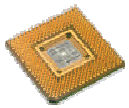
The MPC Parallel Computer

Hardware, Low-level Protocols and Performances

University P. & M. Curie (PARIS)

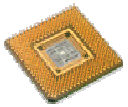
LIP6 laboratory

Olivier Glück



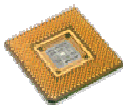
Outline

- Introduction
- Hardware
  - Hardware architecture
  - Hardware components
- Software
  - low-level protocol
  - MPI
- Conclusion

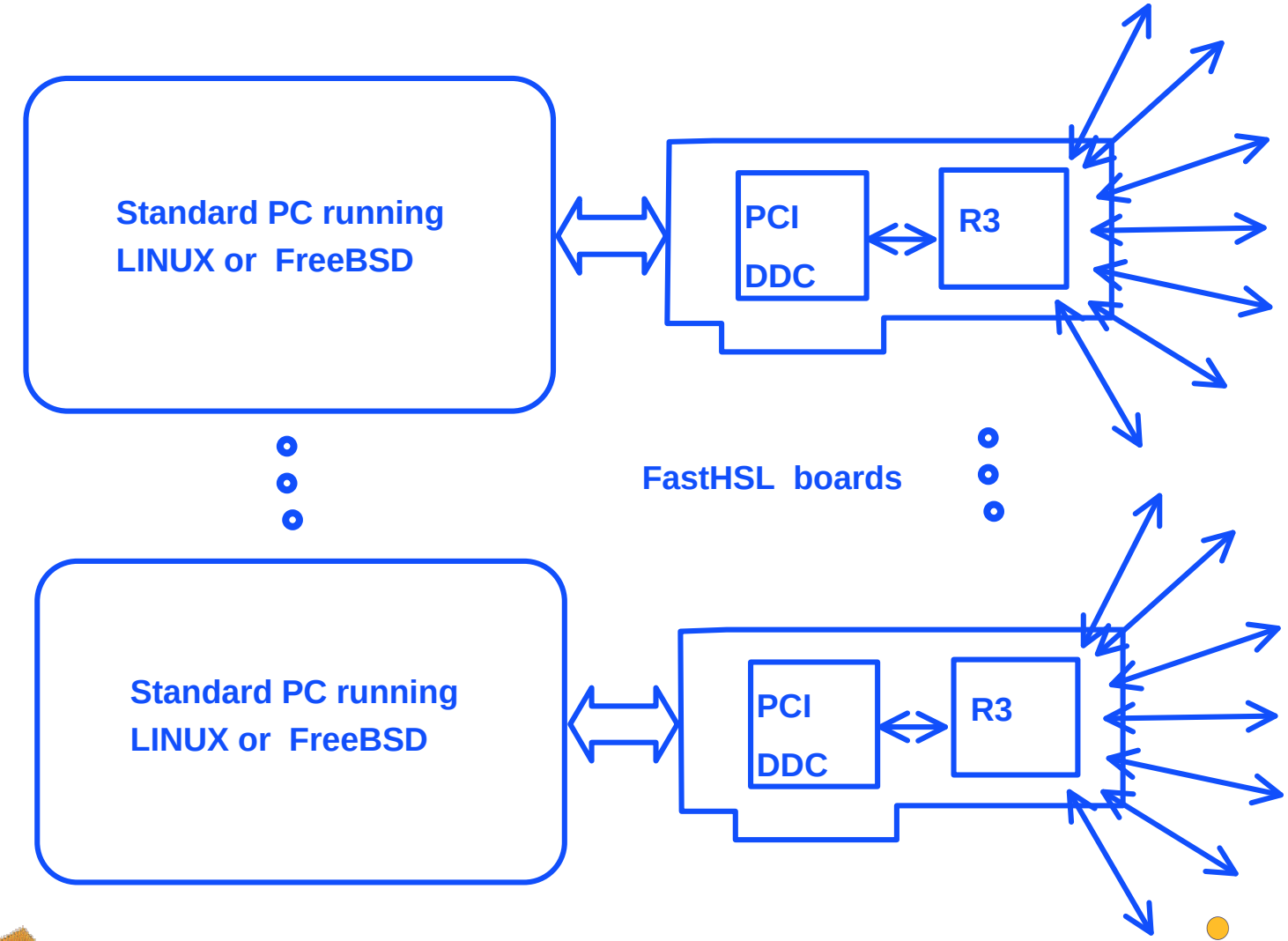


Introduction

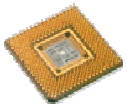
- Very low cost and high performance parallel computer
- PC cluster using optimized interconnection network
- A PCI network board (FastHSL) developed at LIP6 :
  - High speed communication network (HSL,1 Gbit/s)
  - RCUBE : router (8x8 crossbar, 8 HSL ports)
  - PCIDDC : PCI network controller (a specific communication protocol)
- Goal : supply efficient soft layers



