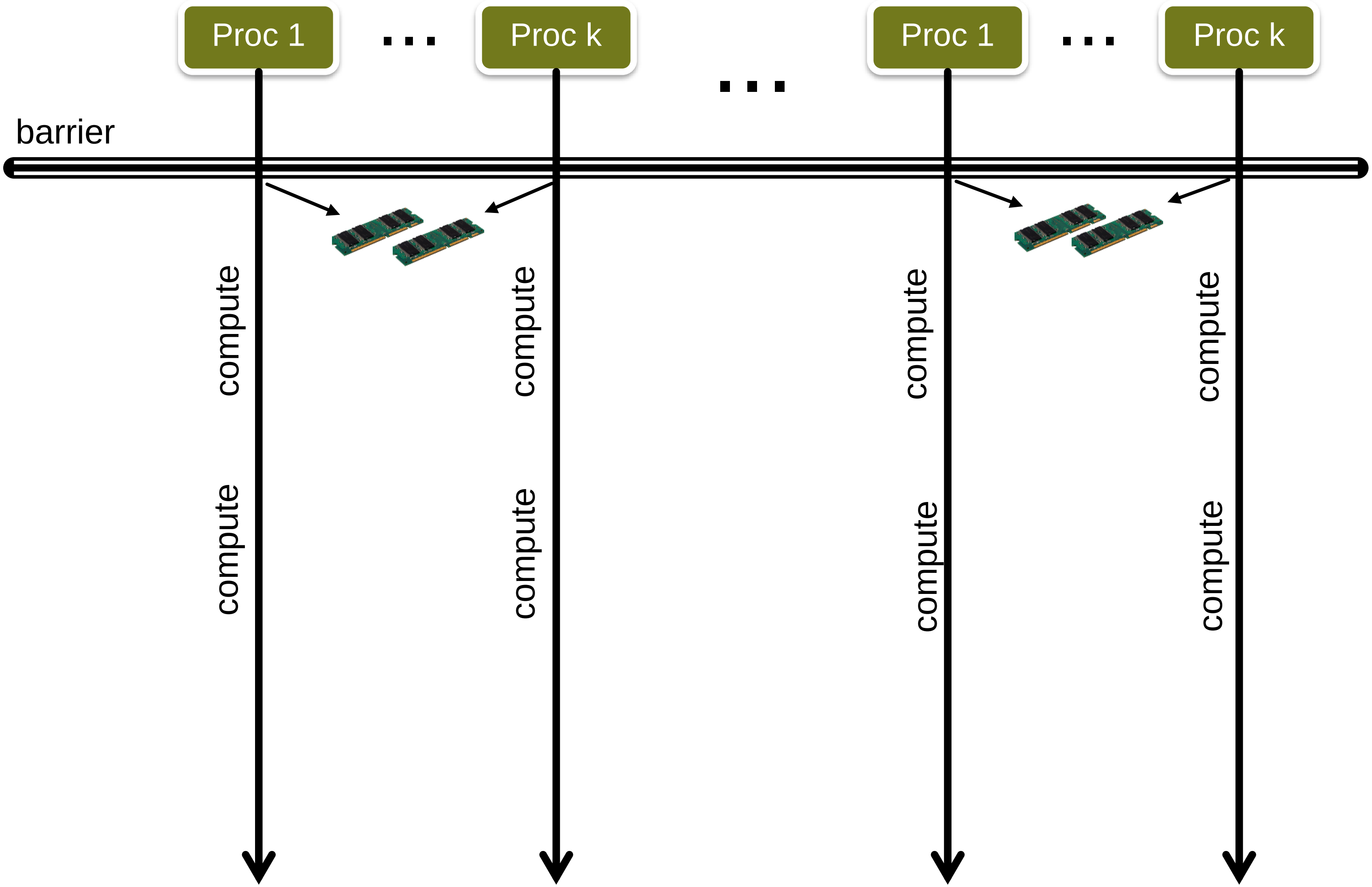
COORDINATED CHECKPOINTING (MP)



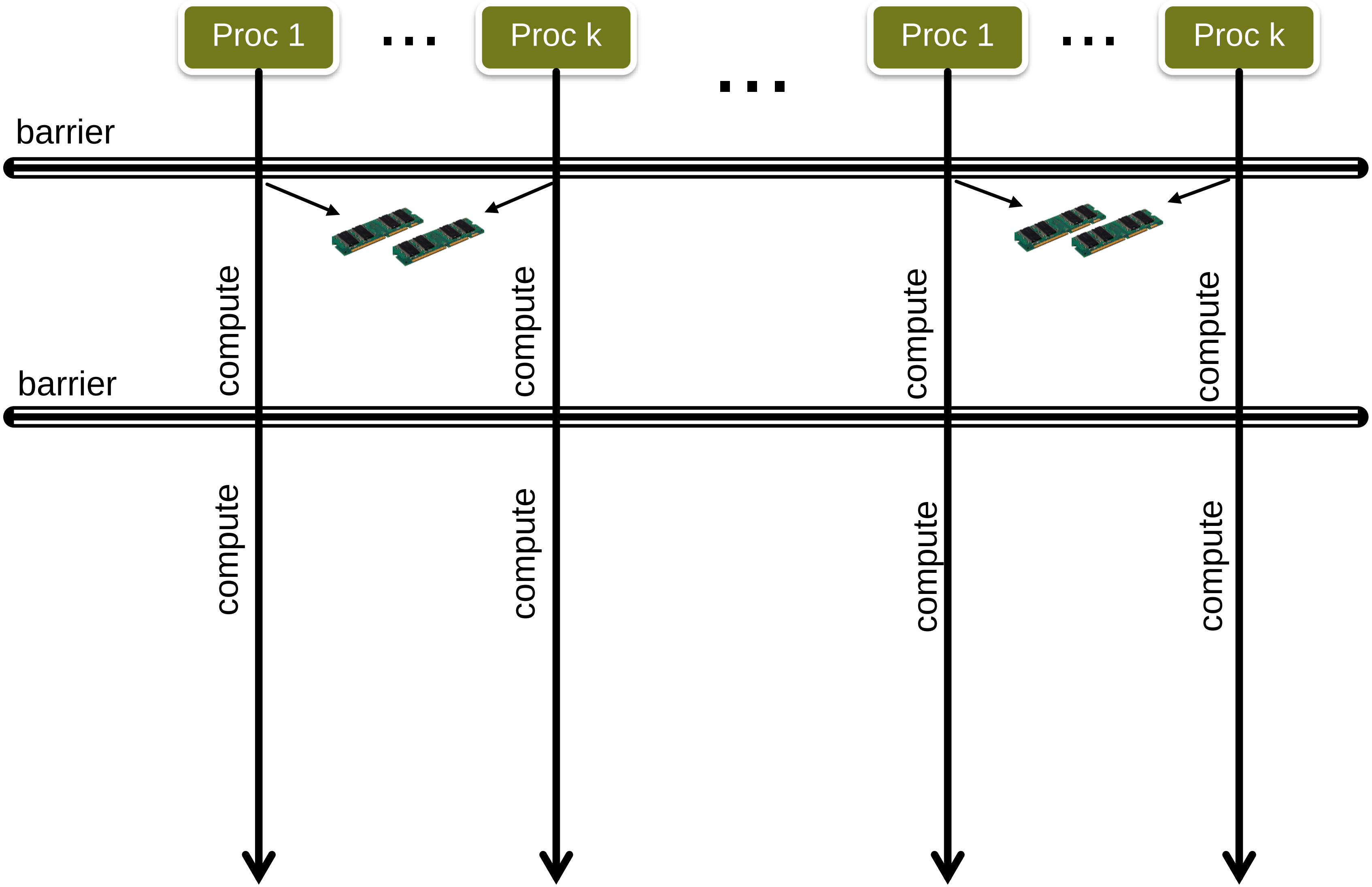
Coordinated Checkpointing (MP)



