IEEE International Conference on Cluster Computing
Newport Beach, California, USA October 8-11, 2001

DECK-SCI: High-Performance Communication and Multithreading for SCI Clusters

Federal University of Rio Grande do Sul
Institute of Informatics
Group of Parallel and Distributed Processing
Porto Alegre, RS, Brazil

Catholic University of Rio Grande do Sul
High-Performance Research Center
Porto Alegre, RS, Brazil

Fábio Abreu Dias de Oliveira
Rafael Bohrer Ávila
Marcos Ennes Barreto
Philippe Olivier Alexandre Navaux

César Augusto Fonticielha De Rose

Outline

- ⇒ Introduction and motivations
- ⇒ Overview of DECK-SCI
- ⇒ Key characteristics of DECK-SCI's communication protocols
- ⇒ Description and evaluation of the communication protocols
- ⇒ Concluding remarks

Introduction and motivations (1)

⇒ SCI (Scalable Coherent Interface) as a high-performance network for clusters

- ⇒ low latency (direct access to remote memory addresses)
  - communication via CPU loads and stores
- ⇒ high bandwidth

⇒ Several efforts have been made towards offering message-passing libraries for SCI clusters

- ⇒ SCIPVM
- ⇒ PVM-SCI
- ⇒ ScaMPI
- ⇒ SCI-MPICH
- ⇒ CML (Common Messaging Layer)

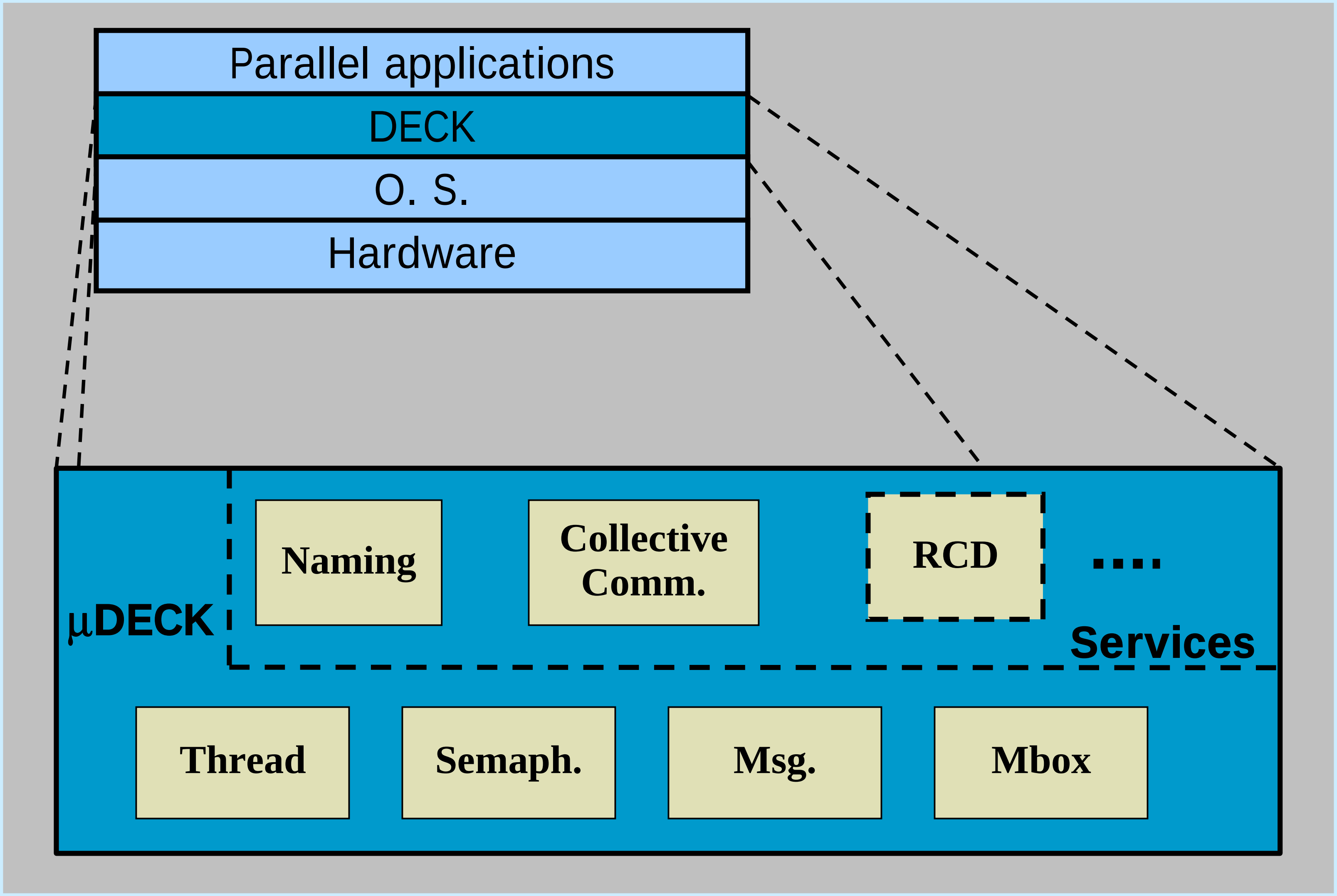
Introduction and motivations (2)

⇒ Previous implementations of DECK
  ⇒ Fast Ethernet
  ⇒ Myrinet
⇒ Main motivations for developing DECK-SCI
  ⇒ to allow the integration between Myrinet and SCI clusters
    - to support to the “MultiCluster” model
  ⇒ to devise a new library for programming SCI clusters based on
    - very efficient message-passing
    - multithreading at the API level

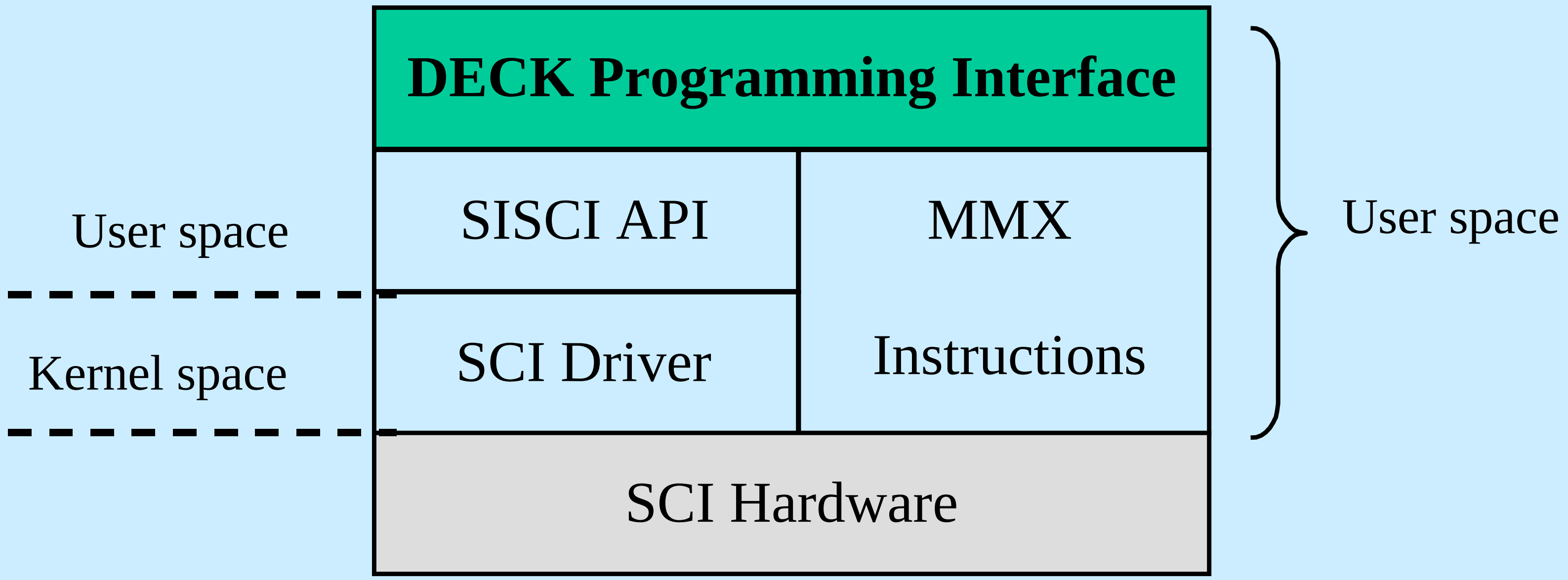
Overview of DECK (1)