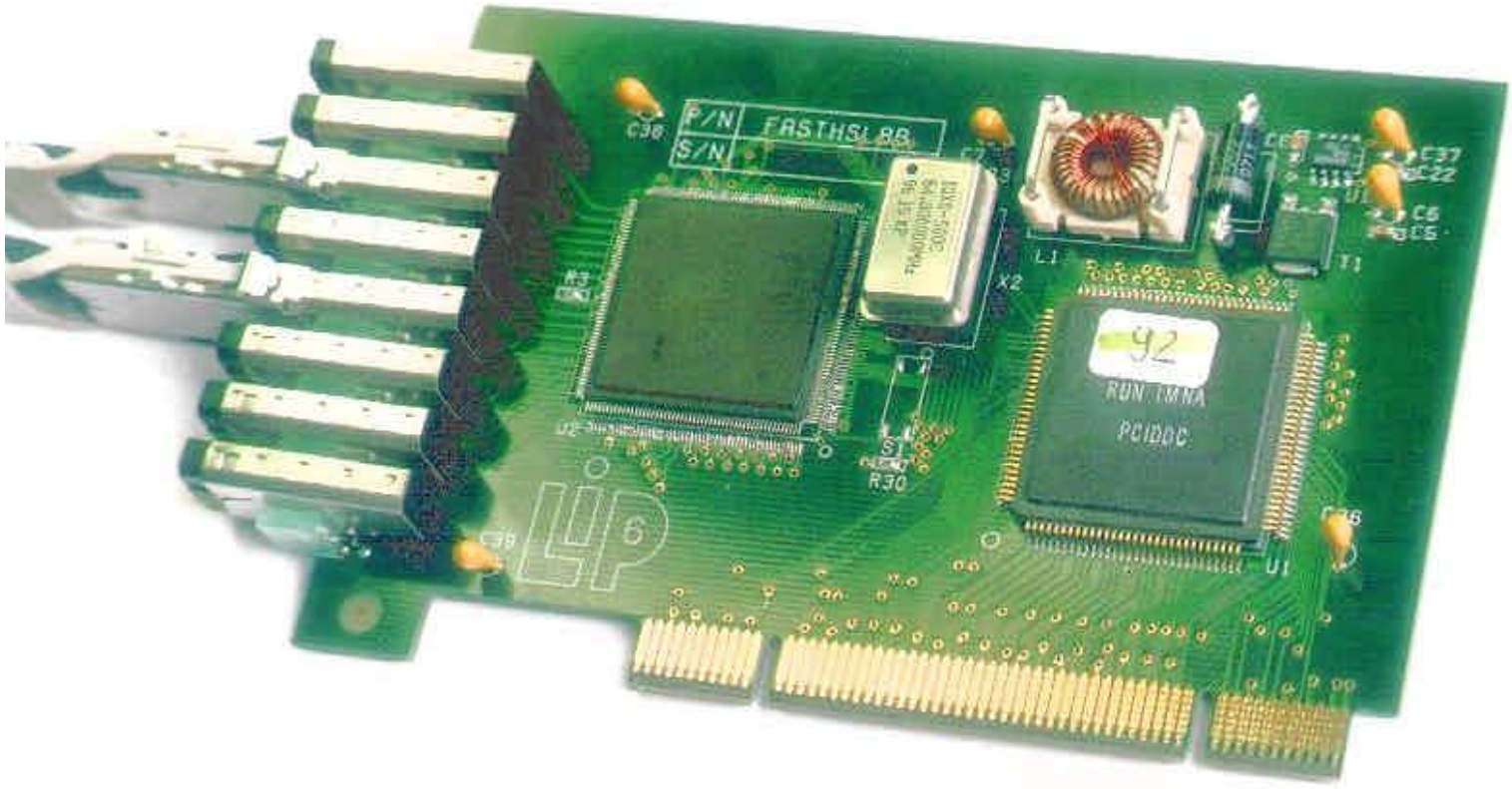
Hardware architecture

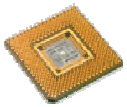


The MPC machine



The FastHSL board





Hardware layers

HSL link (1 Gbit/s)