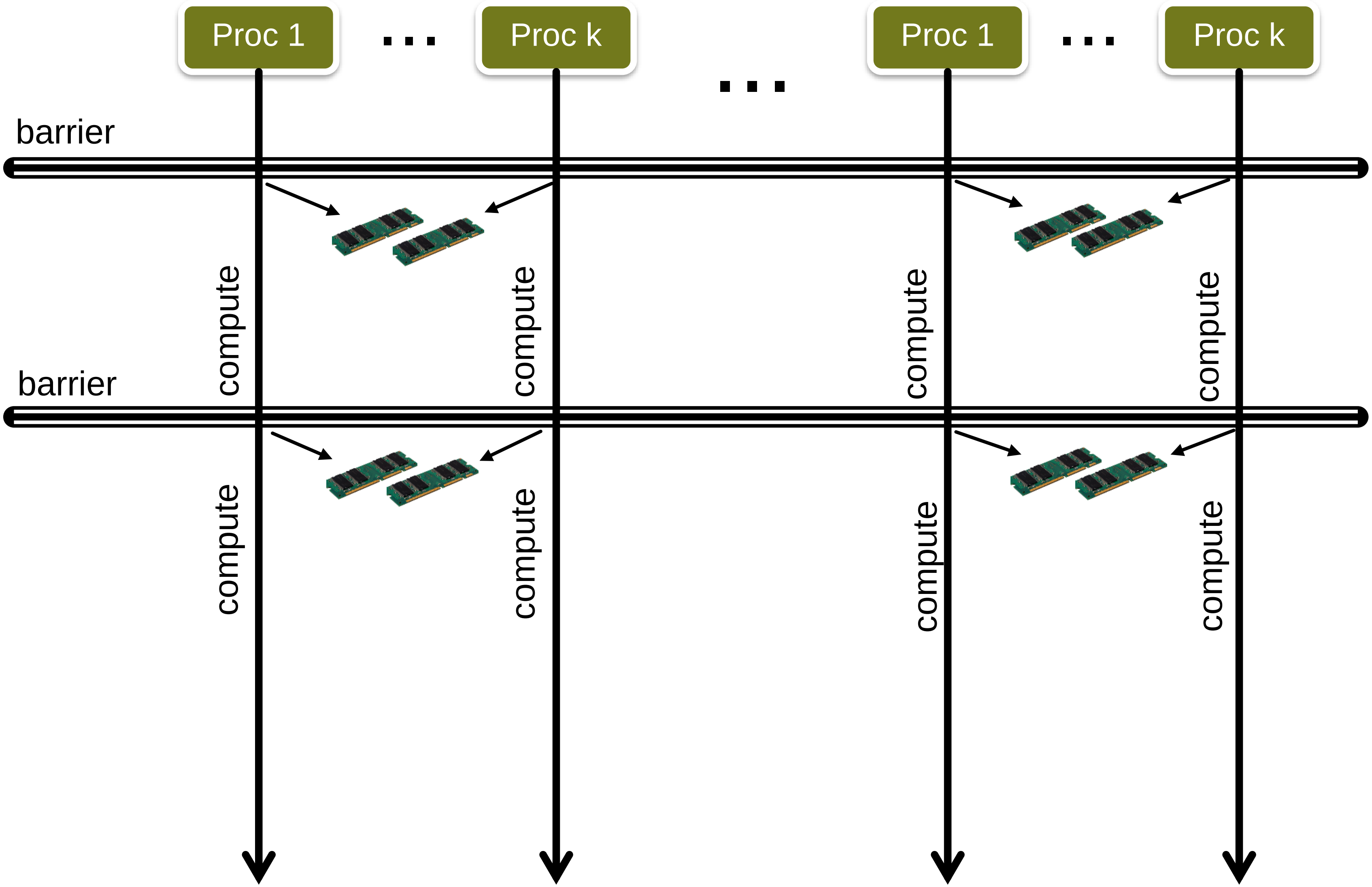
Coordinated Checkpointing (MP)