- ⇒ DECK: ***Distributed and Execution Communication Kernel***
- ⇒ SPMD model + multithreading
- ⇒ Basic abstractions
  - ⇒ messages and mail boxes
    - point-to-point communication
  - ⇒ threads and semaphores

Overview of DECK (2)



Overview of DECK-SCI



Devised and implemented communication protocols

⇒ Objectives

- ⇒ to compose a communication kernel aimed at fully exploiting the high-performance capabilities of the SCI network
- ⇒ to obtain a performance near the limits of the SCI hardware
  - low latency for short messages
  - high bandwidth for large ones

⇒ Hardware platform

- ⇒ Cluster composed of four PCs equipped with dual Pentium-III (500 MHz), 256 Mbytes of RAM and SCI NICs for the PCI bus (32 bits and 33 MHz)

⇒ Three different protocols transparently chosen according to the size of the message to be sent

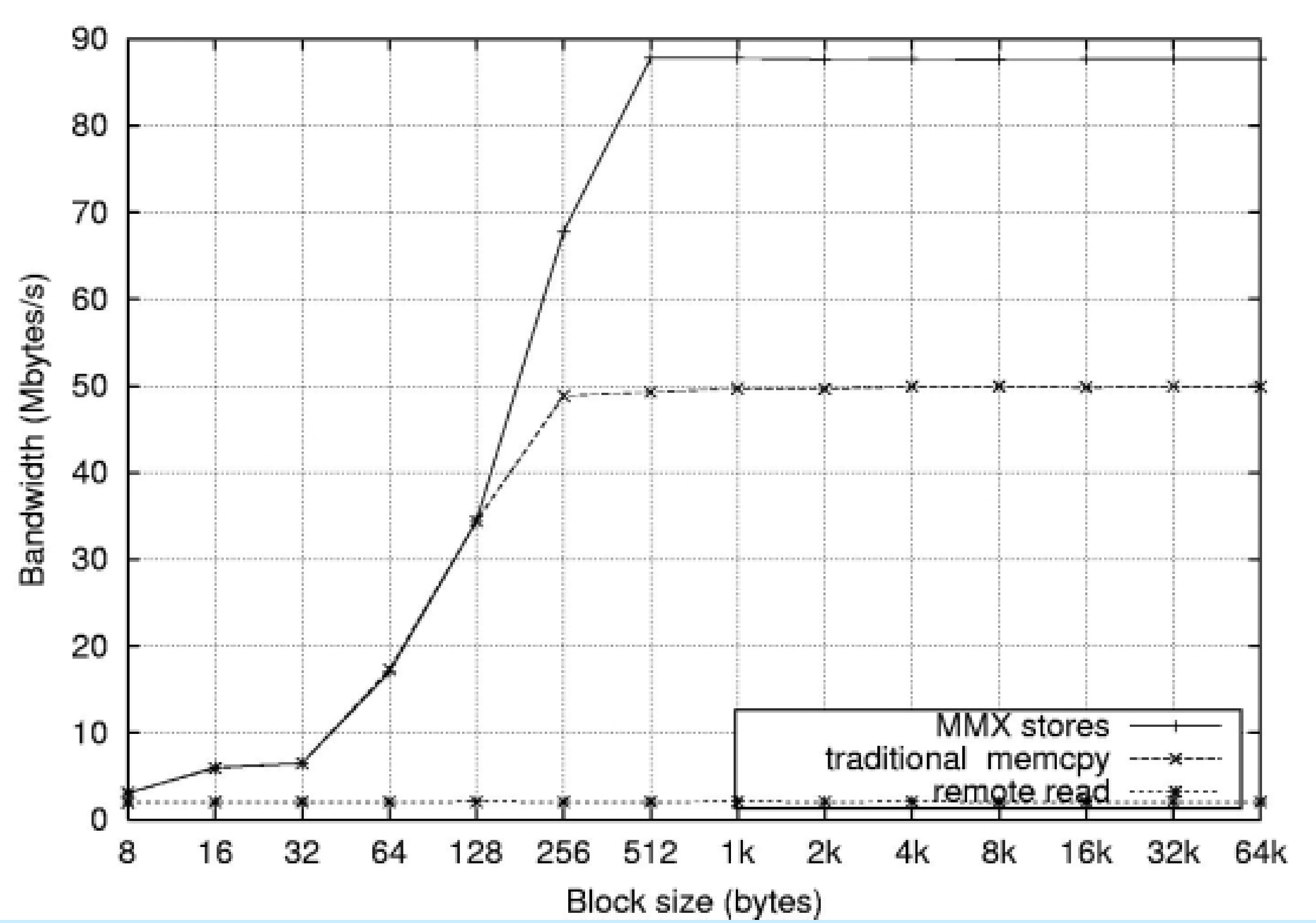
Key features of the implemented communication protocols (1)

⇒ Communication strictly accomplished by remote writes

- ⇒ **write-only protocols**

⇒ Use of MMX instructions for message transfers, in order to increase the maximum achievable bandwidth, by

- ⇒ **avoiding the traditional memcpy routine**
- ⇒ **generating a write burst on the PCI bus**