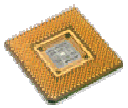
- coaxial cable, point to point, full duplex
- data encoded on 12 bits
- low-level flow control

RCUBE

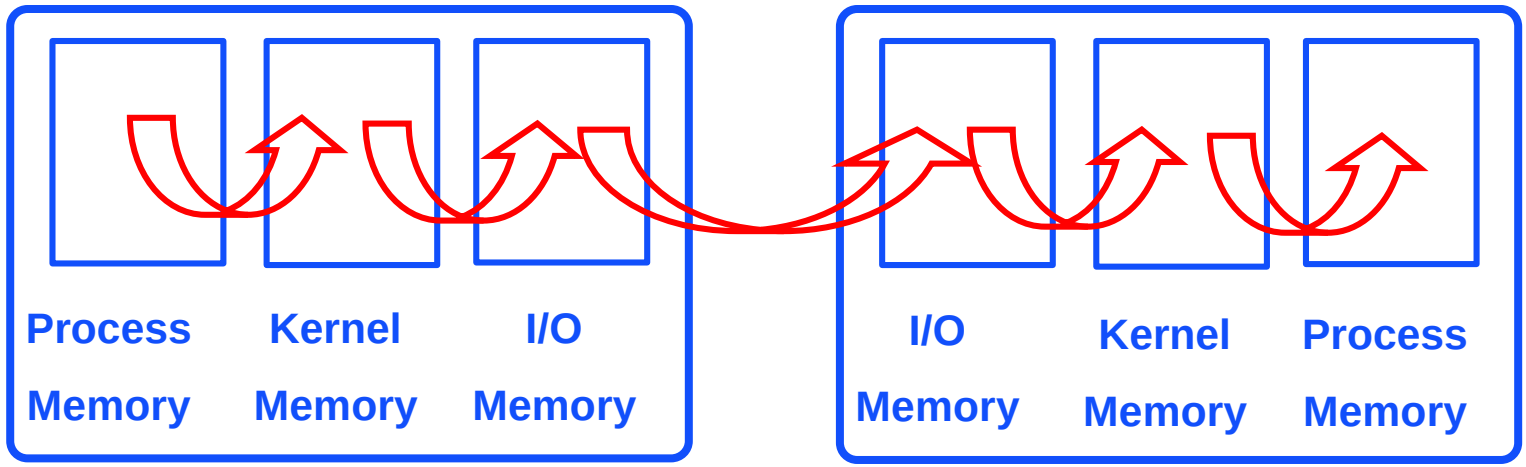
- Rapid Reconfigurable Router, extensibility
- Latency : 150 ns
- wormhole strategy, interval routing schemes

PCIDDC

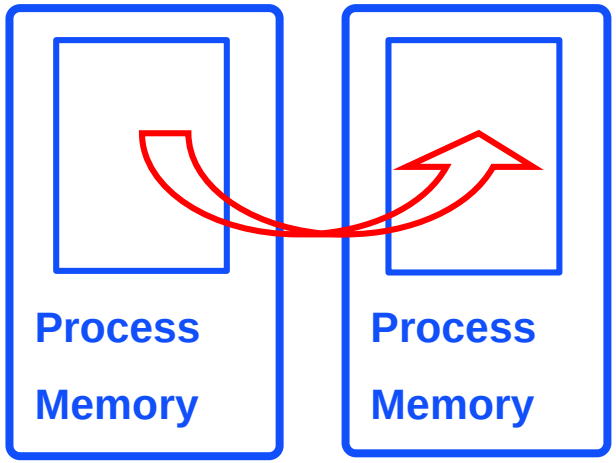
- the network interface controller
- implements communication protocol : Remote DMA
- zero copy

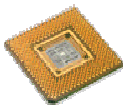


Low-level communication protocol



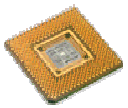
- Zero-copy protocol (Direct deposit protocol)
- FastHSL board accesses directly to host memory





