Federal University of Santa Catarina
Software/Hardware Integration Laboratory

# An application-oriented low-level communication architecture


LISHA/UFSC

Thiago Robert

robert@lisha.ufsc.br

http://www.lisha.ufsc.br/~robert


May 27, 2004

# Motivation

- The use of parallelism is commonplace
  - **in** processors: arithmetic, bit and instruction level parallelism
  - **between** processors: Clusters, MPPs
- Widespread of cluster computing
  - HPC
  - Industry
- Cluster's performance relies on efficient communication mechanisms

# Efficient communication mechanisms

- Hardware is no longer an issue
  - Myrinet
  - SCI
  - Infiniband
  - ...

- Software overhead is the main obstacle
  - User-Level Communication

# User-Level Communication systems

- ULC systems differ in how they are implemented
  - especially on networks with programmable NICs
- Major points of difference:
  - the way host and NIC interact
  - low-level communication protocols
  - parameter settings

Federal University of Santa Catarina
Software/Hardware Integration Laboratory

# So... what is my role?

- To develop a flexible, high-performance communication system to support our PPS
  - following the design of EPOS communication system
  - improving it when necessary
- Focus on Myrinet networking technology
  - high-bandwidth
  - low latency
  - programmable NIC
  - open standards

# SNOW Project Goal

- *To develop a comprehensive programming environment to support parallel computing on clusters of commodity workstations*
- This software environment shall include:
  - A parallel programming language
  - Run-time system
  - Support to standard interfaces (POSIX and MPI)
  - Management tools
  - Parallel applications

# SNOW – The run-time system

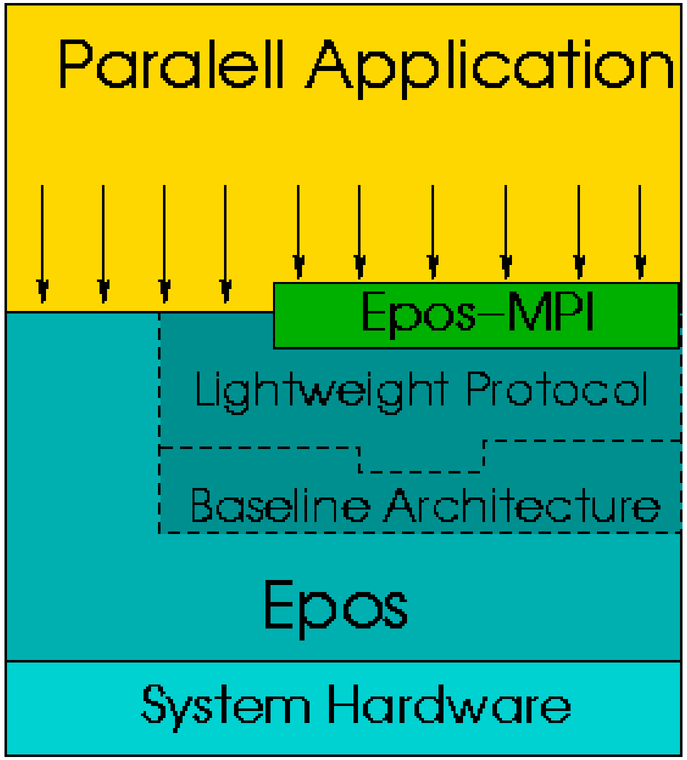
- Epos – Embedded Parallel Operating System
  - AOSD - Application-Oriented
- A set of reusable and adaptable components
  - result of a domain analysis of the **high-performance dedicated computing** domain
- Mechanisms that allow combining these components into run-time systems

# SNOW – Other features

- Support for standard interfaces
  - POSIX – supported by EPOS natively
  - MPI – Epos-MPI
- Management tools
  - CODINE
- Parallel programming language
  - DPC++
- Parallel applications
  - Bioinformatics and