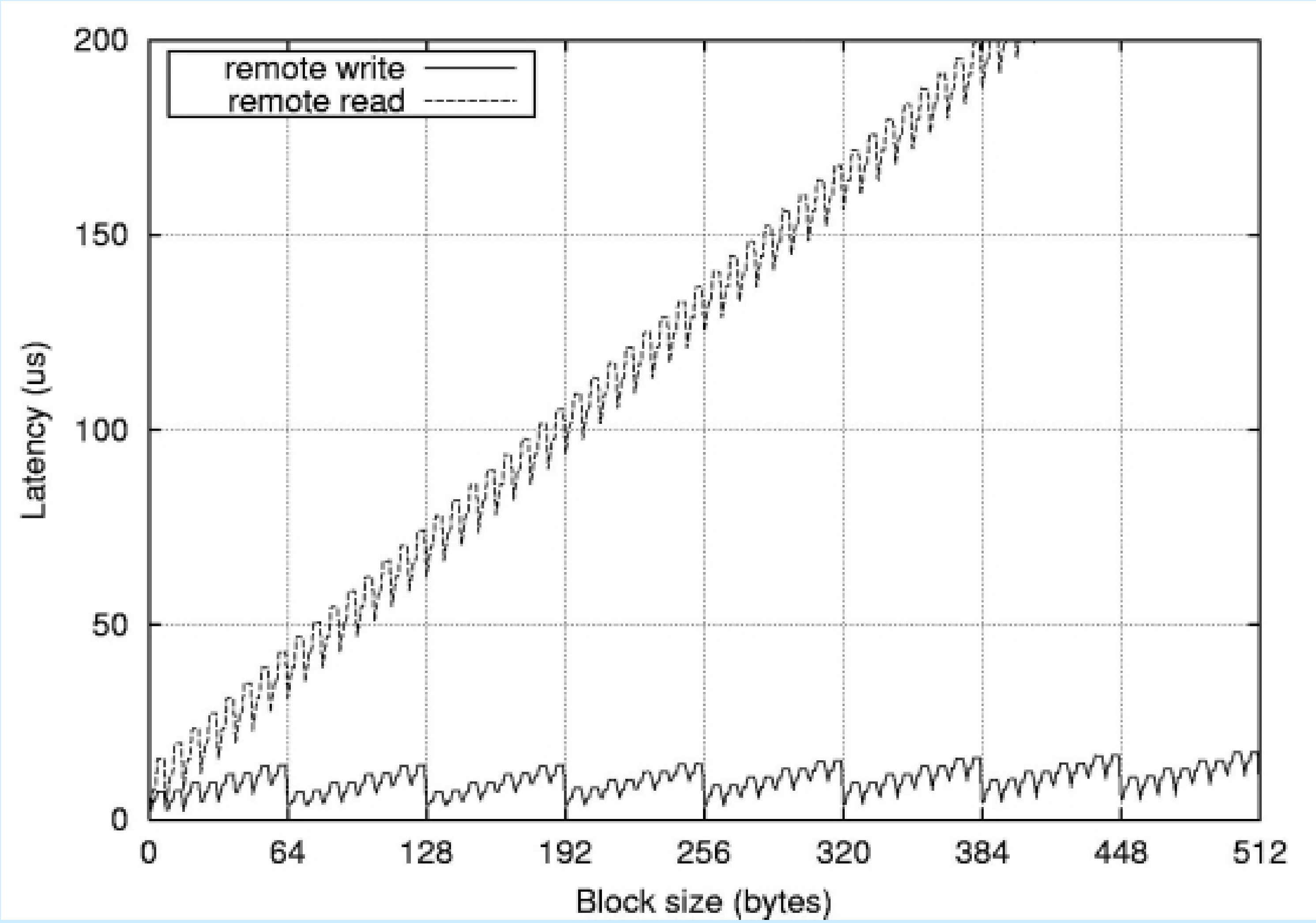
Bandwidth (Mbytes/s)
0
10
20
30
40
50
60
70
80
90
8
16
32
64
128
256
512
1k
2k
4k
8k
16k
32k
64k
Block size (bytes)
MMX stores
traditional memcpy
remote read

Key features of the implemented communication protocols (2)

⇒ Exploitation of the PCI-SCI adapter's stream buffers

⇒ eight stream buffers with 64 bytes

- pipeline with eight stages



remote write
remote read
Latency (us)
200
150
100
50
0
0
64
128
192
256
320
384
448
512
Block size (bytes)

Key features of the implemented communication protocols (3)

- ⇒ Polling-based message passing
  - ⇒ **avoid interrupts for signaling the arrival of a message**
- ⇒ Thread-awareness and thread safety

“Protocol 1”: short messages

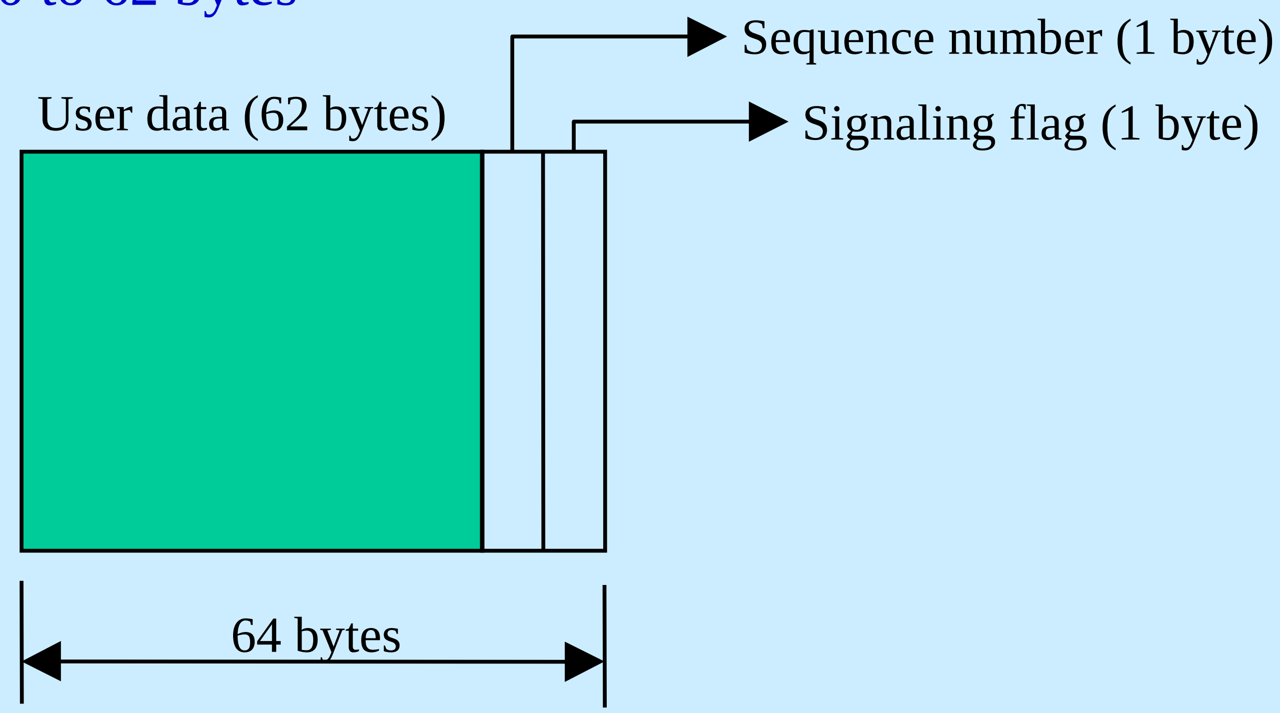
⇒ Objective: very low-latency for short messages