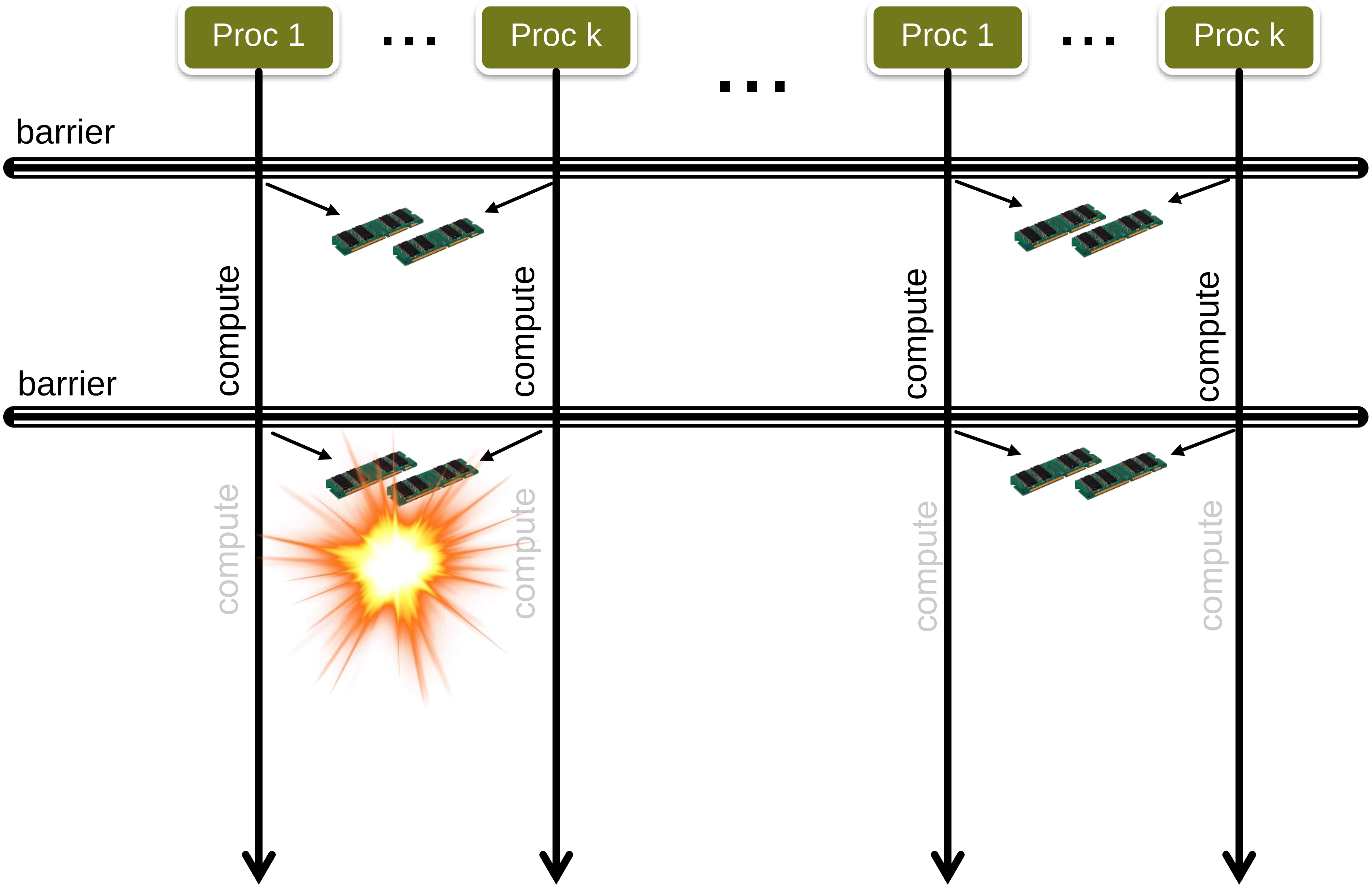
Coordinated Checkpointing (MP)



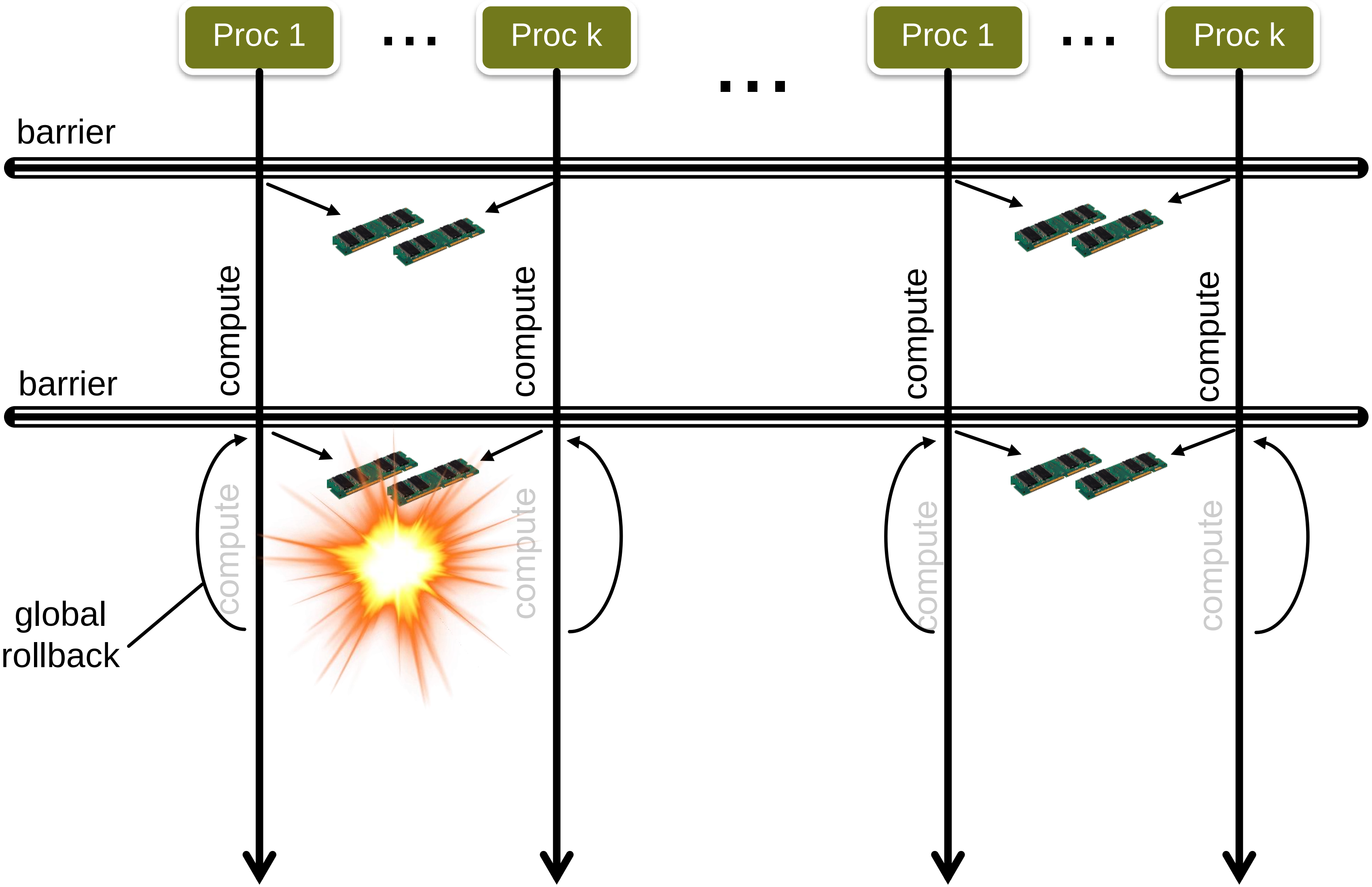
Coordinated Checkpointing (MP)



