Application Transparent Near-Memory Processing Architecture with Memory Channel Network

Mohammad Alian, Seung Won Min, Hadi Asgharimoghaddam, Ashutosh Dhar, Dong Kai Wang, Thomas Roewer, Adam McPadden, Oliver O'Halloran, Deming Chen, Jinjun Xiong, Daehoon Kim, Wen-mei Hwu, Nam Sung Kim

University of Illinois Urbana-Champaign

IBM Research and Systems

ILLINOIS

Electrical & Computer Engineering

COLLEGE OF ENGINEERING

C3SR center for cognitive computing systems research

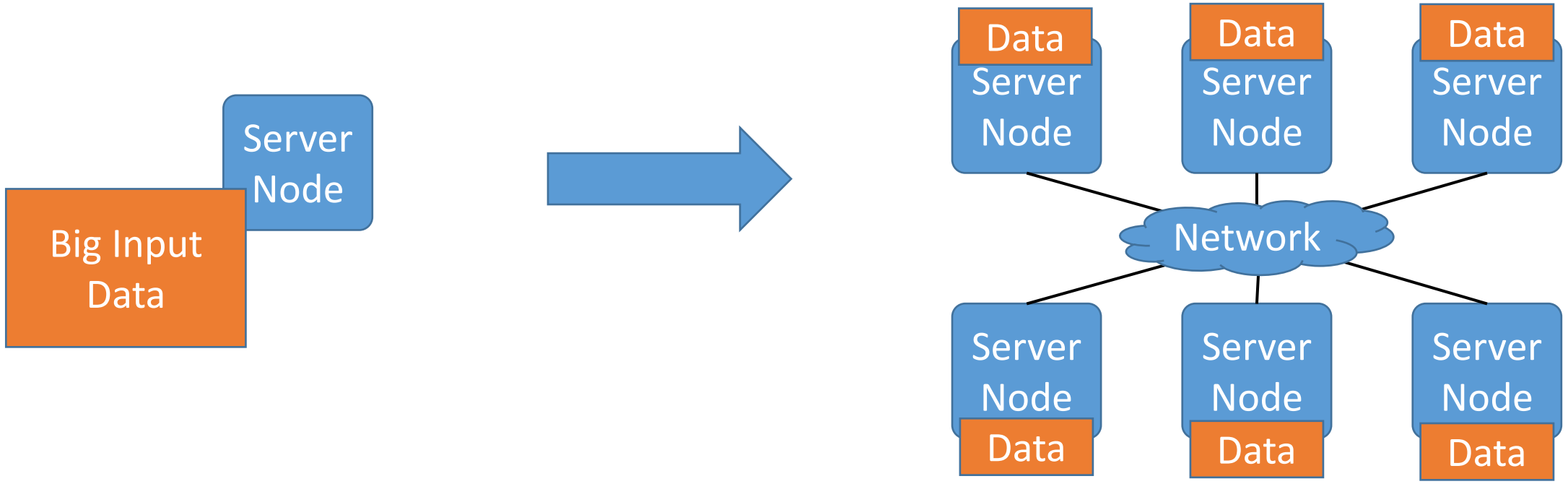
IBM | ILLINOIS

Executive Summary

- Near-memory processing concept is around for two decades
  - ✓EXECUBE'94, IRAM'97, ActivePages'98, FlexRAM'99, DIVA'99, SmartMemories'00, …
- **Question**: Why it is not commercialized yet?
  - ✓Significant **changes** is required in **applications** or **processor architecture**
  - ✓Industry resists to such changes
- **Solution**: Near memory processing using **Memory Channel Network (MCN)**
  - ✓**Robust**: no change required in the host processor architecture
  - ✓**Application-Transparent**: no change required in the application
  - ✓**Scalable**: seamless integration with distributed computing frameworks
- Implementation of the MCN HW/SW components on an experimental near-memory platform and a full-system simulator shows:
  - ✓Feasibility of the proposal
  - ✓Performance/power improvements: 8.17x higher aggregate DRAM bandwidth

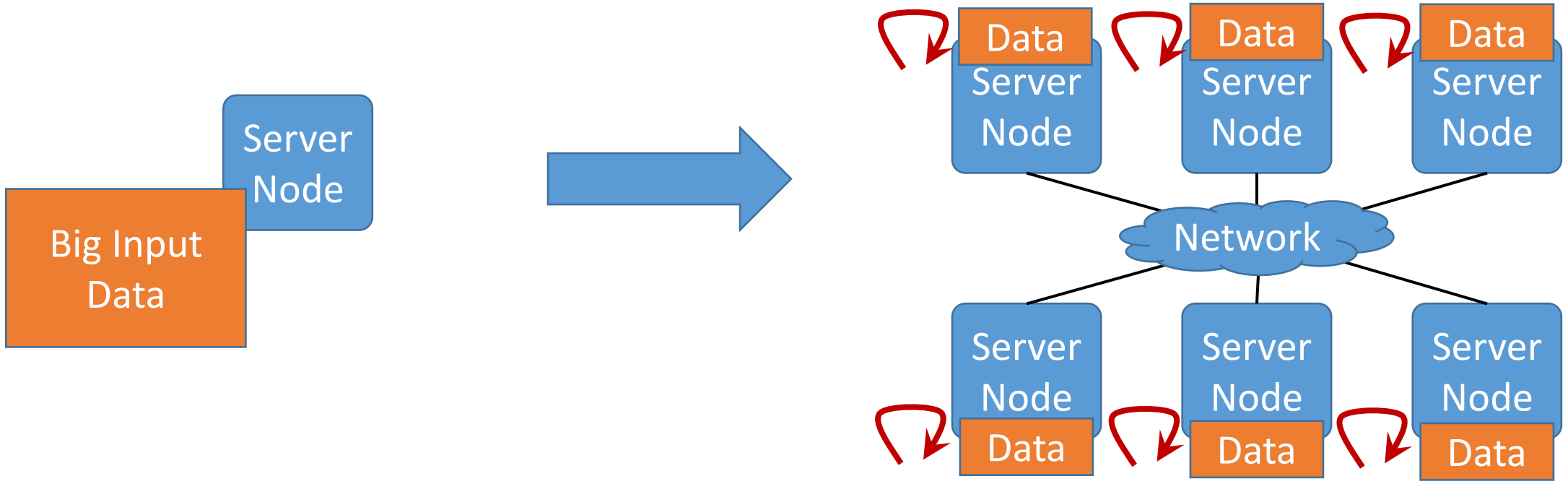
Big-Data Processing

- Big-Data processing model: **single node** vs. **distributed computing**



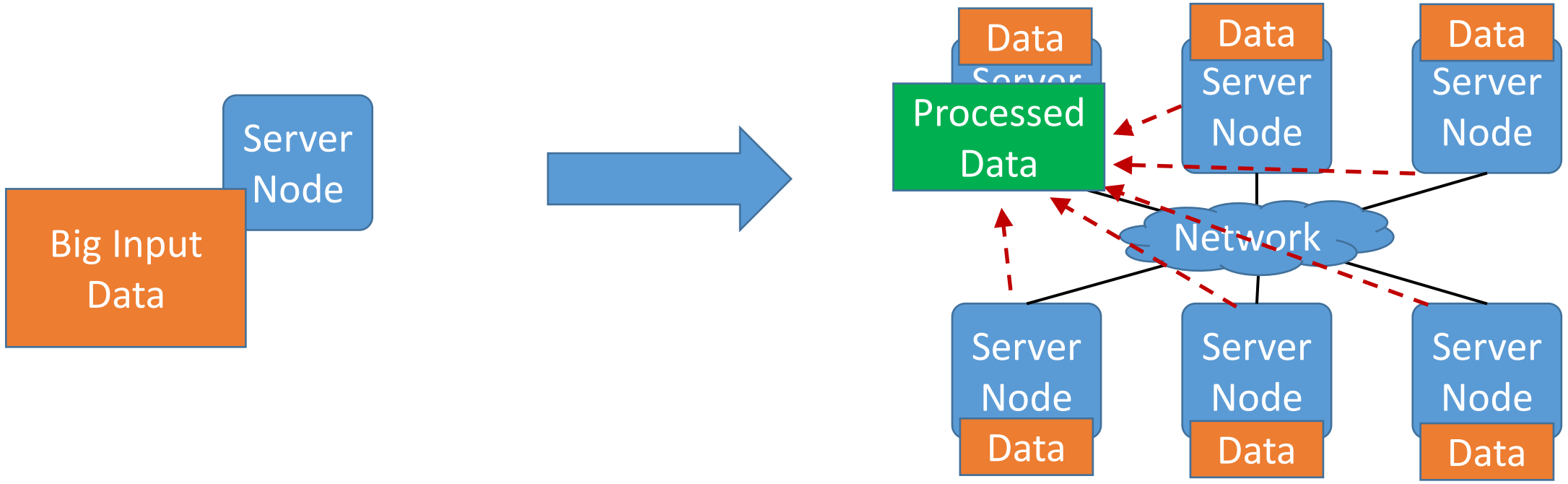
Big-Data Processing

- Big-Data processing model: **single node** vs. **distributed computing**



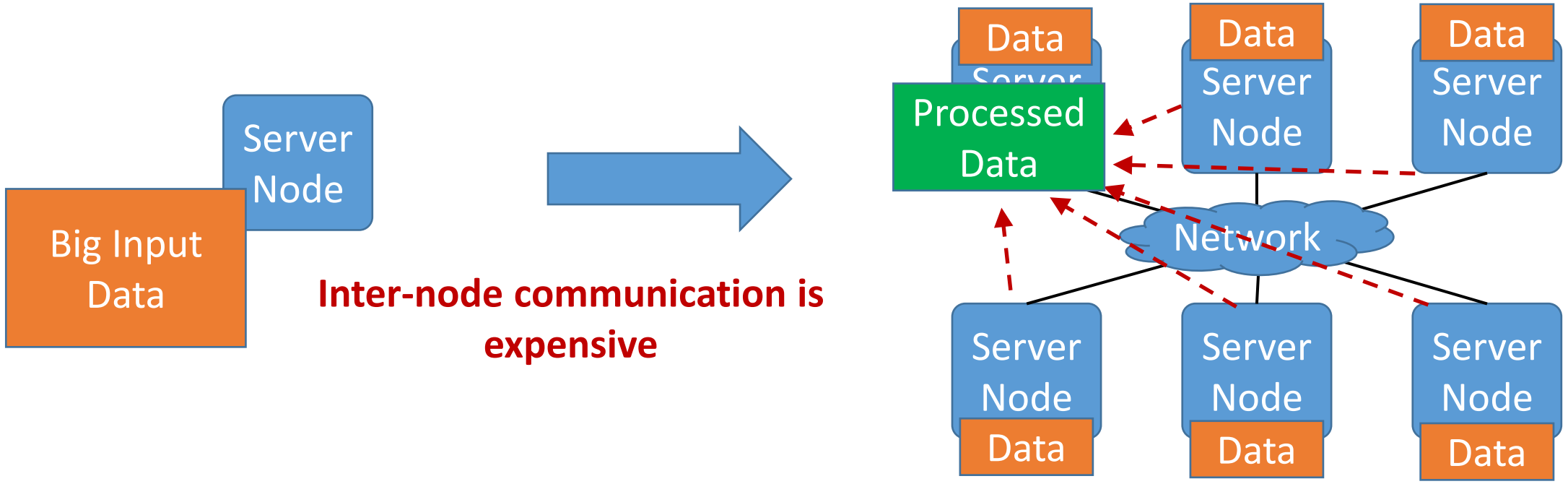
Big-Data Processing

- Big-Data processing model: **single node** vs. **distributed computing**



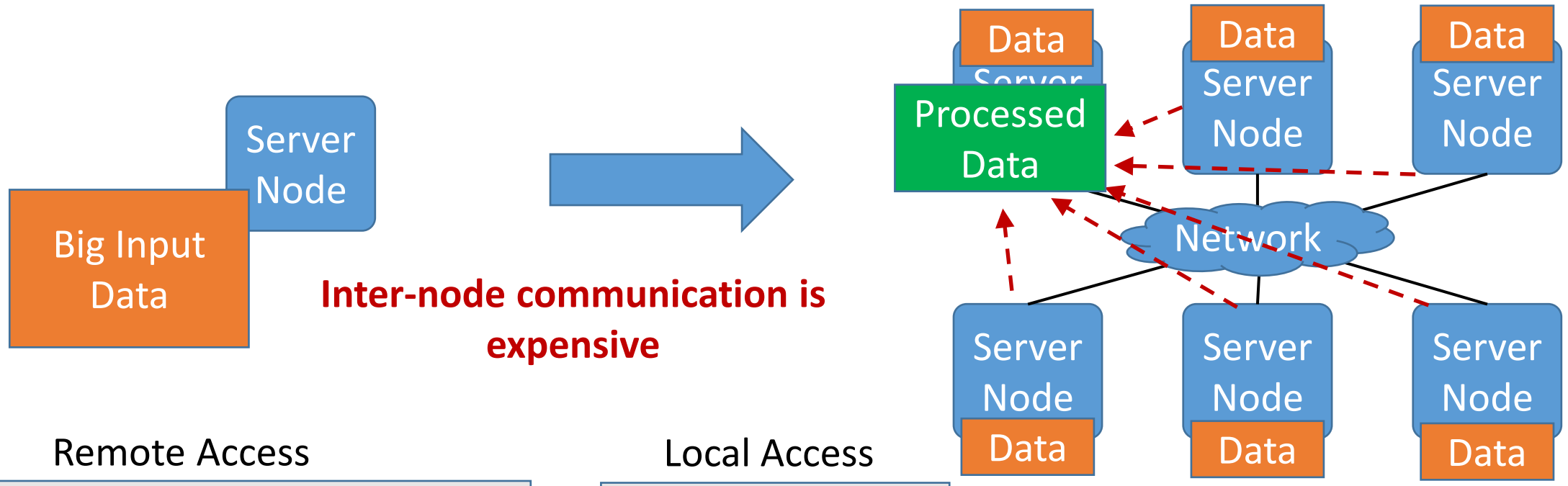
Big-Data Processing

- Big-Data processing model: **single node** vs. **distributed computing**



Big-Data Processing

- Big-Data processing model: **single node** vs. **distributed computing**

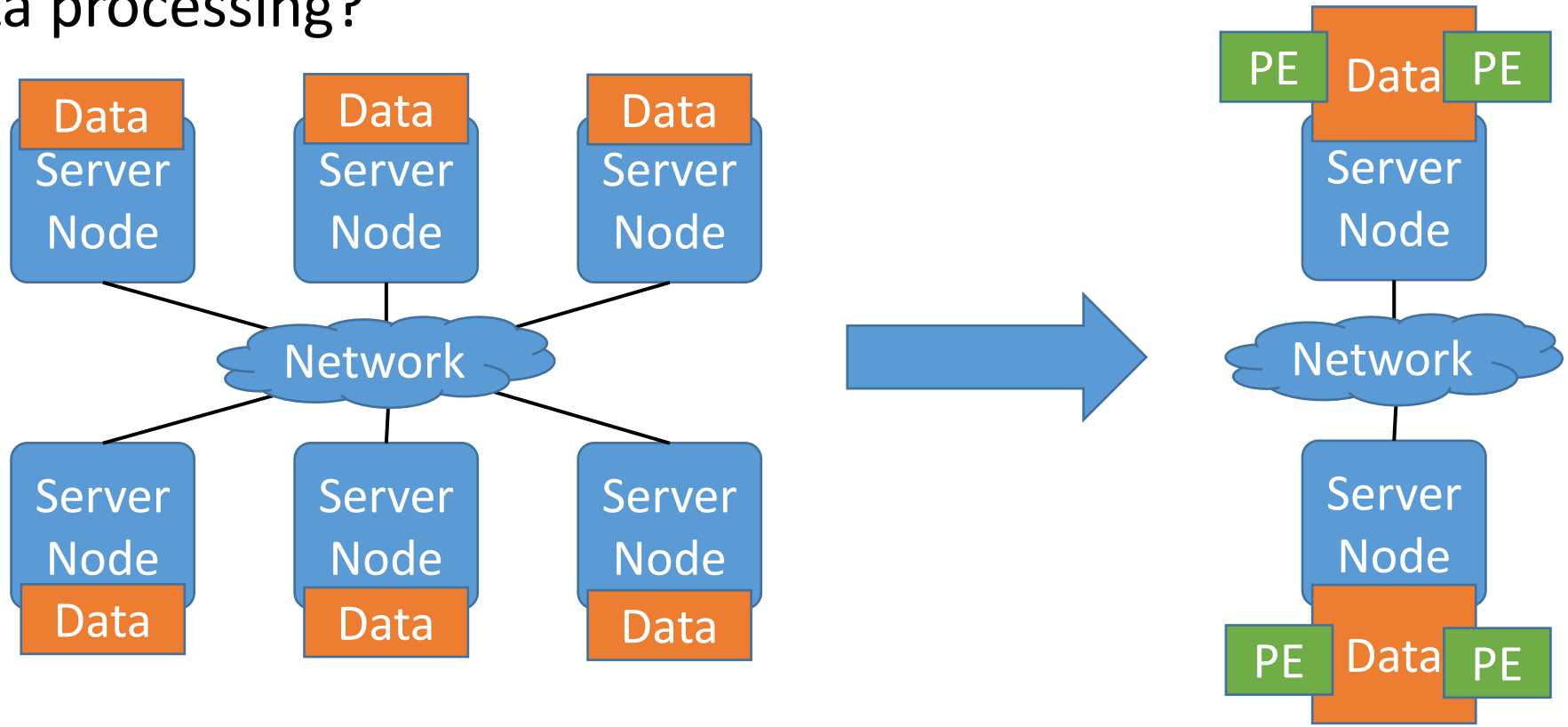


How to reduce inter-node communication?

- Near-data processing?

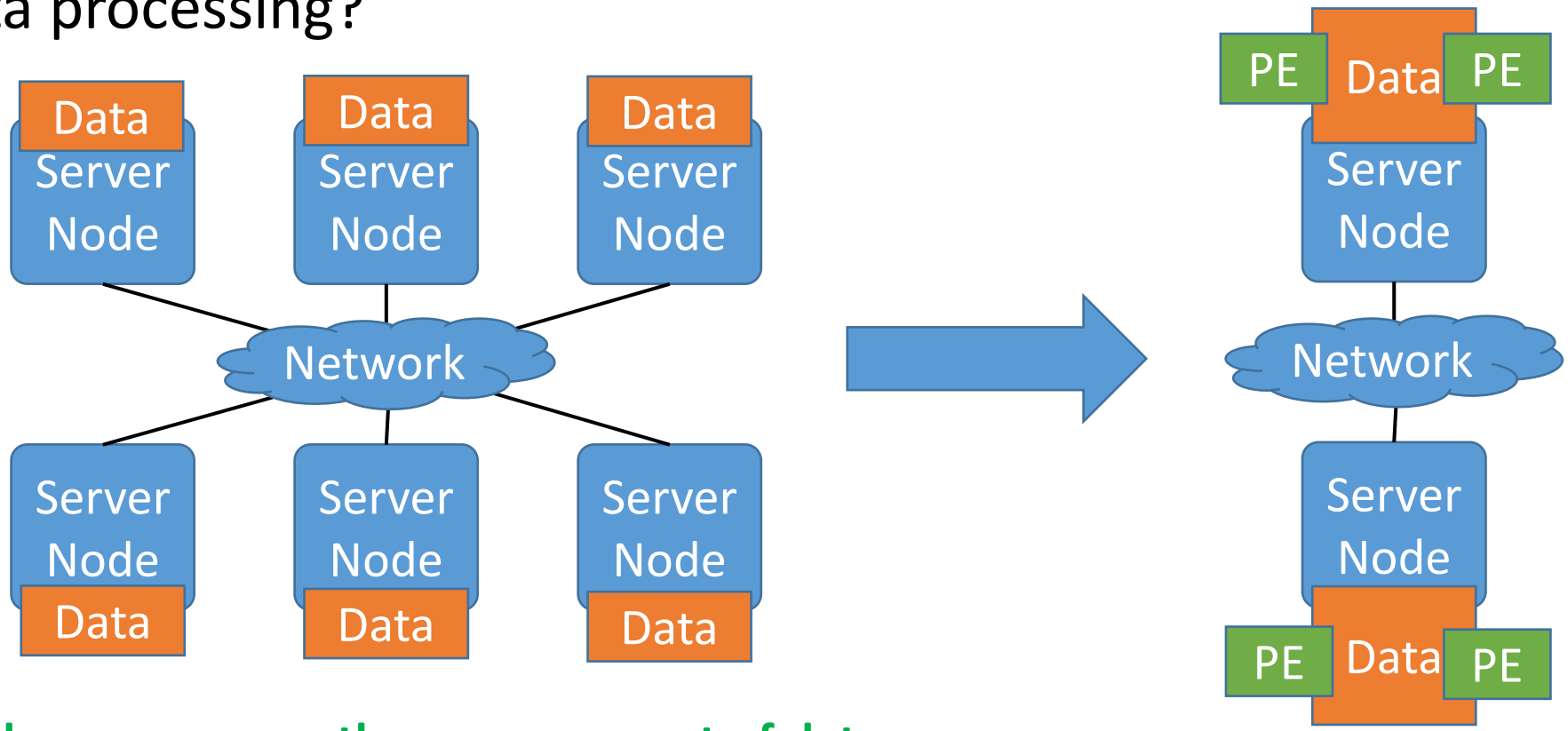
How to reduce inter-node communication?

- Near-data processing?



How to reduce inter-node communication?

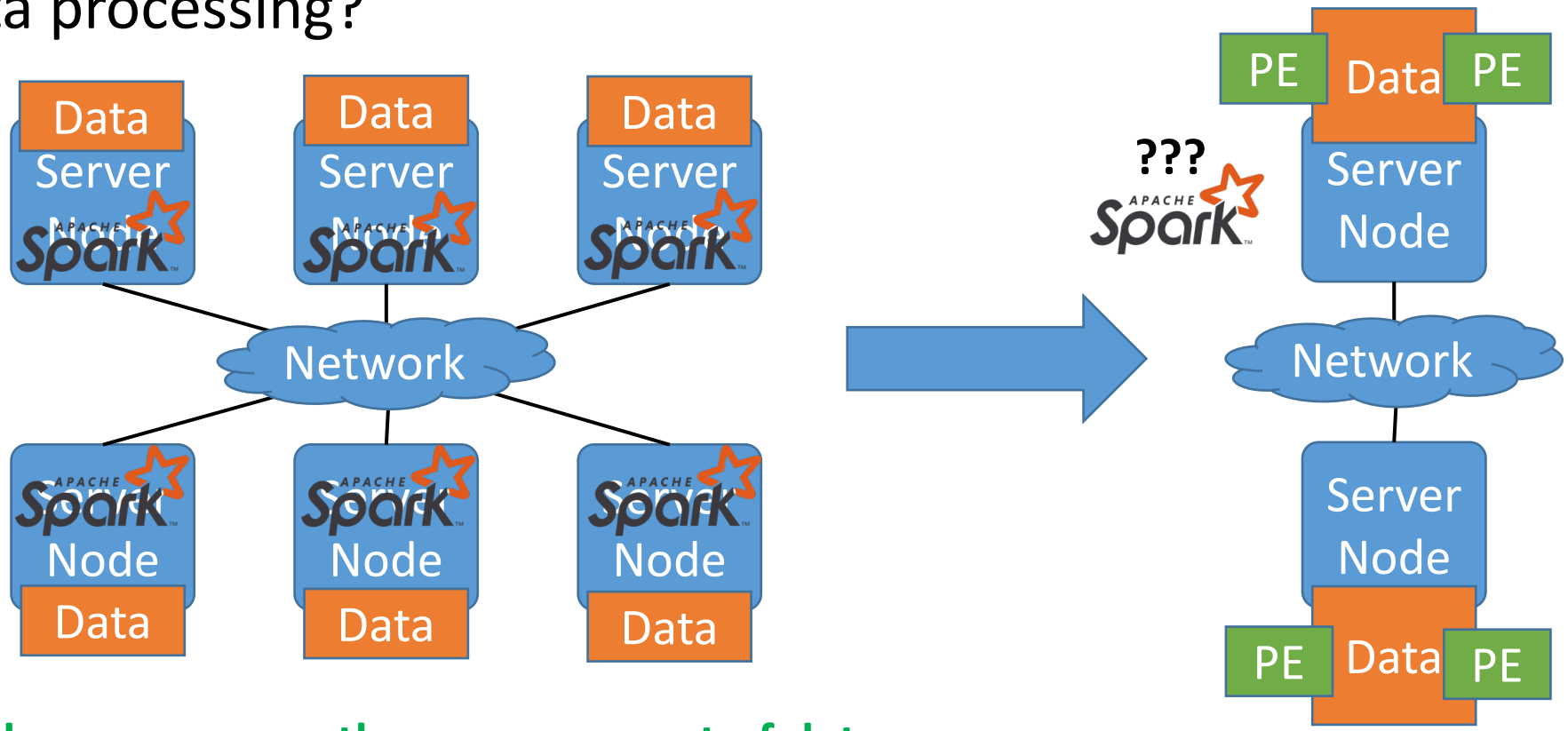
- Near-data processing?



✓ **Lesser nodes can process the same amount of data**

How to reduce inter-node communication?

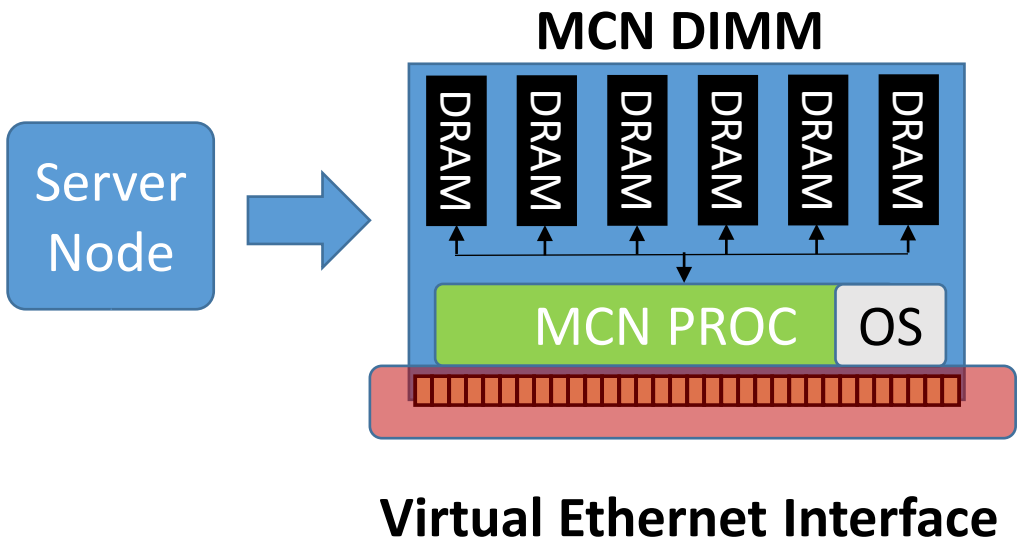
- Near-data processing?