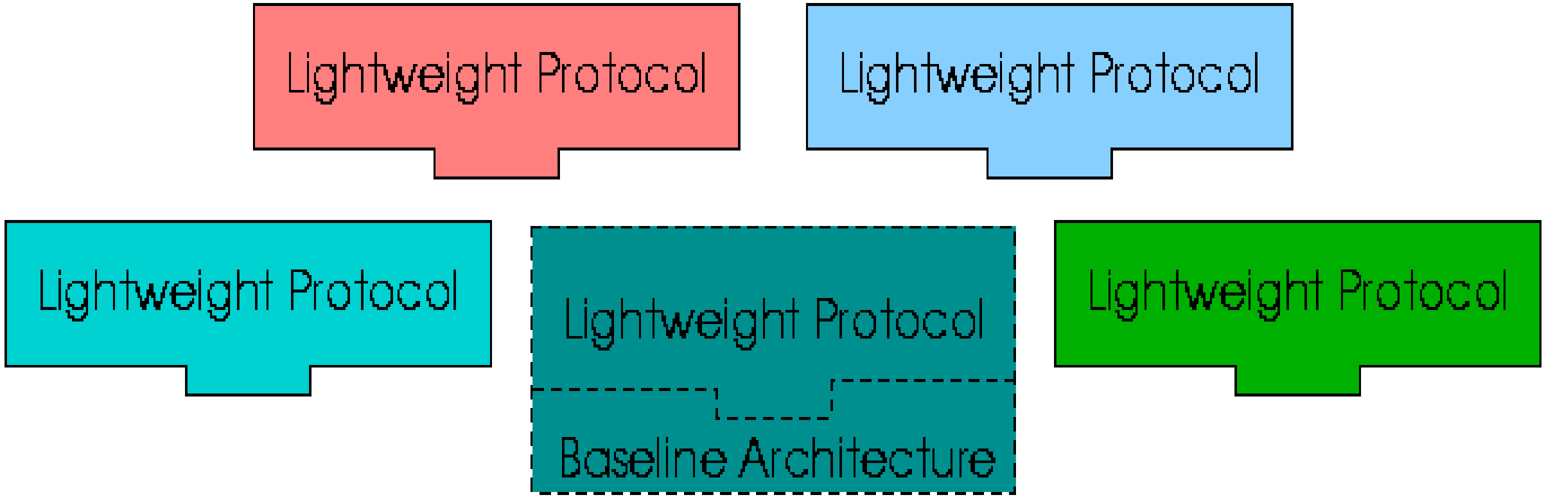
Federal University of Santa Catarina
Software/Hardware Integration Laboratory

# PPS architecture




- Application interfaces with:
  - Epos-MPI
  - Epos
  - directly with the communication when necessary
- The whole run-time system is application-oriented