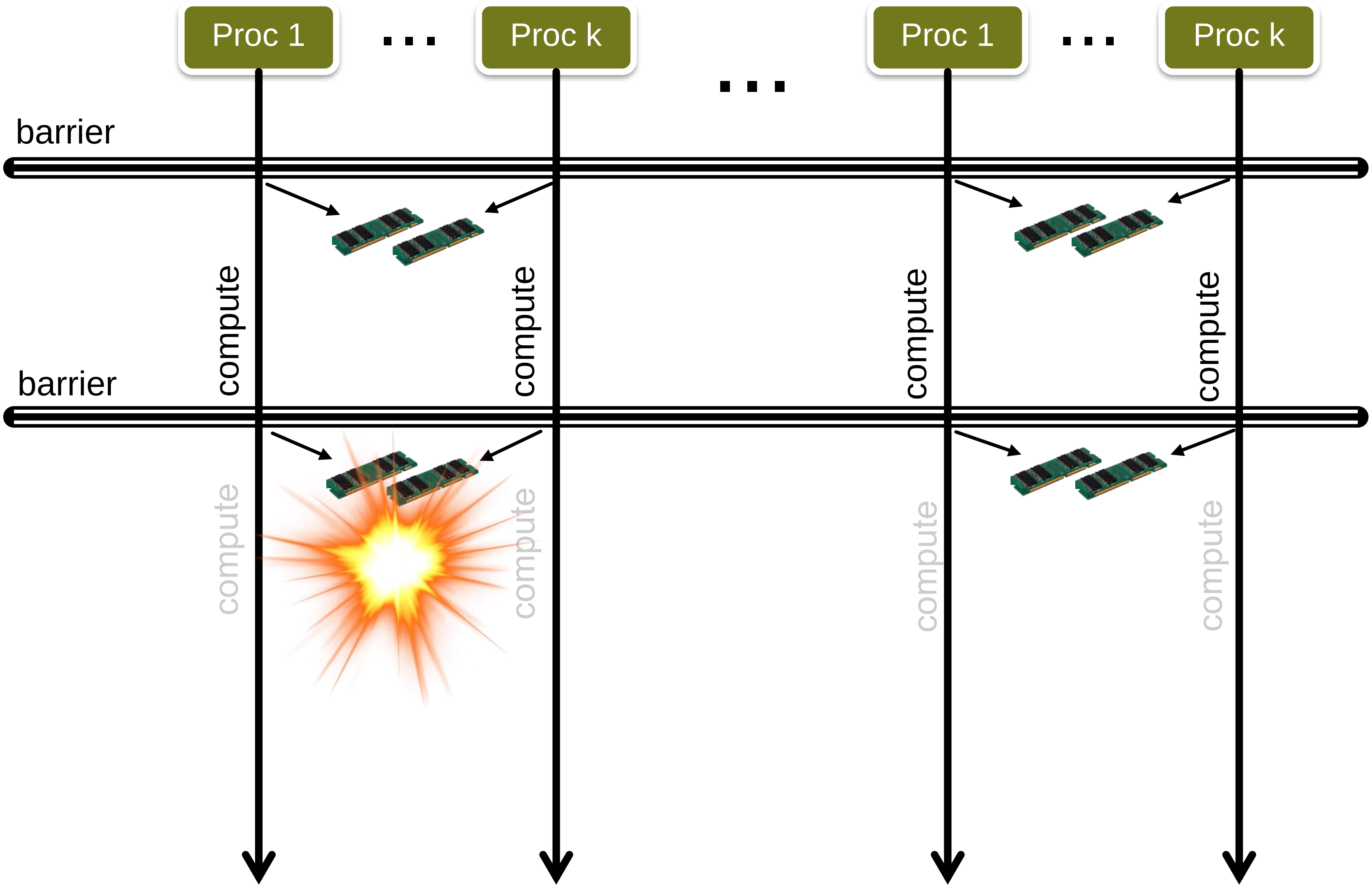
Coordinated Checkpointing (MP)