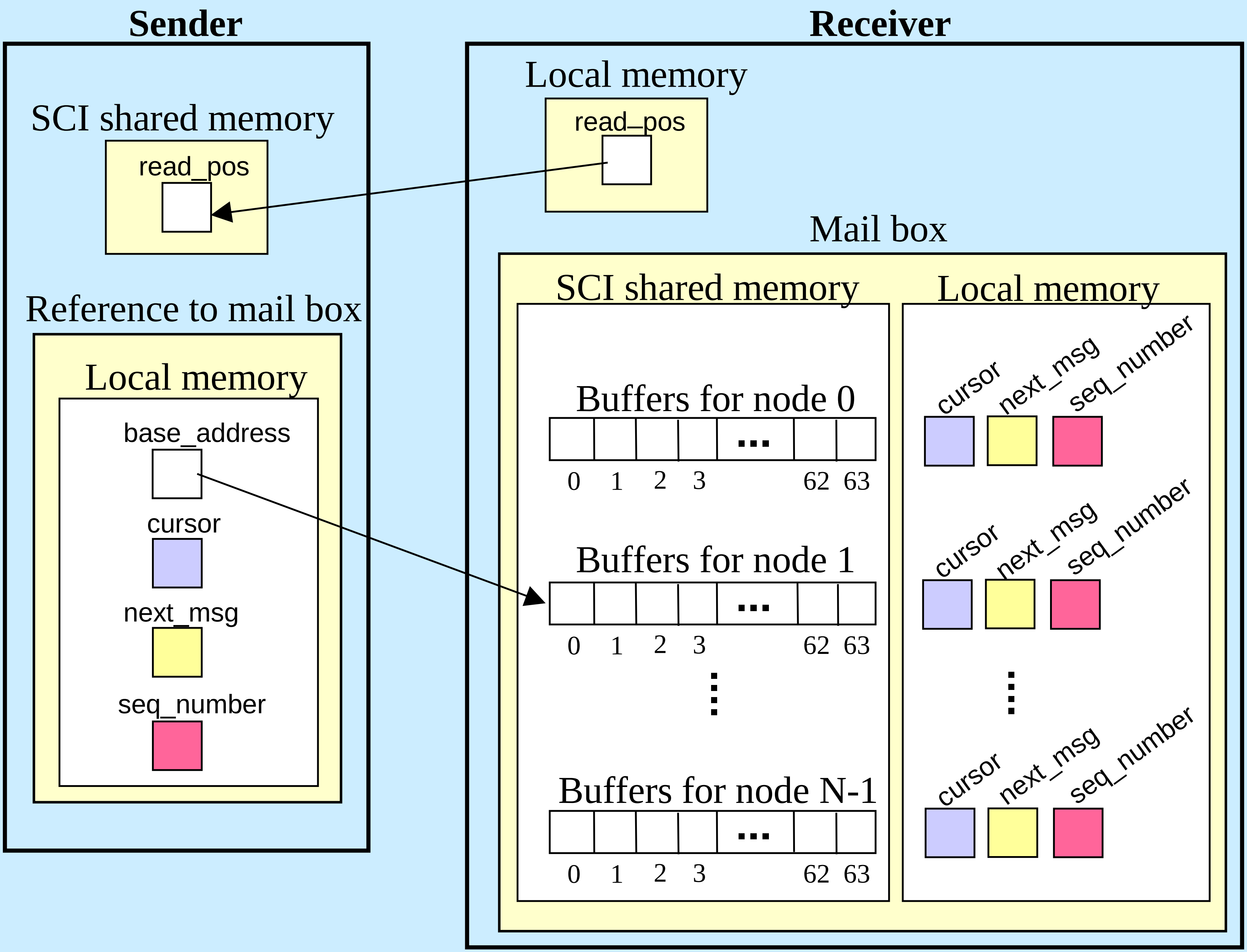
⇒ **minimal overhead**

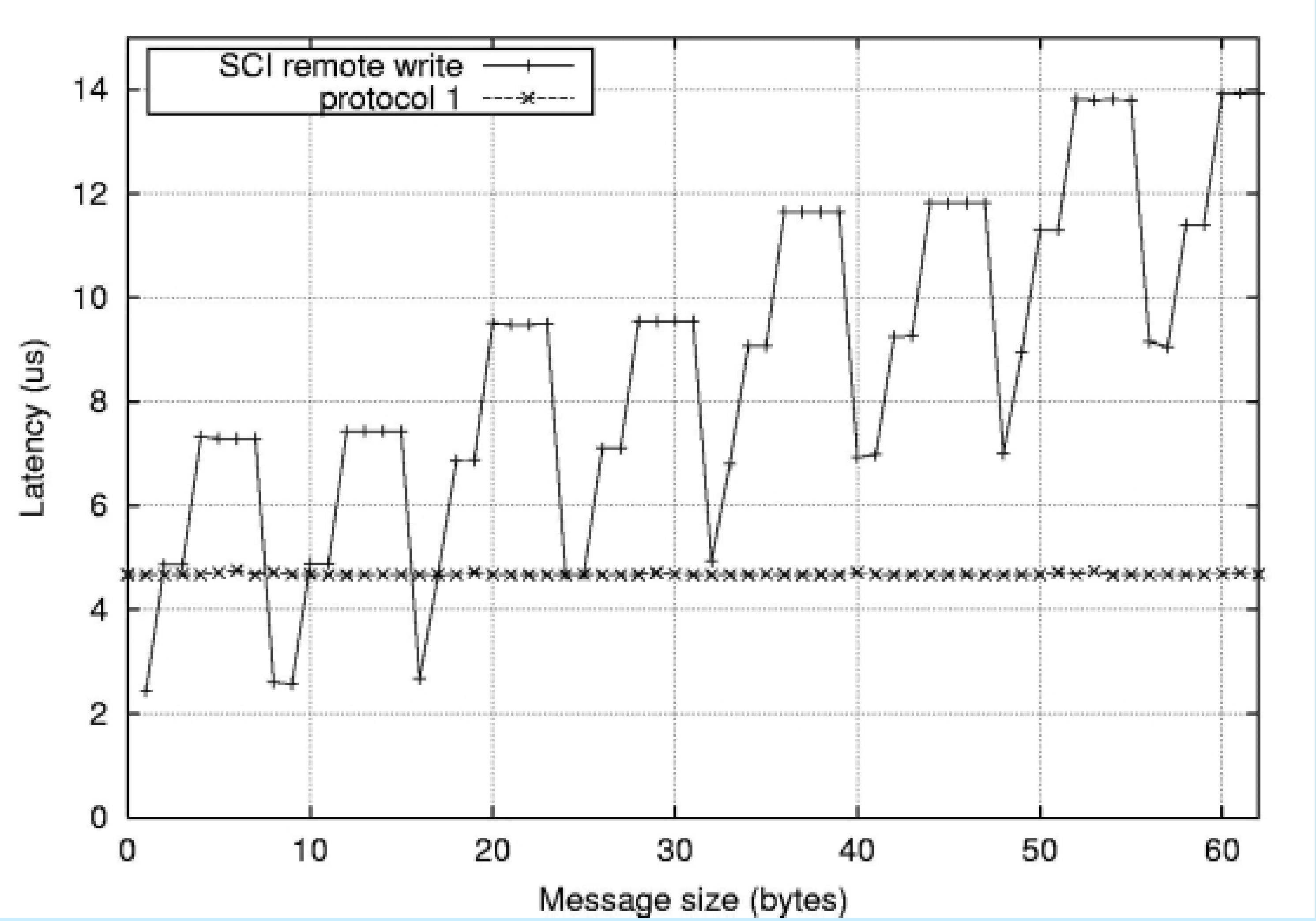
⇒ **signaling and communication using a single SCI packet**

⇒ Specialized in exchanging short messages

⇒ 0 to 62 bytes




Sender
SCI shared memory
read_pos
Reference to mail box
Local memory
base_address
cursor
next_msg
seq_number
Receiver
Local memory
read_pos
Mail box
SCI shared memory
Buffers for node 0
0 1 2 3 ... 62 63
Buffers for node 1
0 1 2 3 ... 62 63
Buffers for node N-1
0 1 2 3 ... 62 63
Local memory
cursor
next_msg
seq_number