# Communication system architecture



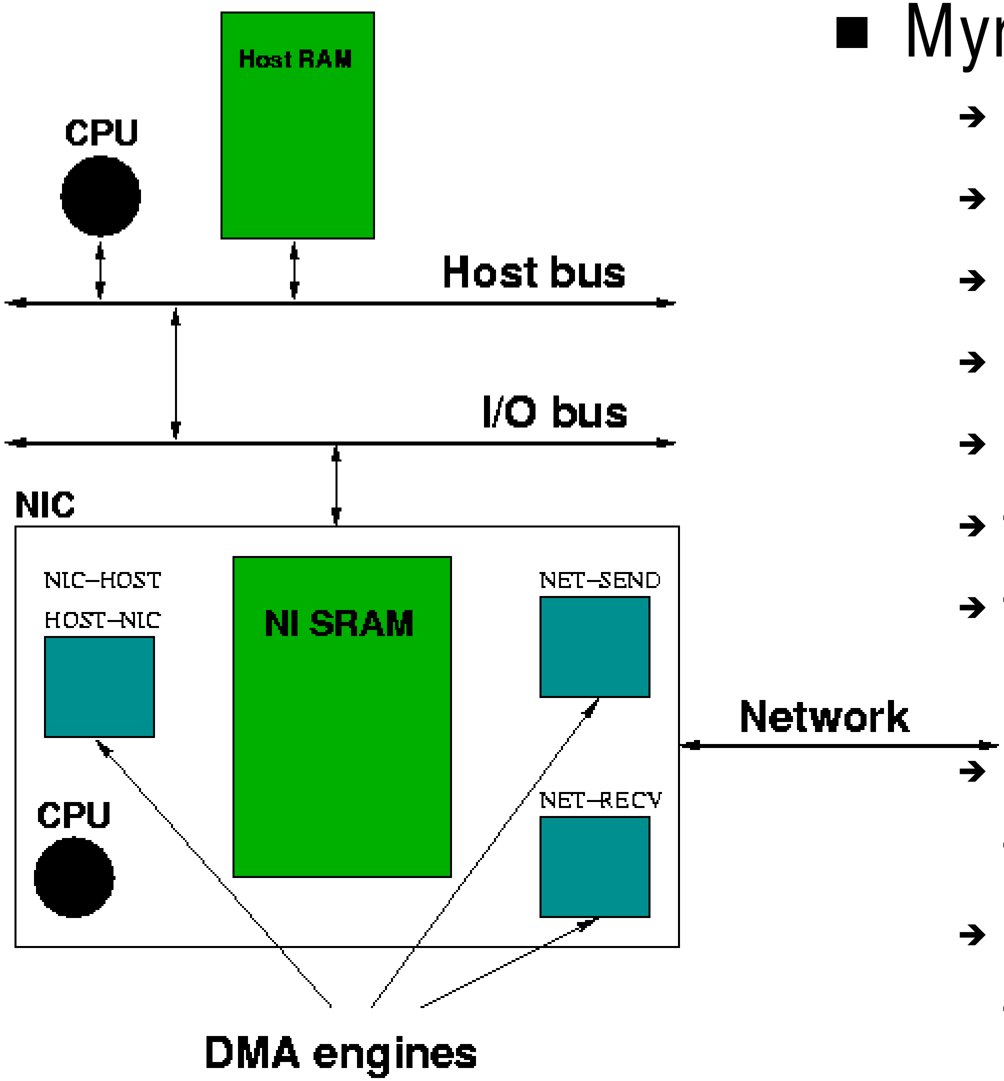

- Different low-level lightweight protocols can be implemented from a lean baseline architecture
- Each one of these protocols meets the communication requirements of a specific parallel application
- Lightweight protocols are grouped into Epos families
- Protocols can be implemented on demand

Federal University of Santa Catarina
Software/Hardware Integration Laboratory