✓ **Lesser nodes can process the same amount of data**

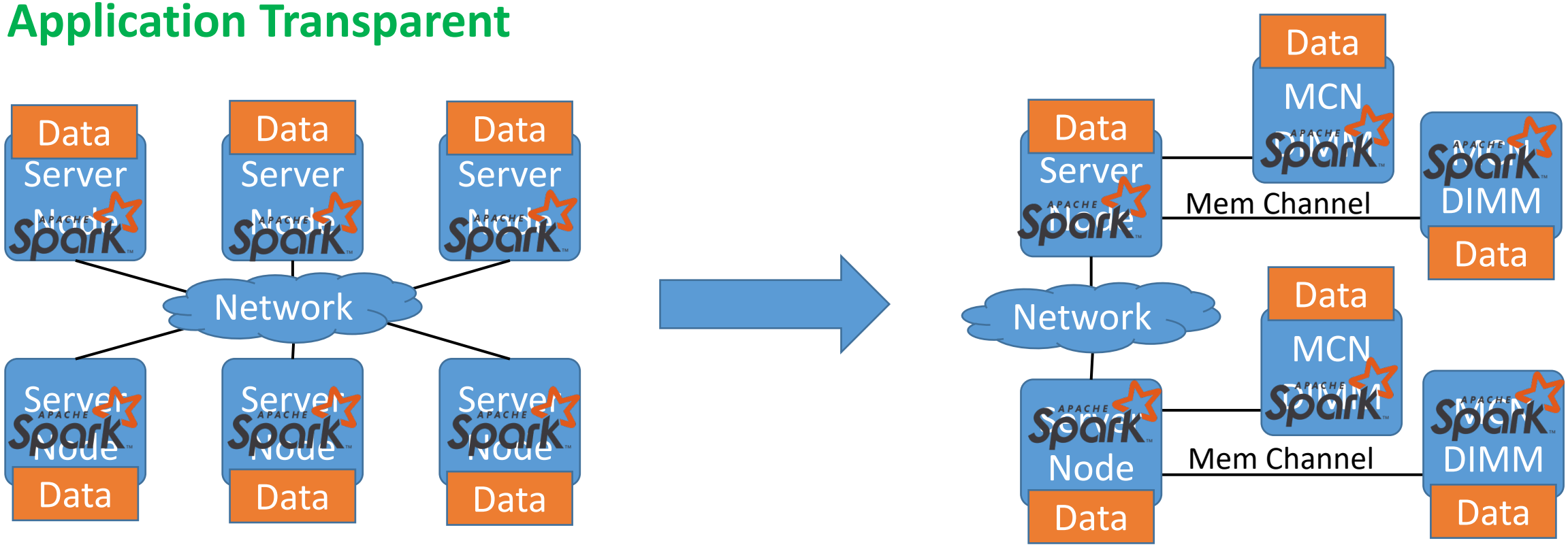
× **Requires application modification**

Memory Channel Network



Memory Channel Network

- Unify **near-memory processing** within a server and **distributed computing** across servers
- **Application Transparent**



Outline

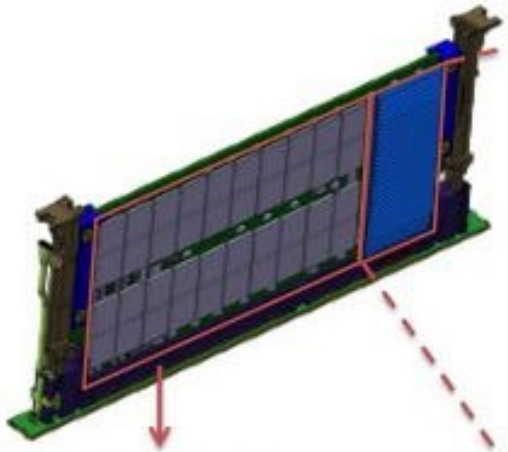
- **MCN Architecture**
  - ✓Background – Buffered DIMM
  - ✓Design Overview
  - ✓MCN DIMM Architecture
  - ✓MCN Packet Routing
- **Proof of Concept – Hardware Demonstration**
- **Design Optimizations**
- **Evaluation**
  - ✓Simulation Methodology
  - ✓Bandwidth and Latency
- **Conclusion**

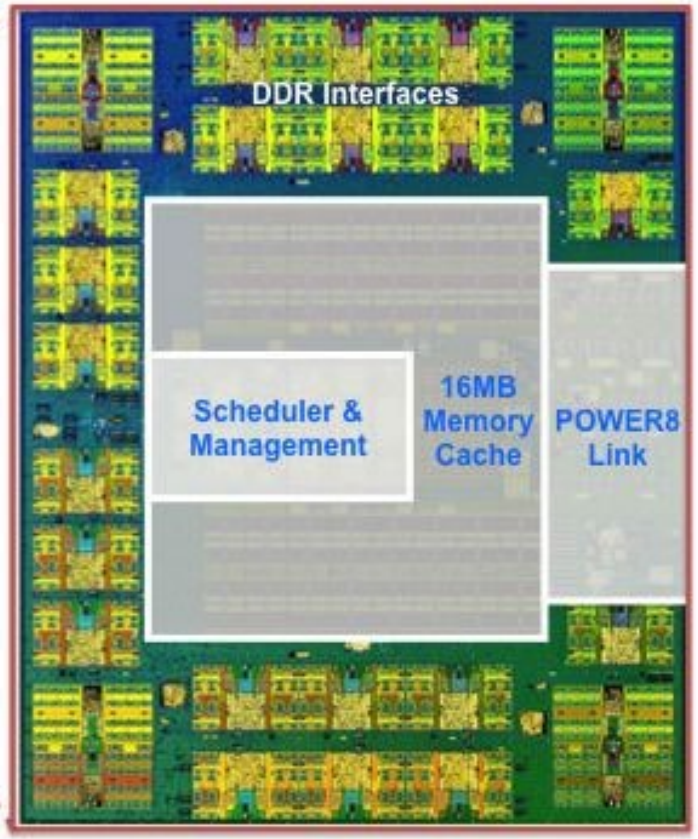
Outline

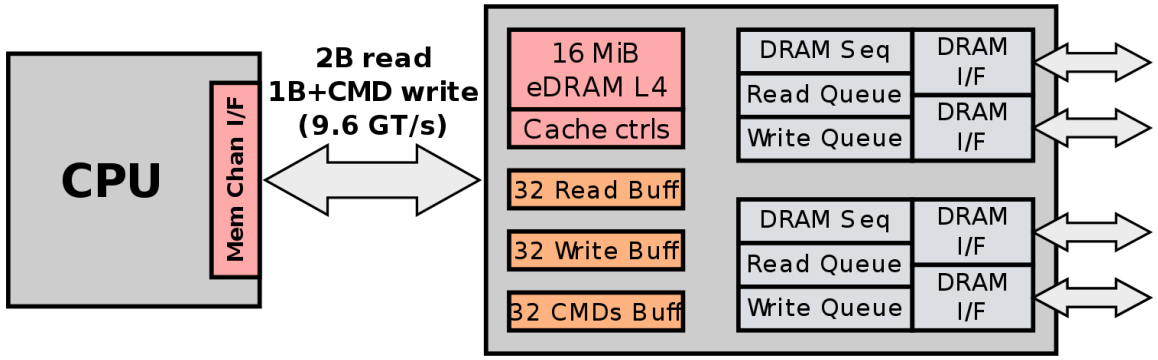
- **MCN Architecture**
  - ✓Background – Buffered DIMM
  - ✓Design Overview
  - ✓MCN DIMM Architecture
  - ✓MCN Packet Routing
- **Proof of Concept – Hardware Demonstration**
- **Design Optimizations**
- **Evaluation**
  - ✓Simulation Methodology
  - ✓Bandwidth and Latency
- **Conclusion**

Background: Buffered DRAM Module

- Employ a buffer per DIMM to reduce capacitive load
  - ✓Centaur DIMM
- 16MB eDRAM, L4 cache
- Memory management logic
- DDR-DMI interfaces



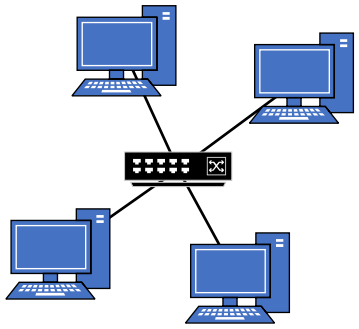




https://en.wikichip.org/wiki/ibm/centaur

MCN Design Overview

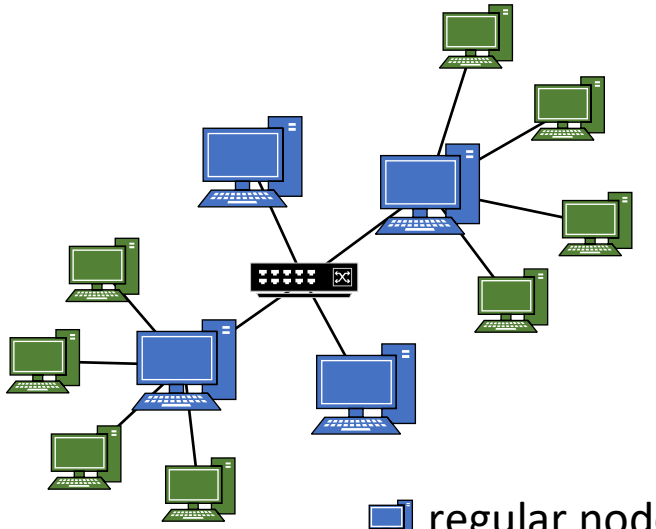
conventional distributed computing



regular node

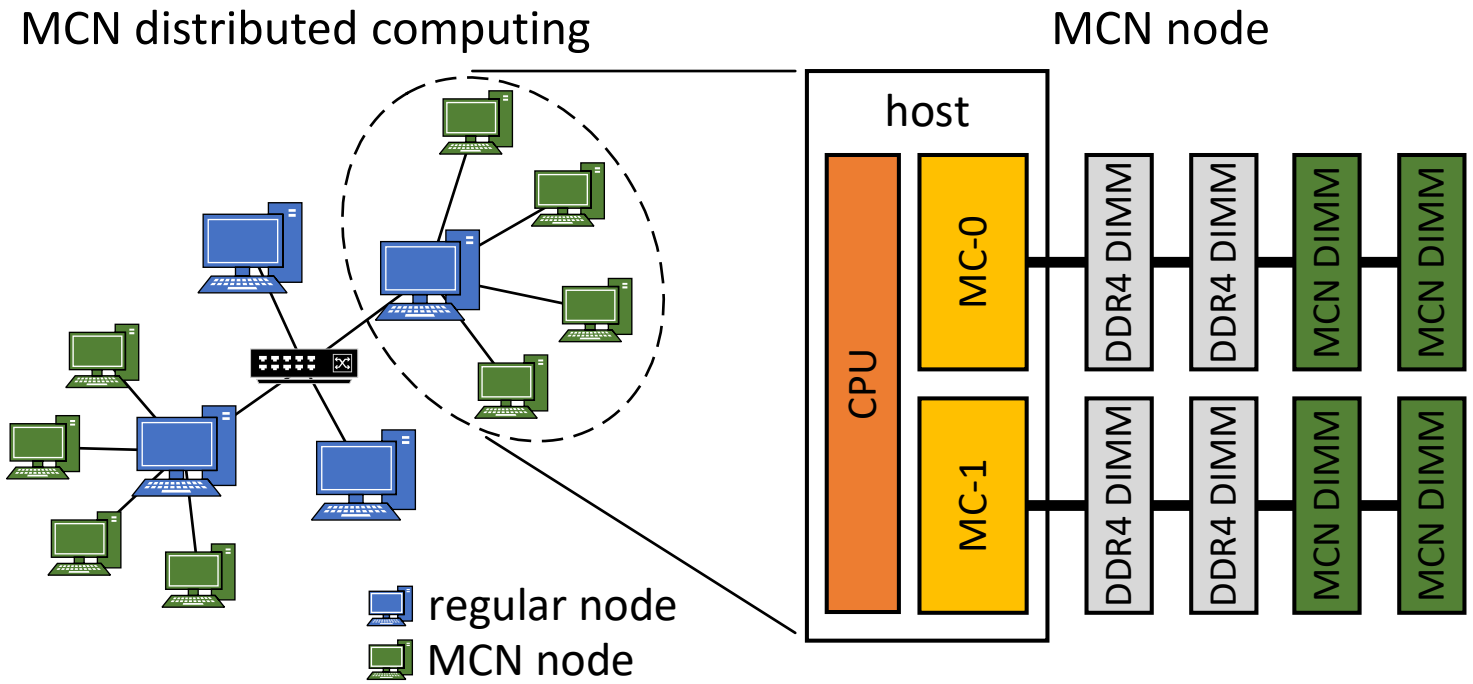
MCN Design Overview

MCN distributed computing

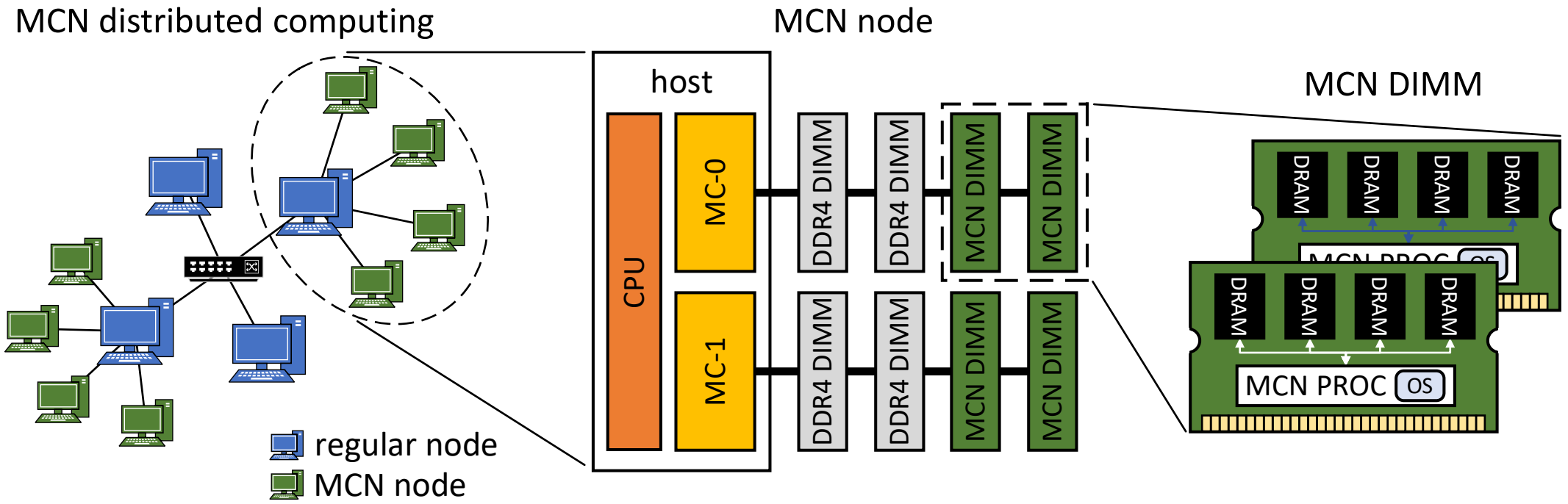


regular node
MCN node

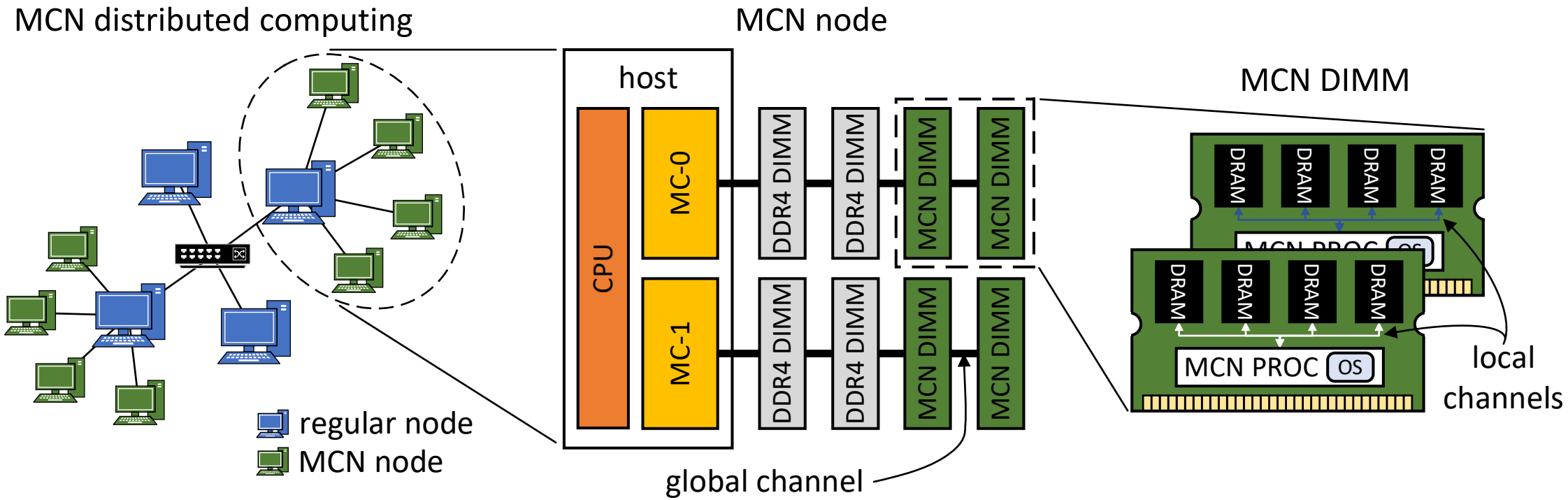
MCN Design Overview



MCN Design Overview

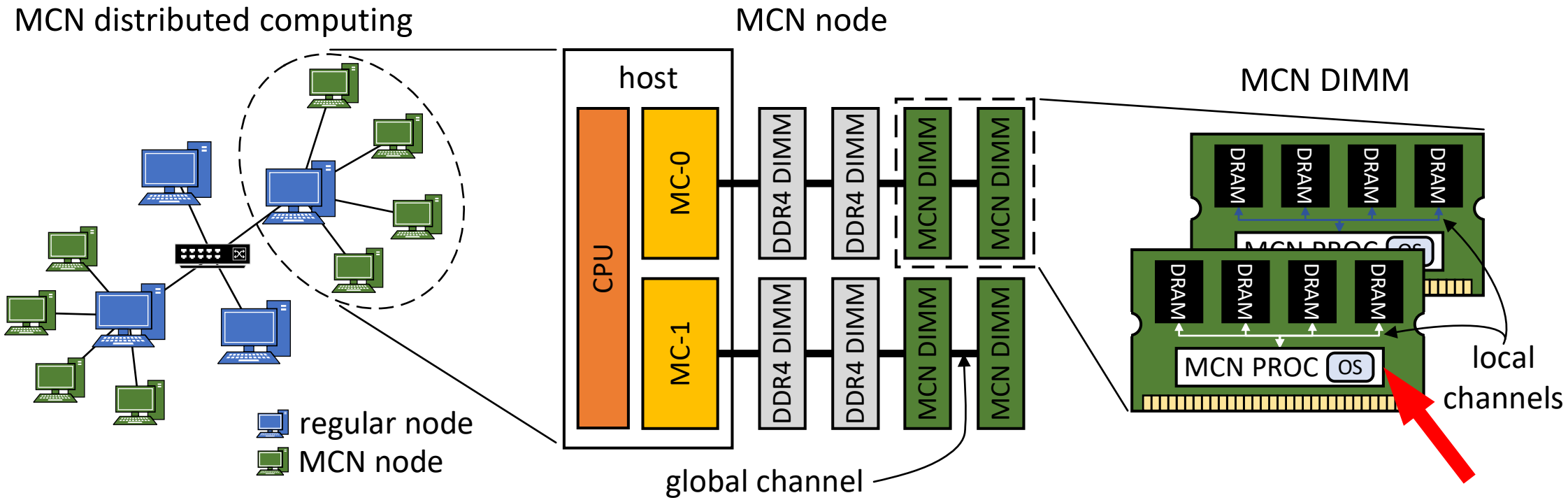


MCN Design Overview



- Individual local channels can be accessed in parallel by each MCN processor

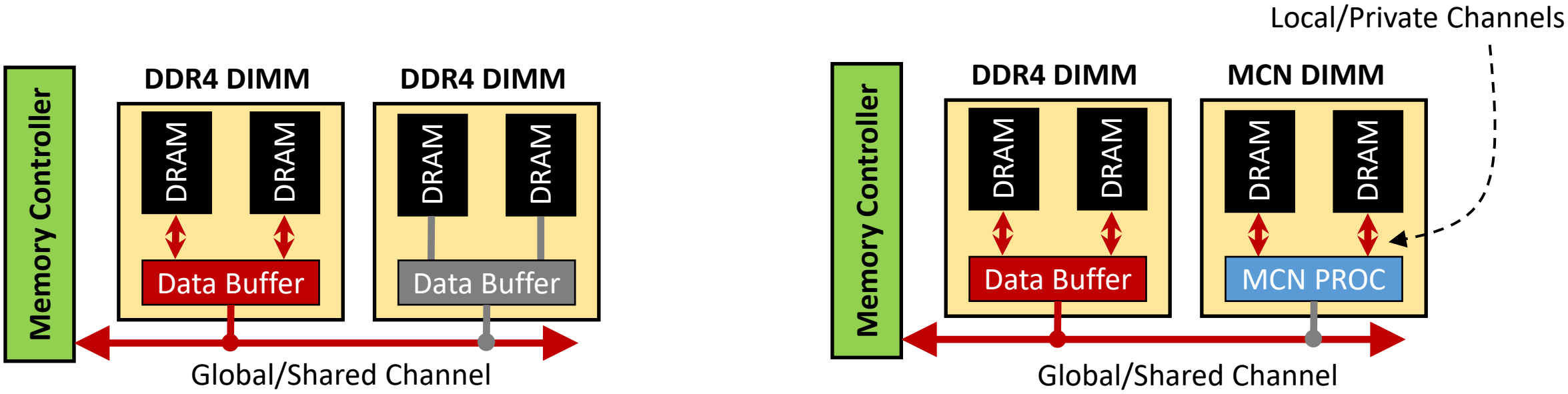
MCN Design Overview



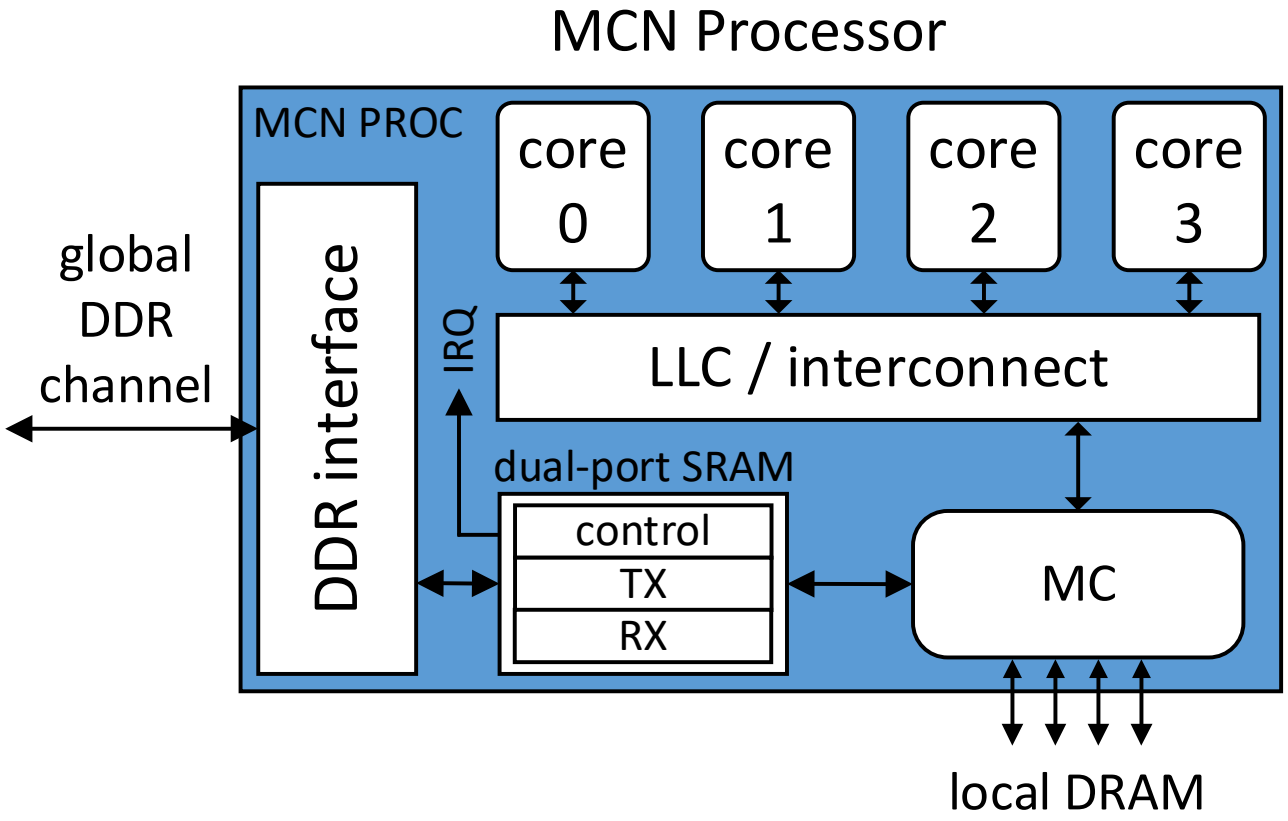
- Individual local channels can be accessed in parallel by each MCN processor

Transcending Limitation of DIMMs

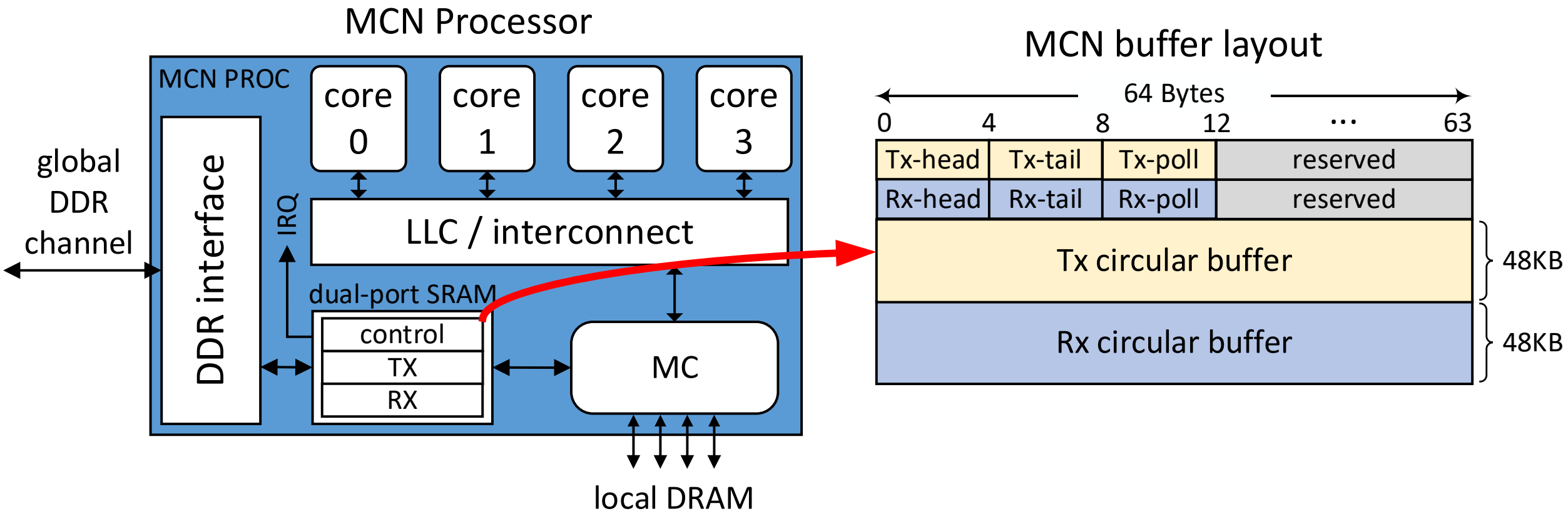
- DRAM device connected to MCN processor is electrically disconnected from global/shared memory channel
  - ✓Aggregate memory bandwidth scales with #MCN DIMMs