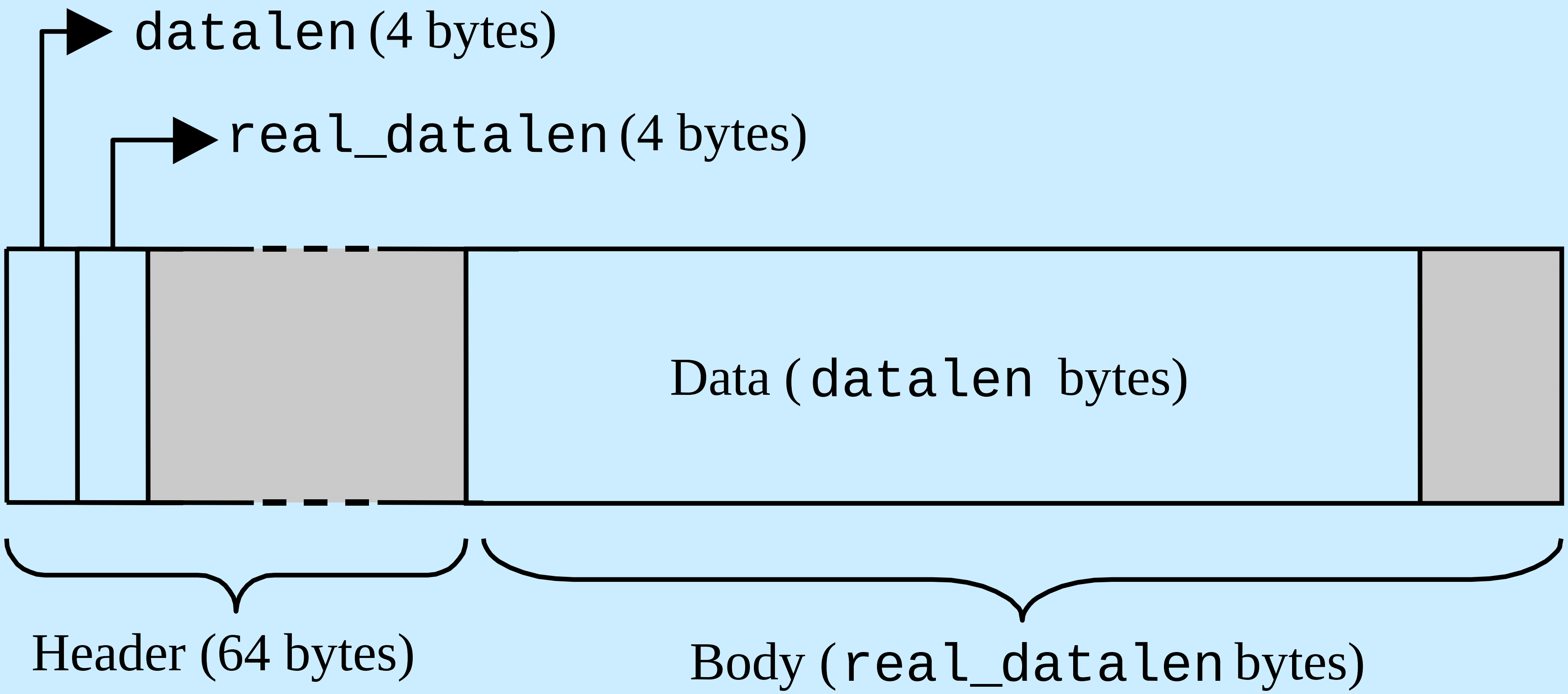


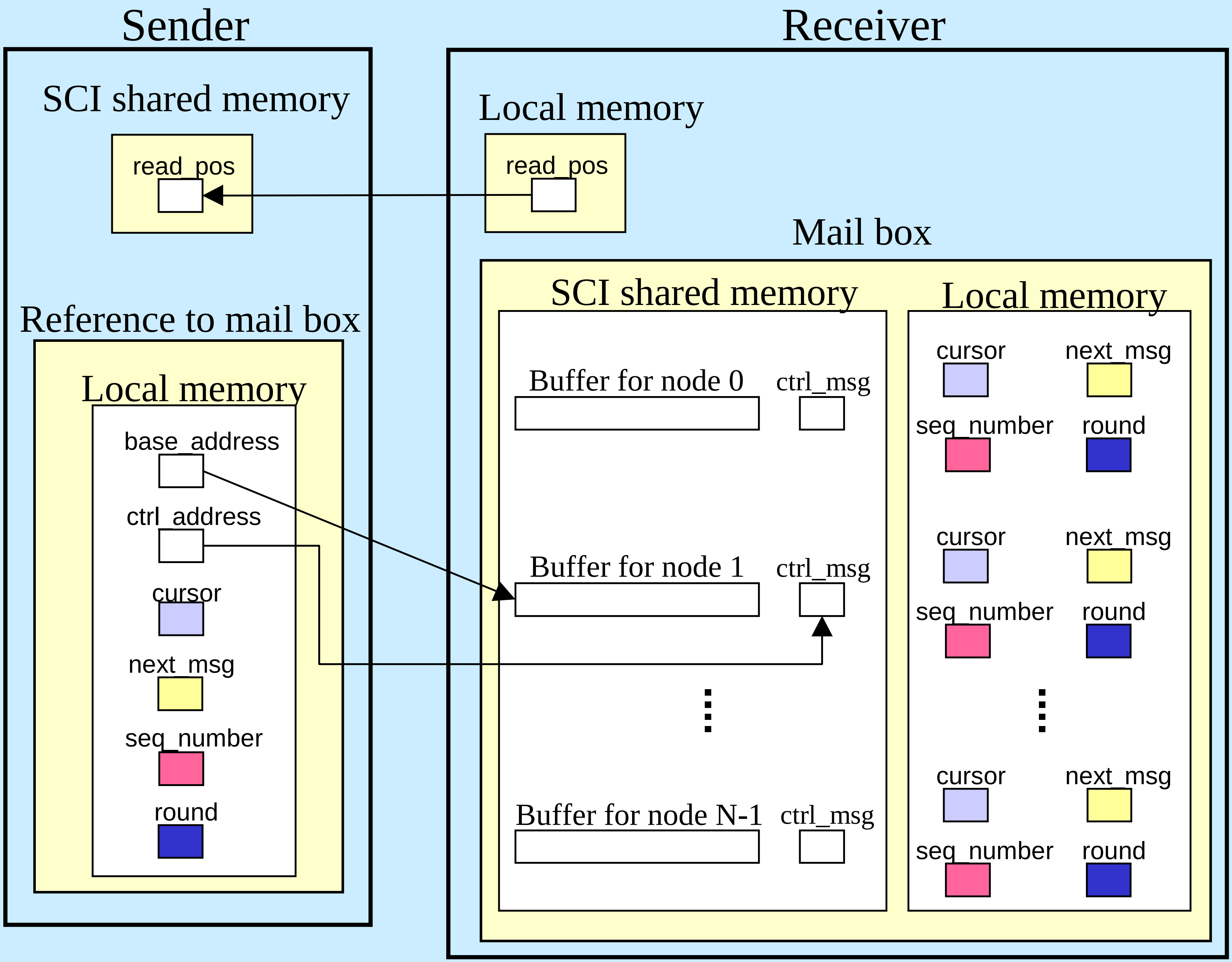
SCI remote write
protocol 1
Latency (us)
0
2
4
6
8
10
12
14
0
10
20
30
40
50
60
Message size (bytes)

“Protocol 2”: general-purpose

⇒ It handles messages greater than 62 bytes

⇒ Signaling with an additional SCI packet




Sender
SCI shared memory
read_pos
Reference to mail box
Local memory
base_address
ctrl_address
cursor
next_msg
seq_number
round
Receiver
Local memory
read_pos
Mail box
SCI shared memory
Buffer for node 0
ctrl_msg
Buffer for node 1
ctrl_msg
Buffer for node N-1
ctrl_msg
Local memory
cursor
next_msg
seq_number
round
cursor
next_msg
seq_number
round
cursor
next_msg
seq_number
round

“Protocol 3”: zero-copy mechanism (1)

⇒ Objective

  ⇒ to get as much as possible from the SCI network, boosting the bandwidth near the limits imposed by the SCI hardware