# Baseline architecture

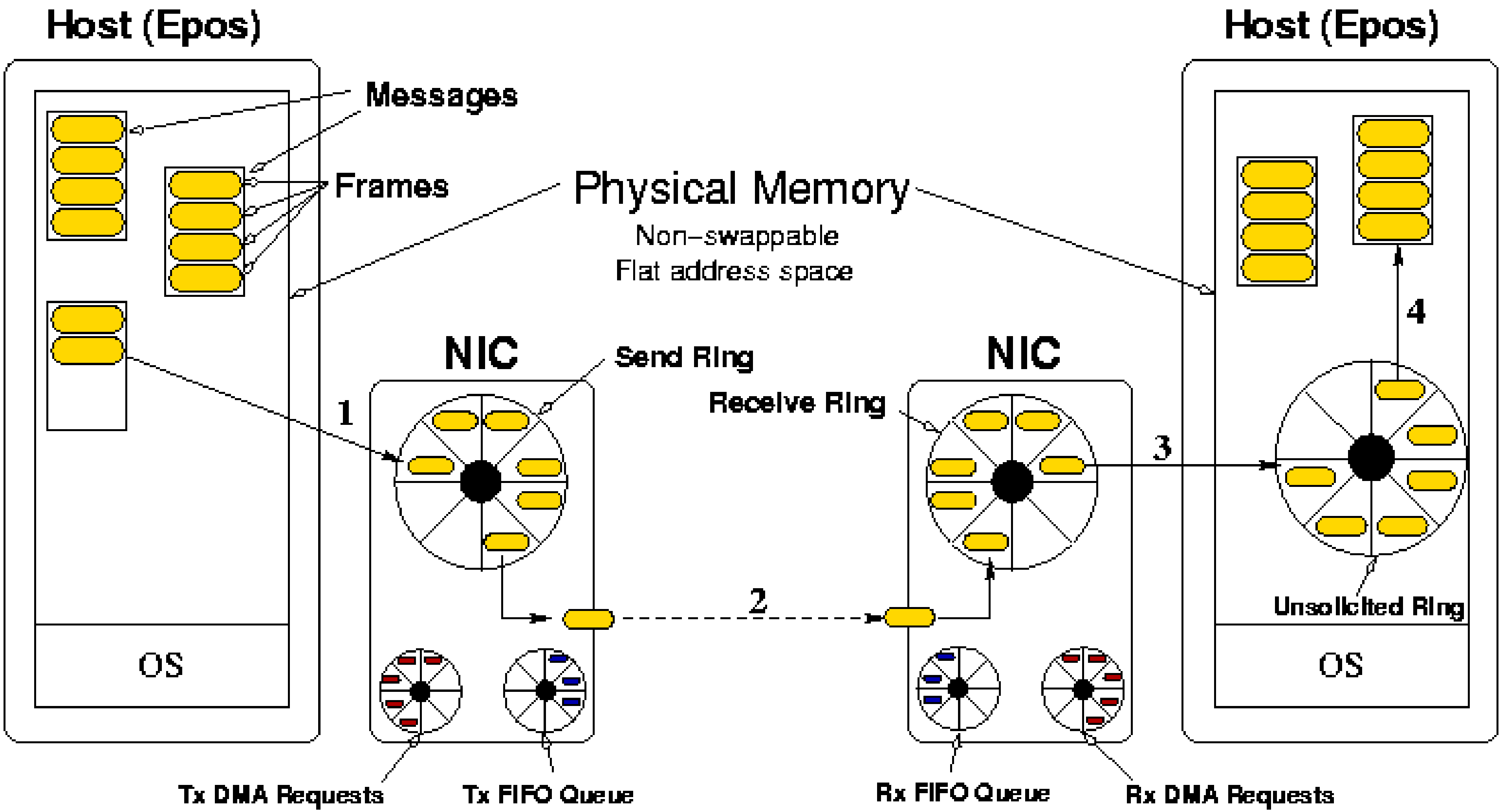
- It must be simple, flexible and configurable
  - not to hinder the design and implementation of specific protocols
- The highest bandwidth and lowest latency possible are desired
  - complex protocol implementations will affect both of them
- Maximum utilization of the NIC resources
  - improves performance
  - doorbell mechanism
- Host and NIC work asynchronously