MCN DIMM Architecture

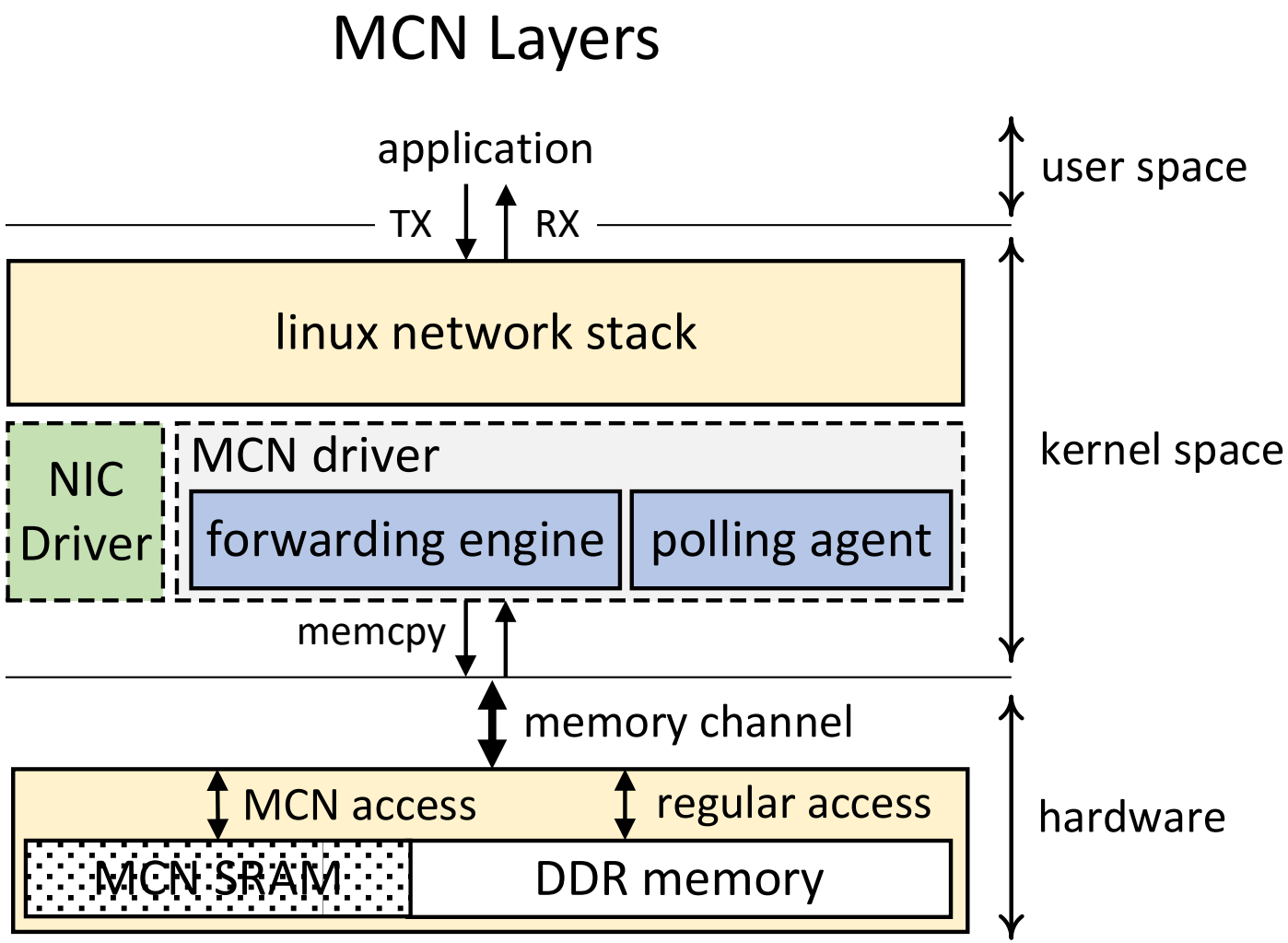


MCN DIMM Architecture



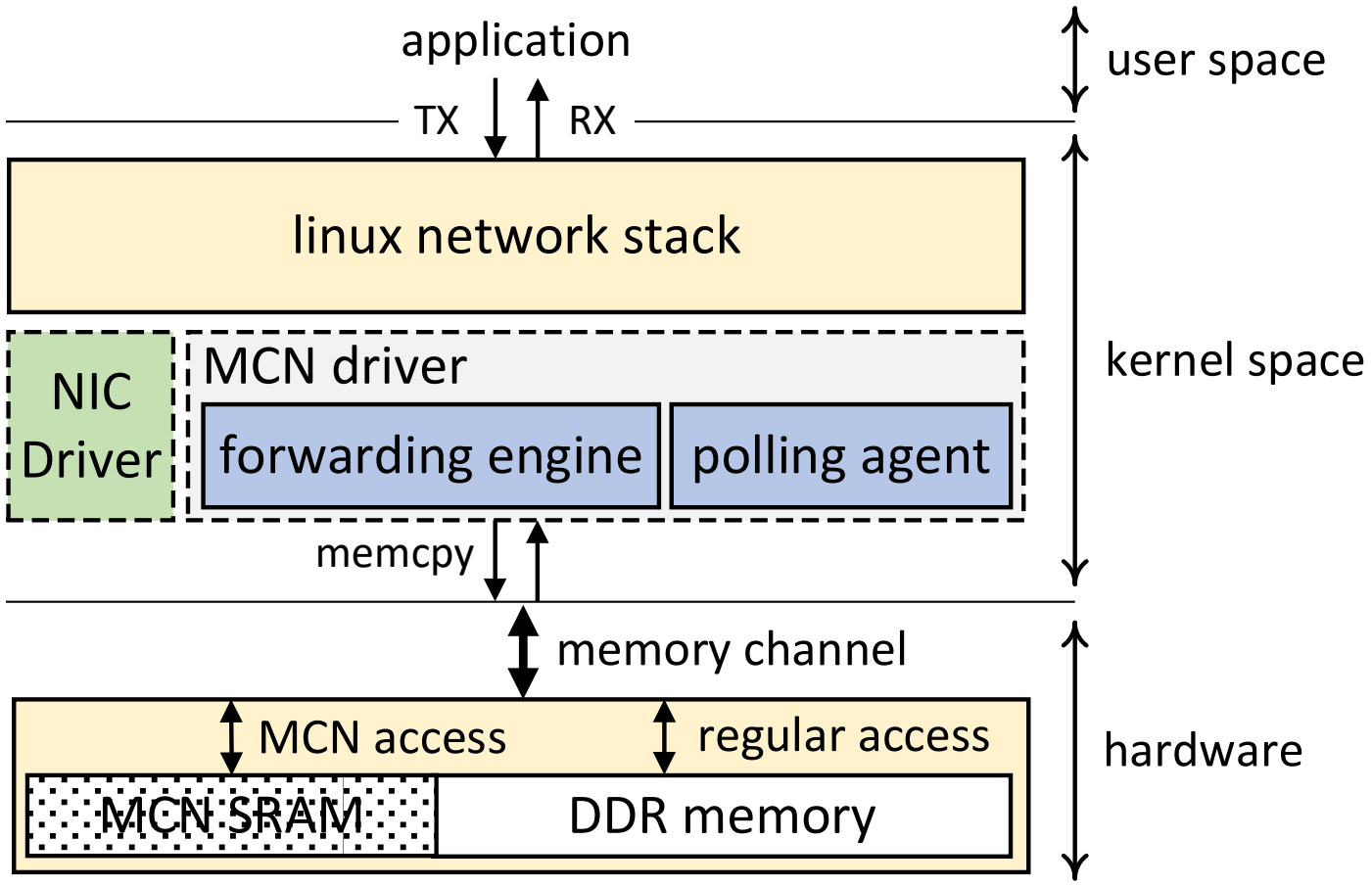
MCN Hardware/Software Architecture

MCN Hardware/Software Architecture

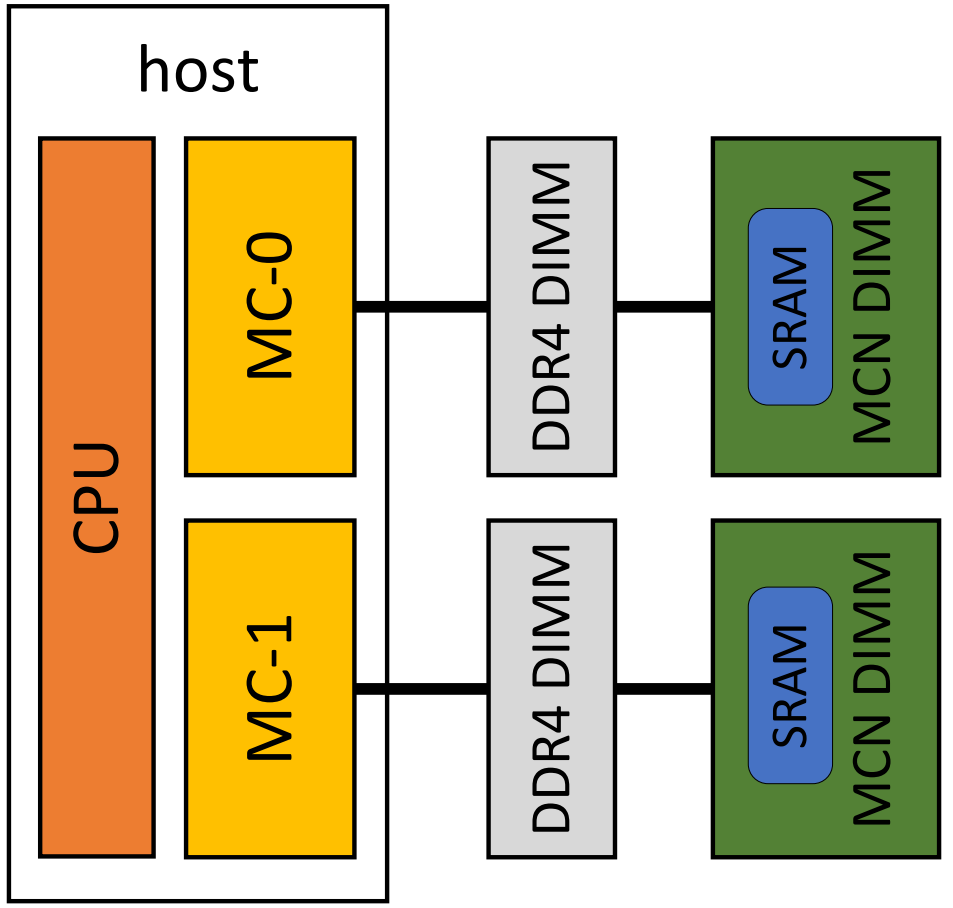


MCN Hardware/Software Architecture

MCN Layers

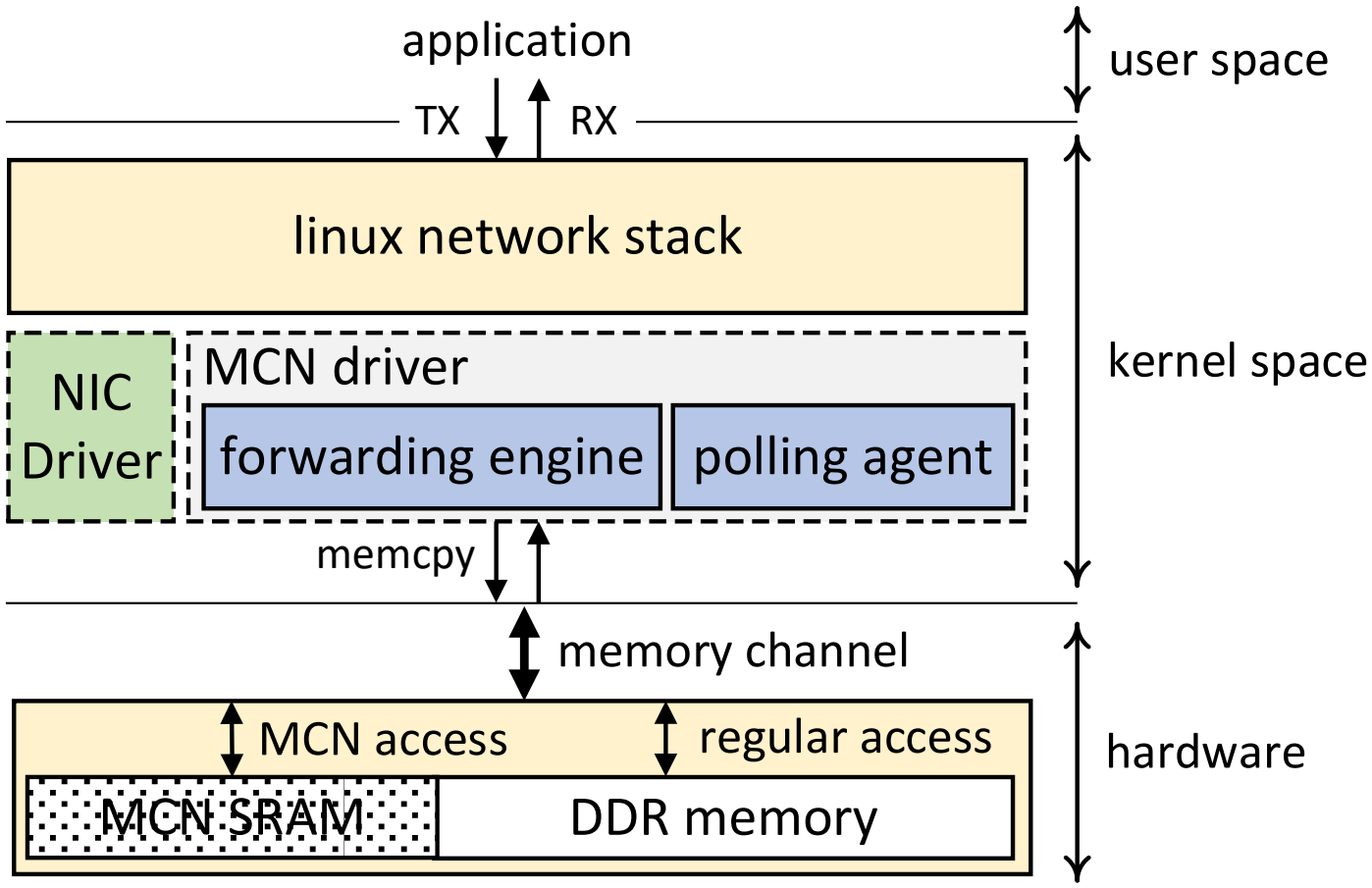


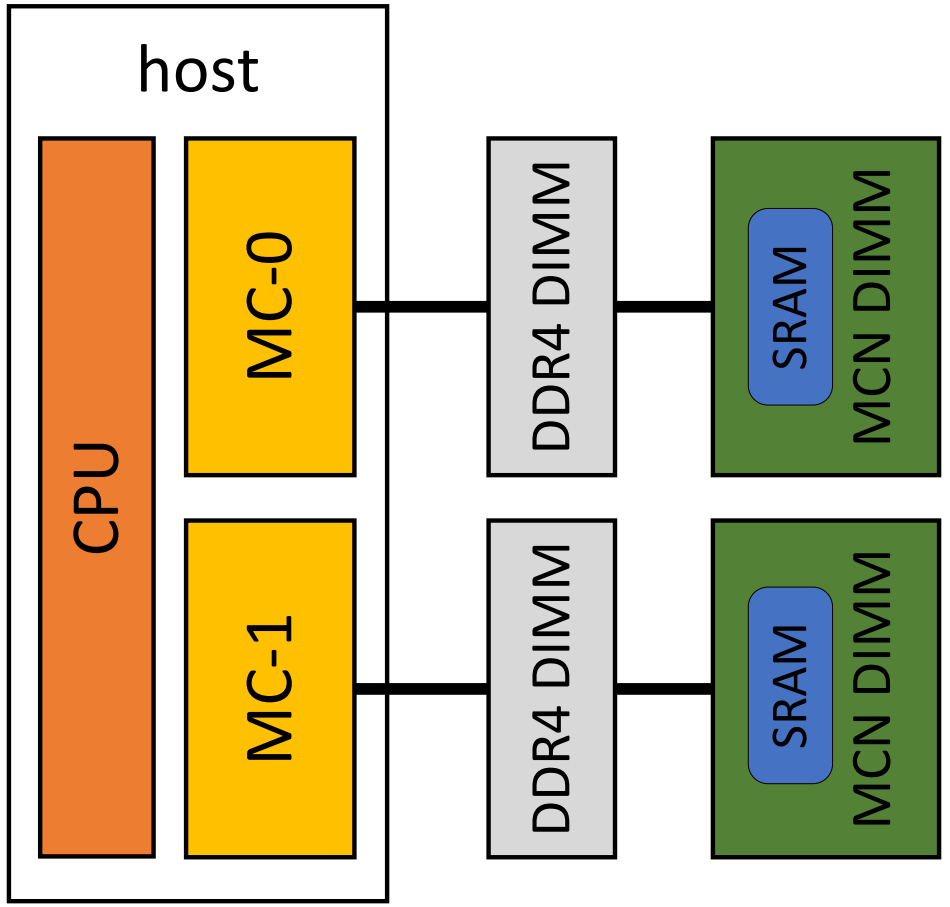
MCN Hardware Configuration



MCN Packet Routing

- MCN → Host

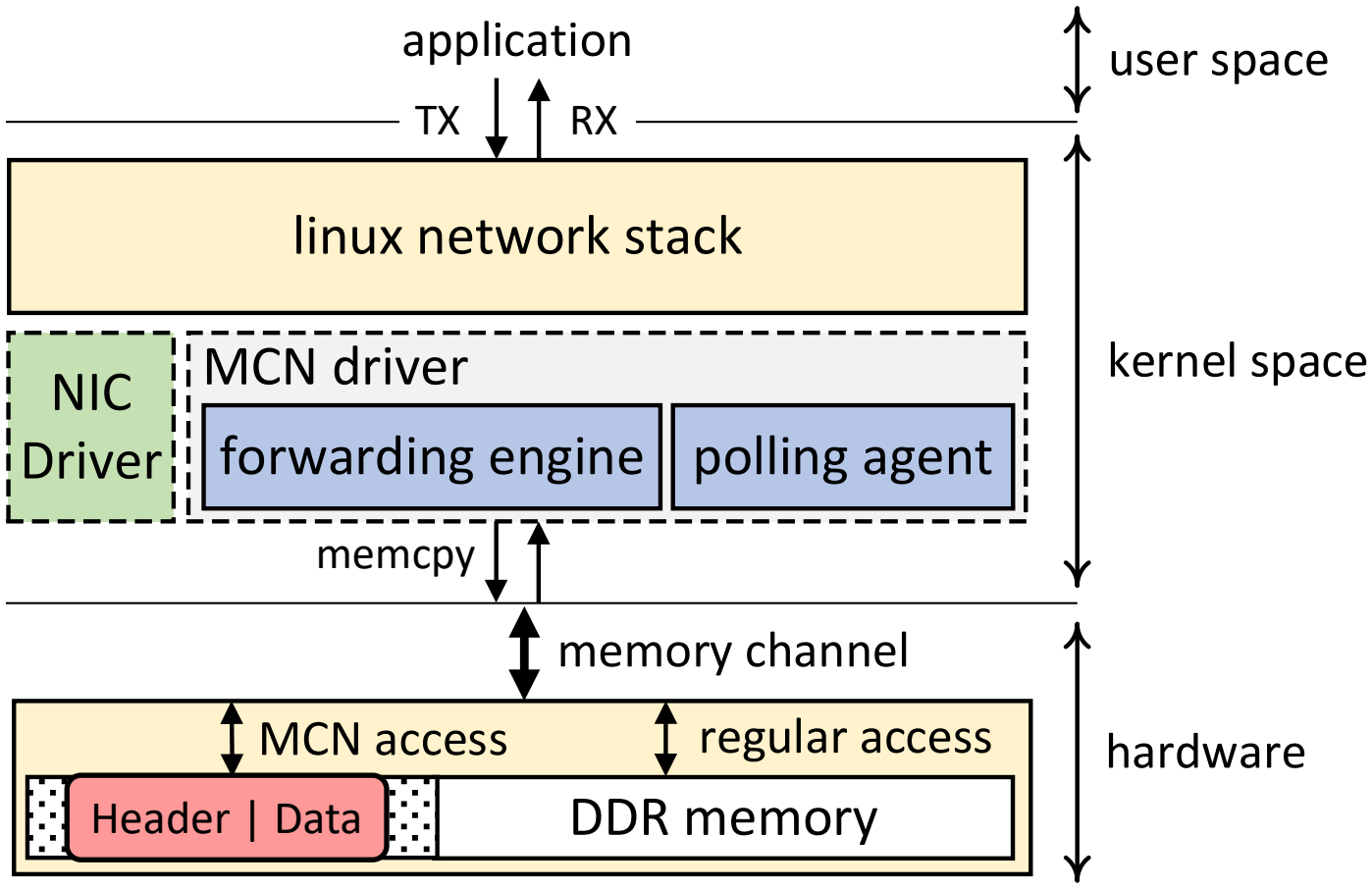


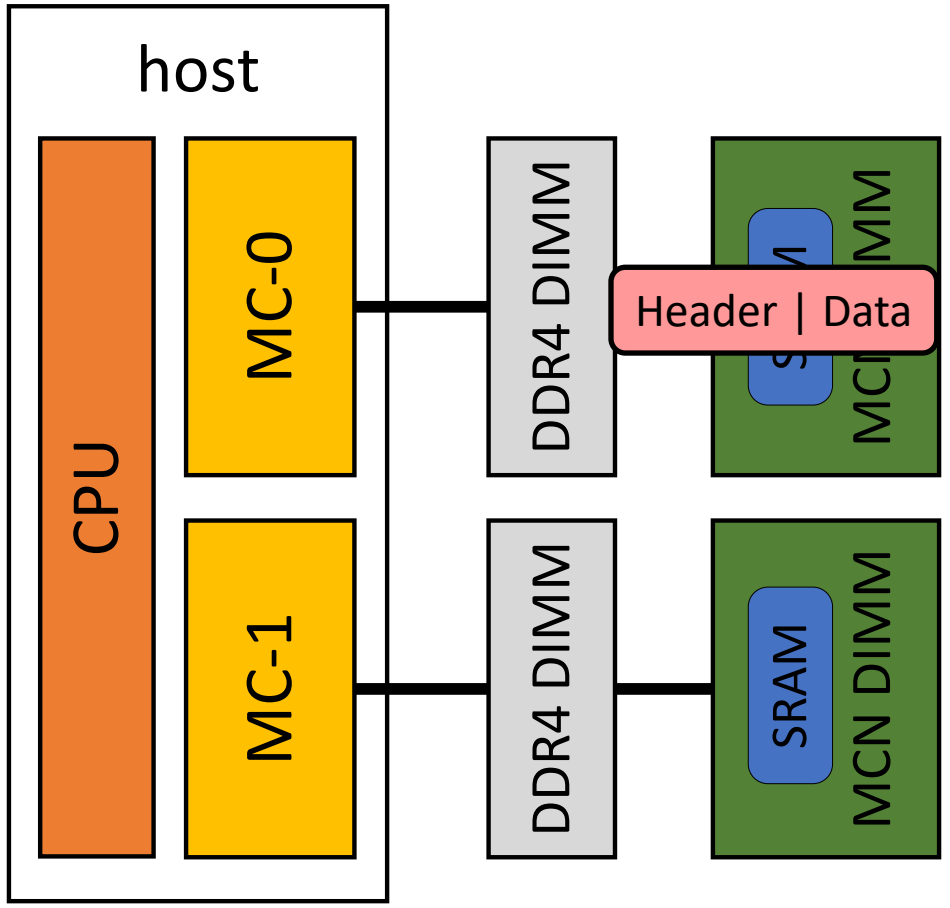


MCN Packet Routing

- MCN → Host

1. MCN DIMM writes Ethernet Packet to its SRAM buffer and set RX-poll bit

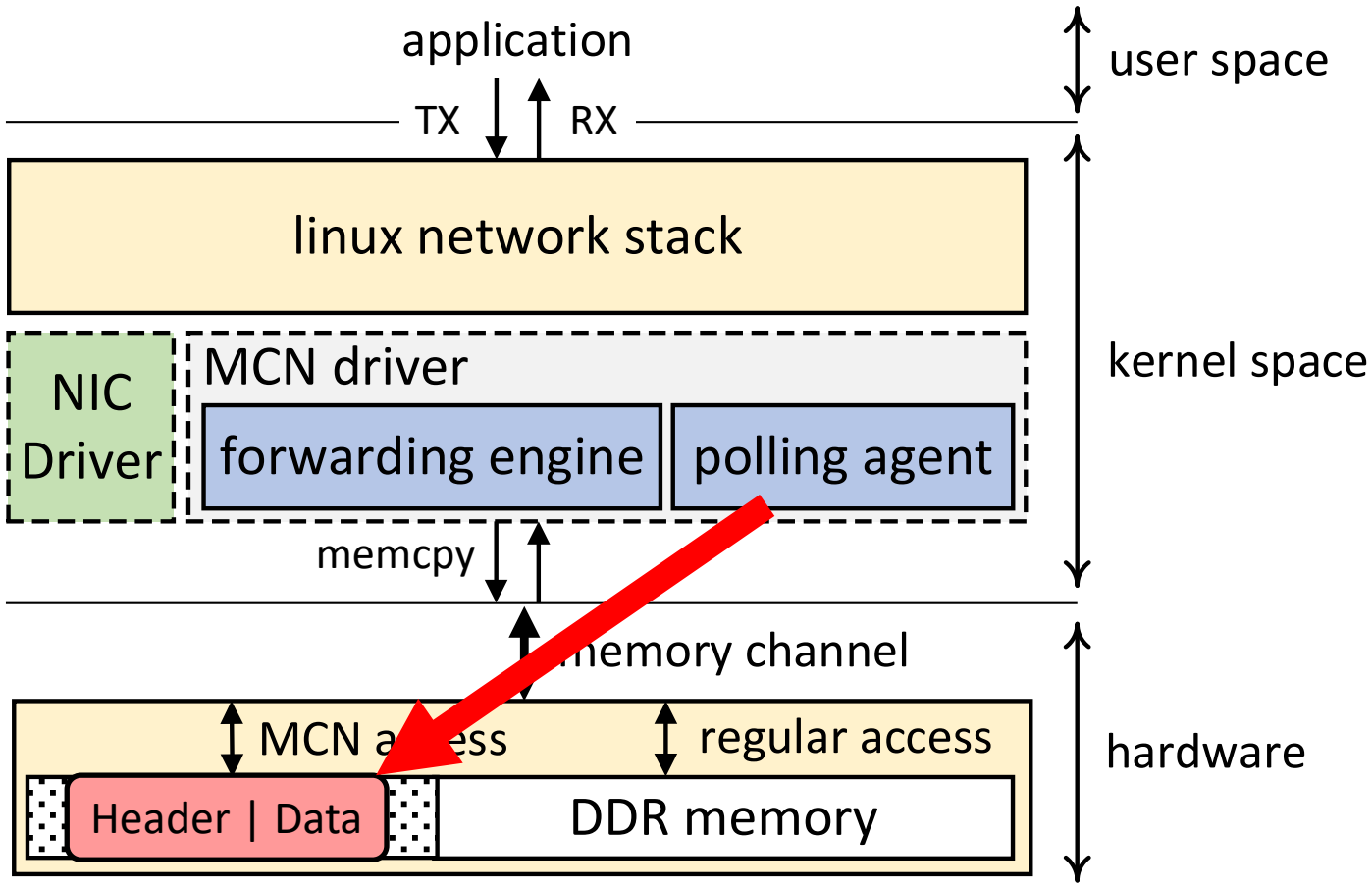


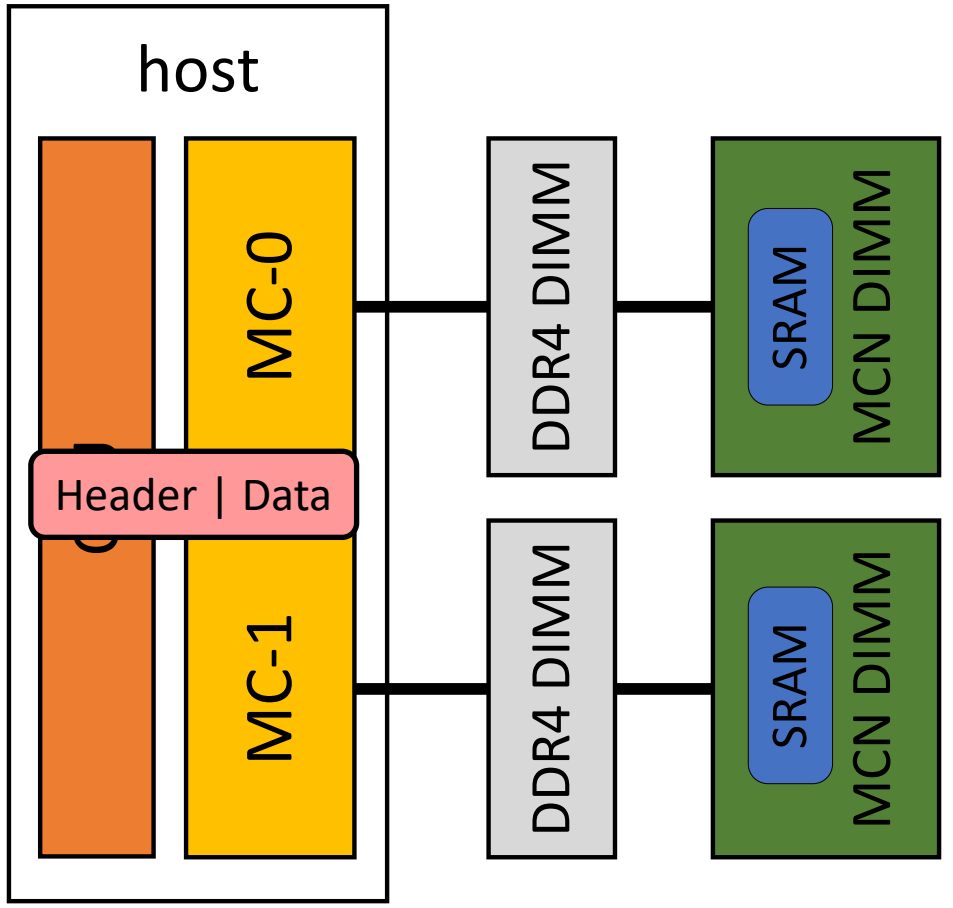


MCN Packet Routing

- MCN → Host

2. Polling agent from the host recognize the RX-poll bit is set and read the incoming packet

