Communication Performance in Network-on-Chips

Axel Jantsch
Royal Institute of Technology, Stockholm

November 24, 2004
Overview

Introduction
Communication Performance
Organizational Structure
Interconnection Topologies
Trade-offs in Network Topology
Routing
**Introduction**

- **Topology**: How switches and nodes are connected.
- **Routing algorithm**: Determines the route from source to destination.
- **Switching strategy**: How a message traverses the route.
- **Flow control**: Schedules the traversal of the message over time.
Basic Definitions

**Message** is the basic communication entity.

**Flit** is the basic flow control unit. A message consists of 1 or many flits.

**Phit** is the basic unit of the physical layer.

**Direct network** is a network where each switch connects to a node.

**Indirect network** is a network with switches not connected to any node.

**Hop** is the basic communication action from node to switch or from switch to switch.

**Diameter** is the length of the maximum shortest path between any two nodes measured in hops.

**Routing distance** between two nodes is the number of hops on a route.

**Average distance** is the average of the routing distance over all pairs of nodes.
Basic Switching Techniques

**Circuit Switching** A real or virtual circuit establishes a direct connection between source and destination.

**Packet Switching** Each packet of a message is routed independently. The destination address has to be provided with each packet.

**Store and Forward Packet Switching** The entire packet is stored and then forwarded at each switch.

**Cut Through Packet Switching** The flits of a packet are pipelined through the network. The packet is not completely buffered in each switch.

**Virtual Cut Through Packet Switching** The entire packet is stored in a switch only when the header flit is blocked due to congestion.

**Wormhole Switching** is cut through switching and all flits are blocked on the spot when the header flit is blocked.
Latency

\[ \text{Time}(n) = \text{Admission} + \text{ChannelOccupancy} + \text{RoutingDelay} + \text{ContentionDelay} \]

**Admission** is the time it takes to emit the message into the network.

**ChannelOccupancy** is the time a channel is occupied.

**RoutingDelay** is the delay for the route.

**ContentionDelay** is the delay of a message due to contention.
Channel Occupancy

\[
\text{Channel Occupancy} = \frac{n + n_E}{b}
\]

- \(n\) ... message size in bits
- \(n_E\) ... envelop size in bits
- \(b\) ... raw bandwidth of the channel
Routing Delay

Store and Forward:

\[ T_{sf}(n, h) = h\left(\frac{n}{b} + \Delta\right) \]

Circuit Switching:

\[ T_{cs}(n, h) = \frac{n}{b} + h\Delta \]

Cut Through:

\[ T_{ct}(n, h) = \frac{n}{b} + h\Delta \]

Store and Forward with fragmented packets:

\[ T_{sf}(n, h, n_p) = \frac{n-n_p}{b} + h\left(\frac{n_p}{b} + \Delta\right) \]

\( n \) ... message size in bits
\( n_p \) ... size of message fragments in bits
\( h \) ... number of hops
\( b \) ... raw bandwidth of the channel
\( \Delta \) ... switching delay per hop
Routing Delay: Store and Forward vs Cut Through

SF vs CT switching; d=2, k=10, b=1

SF vs CT switching; d=2, k=10, b=32

SF vs CT switching, k=2, m=8

SF vs CT switching, d=2, m=8

A. Jantsch, KTH
Local and Global Bandwidth

**Local bandwidth**

\[
\text{Local bandwidth} = b \left( \frac{n}{n + n_E + w\Delta} \right)
\]

**Total bandwidth**

\[
\text{Total bandwidth} = C b \text{[bits/second]} = C w \text{[bits/cycle]} = C \text{[phits/cycle]}
\]

**Bisection bandwidth**

... minimum bandwidth to cut the net into two equal parts.