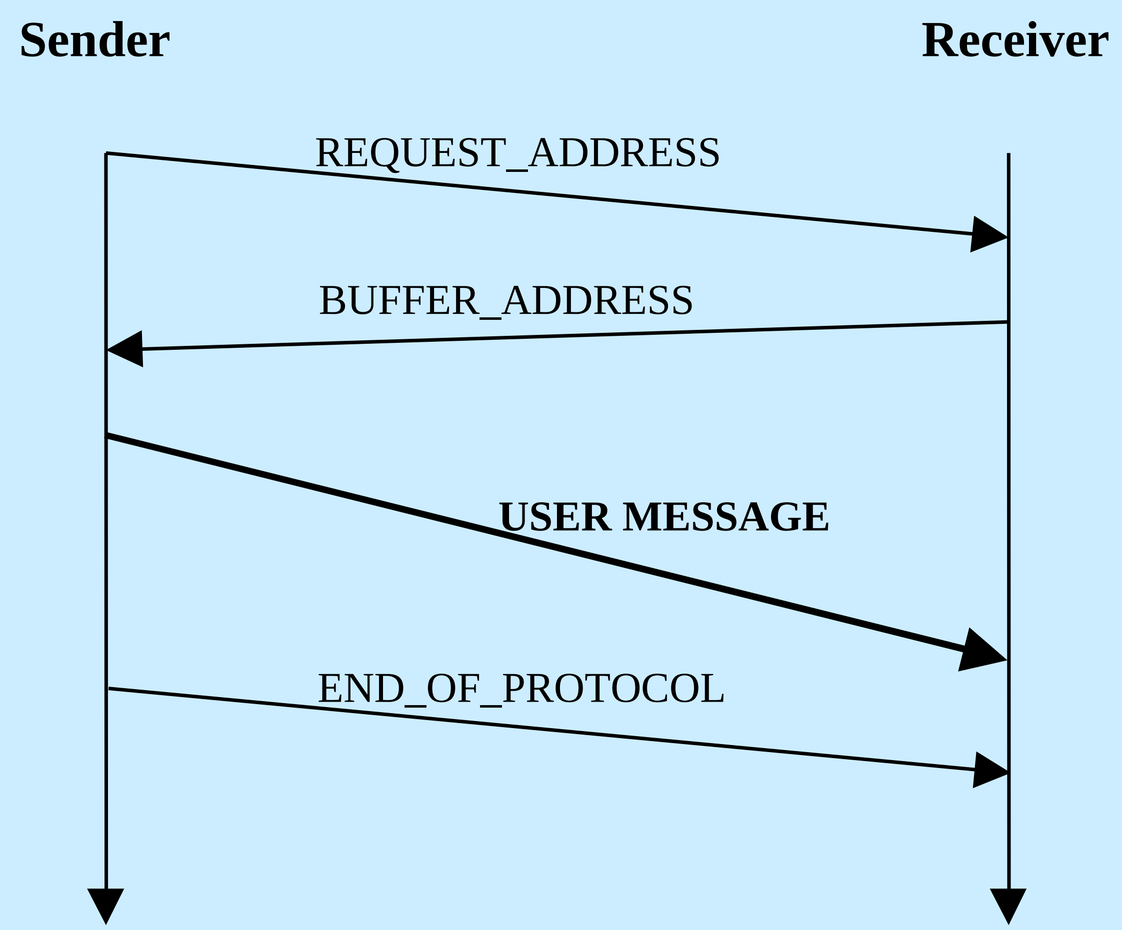
⇒ Messages are directly sent to the user buffer, without additional data movement into local memory

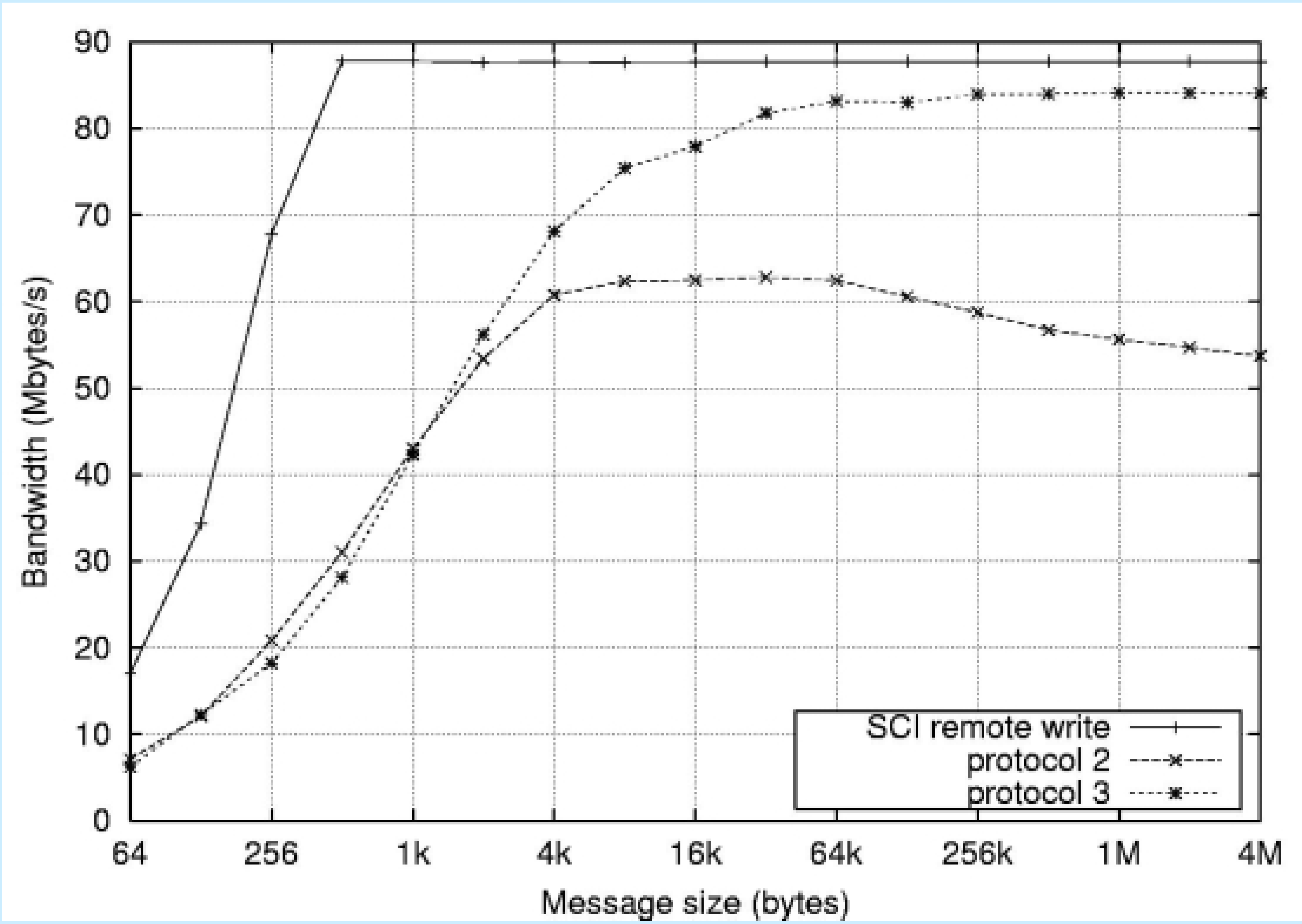
⇒ Handshaking between sender and receiver

