An Introduction to Interconnection Networks & Xmulator

Sina Meraji

March 2008
Outline

- Introduction
  - Interconnection networks
  - Topology
  - Switching
  - Routing

- Xmulator
  - Multi-Layer Architecture
  - Listener-Based Integration

- Conclusions and future work
Interconnection Network (IN)

- A set of processors with local memories which communicate through a network

Terminology

- **Topology:** The way nodes are interconnected
- **Routing Algorithm:** Determines the path from source to destination
- **Switching Method:** Specifies when input channel is connected to the output channel selected by the routing algorithm
Topology

- **Direct:** Each node directly connects to its neighbor nodes

- **Indirect:** Nodes are connected through a set of switches
Indirect IN

- Example: A butterfly constructed from 2x2 switches
Direct IN

- Example: Hypercube

\[ H_1 \]

\[ H_2 \]

\[ H_3 \]
Switching Methods

Store & Forward (Packet Switching)
Sending the message after receiving whole of the message

Disadvantages
Large buffer space, Increasing time to transmit a message in network

Wormhole Switching
Messages are divided into flits (= a few bytes)
Each node has buffer for only one flit of a message

Advantages
Latency is almost insensitive to distance
Fast and simple routers

Disadvantages
Prone to deadlock
A blocked message remains in the network. Hence, a bottleneck in a single point can increase the traffic of neighboring area

www.wikipedia.org
Switching Methods

- **Store & Forward**
  - Buffers for data packets
  - Requirement: buffers must be sized to hold entire packet

- **Wormhole**
  - Buffers for flits: packets can be larger than buffers
Switching Methods

- **Store & Forward**
  - Source
  - Packet completely stored at the switch
  - Busy Link
  - Destination end node

- **Wormhole**
  - Source
  - Packet stored along the path
  - Busy Link
  - Destination end node

Buffers for data packets:
Requirement: buffers must be sized to hold entire packet

Buffers for flits: packets can be larger than buffers
Routing Algorithms

- Topologies are simple and regular
- Very low delay and extra high bandwidth are needed
- Simple routing algorithms are developed for INs (deadlock is an important concern)
- Example: e-cube routing in hypercube networks
  Messages are routed along first dimension, then routing continues in other dimensions.
e-Cube

Source

Destination
Xmulator: An Object Oriented Listener-based Network Simulator

- XML + Simulator
- Designed in HPCAN lab. of Sharif University of Technology in 2006

Xmulator: An Object Oriented Listener-based Network Simulator

Motivations

- Developing a simulator for simulating interconnection networks:
  - Possibility of defining details of hardware and algorithms with performance considerations
  - Applying wormhole, store & forward, and other switching techniques
- Using new software architecture paradigms (Object Oriented and Multi-Layered Architecture)
- Using new programming languages (C#, java, XML)
  - Xmulator=XML+Simulator
Xmulator: An Object Oriented Listener-based Network Simulator

Motivations

- Using XML format for defining topologies, parameters, and outputs
- No need to learn a new programming or scripting languages and using the previous abilities of users in java and C# languages
- Ease of developing new packages, new tools for existing packages, or defining routing algorithms
- Possibility of debugging even into the core of simulator
- Logging capabilities
Simulator Engine

- Managing the events list:
  - Eliminating the event with less time
  - Call the ProcessXEvent of the component which the event belongs to it
  - The above method can produce other events and register them in simulator engine

- Xmulator use Red-Black tree to keep the concurrent event queues
  - Save the balanced tree during the work
  - Eliminate the smallest element with $O(\log N)$
  - Insert the element with $O(\log N)$
Xmulator Multi-Layer Architecture

UIControls

QueuingNetwork
InterconnectionNetwork

BaseComponents
Distributions (from BaseC...)

Common
Log4net
Util
Layers

- **First Layer**: Util – Common – Log4net
- **Second Layer**: BaseComponents – Distributions
- **Third Layer**: InterconnectionNetwork – QueuingNetwork
- **Forth Layer**: Presentation
Listener Based Integrating

- Increase the Extensibility

- Example: Assume a queue and a server
  - The server must know the queue’s signature
  - When an element enters the queue, server must serve it
  - Server can’t check the queue at all the times

- Solutions
  - Queue knows the server behavior (min extensibility)
  - Simulator’s engine keeps track the way (min extensibility)
Listener Based Integrating

- Using the listener: Queue defines an event and server listens to it
  - The listener is called every time a new object is received by queue
Conclusions & Future work

- Design a user interface for Xmulator using AToM³
- Using model transformation to generate the input XML file for Xmulator
- Developing some routing protocols for interconnection networks
- User can change input parameters to see their effects on performance
References


- http://www.cm.cf.ac.uk/Parallel/Year2/section5.html

- www.wikipedia.org
Questions