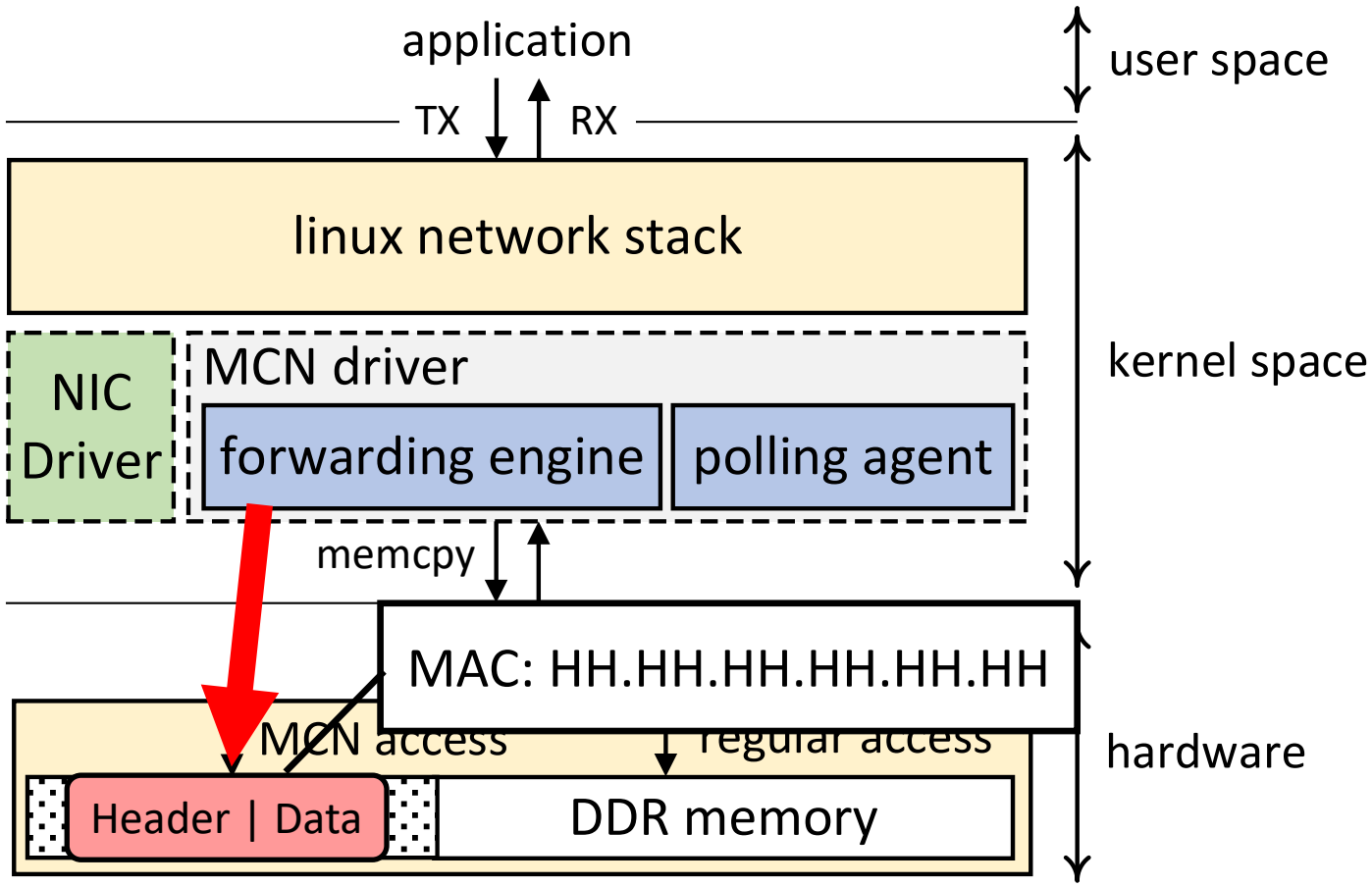


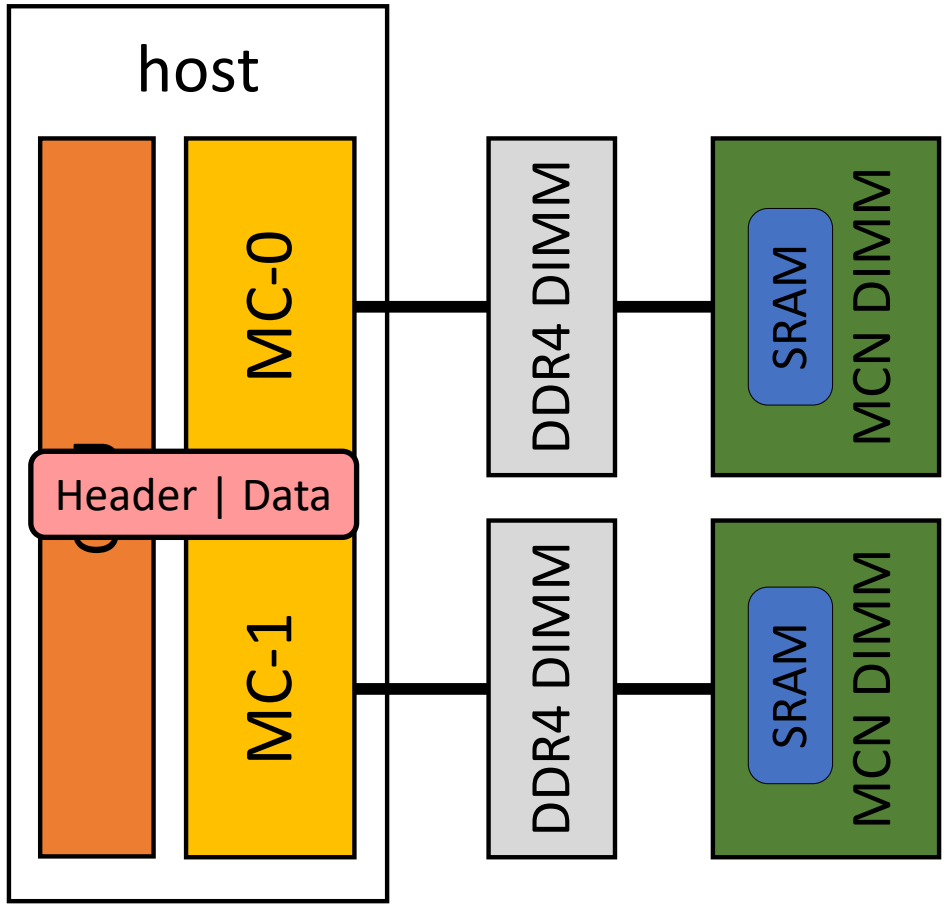


MCN Packet Routing

- MCN → Host

3. Forwarding engine checks the MAC address and determine the destination

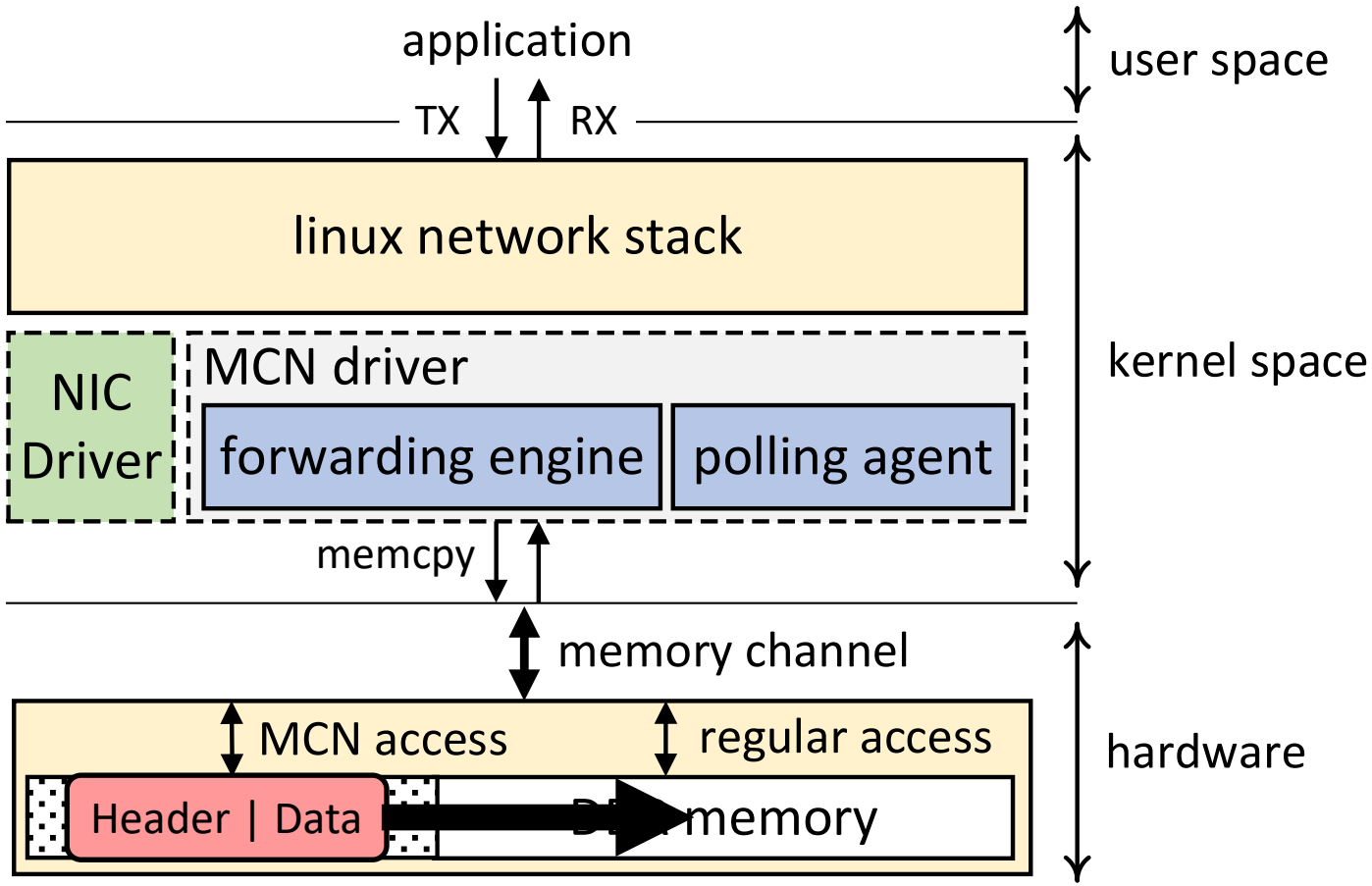


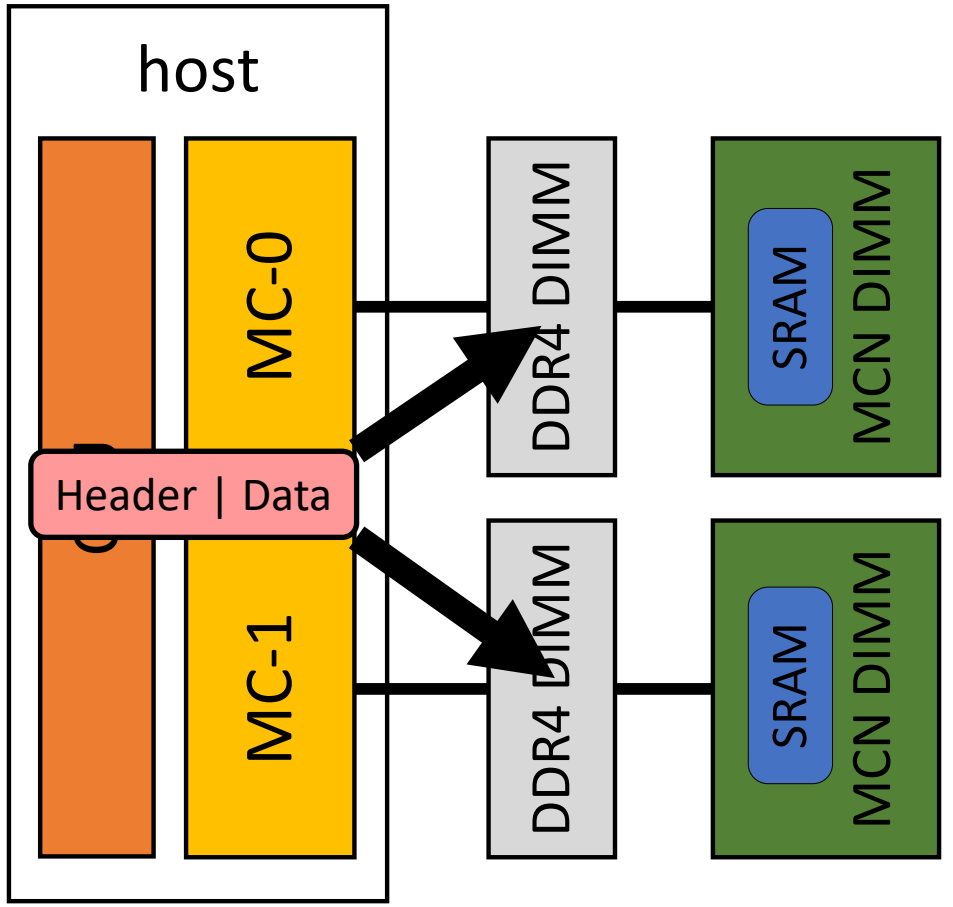


MCN Packet Routing

- MCN → Host

4. If this is for the host, it copies the Ethernet packet to the host SKB which ends up in the host DRAM

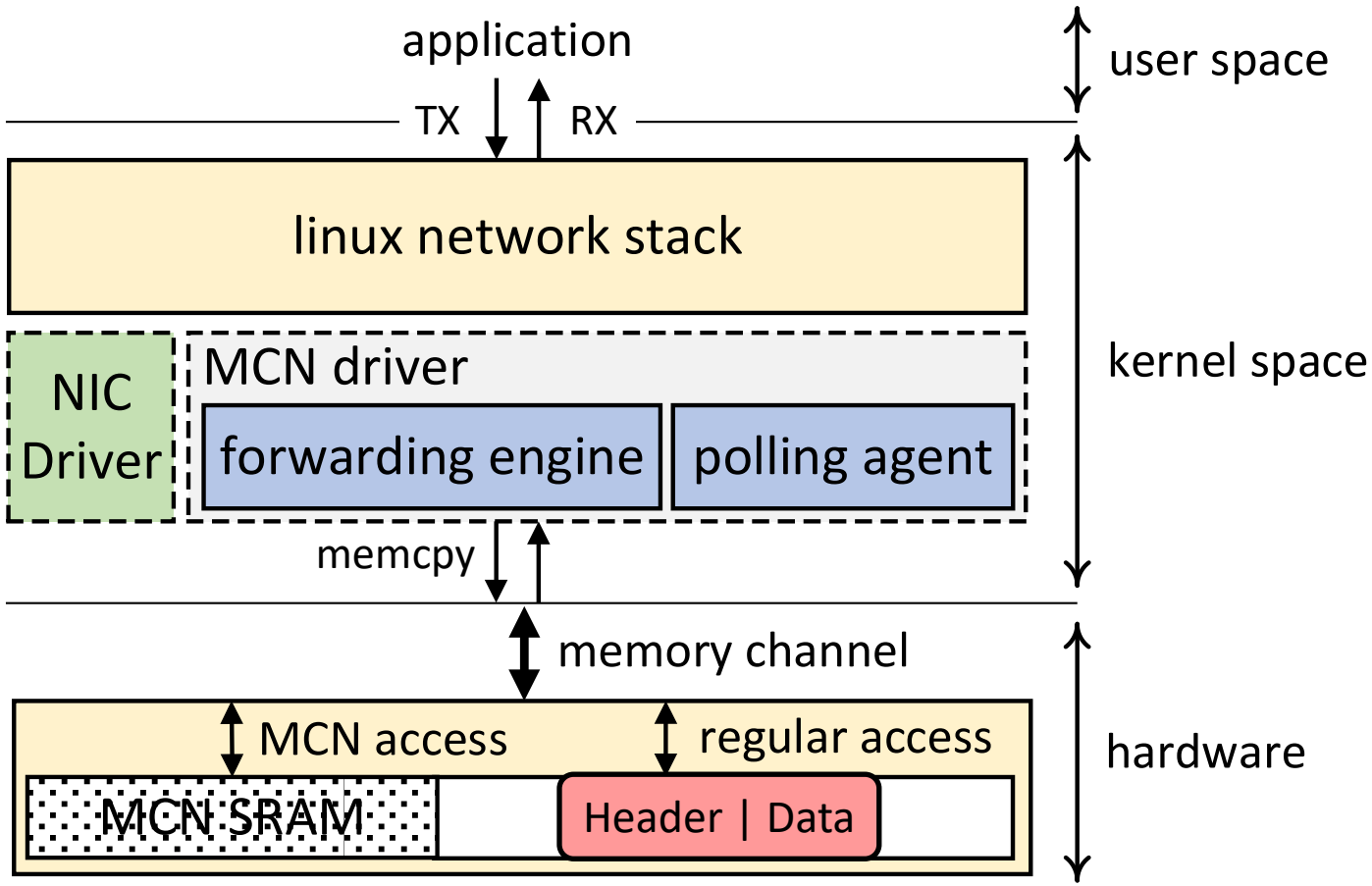


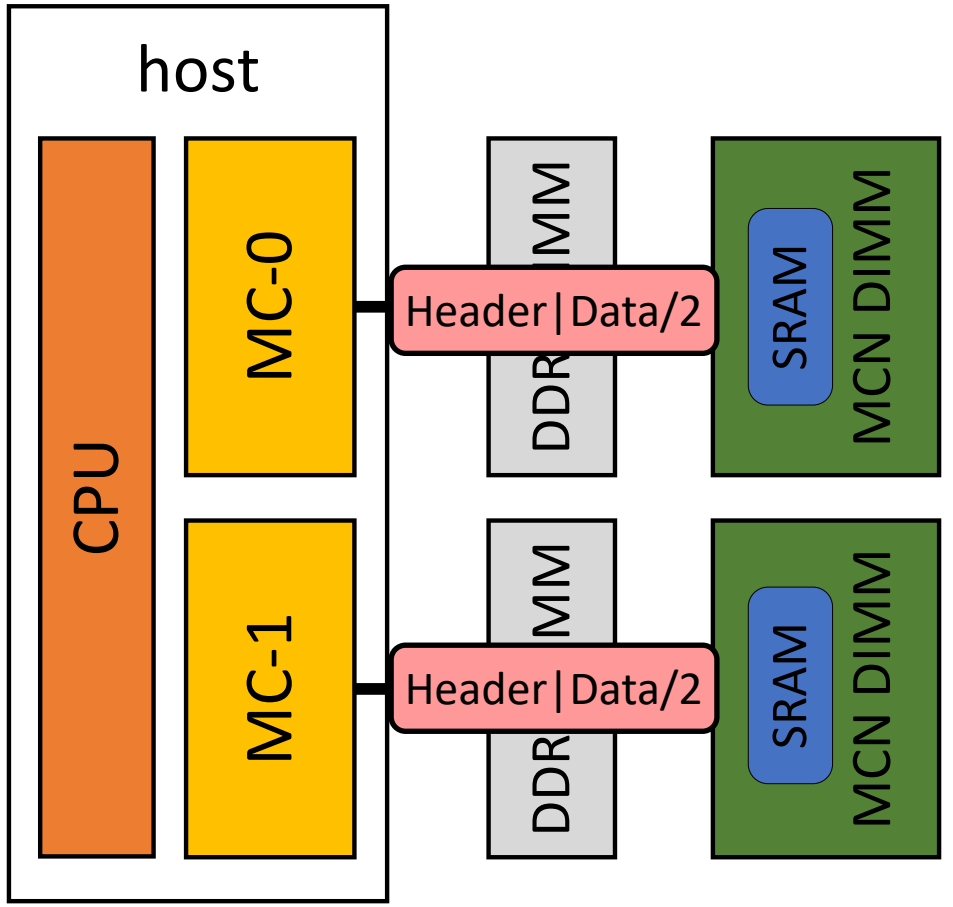


MCN Packet Routing

- MCN → Host

4. If this is for the host, it copies the Ethernet packet to the host SKB which ends up in the host DRAM

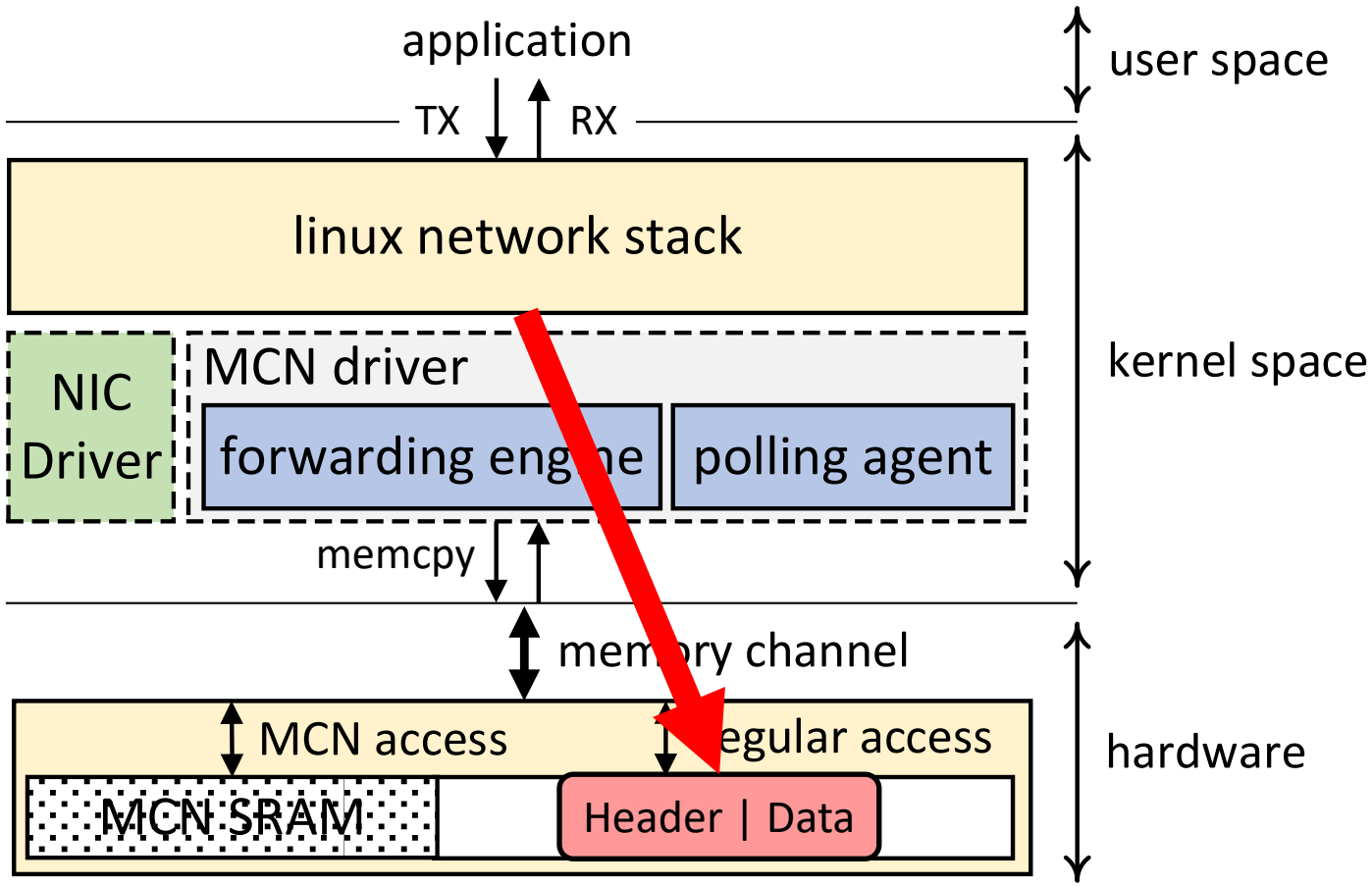


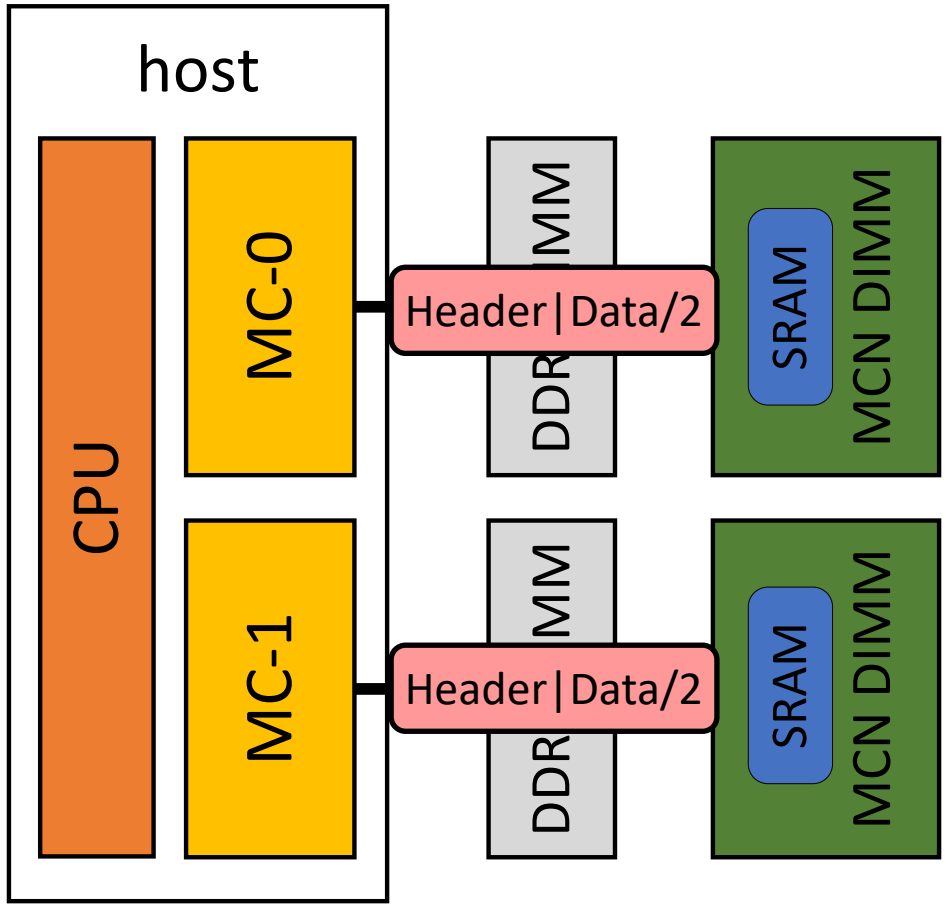


MCN Packet Routing

- MCN → Host

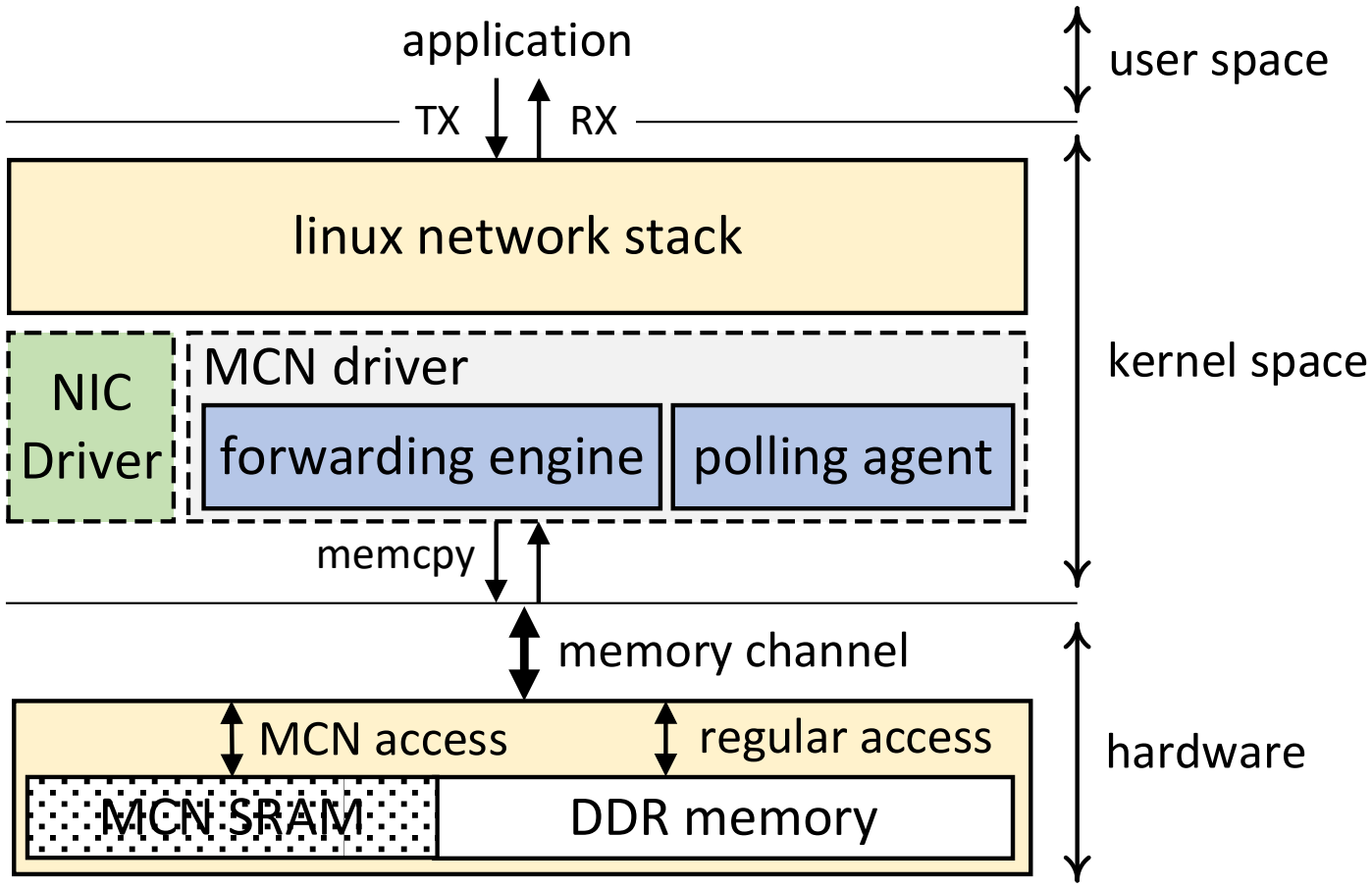
5. Now finally the packet is passed to the host network stack (e.g. TCP/IP)