  ⇒ synchronous protocol

  ⇒ there is no need of flow control

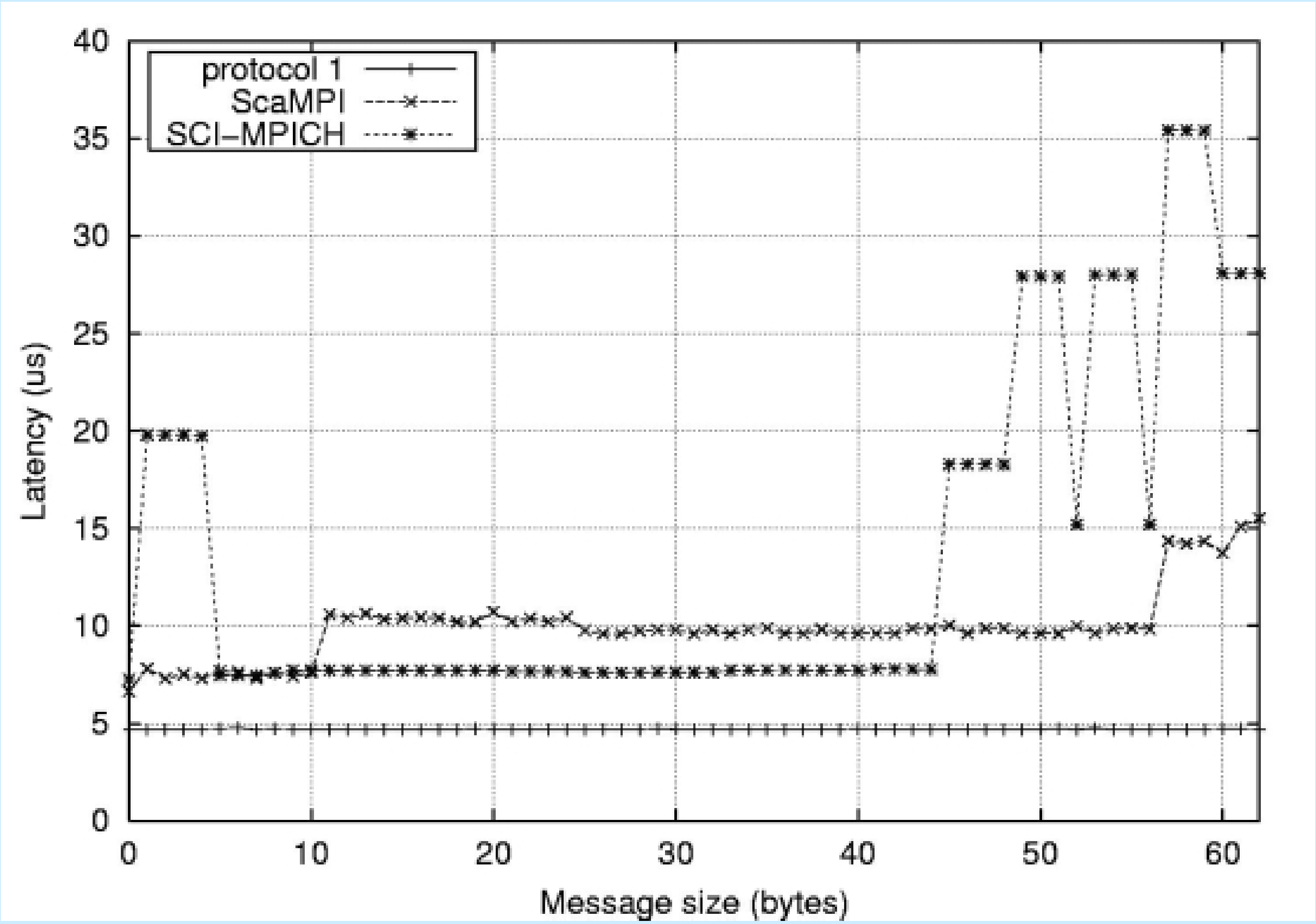
“Protocol 3”: zero-copy mechanism (2)



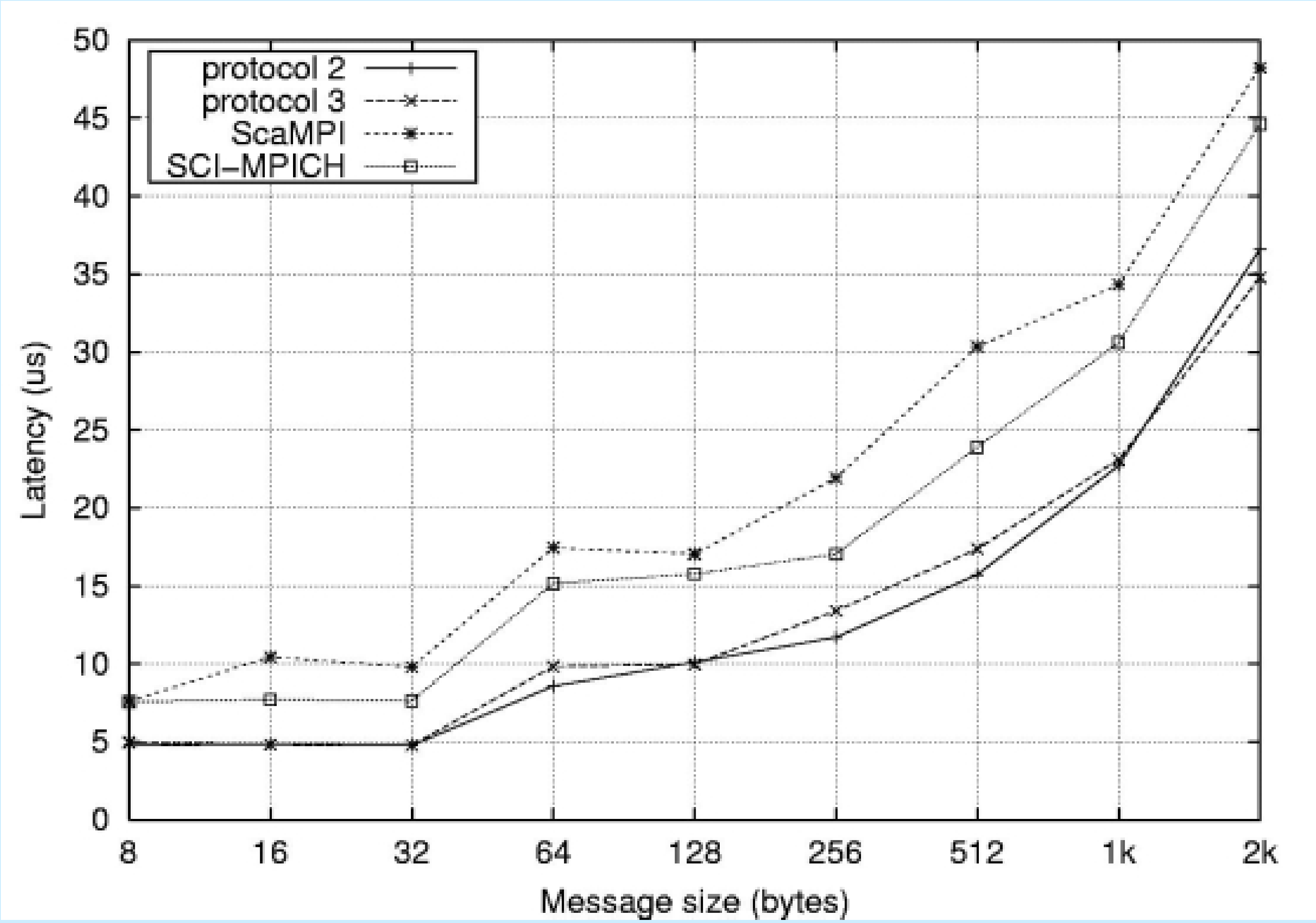


90
80
70
60
50
40
30
20
10
0
Bandwidth (Mbytes/s)
64
256
1k
4k
16k
64k
256k
1M
4M
Message size (bytes)
SCI remote write
protocol 2
protocol 3