PUT : the lowest level software API

- Unix based layer : FreeBSD or Linux
- Zero-copy strategy
- Provides a basic kernel API using the PCIDDC remote-write
- Parameters of a PUT() call :
  - remote node
  - local physical address
  - remote physical address
  - size of data
  - message identifier
  - callback functions for signaling



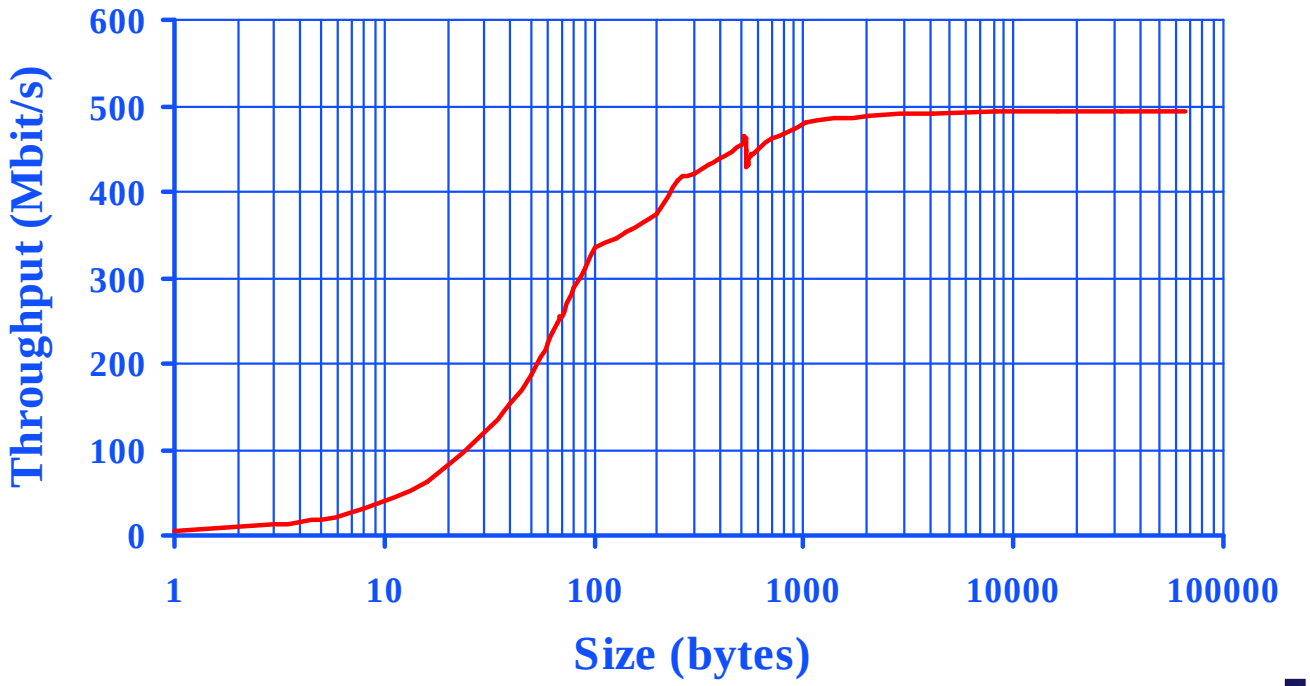
PUT performances

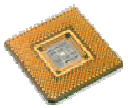
PC Pentium II 350MHz

Throughput : 494 Mbit/s

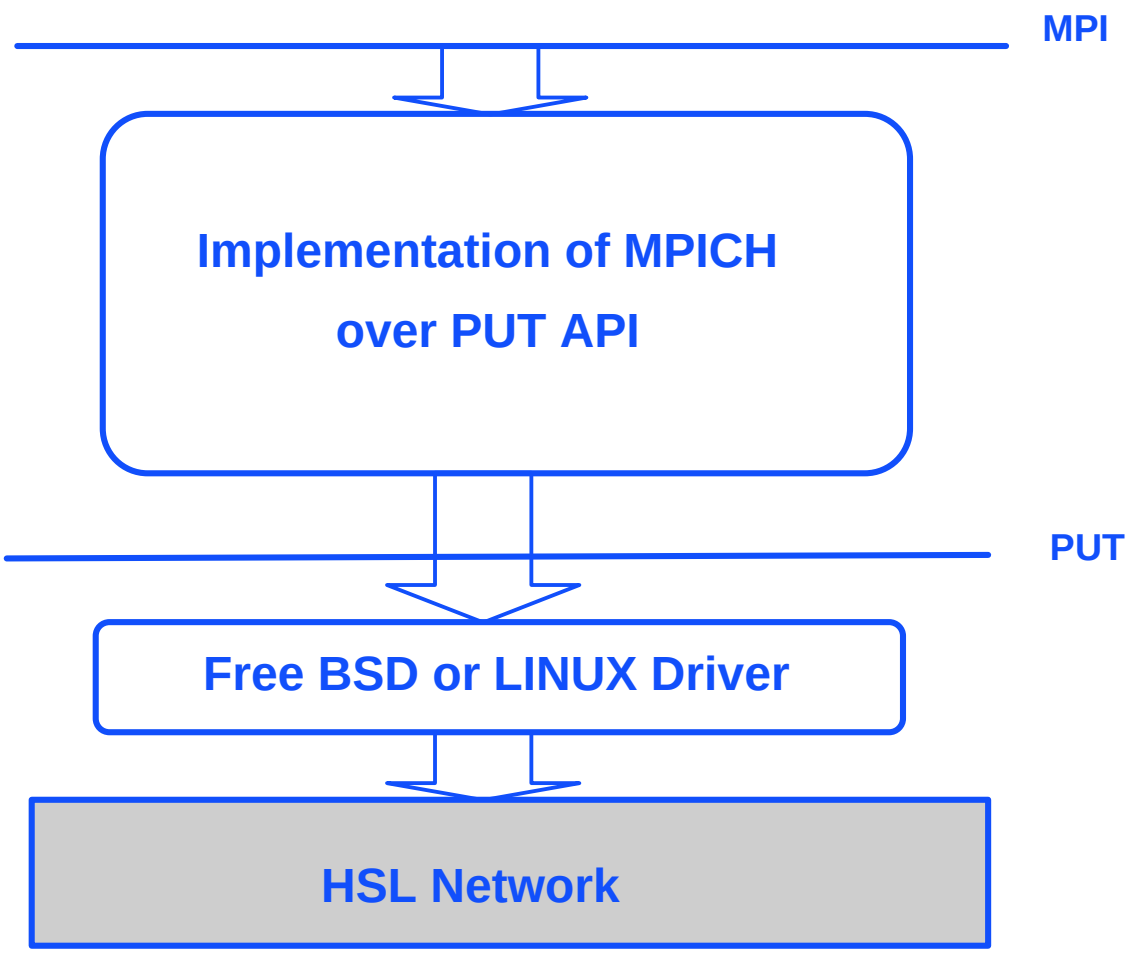
Half-throughput : 66 bytes

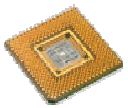
Latency : 4 µs (without system call)





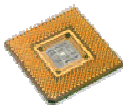