- \( b \) ... raw bandwidth of a link;
- \( n \) ... message size;
- \( n_E \) ... size of message envelope;
- \( w \) ... link bandwidth per cycle;
- \( \Delta \) ... switching time for each switch in cycles;
- \( w\Delta \) ... bandwidth lost during switching;
- \( C \) ... total number of channels;

For a \( k \times k \) mesh with bidirectional channels:

\[
\text{Total bandwidth} = (4k^2 - 4k)b
\]

\[
\text{Bisection bandwidth} = 2kb
\]
Link and Network Utilization

**total load on the network:** \( L = \frac{Nhl}{M} \) [phits/cycle]

**load per channel:** \( \rho = \frac{Nhl}{MC} \) [phits/cycle] \( \leq 1 \)

- \( M \) ... each host issues a packet every \( M \) cycles
- \( C \) ... number of channels
- \( N \) ... number of nodes
- \( h \) ... average routing distance
- \( l = n/w \) ... number of cycles a message occupies a channel
- \( n \) ... average message size
- \( w \) ... bitwidth per channel
Typical saturation points are between 40% and 70%.
The saturation point depends on

- Traffic pattern
- Stochastic variations in traffic
- Routing algorithm
Organizational Structure

- Link
- Switch
- Network Interface
Link

**Short link**  At any time there is only one data word on the link.

**Long link**  Several data words can travel on the link simultaneously.

**Narrow link**  Data and control information is multiplexed on the same wires.

**Wide link**  Data and control information is transmitted in parallel and simultaneously.

**Synchronous clocking**  Both source and destination operate on the same clock.

**Asynchronous clocking**  The clock is encoded in the transmitted data to allow the receiver to sample at the right time instance.
Switch Design Issues

**Degree:** number of inputs and outputs;

**Buffering**
- Input buffers
- Output buffers
- Shared buffers

**Routing**
- Source routing
- Deterministic routing
- Adaptive routing

**Output scheduling**

**Deadlock handling**

**Control flow**
Network Interface

- Admission protocol
- Reception obligations
- Buffering
- Assembling and disassembling of messages
- Routing
- Higher level services and protocols
Interconnection Topologies

• Fully connected networks
• Linear arrays and rings
• Multidimensional meshes and tori
• Trees
• Butterflies
**Fully Connected Networks**

**Bus:**
- Switch degree: $N$
- Diameter: 1
- Distance: 1
- Network cost: $O(N)$
- Total bandwidth: $b$
- Bisection bandwidth: $b$

**Crossbar:**
- Switch degree: $N$
- Diameter: 1
- Distance: 1
- Network cost: $O(N^2)$
- Total bandwidth: $Nb$
- Bisection bandwidth: $Nb$
Linear Arrays and Rings

Linear array:
- switch degree: 2
- diameter: $N - 1$
- distance: $\sim \frac{2}{3}N$
- network cost: $O(N)$
- total bandwidth: $2(N - 1)b$
- bisection bandwidth

Torus:
- switch degree: 2
- diameter: $N/2$
- distance: $\sim \frac{1}{3}N$
- network cost: $O(N)$
- total bandwidth: $2Nb$
- bisection bandwidth: $4b$

A. Jantsch, KTH
Multidimensional Meshes and Tori

$k$-ary $d$-cubes are $d$-dimensional tori with unidirectional links and $k$ nodes in each dimension:

- number of nodes $N = k^d$
- switch degree $= d$
- diameter $= d(k - 1)$
- distance $\sim d^{1/2}(k - 1)$
- network cost $= O(N)$
- total bandwidth $= 2Nb$
- bisection bandwidth $= 2k^{(d-1)}/2$
Routing Distance in $k$-ary $n$-Cubes

Network Scalability wrt Distance

Average distance

Number of nodes

A. Jantsch, KTH
Projecting High Dimensional Cubes

2-ary 2-cube

2-ary 3-cube

2-ary 4-cube

2-ary 5-cube

A. Jantsch, KTH
Binary Trees

- number of nodes $N = 2^d$
- number of switches $= 2^d - 1$
- switch degree $= 3$
- diameter $= 2d$
- distance $\sim d + 2$
- network cost $= O(N)$
- total bandwidth $= 2 \cdot 2(N - 1)b$
- bisection bandwidth $= 2b$
$k$-ary Trees

- Number of nodes $N = k^d$
- Number of switches $\sim k^d$
- Switch degree $= k + 1$
- Diameter $= 2d$
- Distance $\sim d + 2$
- Network cost $= O(N)$
- Total bandwidth $= 2 \cdot 2(N - 1)b$
- Bisection bandwidth $= kb$
Binary Tree Projection