# Host/NIC interface



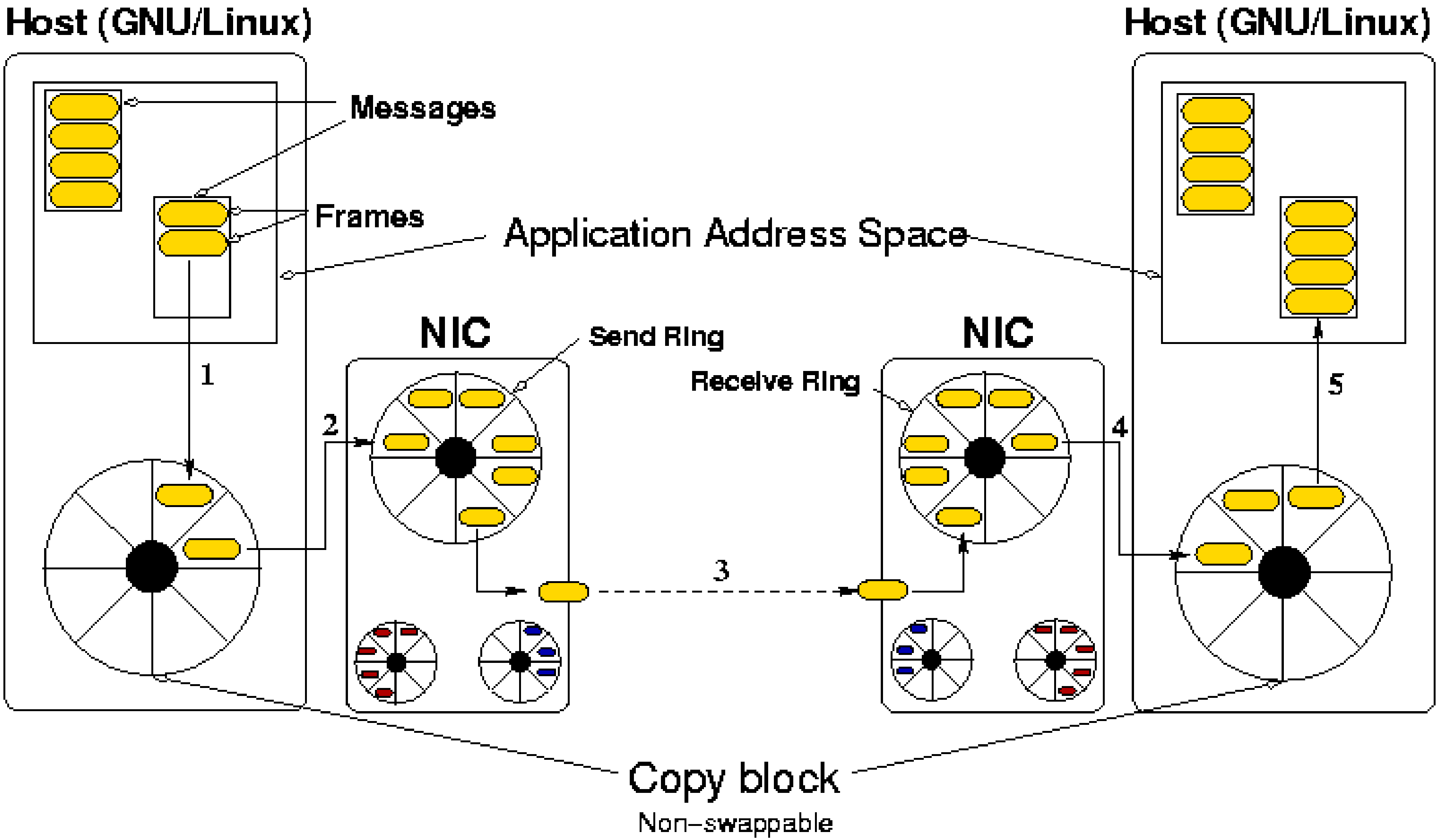

- Myrinet NIC characteristics:
  - LANai processor
  - Host-DMA engine
  - Net-Send DMA engine
  - Net-Recv DMA engine
  - Doorbell mechanism
  - they all can work simmultaneously
  - there is a limit of two memory access per clock cycle
  - all network traffic must go through the NIC SRAM
  - at least three copies are required for each message