MPI over MPC


MPI
Implementation of MPICH
over PUT API
PUT
Free BSD or LINUX Driver
HSL Network



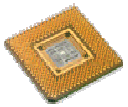
MPI implementation (1)

- 2 main problems :
  - Where to write data in remote physical memory ?
  - PUT only transfers contiguous blocks in physical memory
- 2 kinds of messages :
  - control or short messages
  - data messages



MPI implementation (2)

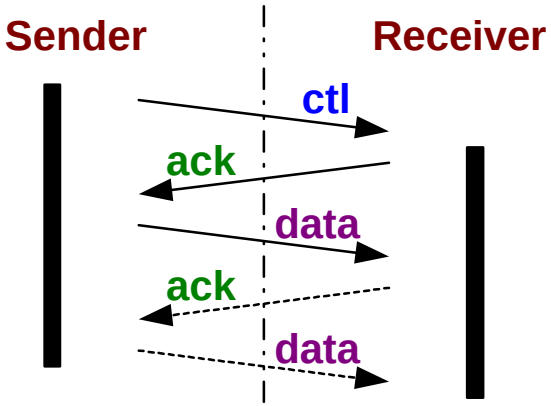
- Short (or control) messages :
  - Control information or limited-size user data
  - Use allocated buffers at starting time, contiguous in physical memory
  - One memory copy in emission and reception