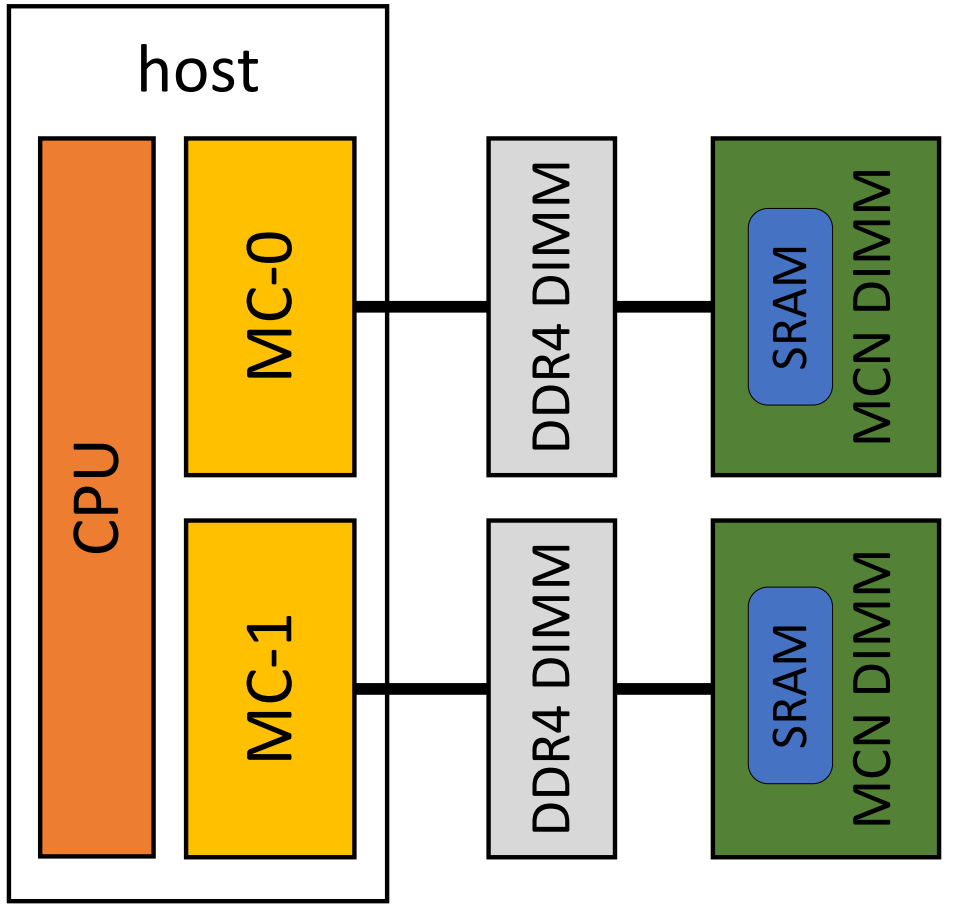




MCN Packet Routing

- Host → MCN

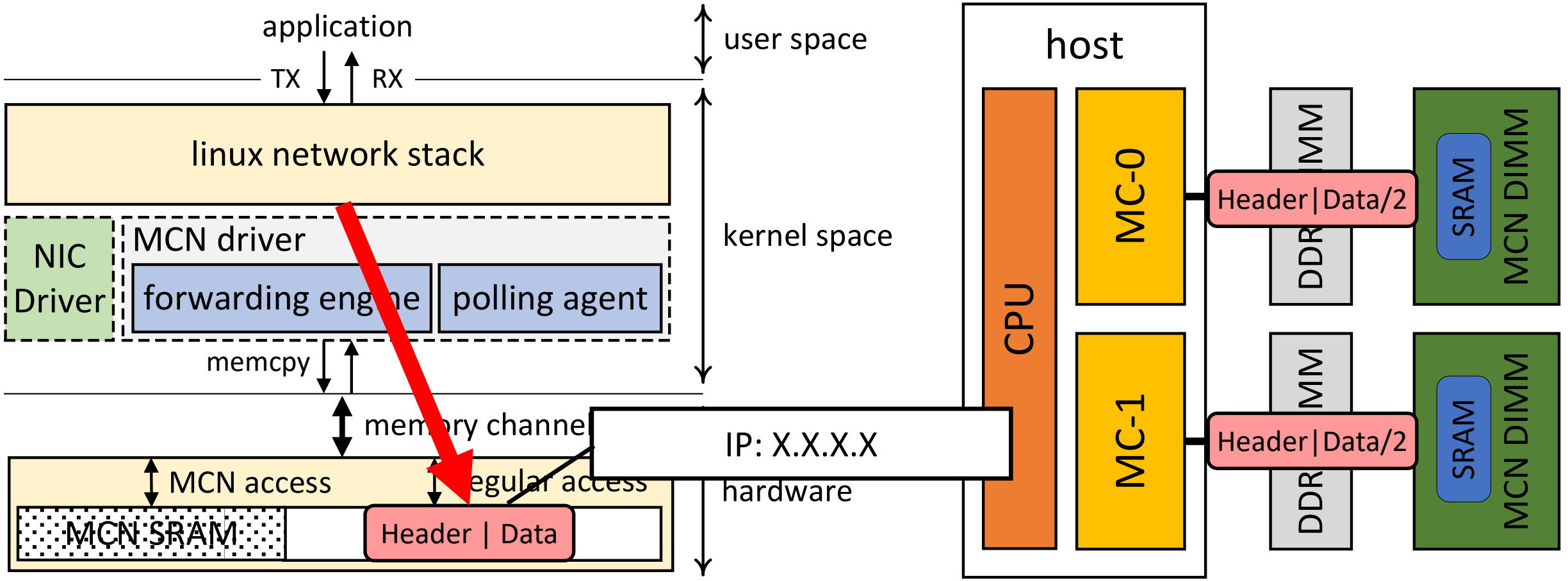




MCN Packet Routing

- Host → MCN

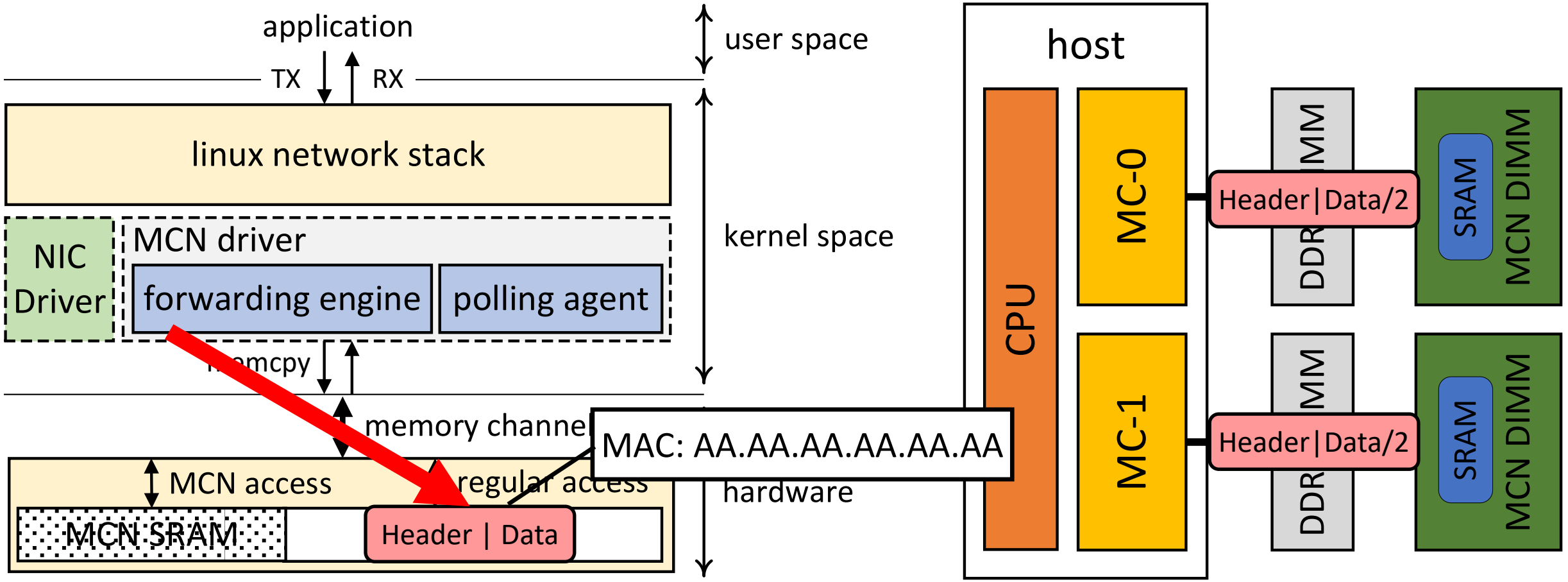
1. Packet is passed from the host network stack and the packet goes to the corresponding Ethernet device



MCN Packet Routing

- Host → MCN

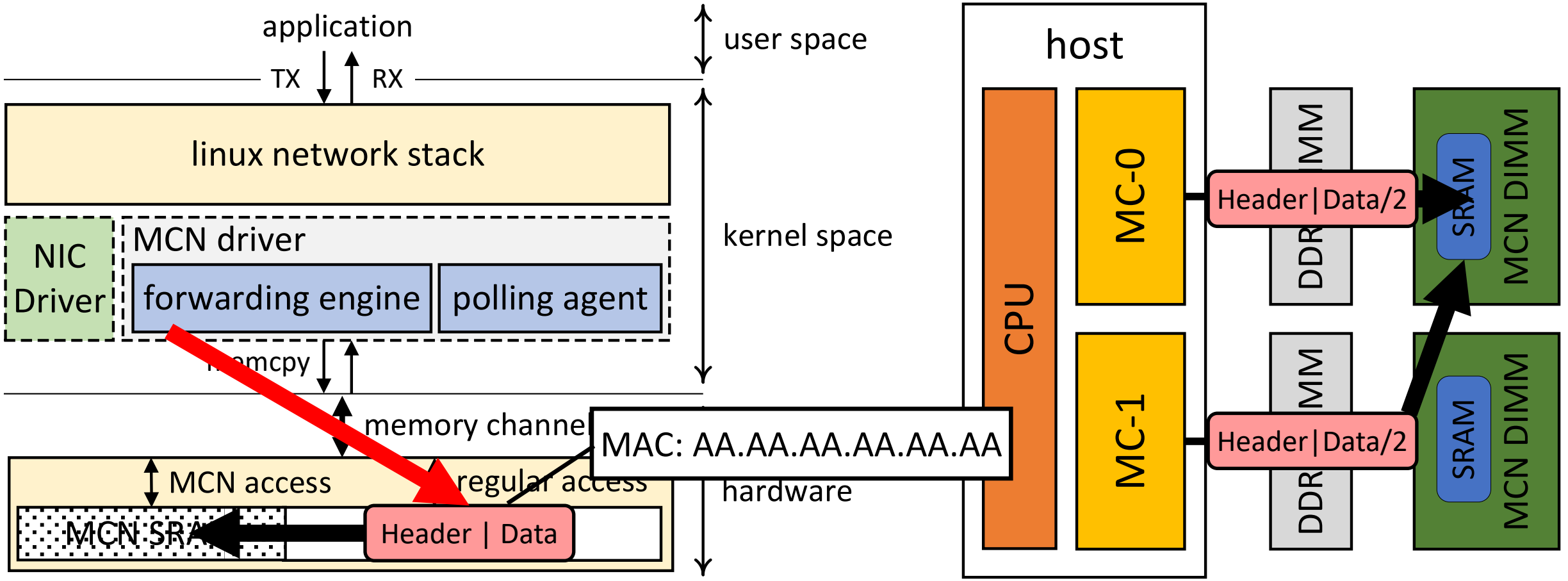
2. If the packet was send to the MCN DIMM, the packet is passed to the forwarding engine to check the MAC



MCN Packet Routing

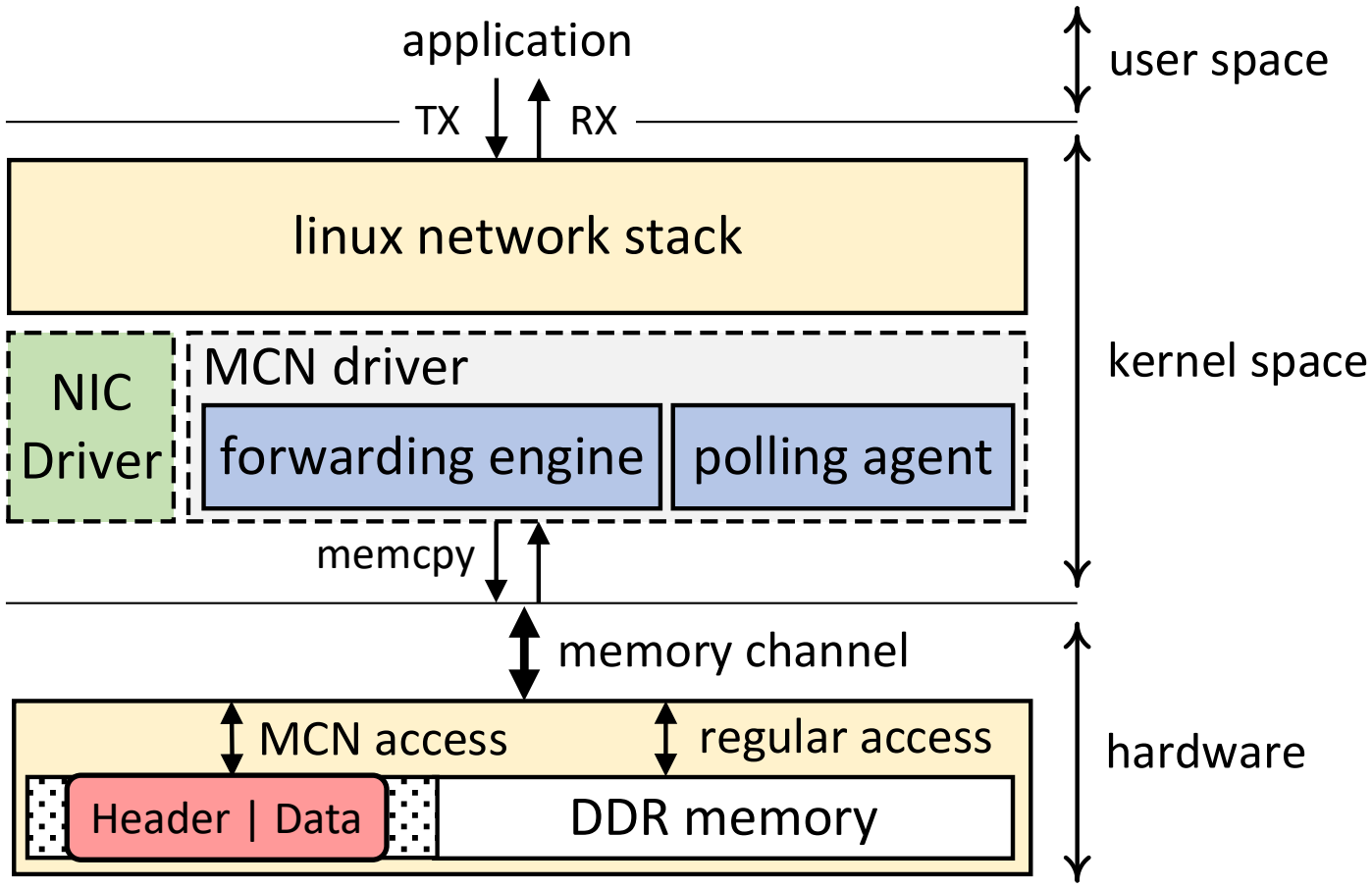
- Host → MCN

3. If the MAC matches with one of the host's MCN DIMMs, the Ethernet packet is copied to the MCN DIMM

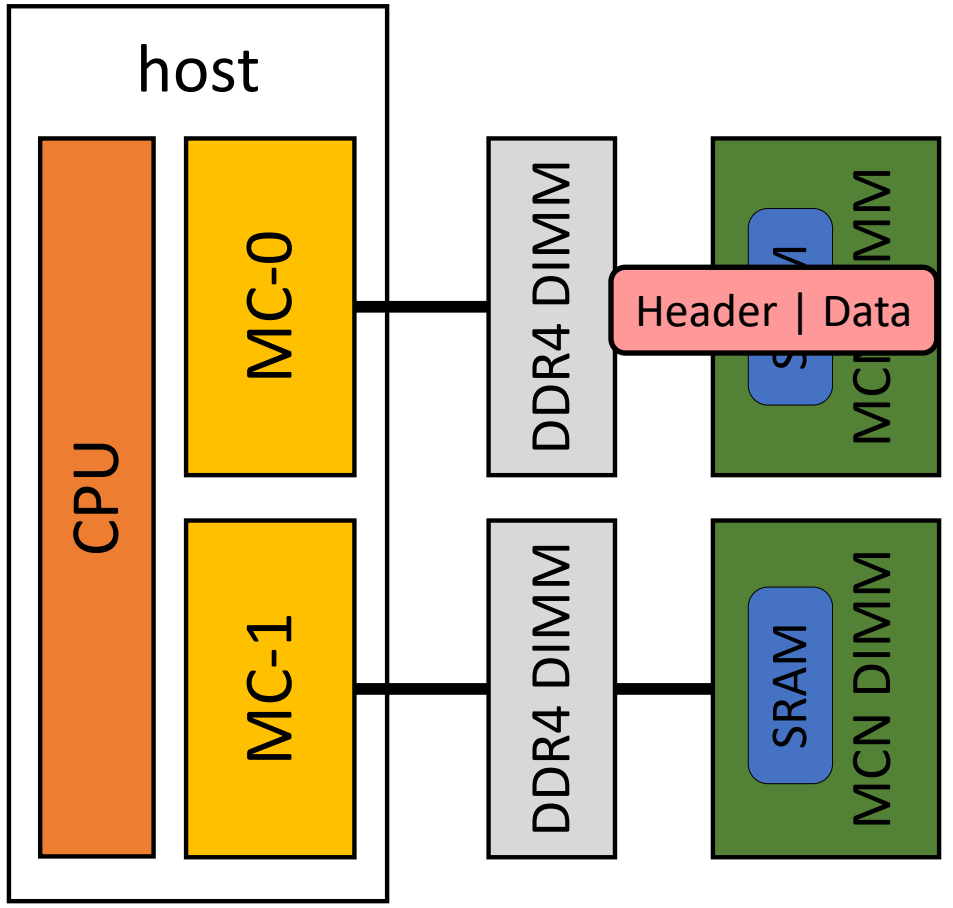


MCN Packet Routing

- Host → MCN



4. This data copy fires IRQ from the MCN DIMM and MCN processor knows there is an incoming packet



Outline

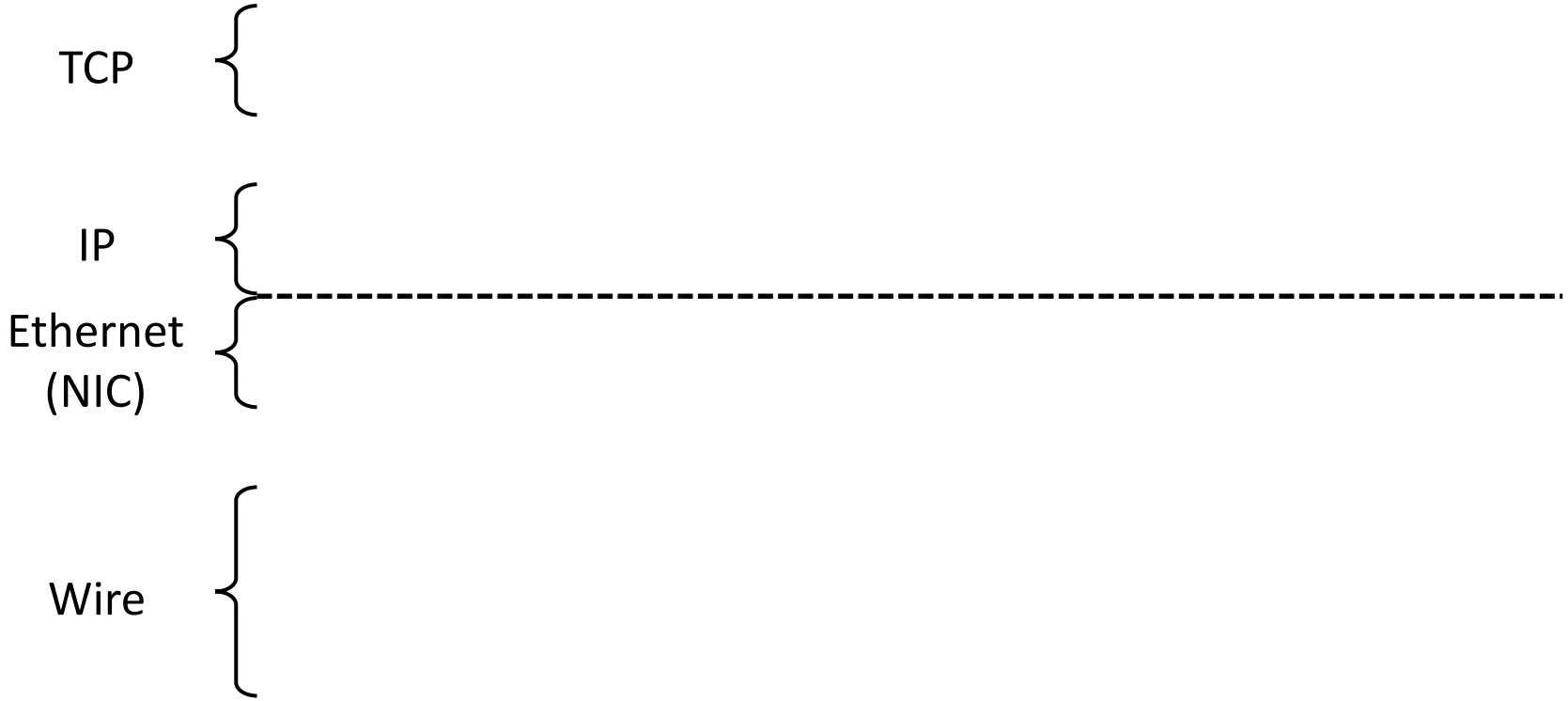
- MCN Architecture
  - ✓Background – Buffered DIMM
  - ✓Design Overview
  - ✓MCN DIMM Architecture
  - ✓MCN Packet Routing
- **Design Optimizations**
- Proof of Concept – Hardware Demonstration
- Evaluation
  - ✓Simulation Methodology
  - ✓Bandwidth and Latency
- Conclusion

Optimizations

- Adopt optimizations from existing network interfaces
  - ✓Offload (or remove) packet fragmentation

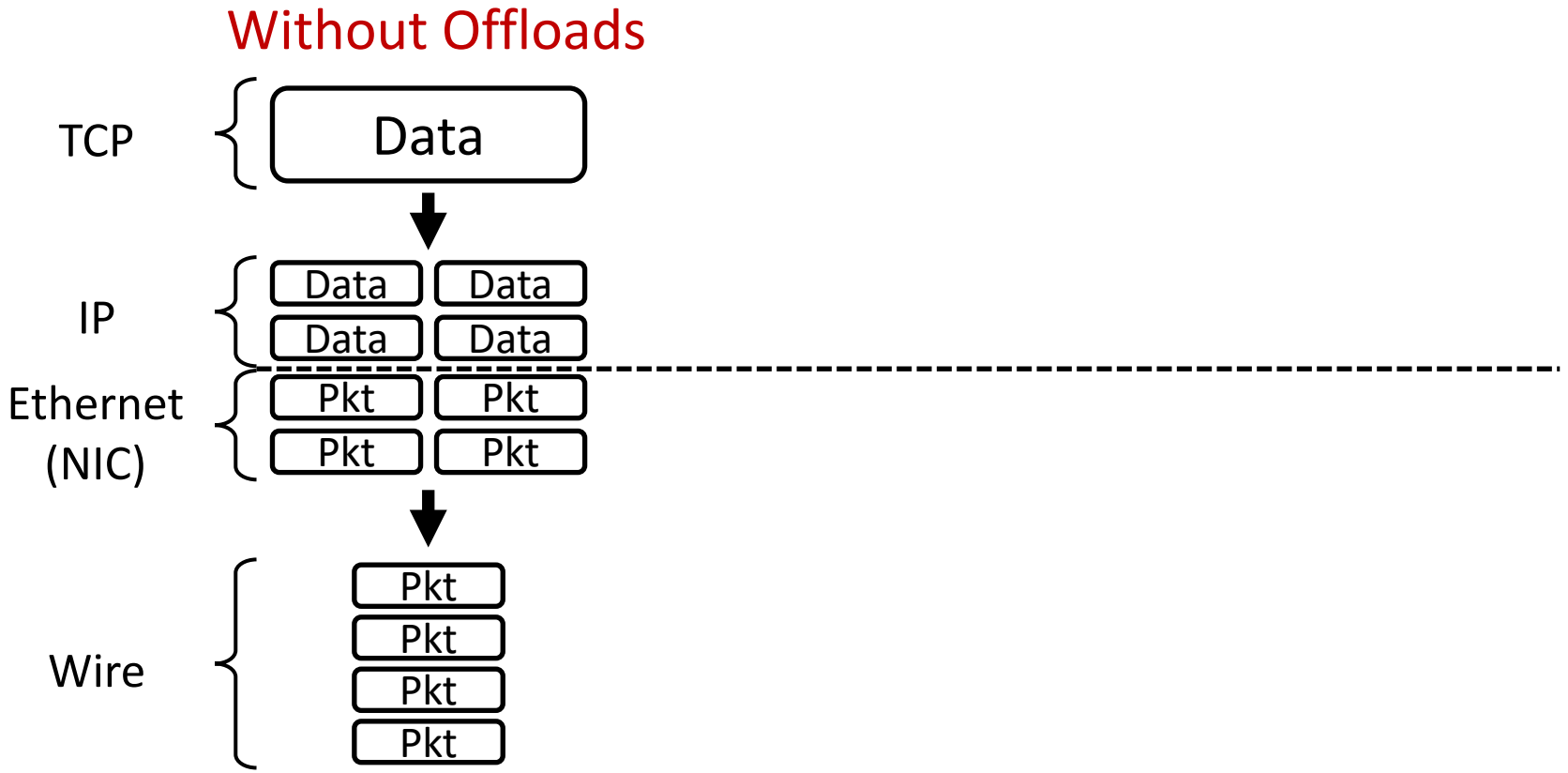
Optimizations

- Adopt optimizations from existing network interfaces
  - ✓Offload (or remove) packet fragmentation (e.g. TCP Segmentation Offload (TSO))



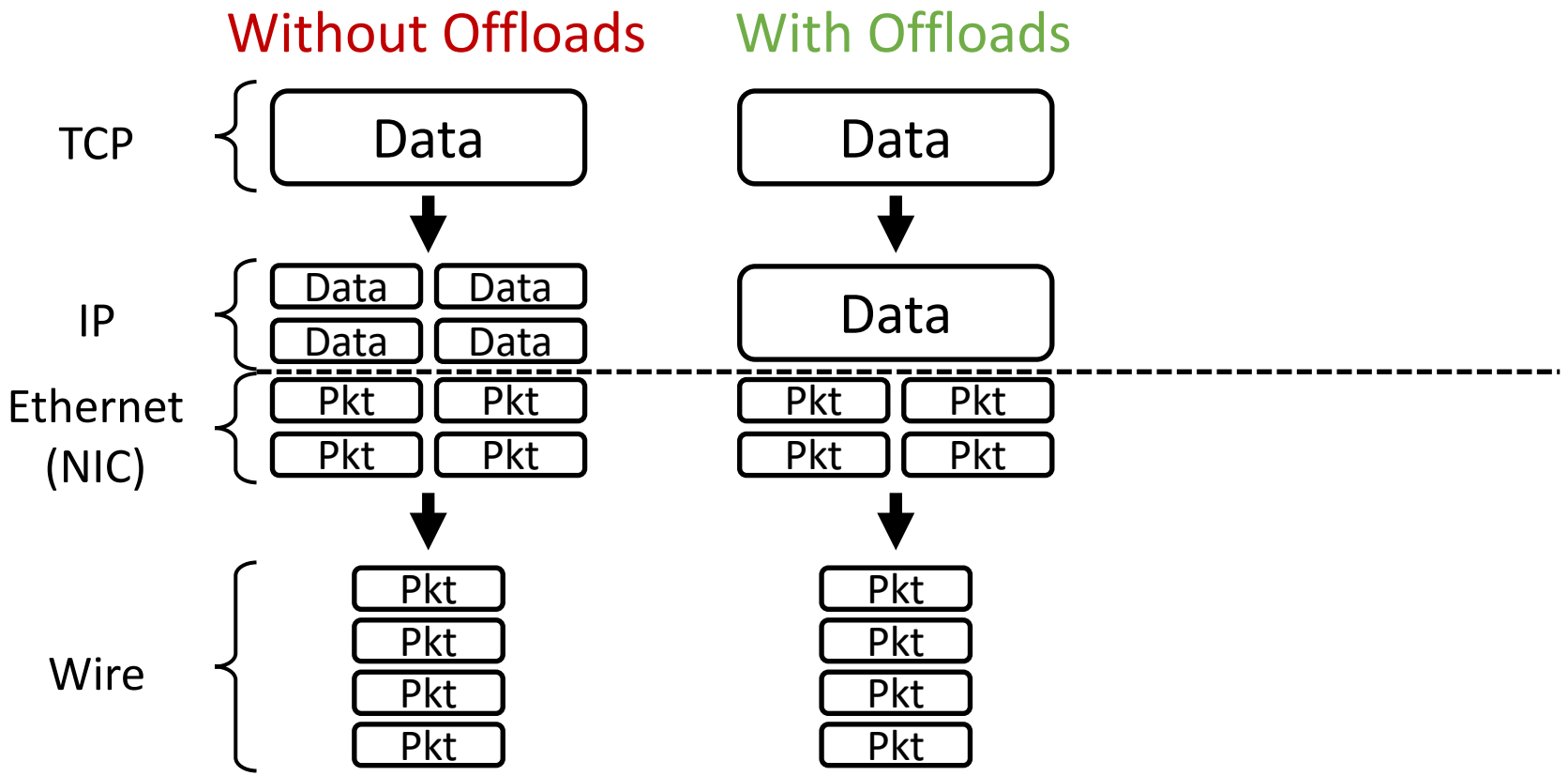
Optimizations

- Adopt optimizations from existing network interfaces
  - ✓Offload (or remove) packet fragmentation (e.g. TCP Segmentation Offload (TSO))