# Memory layout and dynamic data flow

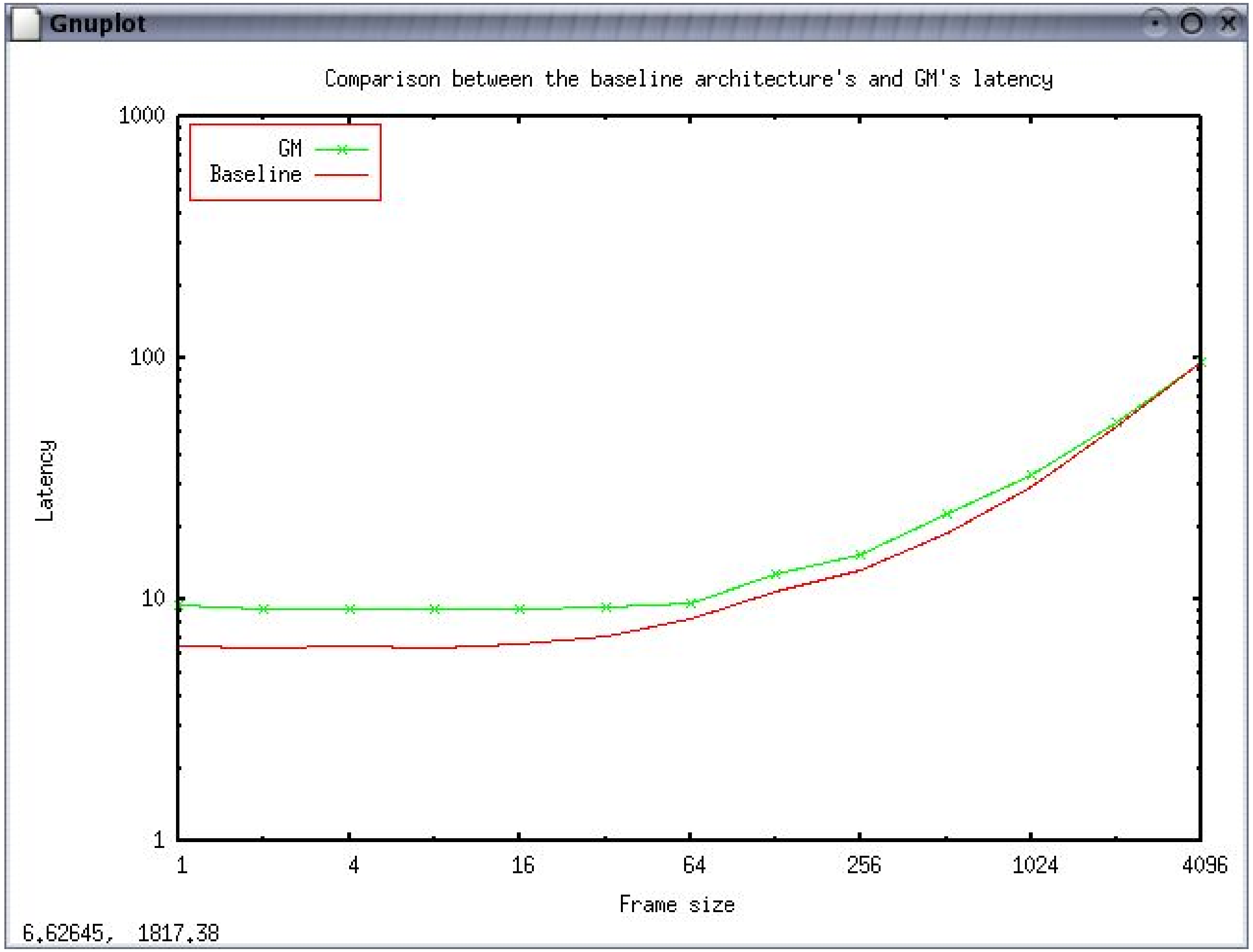


Federal University of Santa Catarina
Software/Hardware Integration Laboratory

# Baseline architecture prototype



# Prototype performance




Latency = RTT / 2

Node 1
```
GetTime();
do
    Send(2);
    Receive();
N times
GetTime();
```

Node 2
```
do
    Receive();
    Send(1);
forever
```

# Other considerations

- Reliability
  - the Myrinet network has a relly low error rate
- Flow control
  - Myrinet's back pressure flow control mechanism
  - FIFO counters
- Protection
  - not provided in the baseline architecture
- More sophisticated mechanisms must be implemented by specific lightweight protocols

# Lightweight protocols

- Infrastructure protocols
  - Multicasting / broadcasting
  - QoS
  - connection management
  - protection schemes
  - reliable delivery
  - flow control mechanisms
- High-performance protocols
  - minimal modifications in the baseline architecture that are required by parallel applications

# Next steps

- Evaluate performance with real applications and lightweight protocols
  - impact of low-level changes in application's performance
- Candidate applications:
  - Available parallel applications (bioinformatics)
- Make the proposed communication system adaptive

# That's it

- Questions?
- Comments?
- Suggestions?

# That's really it

- Thanks

# Myrinet ULC systems – lessons learned

- There is no single best low-level communication protocol
  - performance is affected by applications' communication patterns and run-time system specifics
- The data transfer between host and NIC has a great impact in communication performance
  - DMA x PIO
- It's hard to find the right balance when offloading communication tasks to the NIC

# Myrinet ULC systems – lessons learned

- The performance indicated by commonly used low-level benchmarks (e. g., LogP) is a poor predictor for application performance
- Issues that are sometimes ignored by ULC systems can impact the usability of a communication system and the performance of real applications
  - flow control
  - reliability
  - run-time system requirements

Federal University of Santa Catarina
Software/Hardware Integration Laboratory

# Dynamic data flow

- Send and Receive Ring hold the frames before they are accessed by the Net DMA engine
- Rx and Tx DMA Requests are circular chains of DMA control blocks
- Rx and Tx FIFO Queue are circular FIFO queues used by the host and LANai to signal for each other the arrival of a new frame

- The sender host fills up an entry in the Rx DMA Requests using programmed I/O and triggers the doorbell, creating a new entry in Tx DMA Requests and signaling to the LANai that a new frame must be sent

1. The transmission of frames between host and NIC memory is carried out asynchronously by the Host/NIC DMA engine
2. A frame is sent as soon as possible after the corresponding DMA finishes

- When a frame arrives from the network the LANai receives the frame and fills up an entry in the Rx DMA Requests DMA chain

3. The message is assembled asynchronously in the Unsolicited Ring circular buffer in host memory
4. The receiving side is responsible for copying the whole message from the Unsolicited Ring before it is overwritten by other messages