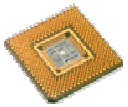


MPI implementation (3)

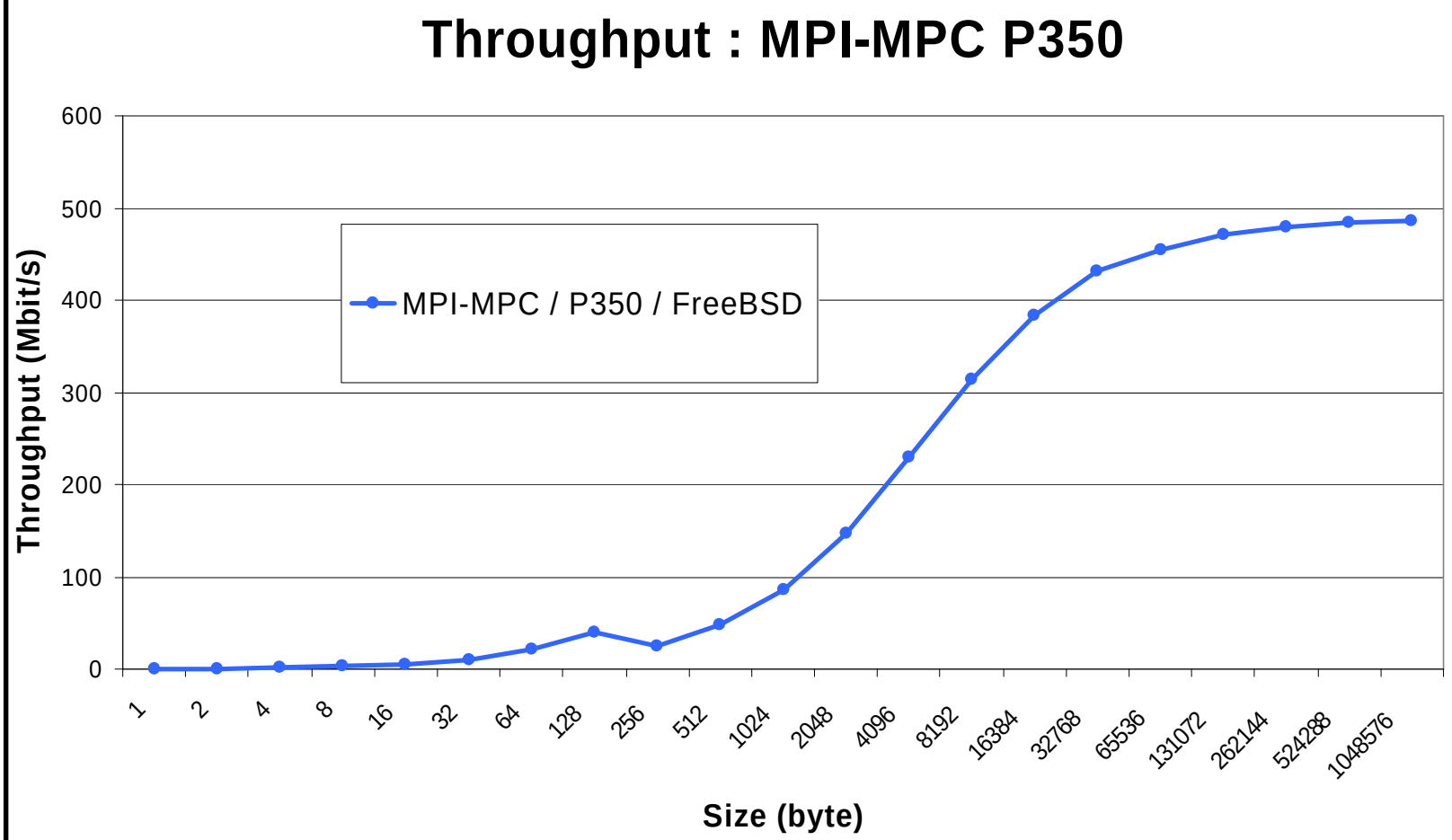
- Data messages :
  - transfer data larger than the maximum size of a control message or for specific MPI functions (e.g. MPI_Ssend)
  - RDV protocol
  - manage zero-copy transfer