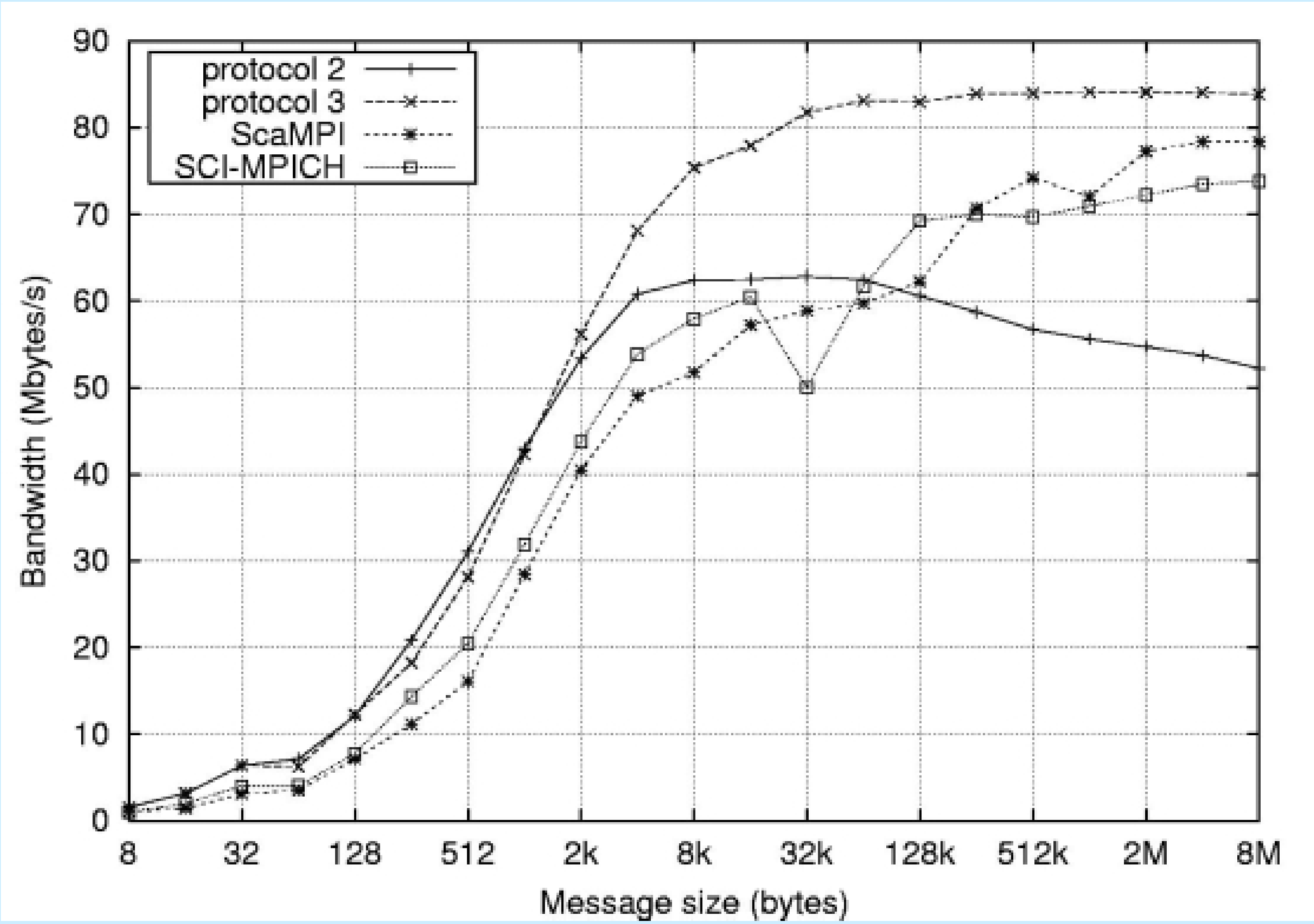
DECK-SCI, SCI-MPICH and ScaMPI (1)



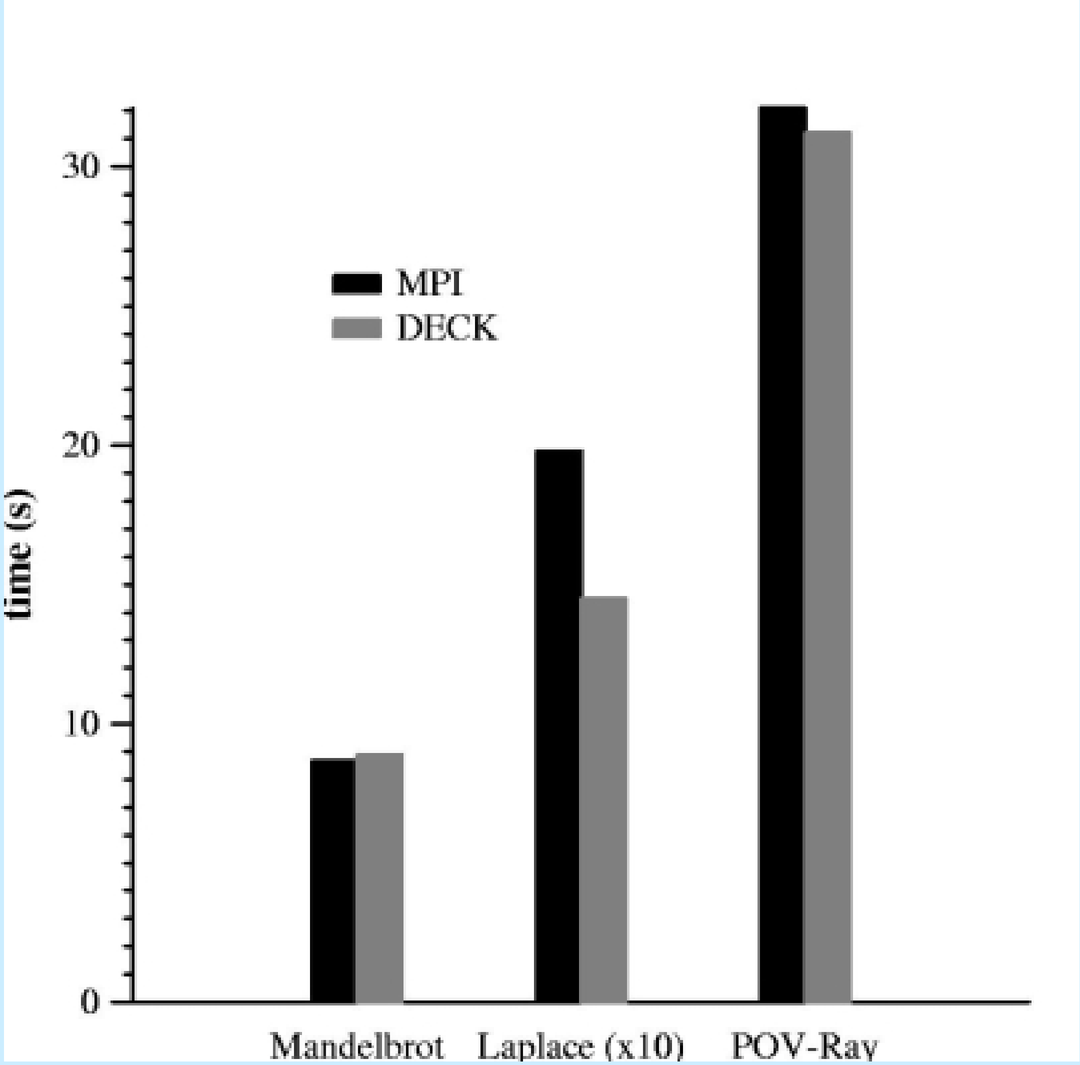
DECK-SCI, SCI-MPICH and ScaMPI (2)



DECK-SCI, SCI-MPICH and ScaMPI (3)



DECK-SCI and ScaMPI



Concluding remarks (1)

Library	Maximum bandwidth	Use of the maximum SCI bandwidth
DECK-SCI	84.12 Mbytes/s	95.9 %
ScaMPI	78.35 Mbytes/s	89.3 %
SCI-MPICH	73.80 Mbytes/s	84.1 %

Library	Minimal latency
DECK-SCI	4.66 µs
ScaMPI	6.63 µs
SCI-MPICH	7.26 µs

Concluding remarks (2)

⇒ Gains obtained with raw communication

- ⇒ maximum bandwidth
  - gain of 7,36% compared to ScaMPI
  - gain of 13,98% compared to SCI-MPICH
- ⇒ minimal latency
  - reduction of 29,71% compared to ScaMPI
  - reduction of 35,81% compared to SCI-MPICH

Concluding remarks (3)

- ⇒ Results with applications show that DECK-SCI performs at least as well as ScaMPI
- ⇒ Another alternative for SCI programming
  - ⇒ integration of communication and multithreading

Contact information

Fábio Abreu Dias de Oliveira

fabreu@inf.ufrgs.br

Federal University of Rio Grande do Sul

Institute of Informatics

Group of Parallel and Distributed Processing

http://www-gppd.inf.ufrgs.br