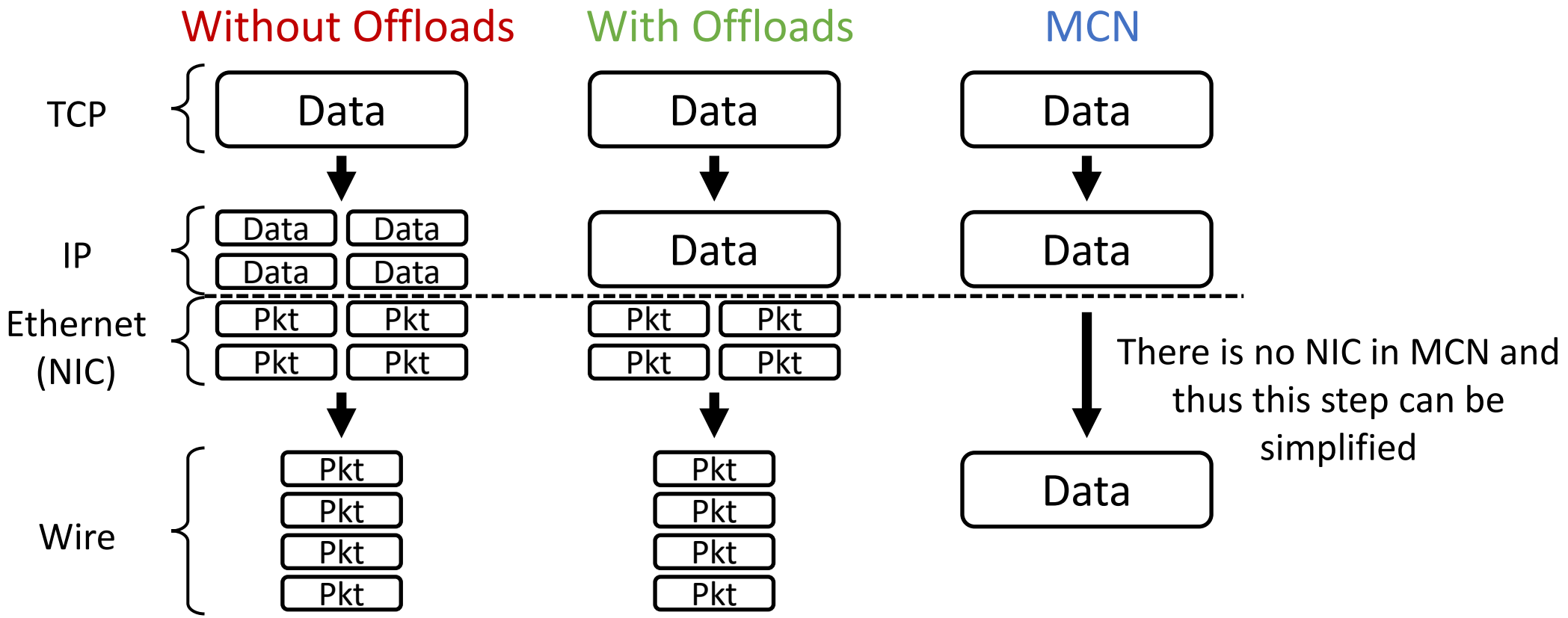
Optimizations

- Adopt optimizations from existing network interfaces
  - ✓Offload (or remove) packet fragmentation (e.g. TCP Segmentation Offload (TSO))



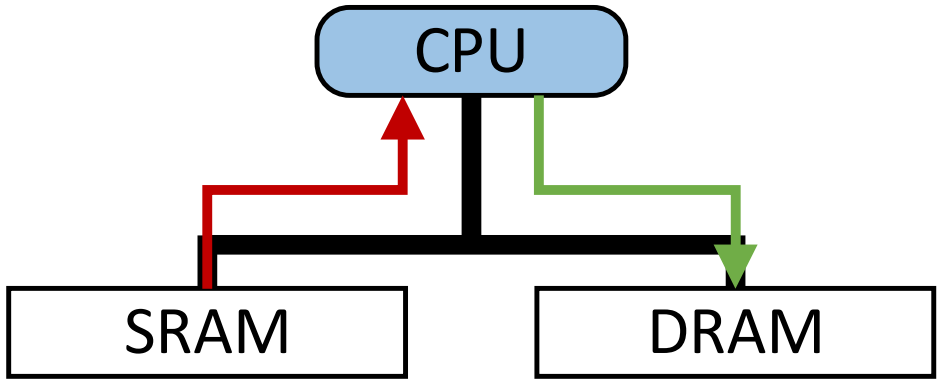
Optimizations

- Adopt optimizations from existing network interfaces
  - ✓Offload (or remove) packet fragmentation (e.g. TCP Segmentation Offload (TSO))

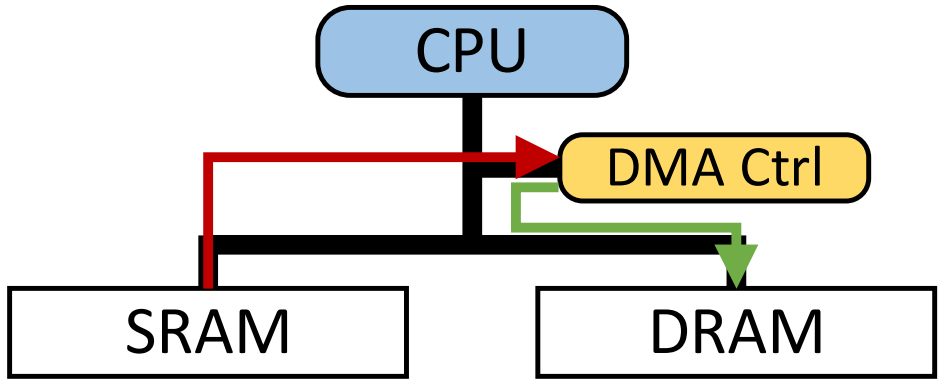


Optimizations

- Adopt optimizations from existing network interfaces
  - ✓Offload (or remove) packet fragmentation
- Add MCN side DMA
  - ✓In the baseline design, MCN processor manually copies data between the SRAM buffer and its own DRAM
  - ✓Reduces CPU memcpy overhead



MCN Baseline



MCN with DMA

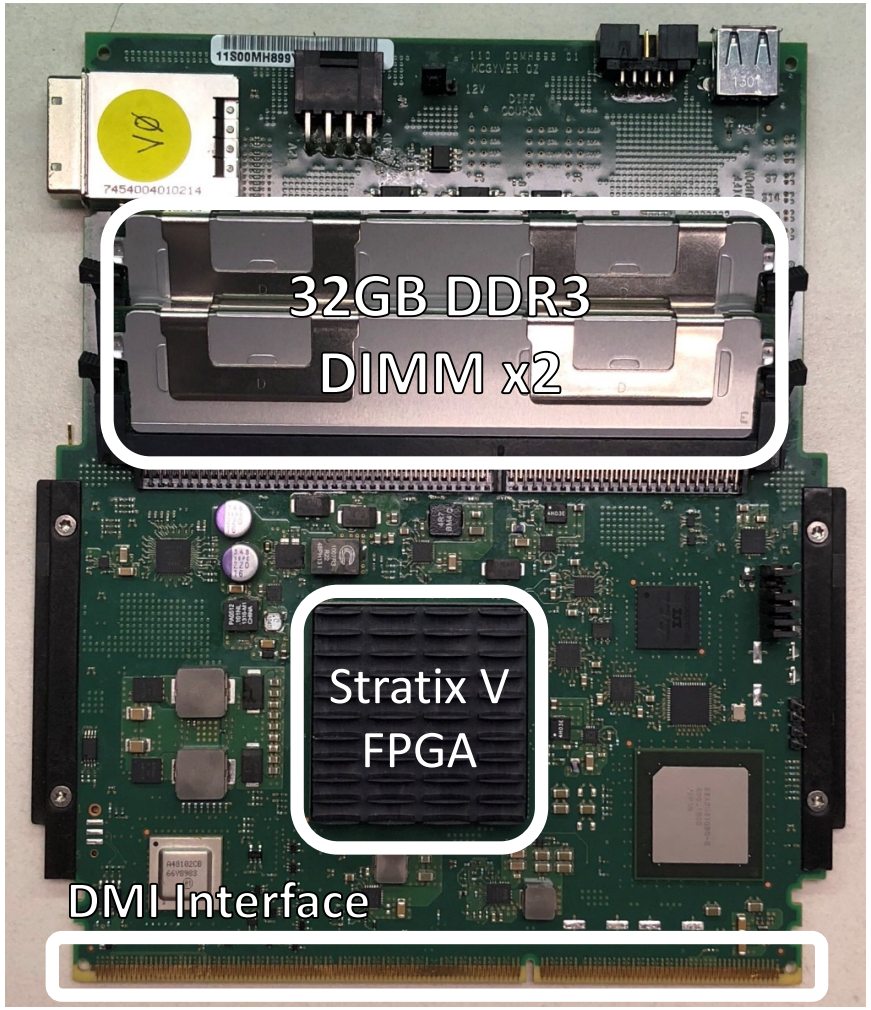
Optimizations

- Adopt optimizations from existing network interfaces
  - ✓Offload (or remove) packet fragmentation
- Add MCN side DMA
  - ✓In the baseline design, MCN processor manually copies data between the SRAM buffer and its own DRAM
  - ✓Reduces CPU memcpy overhead
- Other optimizations attempted:
  - ✓Checksum bypassing
    - Memory channel is ECC protected
  - ✓MCN DIMM interrupt mechanism
    - Leverage ALERT_N in the DDR4 standard

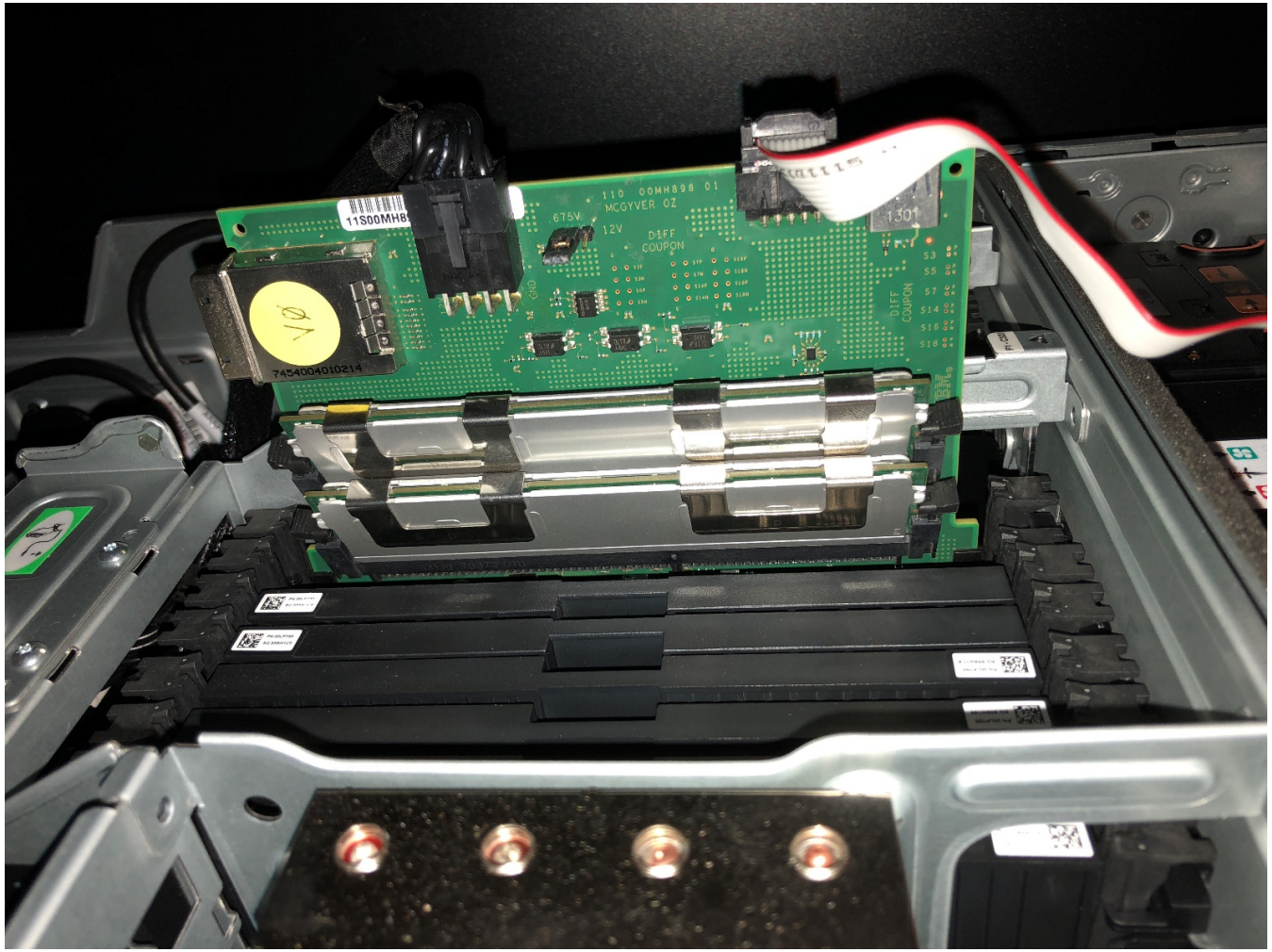
Outline

- MCN Architecture
  - ✓Background – Buffered DIMM
  - ✓Design Overview
  - ✓MCN DIMM Architecture
  - ✓MCN Packet Routing
- Design Optimizations
- **Proof of Concept – Hardware Demonstration**
- Evaluation
  - ✓Simulation Methodology
  - ✓Bandwidth and Latency
- Conclusion

Proof of Concept Hardware Demonstration

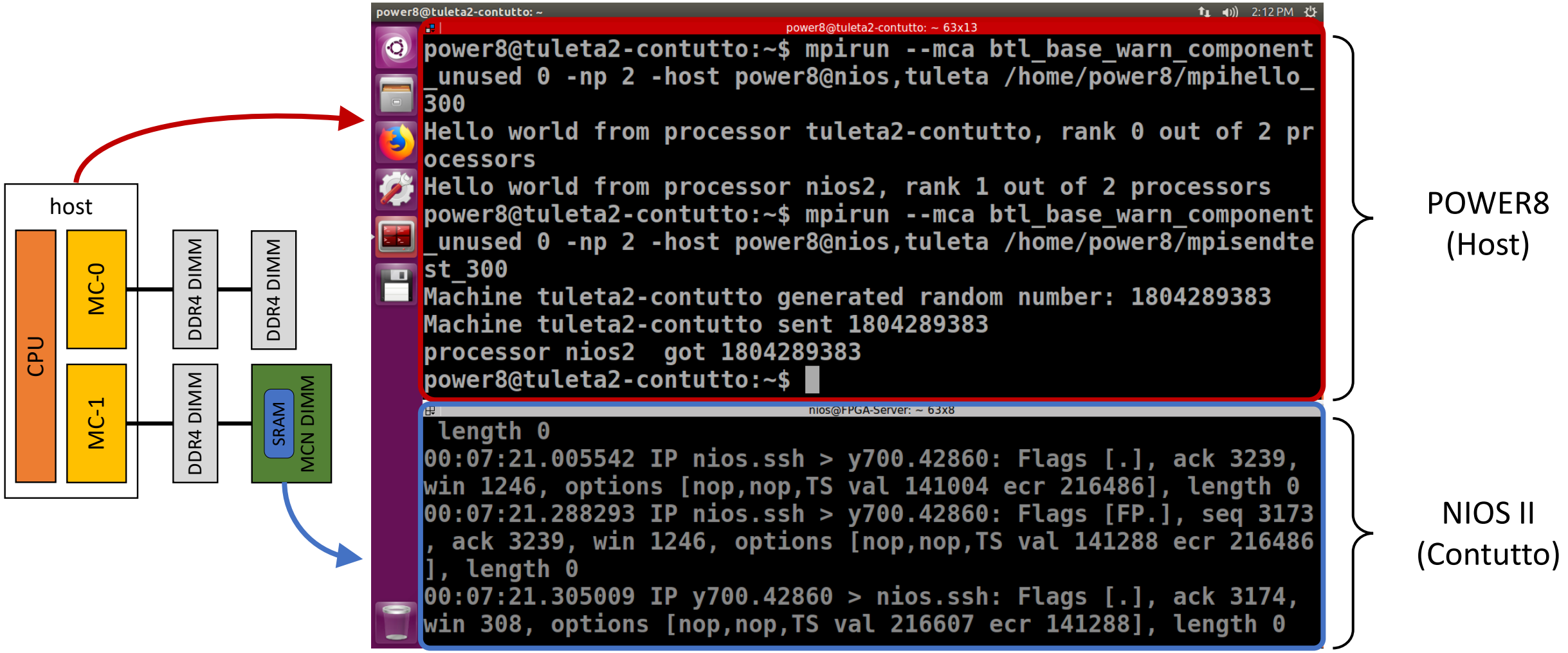


Contutto [2] top view



Contutto plugged in IBM POWER8 server

MPI Demonstration



MPI application running through MCN

Outline

- MCN Architecture
  - ✓Background – Buffered DIMM
  - ✓Design Overview
  - ✓MCN DIMM Architecture
  - ✓MCN Packet Routing
- Design Optimizations
- Proof of Concept – Hardware Demonstration
- **Evaluation**
  - ✓Simulation Methodology
  - ✓Bandwidth and Latency
- Conclusion

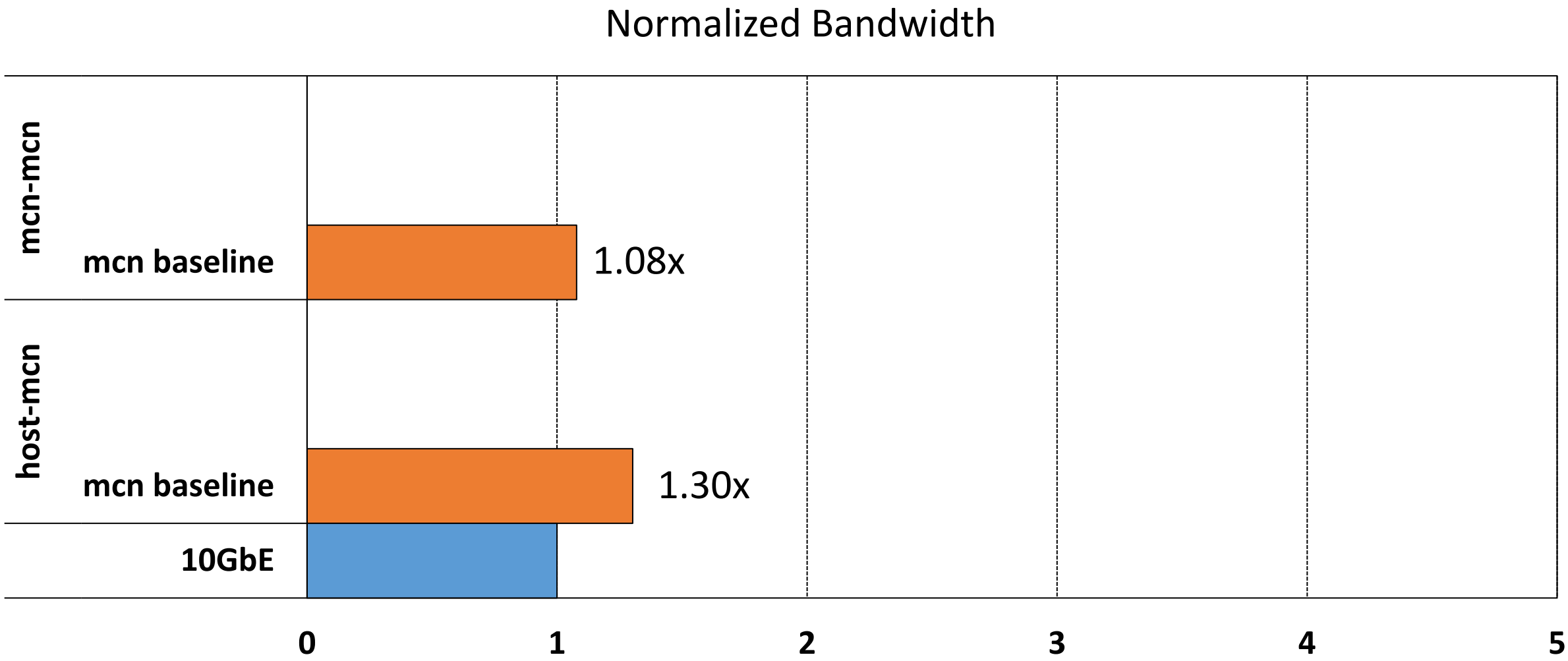
Methodology

- Simulated on dist-gem5 [3]
- Baseline design is only composed of the host system configuration
- Proposed design has 2-8 MCN DIMMs per the host system
- Evaluated with iPerf [4], Ping, Coral [5], Bigdatabench [6], and NPB [7]

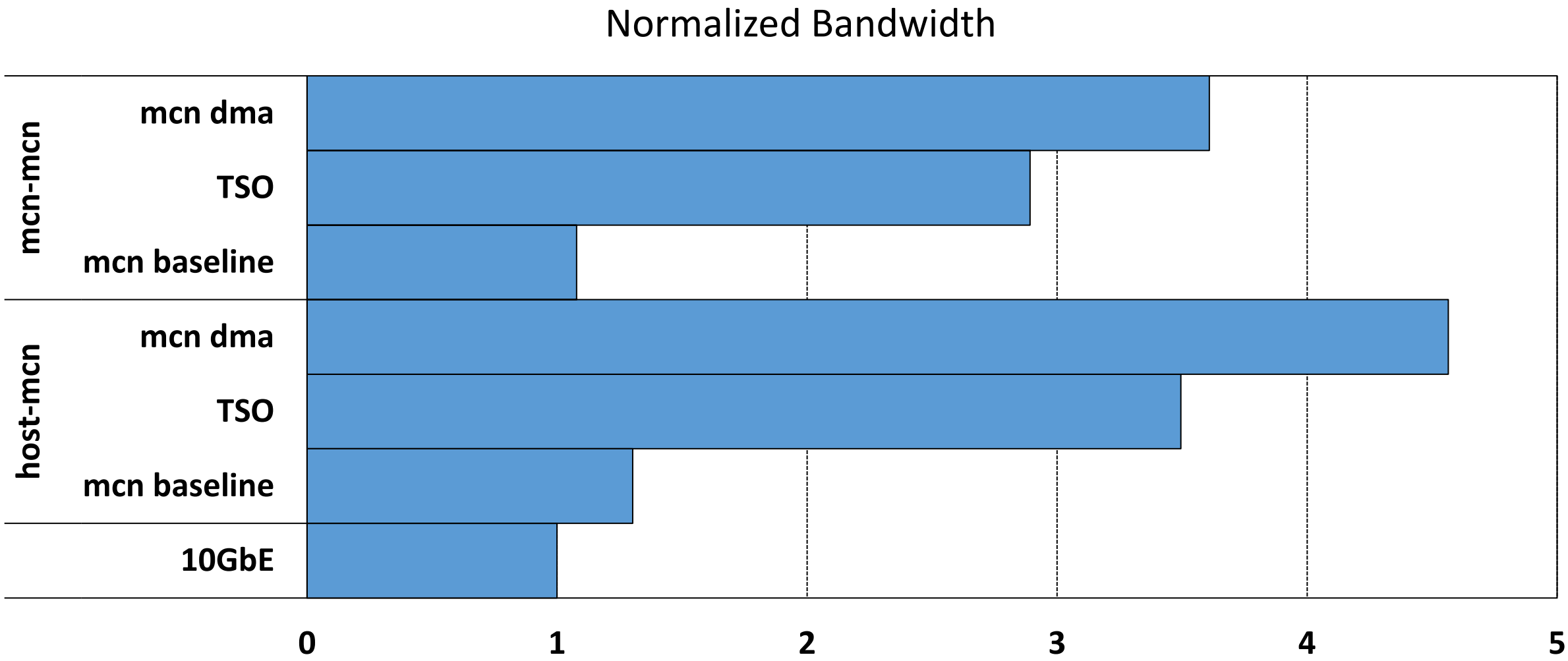
MCN System Configuration	
CPU	ARMv8 Quad Core running @ 2.45GHz
Caches	L1I: 32KB, L1D: 32KB, L2: 1MB
Memory	DDR4-3200
OS	Ubuntu 14.04

Host System Configuration	
CPU	ARMv8 Octa Core running @ 3.4GHz
Caches	L1I: 32KB, L1D: 32KB, L2: 256KB, L3: 8MB
Memory	DDR4-3200
NIC	10GbE/1μs link latency
OS	Ubuntu 14.04

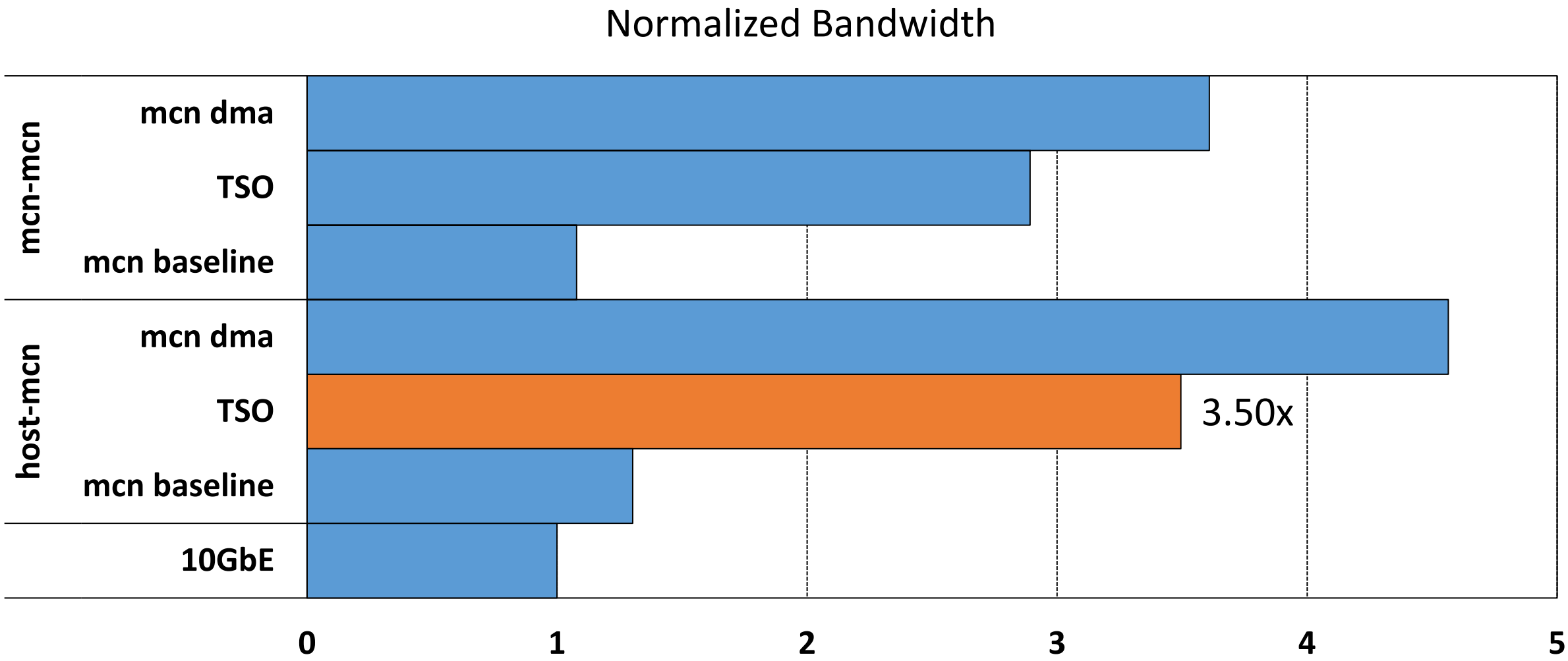
Evaluation – Network Bandwidth (iPerf)



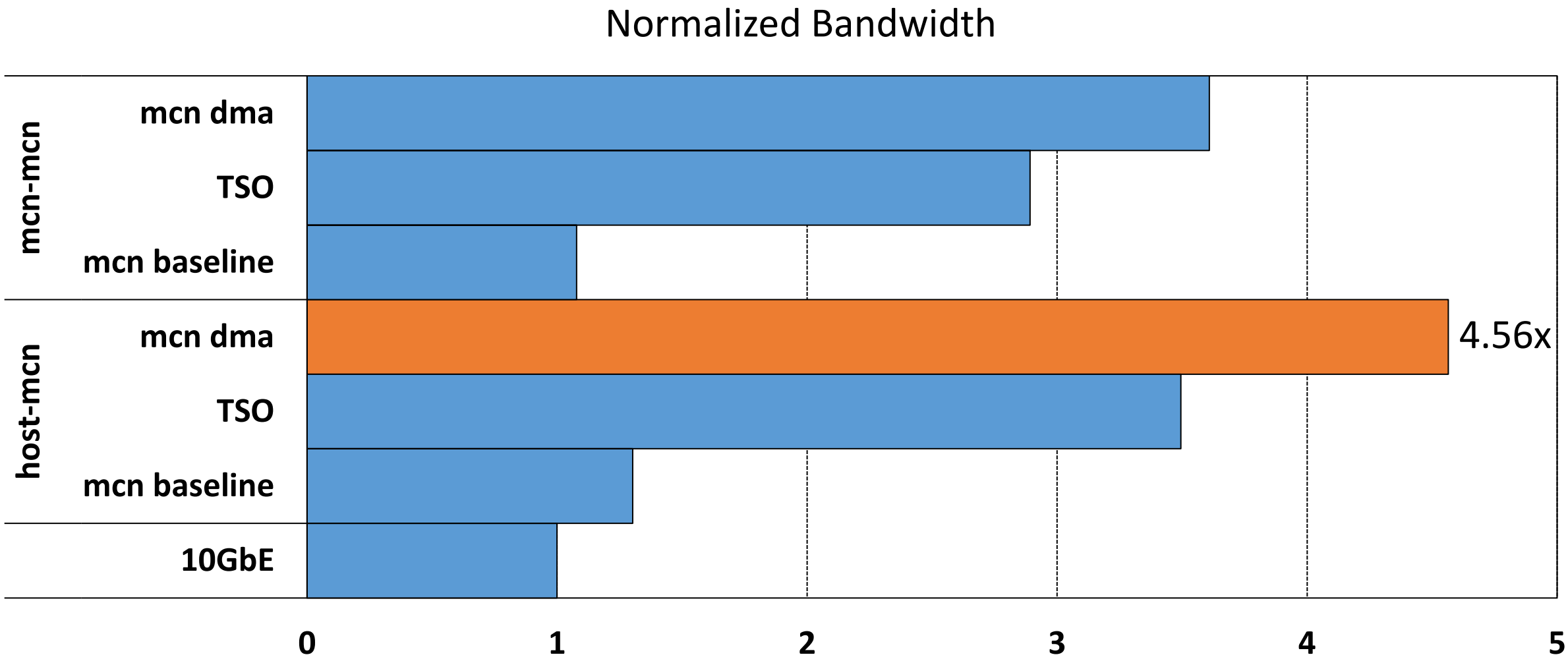
Evaluation – Network Bandwidth (iPerf)