Rendez-vous protocol

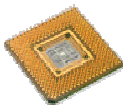




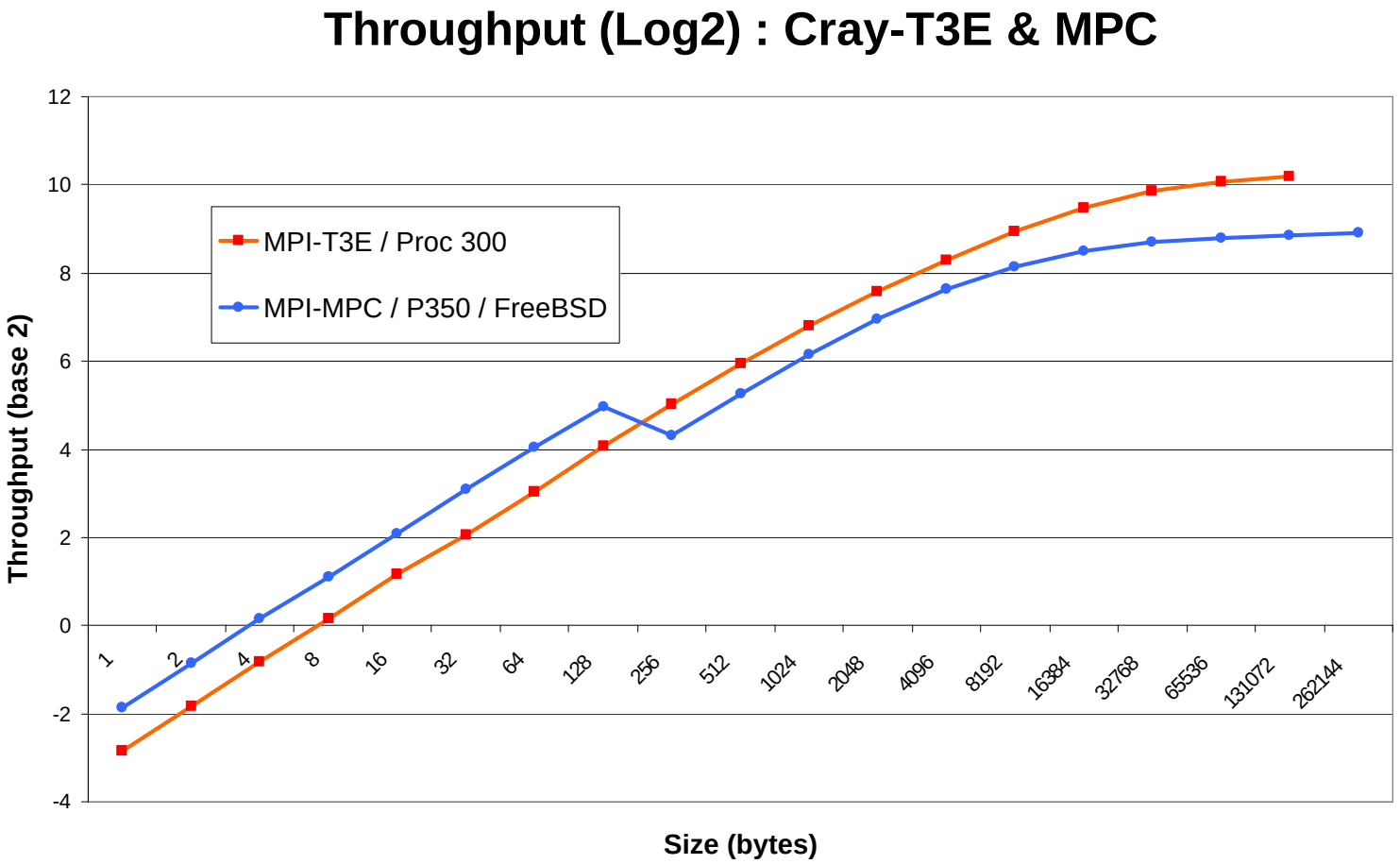
MPI performances (1)



Latency : 26 µs Throughput : 490 Mbit/s

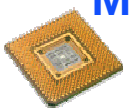


MPI performances (2)