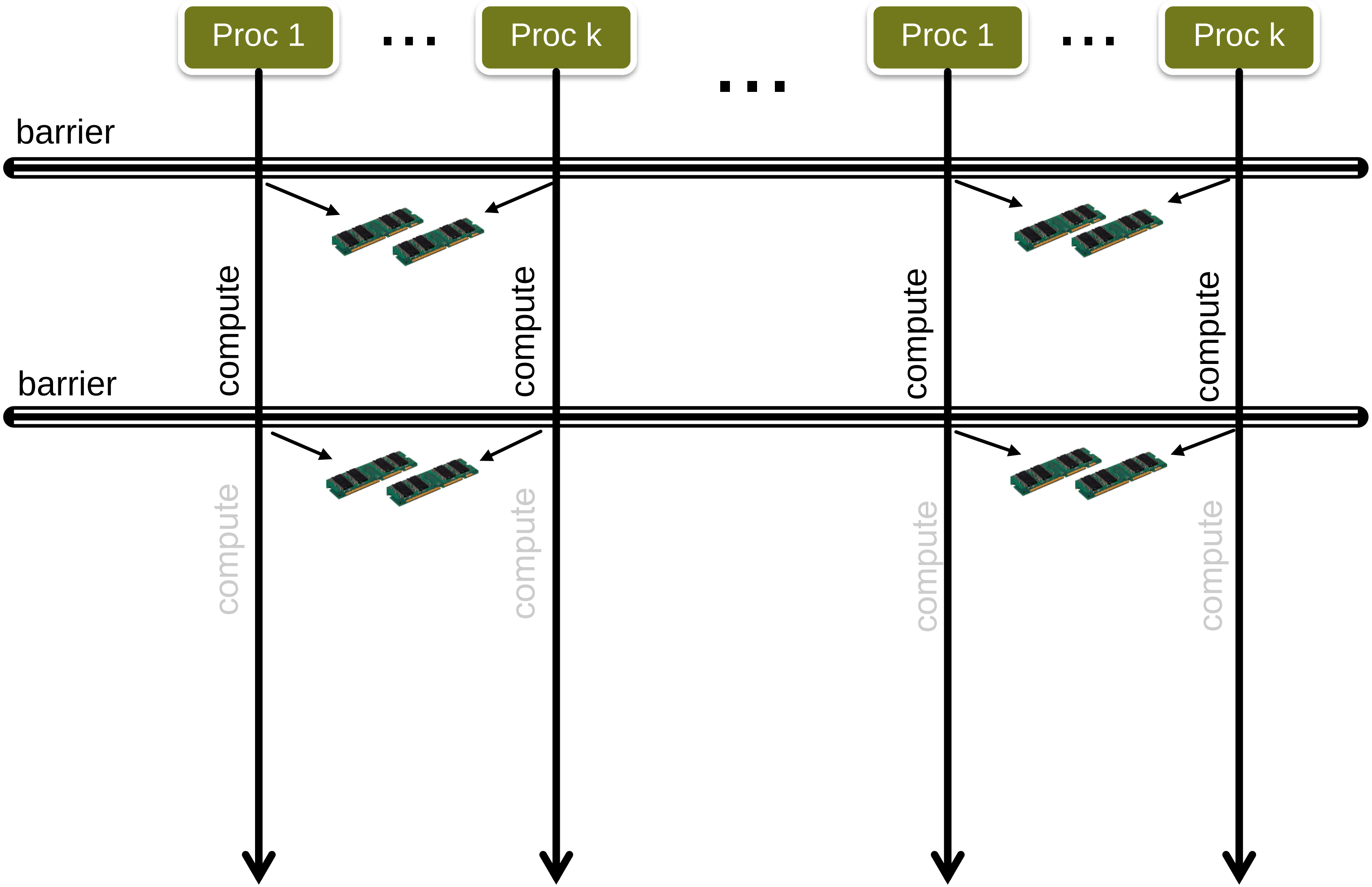
Coordinated Checkpointing (MP)



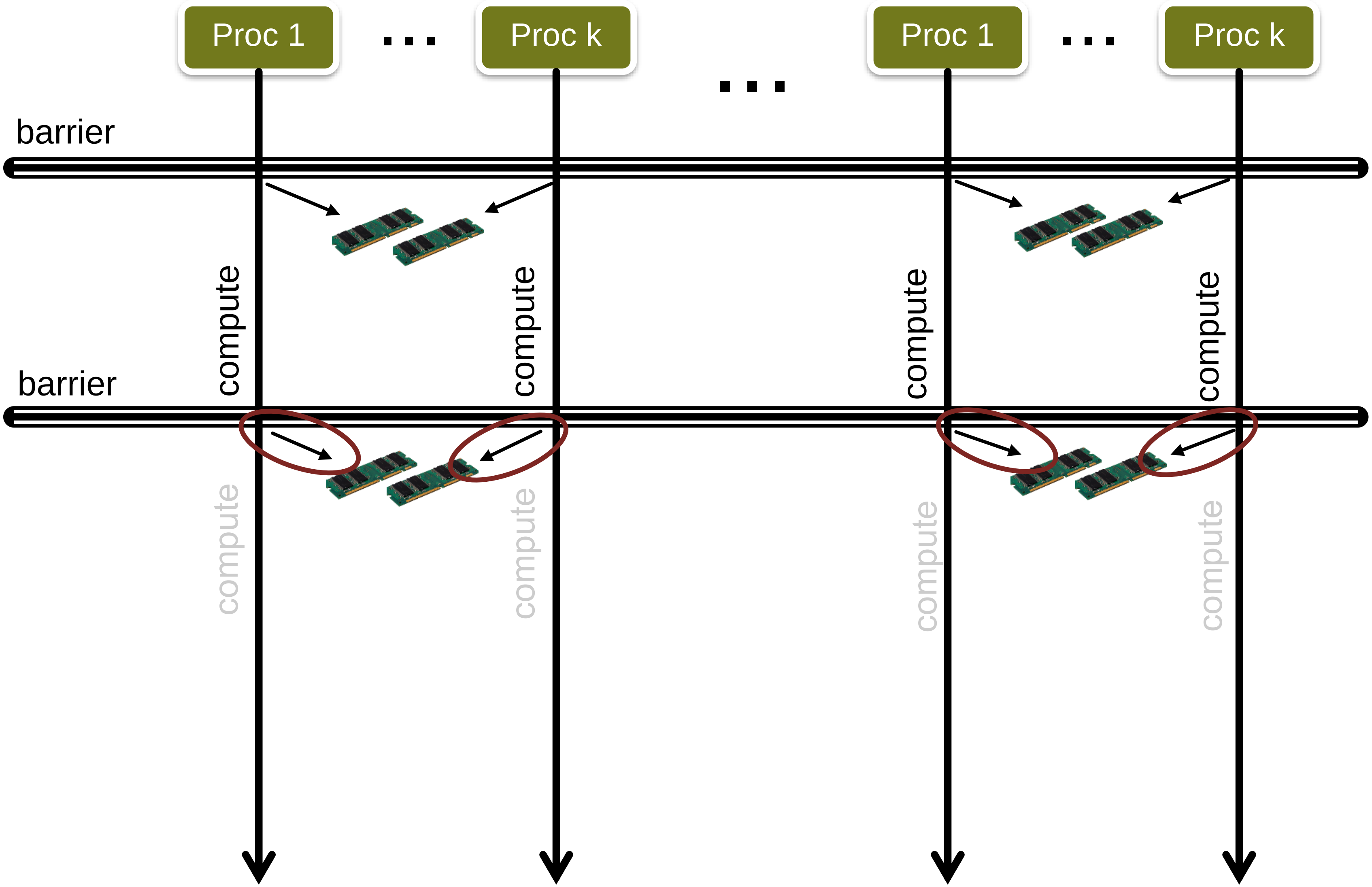
Coordinated Checkpointing (MP)