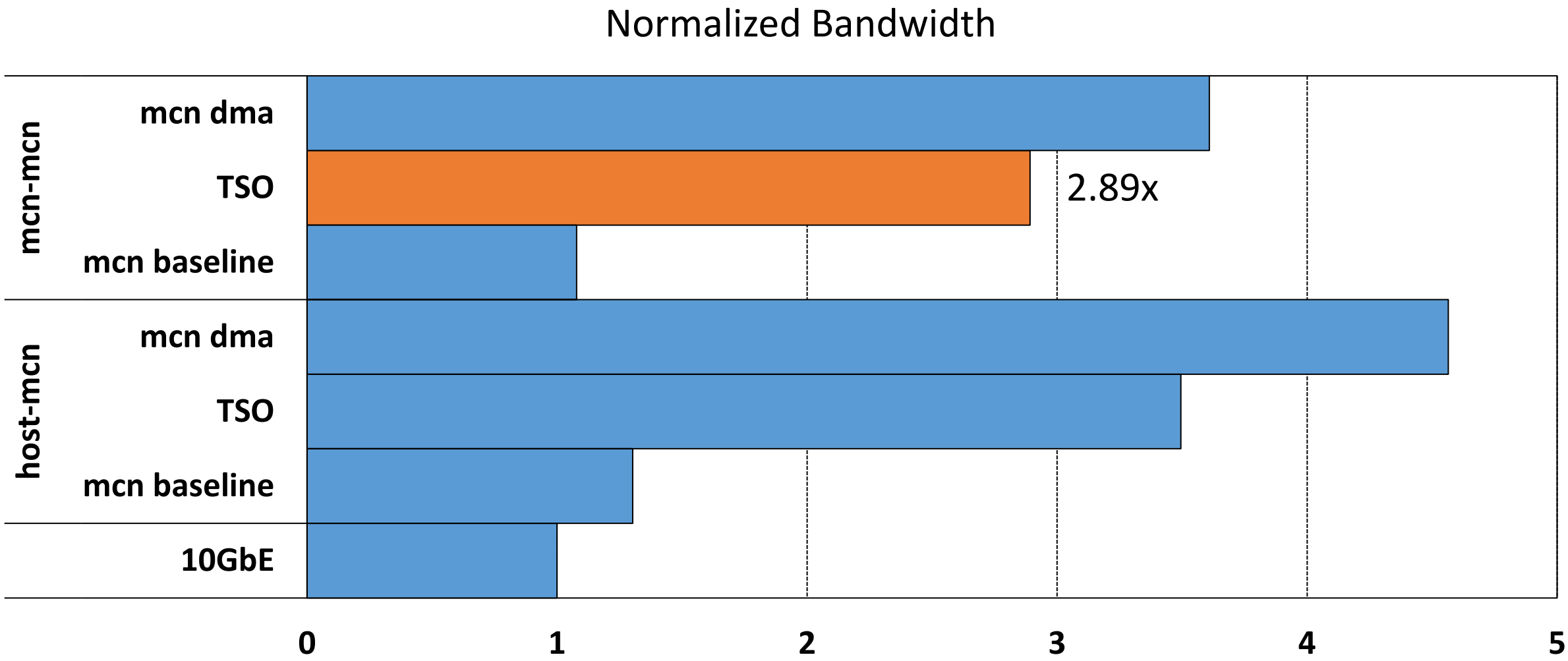
Evaluation – Network Bandwidth (iPerf)



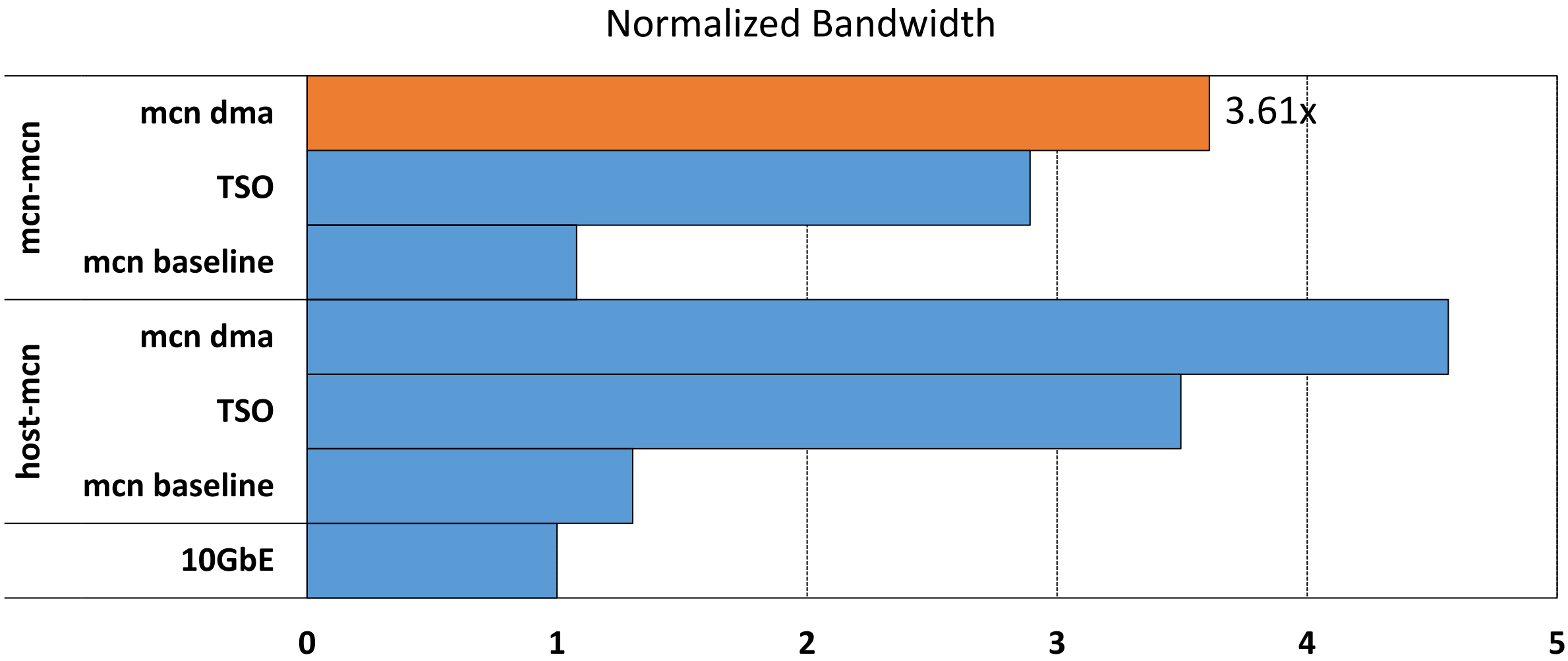
Evaluation – Network Bandwidth (iPerf)



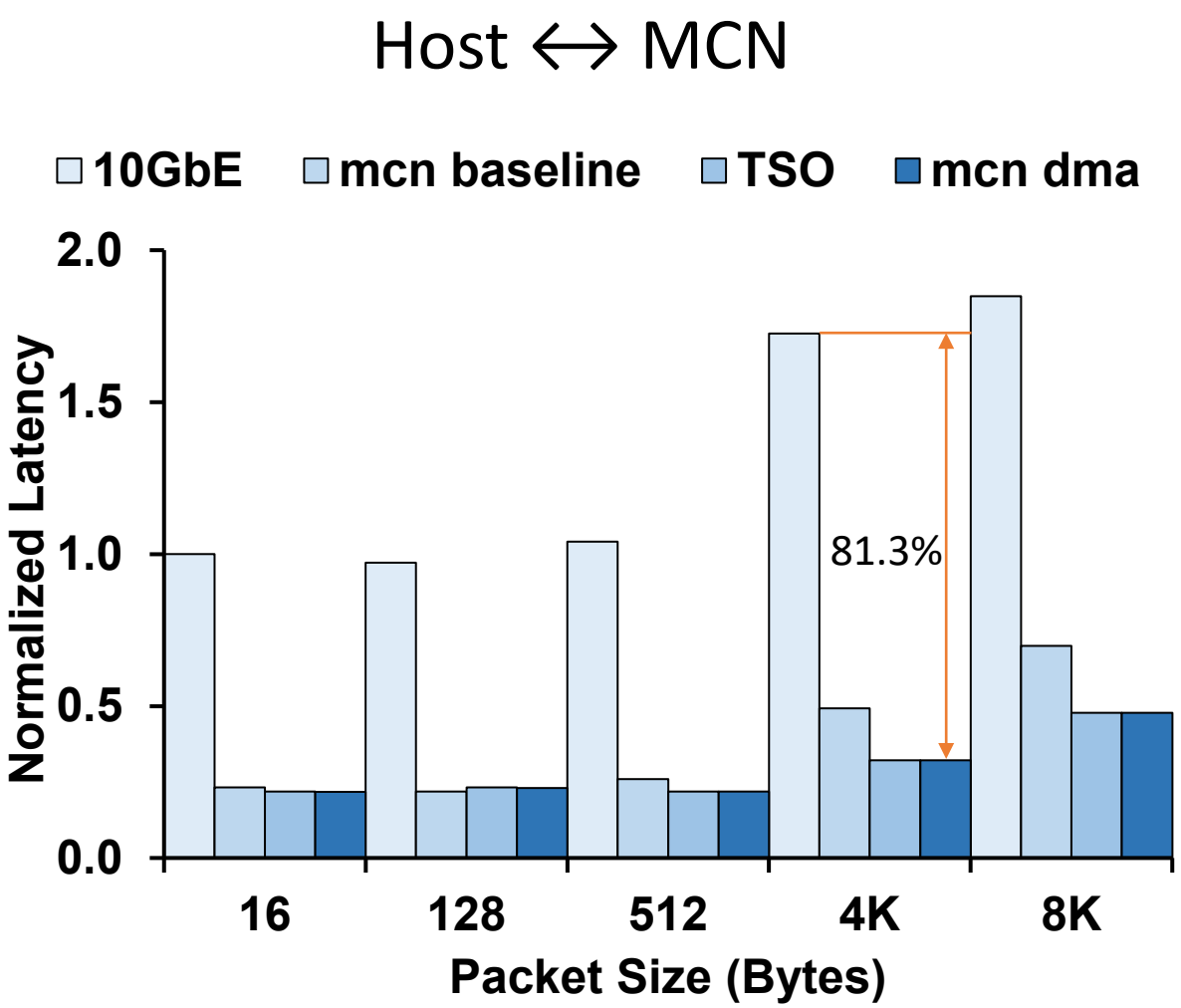
Evaluation – Network Bandwidth (iPerf)

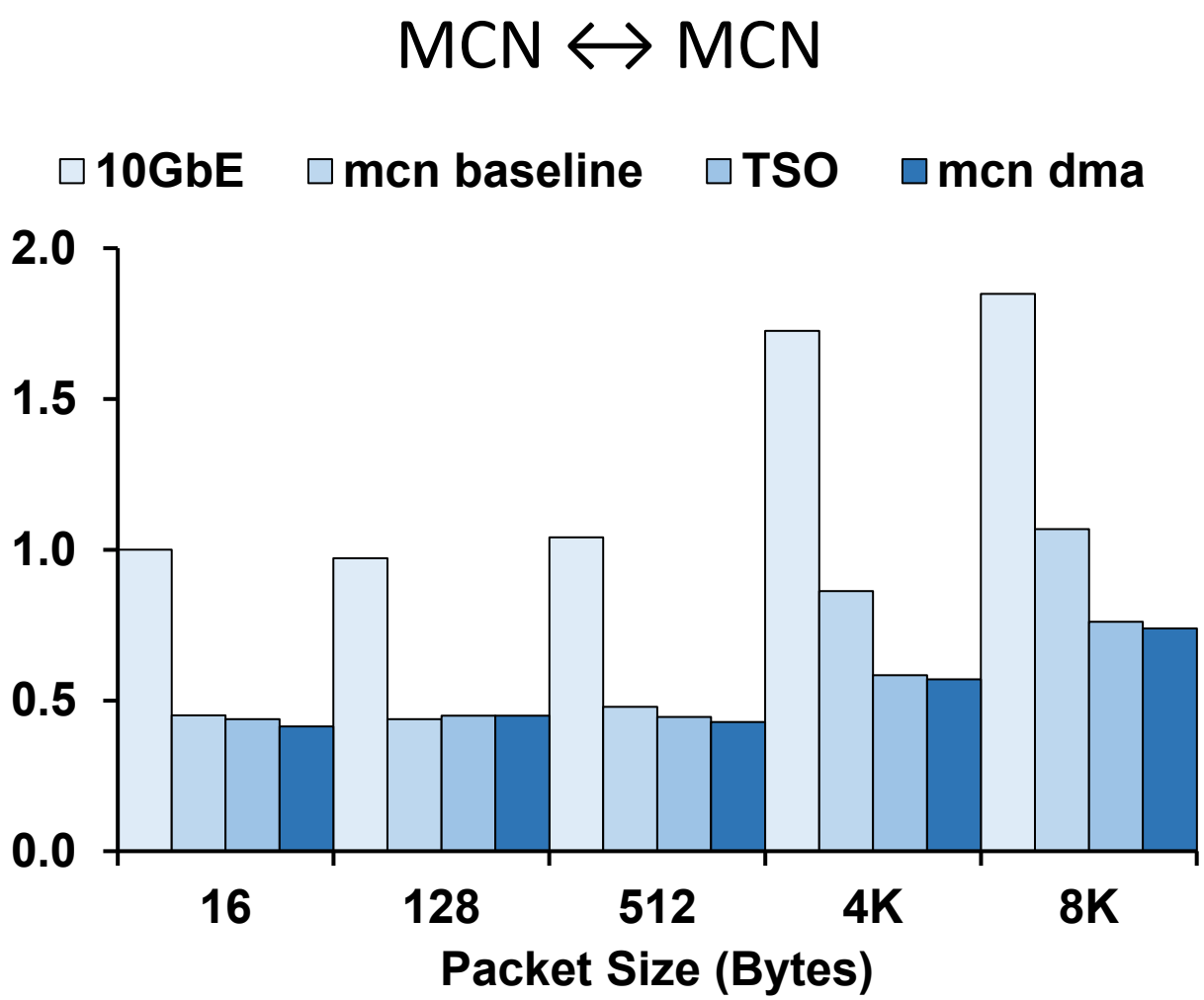


Evaluation – Network Bandwidth (iPerf)

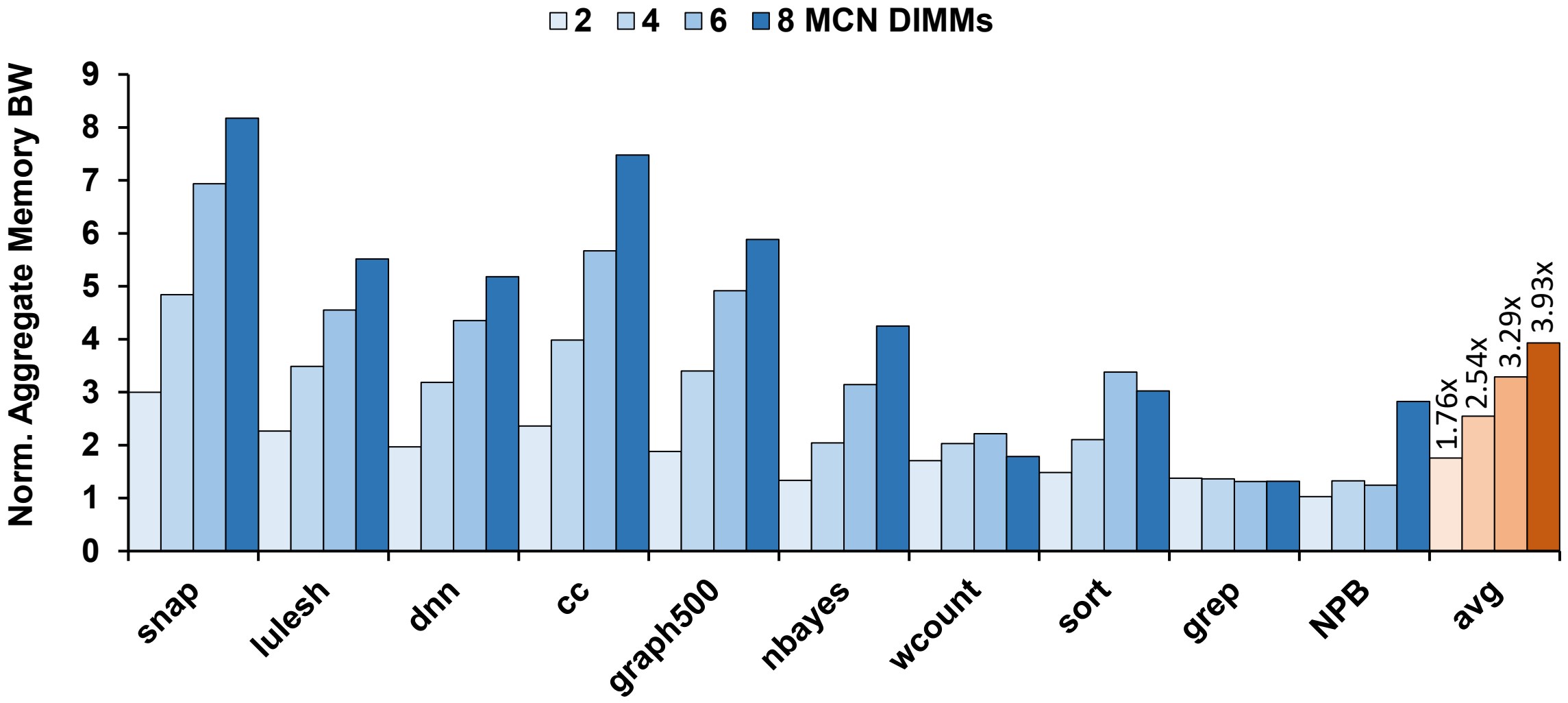


Evaluation – Network Latency (Ping)





Evaluation – Aggregate Memory Bandwidth



Outline

- **MCN Architecture**
  - ✓Background – Buffered DIMM
  - ✓Design Overview
  - ✓MCN DIMM Architecture
  - ✓MCN Packet Routing
- Design Optimizations
- **Proof of Concept – Hardware Demonstration**
- **Evaluation**
  - ✓Simulation Methodology
  - ✓Bandwidth and Latency
- **Conclusion**

Conclusion

- MCN is a novel near-memory processing that:
  - ✓Can run **unmodified** user applications.
  - ✓Requires **no hardware and OS modifications** of the host systems.
  - ✓Supports better **scalability** of distributed computing.
- We showed a proof-of-concept with hardware demonstration
- MCN can provide:
  - ✓4.56x higher network bandwidth
  - ✓78.1% lower network latency
  - ✓8.17x higher aggregate DRAM bandwidth

  than the conventional systems

Application Transparent Near-Memory Processing Architecture with Memory Channel Network

Mohammad Alian, Seung Won Min, Hadi Asgharimoghaddam, Ashutosh Dhar, Dong Kai Wang, Thomas Roewer, Adam McPadden, Oliver O'Halloran, Deming Chen, Jinjun Xiong, Daehoon Kim, Wen-mei Hwu, Nam Sung Kim

University of Illinois Urbana-Champaign

IBM Research and Systems

ILLINOIS

Electrical & Computer Engineering

COLLEGE OF ENGINEERING

C3SR center for cognitive computing systems research

IBM | ILLINOIS

References

[1] Intel, [Online]. Available: https://newsroom.intel.com/editorials/data-centric-innovation-summit
[2] B. Sukhwani et al., “Contutto: A Novel FPGA-based Prototyping Platform Enabling Innovation in the Memory Subsystem of a Server Class Processor,” in Proceedings of the 50th Annual IEEE/ACM International Symposium on Microarchitecture, ser. MICRO-50 ’17. New York, NY, USA: ACM, 2017.

[3] A. Mohammad et al., “dist-gem5: Distributed simulation of computer clusters,” in Performance Analysis of Systems and Software (ISPASS), 2017 IEEE International Symposium on. IEEE, 2017.

[4] “Iperf: The ultimate speed test tool for TCP, UDP and SCTP.” [Online]. Available: https://iperf.fr/

[5] “Coral benchmark codes.” [Online]. Available: https://asc.llnl.gov/CORAL-benchmarks/

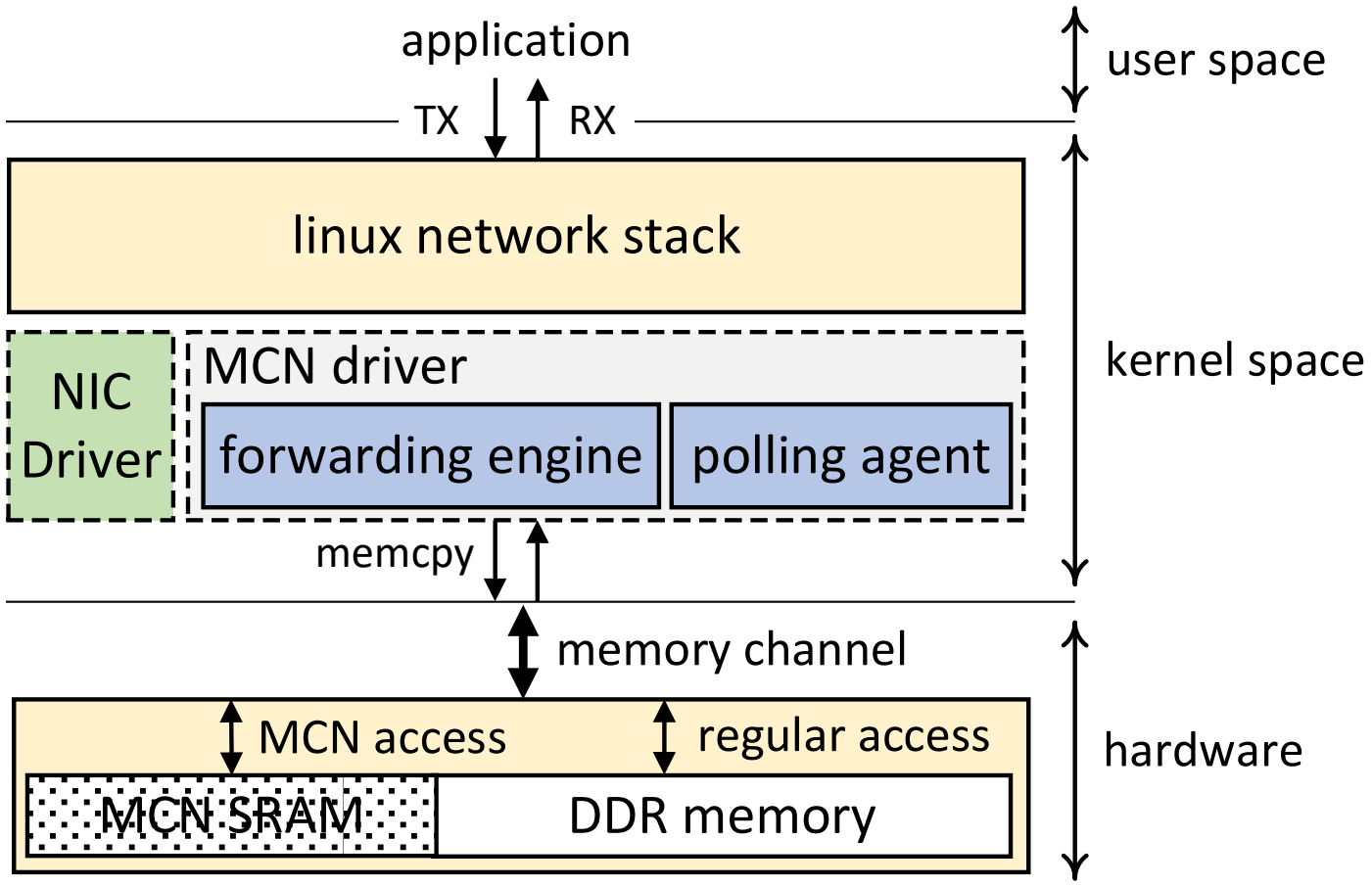
[6] L. Wang et al., “Bigdatabench: A big data benchmark suite from Internet services,” in High Performance Computer Architecture (HPCA), 2014 IEEE 20th International Symposium on. IEEE, 2014.

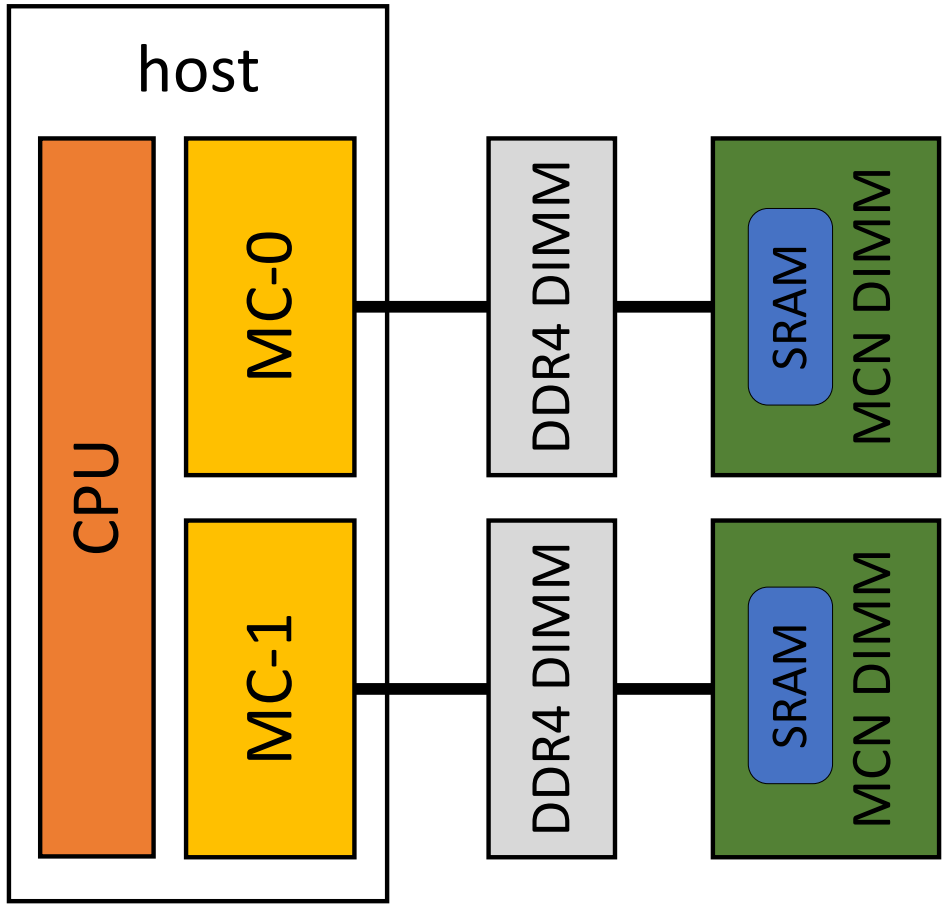
[7] D. H. Bailey et al., “The NAS parallel benchmarks,” The International Journal of Supercomputing Applications, vol. 5, 1991.