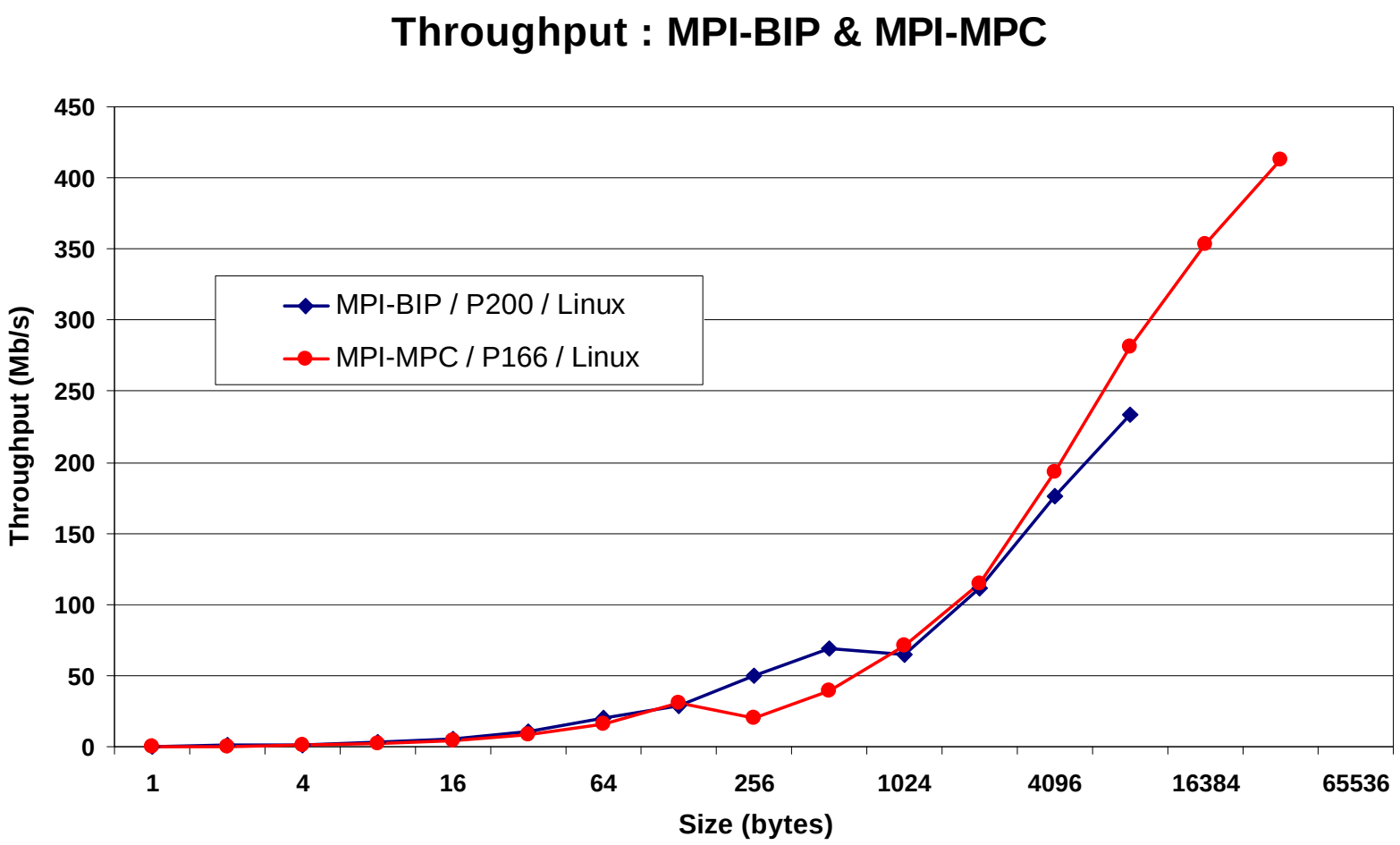


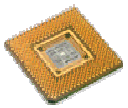
Cray **Latency : 57 µs** **Throughput : 1200 Mbit/s**

MPC **Latency : 26 µs** **Throughput : 490 Mbit/s**



MPI performances (3)





Conclusion

- MPC : a very low cost PC clusters
- Performances : similar to Myrinet clusters
- Very good extensibility (no centralized router)
- Perspectives :
  - a new router
  - an another network controller
  - improvements in MPI over MPC