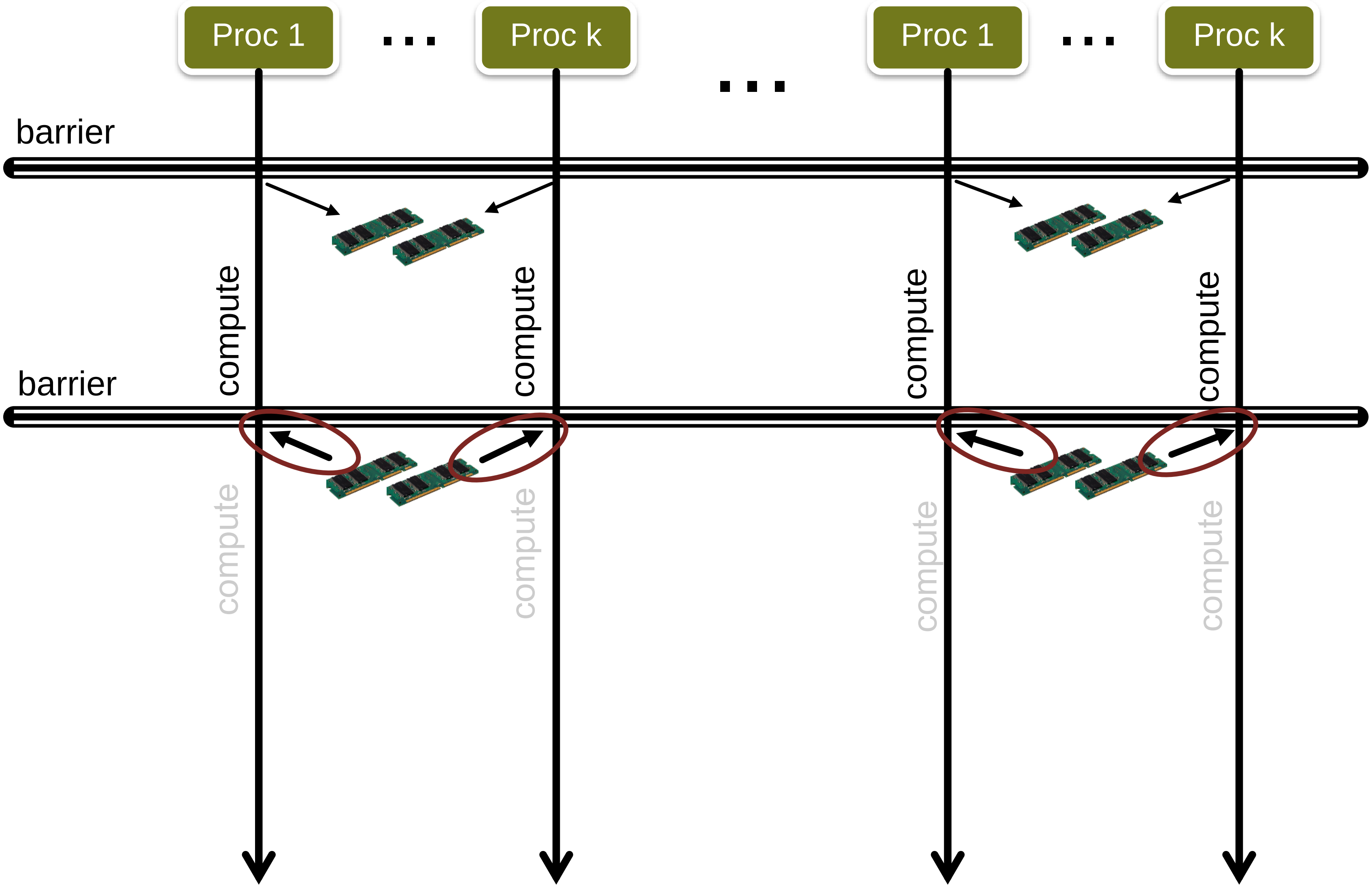
COORDINATED CHECKPOINTING (MP)