Backup slides

MCN Packet Routing

- MCN → MCN

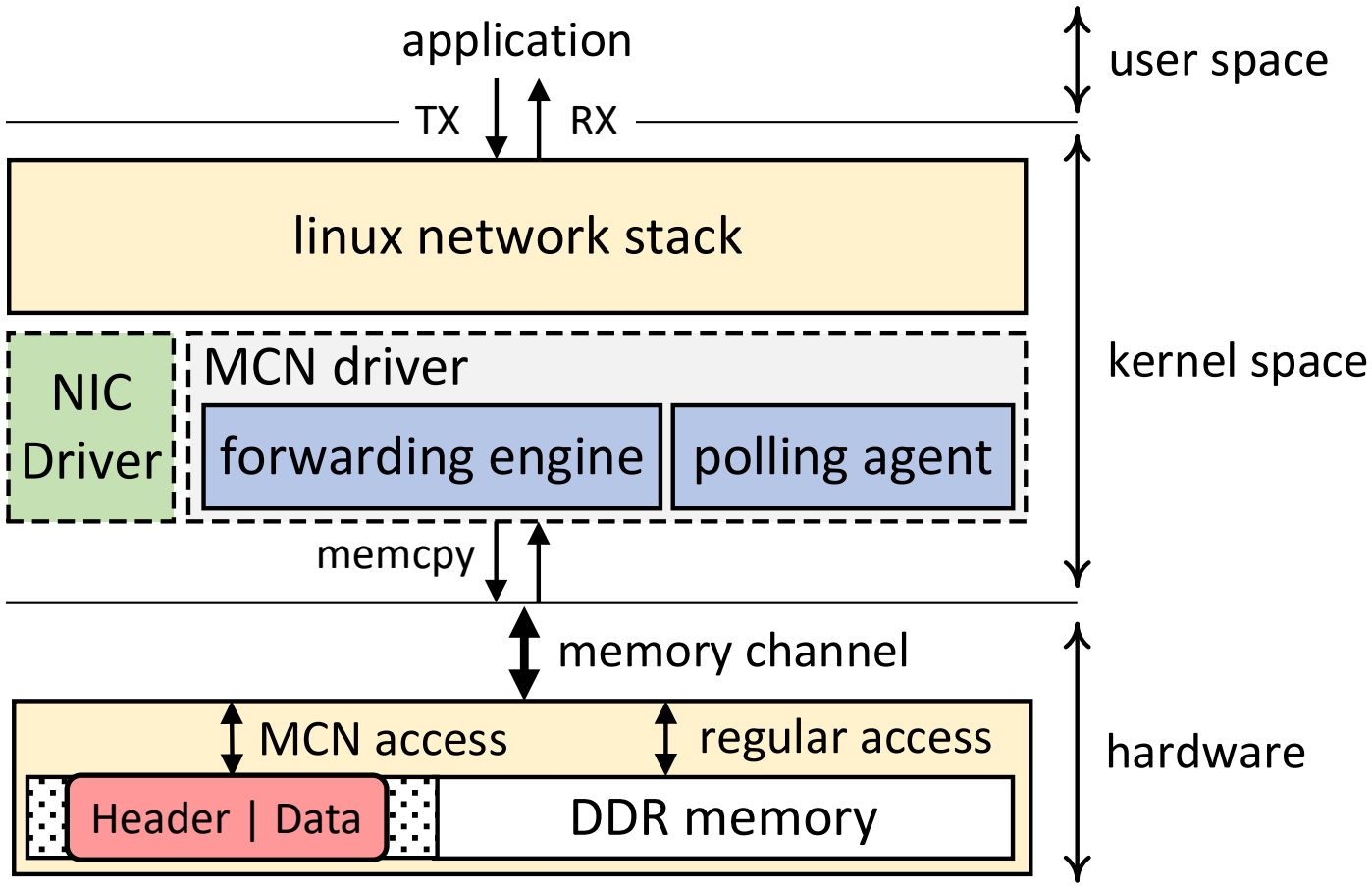


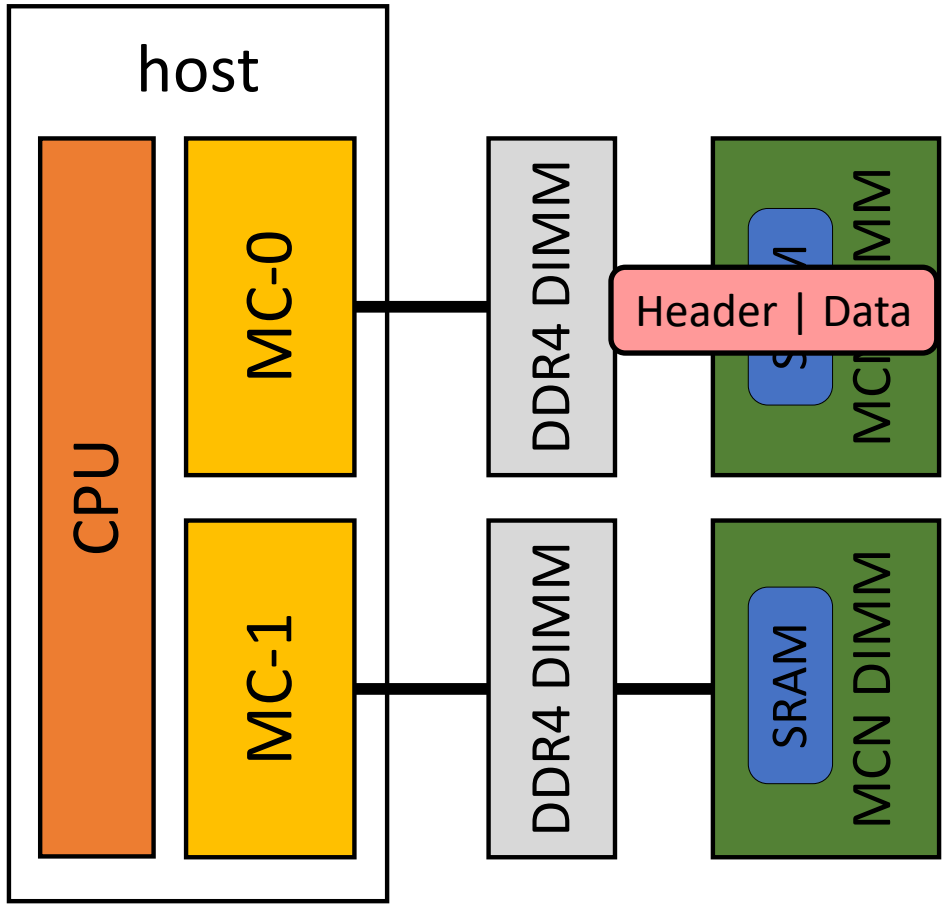


MCN Packet Routing

- MCN → MCN

1. MCN DIMM writes Ethernet Packet to its SRAM buffer and set RX-poll bit

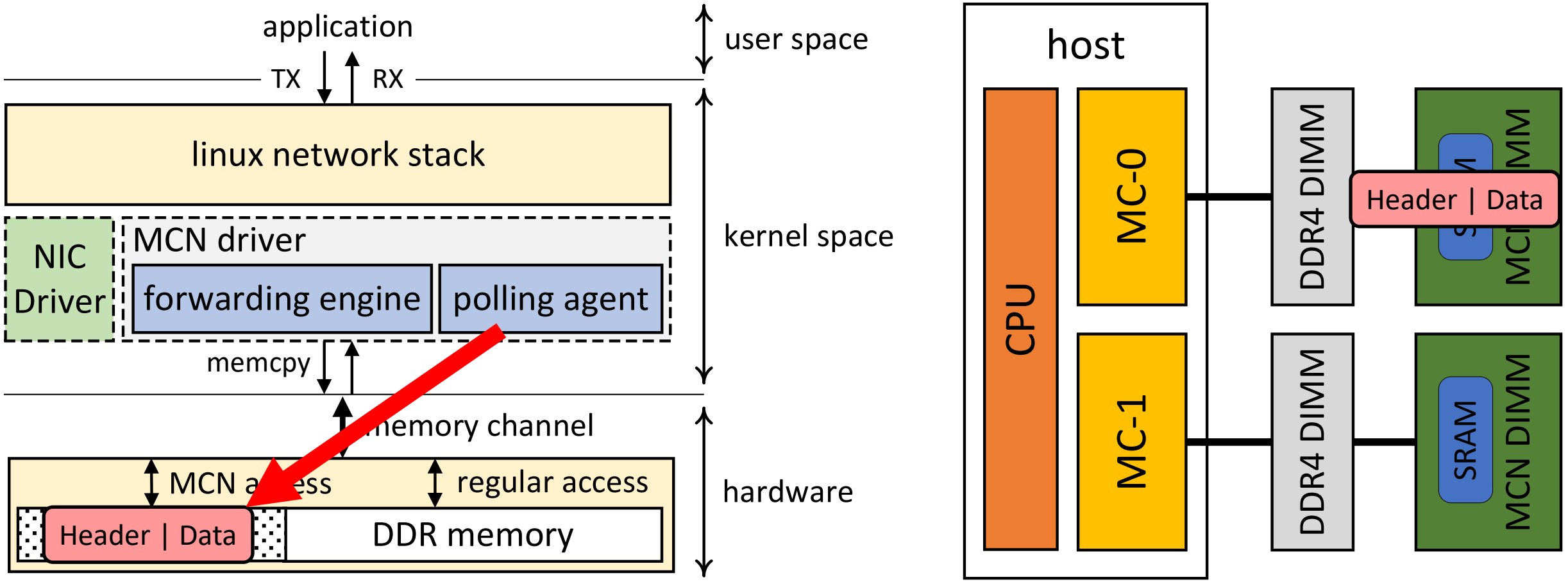




MCN Packet Routing

- MCN → MCN

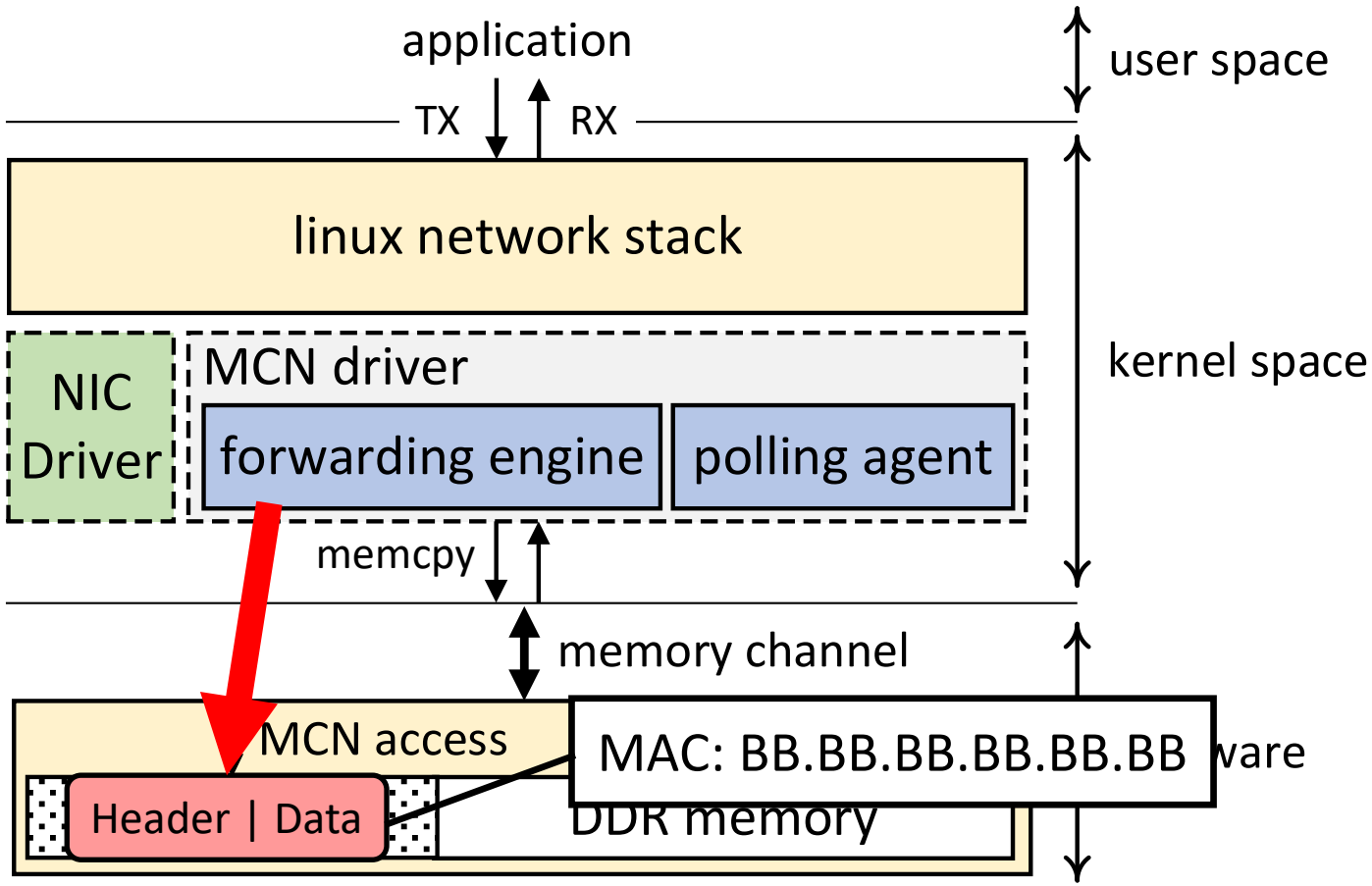
2. Polling agent from the host recognize the RX-poll bit is set and read the incoming packet

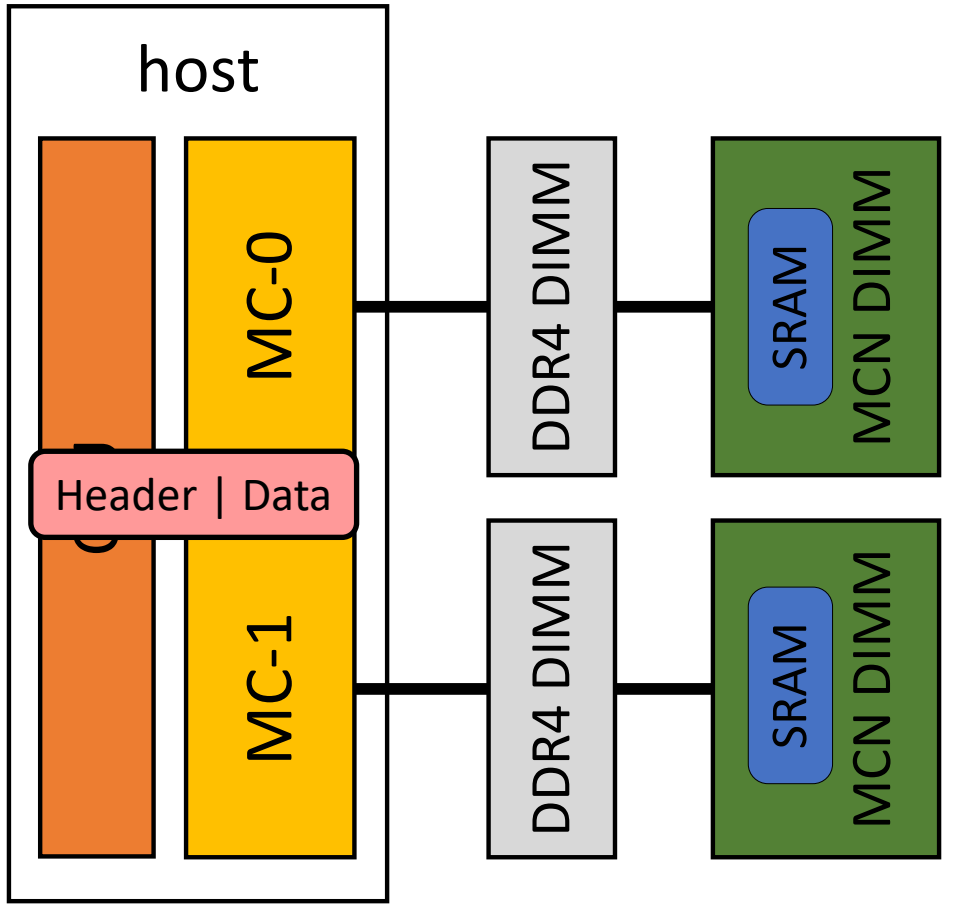


MCN Packet Routing

- MCN → MCN

3. Forwarding engine checks the MAC address and determine the destination

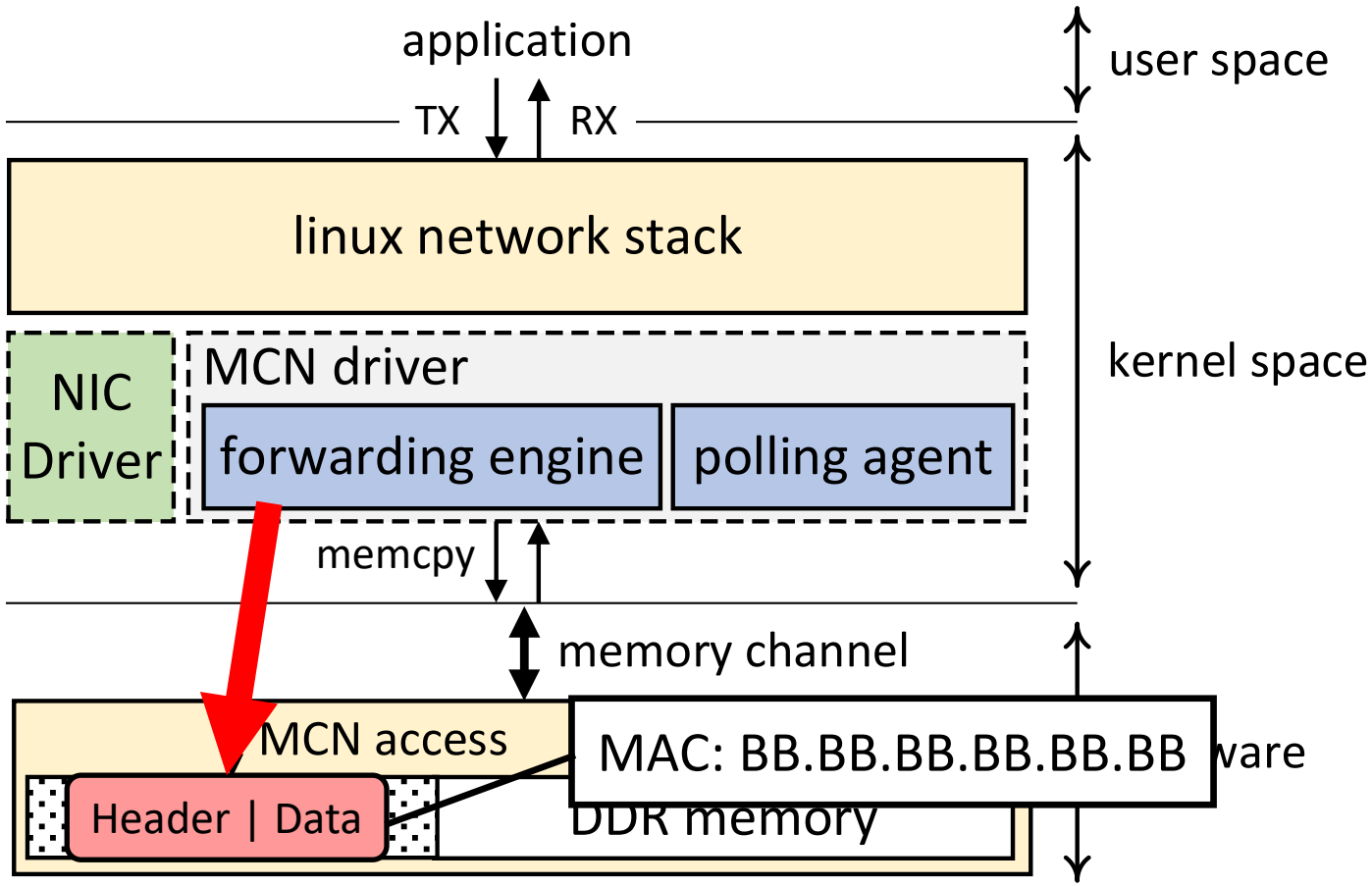


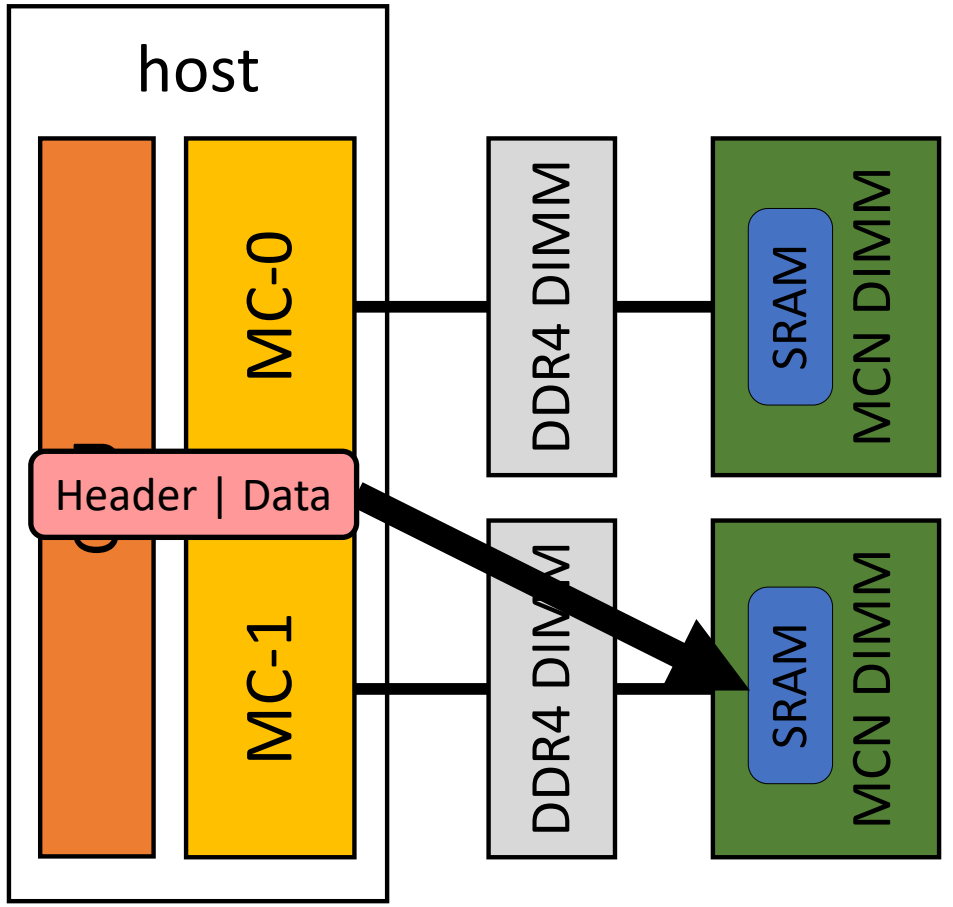


MCN Packet Routing

- MCN → MCN

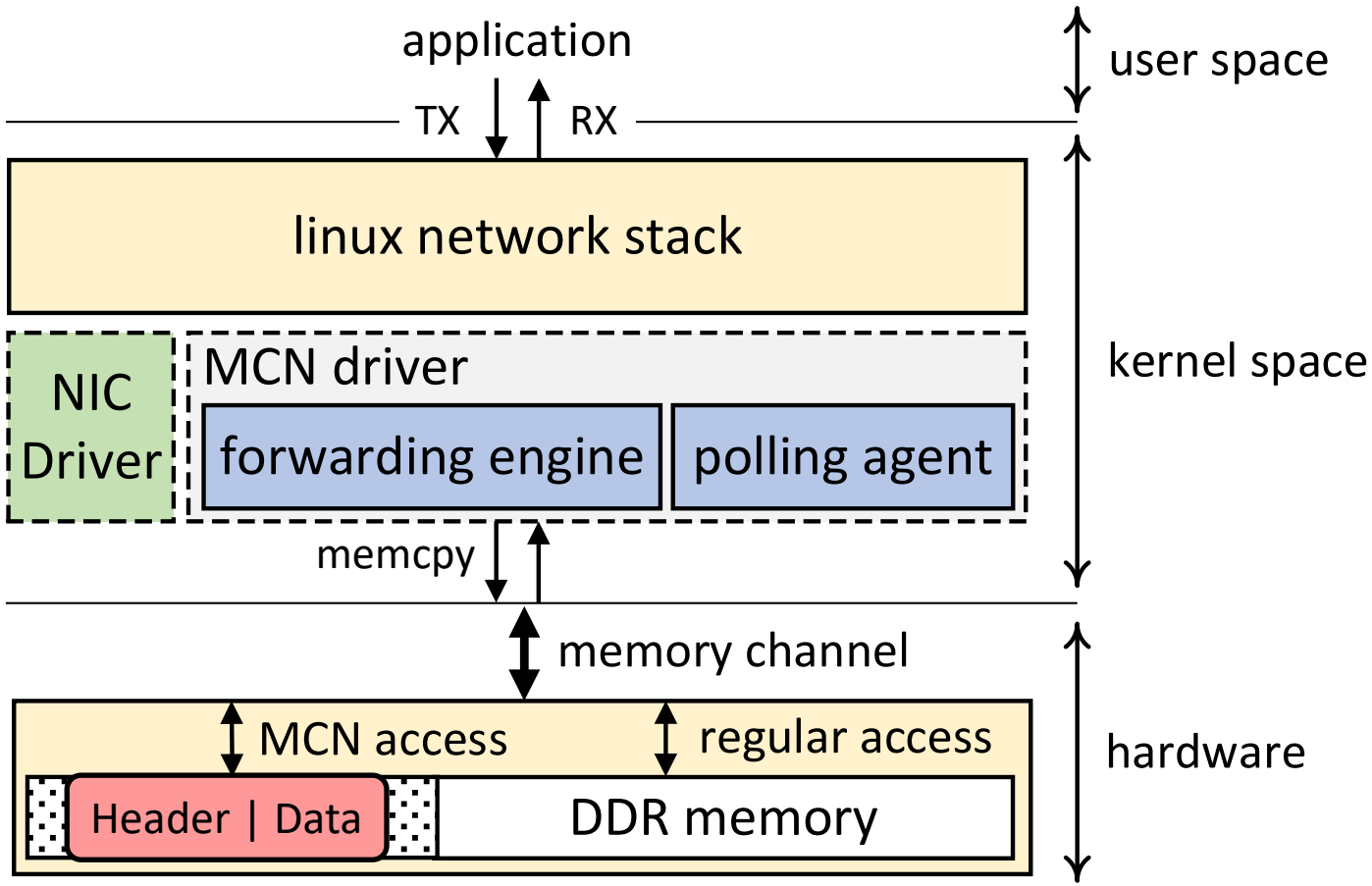
4. If this is for another MCN DIMM, it copies the Ethernet packet to the other MCN DIMM's SRAM buffer



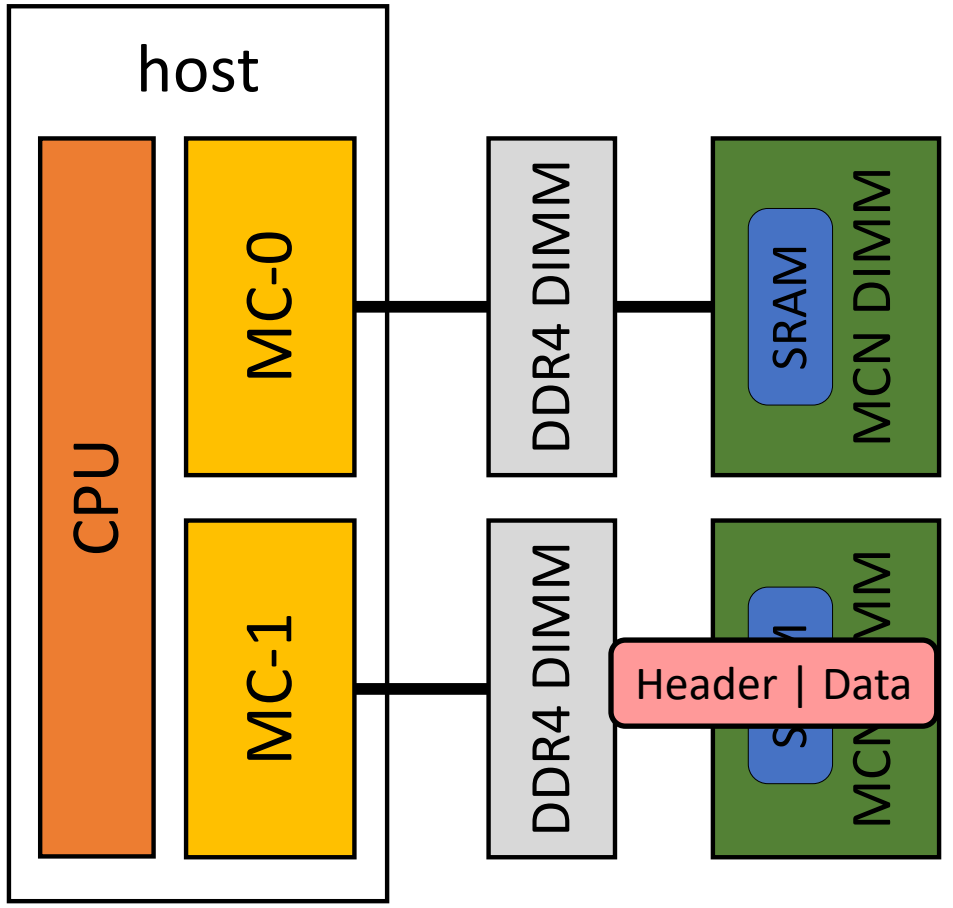


MCN Packet Routing

- MCN → MCN



5. This data copy fires IRQ from the MCN DIMM and MCN processor knows there is an incoming packet



Communication latency breakdown

- DMA operation is expensive
- PCI-Express is not designed for extra low latency communication
  - ✓4KB data transfer
    - **5us** PCIe x8 Gen3 vs. **200ns** DDR4 MC
- Latency breakdown for a 64B TCP/IP packet

Category	Percentage
Link and Switch	11.53
NIC	2.43
DMA over PCIe	38.05
Driver	14.78
Network Stack Processing	25.10
Copy to/from userspace	8.10

52.03%

MCN Bandwidth

MCN-Host

10GbE	MCN baseline	MCN-INT	MCN-CSUM	MCN-JUMBO	MCN-TSO	MCN-DMA
7.22	9.41	9.35	9.63	19.22	25.26	33.00

MCN-MCN

10GbE	MCN baseline	MCN-JUMBO	MCN-TSO	MCN-DMA
7.22	7.79	17.02	20.90	26.08