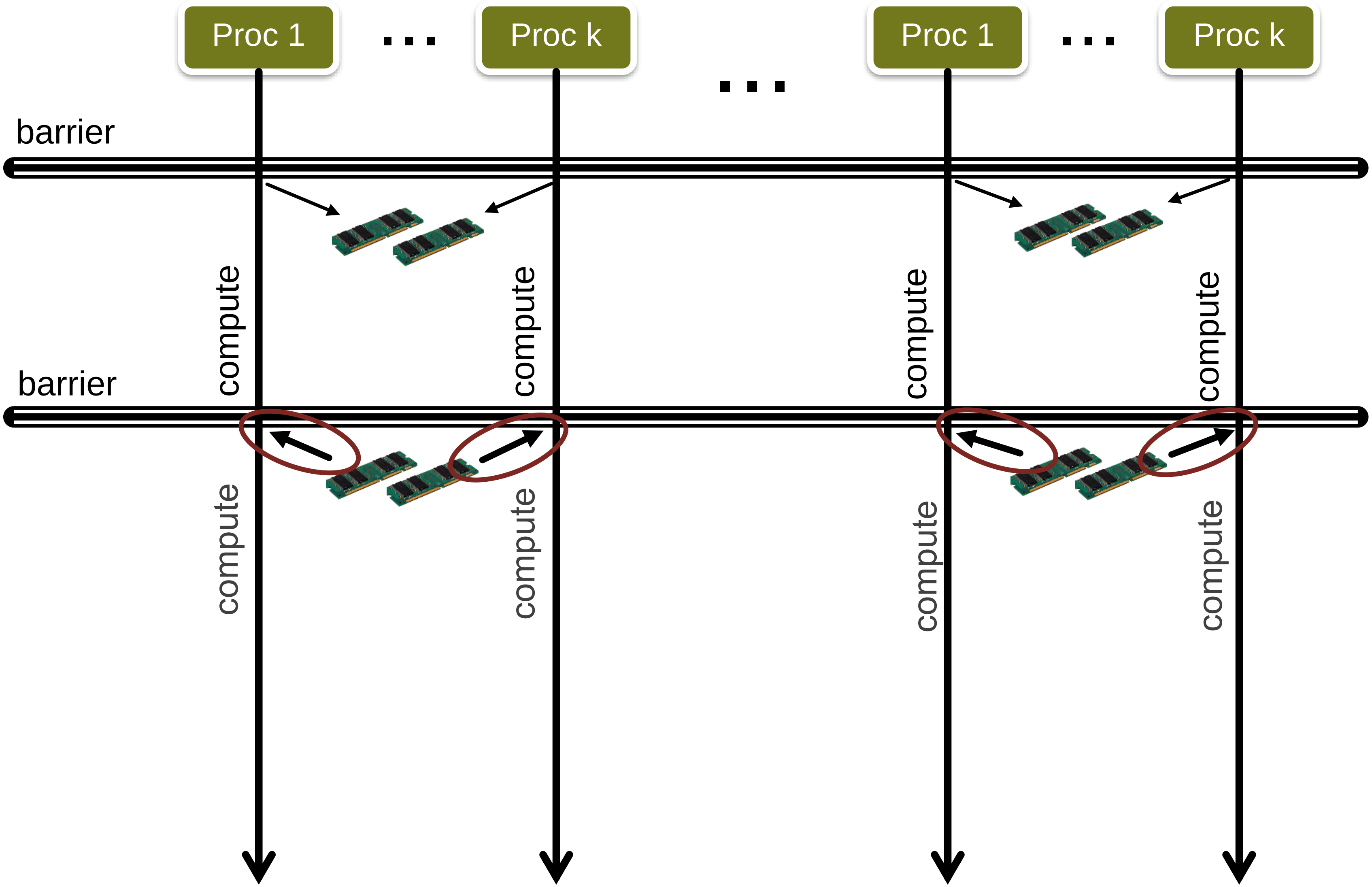
COORDINATED CHECKPOINTING (MP)



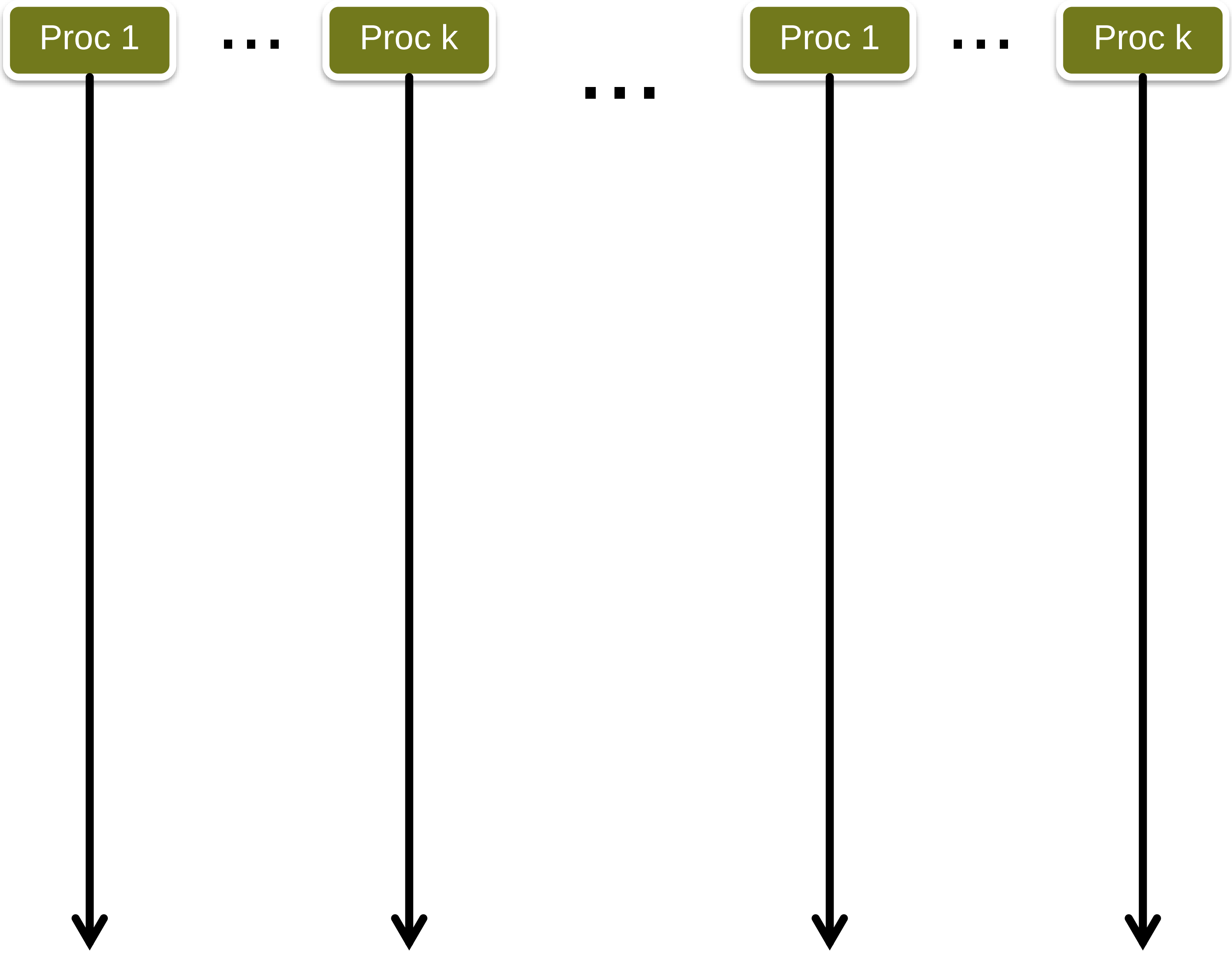
Coordinated Checkpointing (MP)



Coordinated Checkpointing (MP)



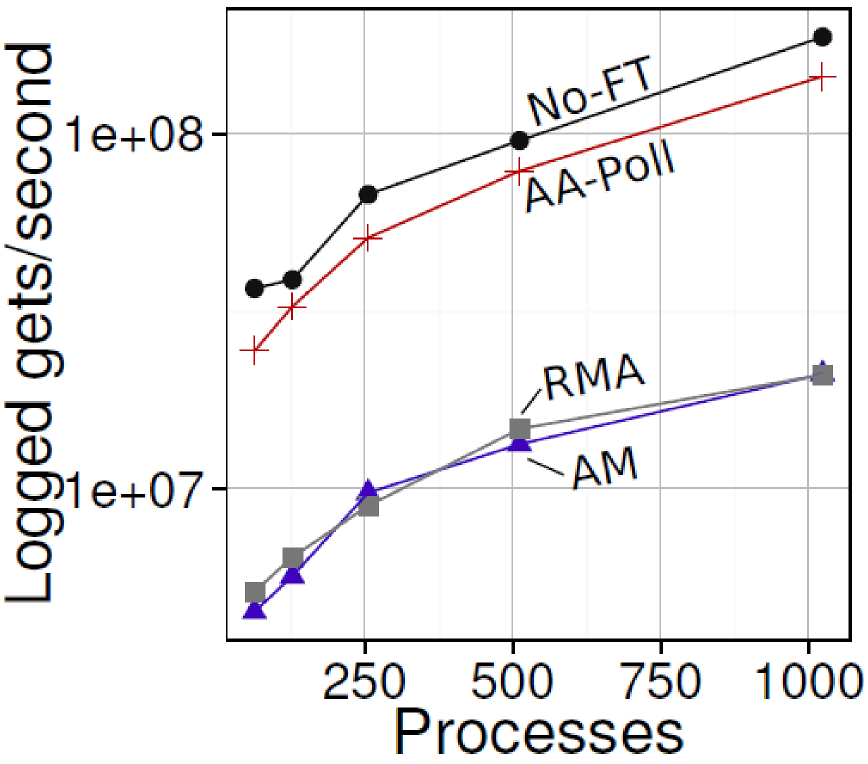
Coordinated Checkpointing (MP)



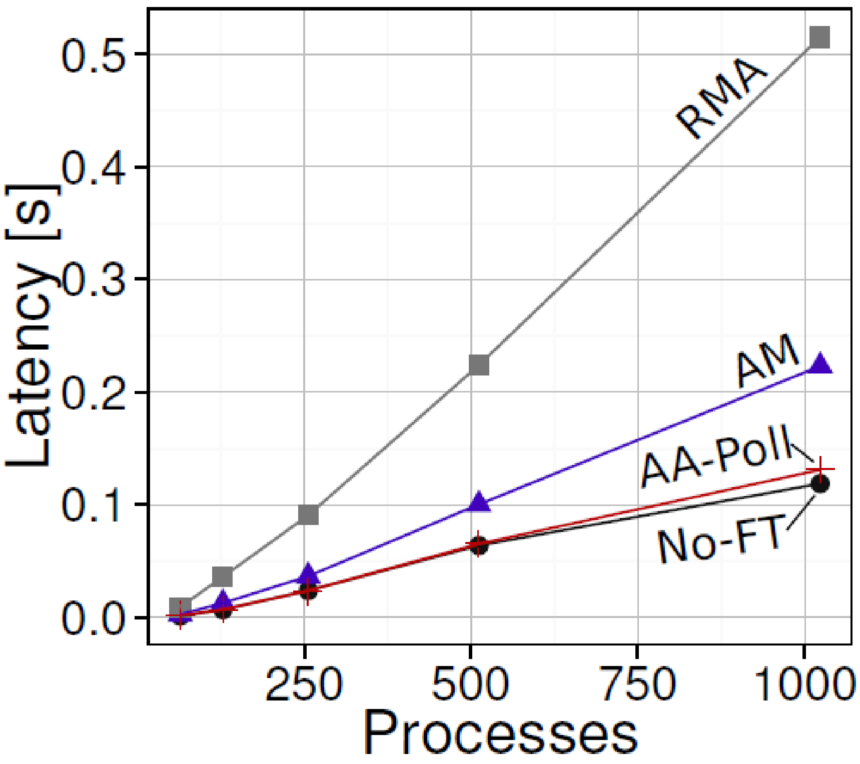
Performance: Large-Scale Codes

Fault Tolerance Scheme

Logging gets:



Sorting time:


Latency [s]
RMA
AM
AA-Poll
No-FT
Processes