- Efficient and regular 2-layout;
- Longest wires in resource width:

$$l_W = 2^\left\lfloor \frac{d-1}{2} \right\rfloor - 1$$

<table>
<thead>
<tr>
<th>$d$</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N$</td>
<td>4</td>
<td>8</td>
<td>16</td>
<td>32</td>
<td>64</td>
<td>128</td>
<td>256</td>
<td>512</td>
<td>1024</td>
</tr>
<tr>
<td>$l_W$</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>8</td>
<td>8</td>
</tr>
</tbody>
</table>
### k-ary n-Cubes versus k-ary Trees

#### k-ary n-cubes:
- Number of nodes \( N = k^d \)
- Switch degree \( = d + 2 \)
- Diameter \( = d(k - 1) \)
- Distance \( \sim d^{1/2}(k - 1) \)
- Network cost \( = O(N) \)
- Total bandwidth \( = 2Nb \)
- Bisection bandwidth \( = 2k^{(d-1)}b \)

#### k-ary trees:
- Number of nodes \( N = k^d \)
- Number of switches \( \sim k^d \)
- Switch degree \( = k + 1 \)
- Diameter \( = 2d \)
- Distance \( \sim d + 2 \)
- Network cost \( = O(N) \)
- Total bandwidth \( = 2 \cdot 2(N - 1)b \)
- Bisection bandwidth \( = kb \)
Butterfly Characteristics

- number of nodes $N = 2^d$
- number of switches $= 2^{d-1}d$
- switch degree $= 2$
- diameter $= d + 1$
- distance $= d + 1$
- network cost $= O(Nd)$
- total bandwidth $= 2^d b$
- bisection bandwidth $= \frac{N}{2} b$
### $k$-ary $n$-Cubes versus $k$-ary Trees vs Butterflies

<table>
<thead>
<tr>
<th></th>
<th>$k$-ary $n$-cubes</th>
<th>binary tree</th>
<th>butterfly</th>
</tr>
</thead>
<tbody>
<tr>
<td>cost per node</td>
<td>$O(N)$</td>
<td>$O(N)$</td>
<td>$O(N \log N)$</td>
</tr>
<tr>
<td>distance</td>
<td>$\frac{1}{2} \sqrt{N} \log N$</td>
<td>$2 \log N$</td>
<td>$\log N$</td>
</tr>
<tr>
<td>links per node</td>
<td>2</td>
<td>2</td>
<td>$\log N$</td>
</tr>
<tr>
<td>bisection</td>
<td>$2N^{\frac{d-1}{d}}$</td>
<td>1</td>
<td>$\frac{1}{2}N$</td>
</tr>
<tr>
<td>frequency limit of random traffic</td>
<td>$\frac{1}{(\sqrt{\frac{N}{2}})}$</td>
<td>$\frac{1}{N}$</td>
<td>$\frac{1}{2}$</td>
</tr>
</tbody>
</table>
Problems with Butterflies

- Cost of the network
  - $O(N \log N)$
  - 2-d layout is more difficult than for binary trees
  - Number of long wires grows faster than for trees.
- For each source-destination pair there is only one route.
- Each route blocks many other routes.
Benes Networks

- Many routes;
- Costly to compute non-blocking routes;
- High probability for non-blocking route by randomly selecting an intermediate node [Leighton, 1992];
Fat Trees

16-node 2-ary fat-tree
$k$-ary $n$-dimensional Fat Tree Characteristics

- Number of nodes $N = k^d$
- Number of switches $= k^{d-1}d$
- Switch degree $= 2k$
- Diameter $= 2d$
- Distance $\sim d$
- Network cost $= O(Nd)$
- Total bandwidth $= 2k^d db$
- Bisection bandwidth $= 2k^{d-1}b$

16-node 2-ary fat-tree
**k-ary n-Cubes versus k-ary d-dimensional Fat Trees**

### k-ary n-cubes:
- **Number of nodes**: \( N = k^d \)
- **Switch degree**: \( d \)
- **Diameter**: \( d(k - 1) \)
- **Distance**: \( d^{\frac{1}{2}}(k - 1) \)
- **Network cost**: \( O(N) \)
- **Total bandwidth**: \( 2Nb \)
- **Bisection bandwidth**: \( 2k^{(d-1)}b \)

### k-ary n-dimensional fat trees:
- **Number of nodes**: \( N = k^d \)
- **Number of switches**: \( k^{d-1}d \)
- **Switch degree**: \( 2k \)
- **Diameter**: \( 2d \)
- **Distance**: \( d \)
- **Network cost**: \( O(Nd) \)
- **Total bandwidth**: \( 2k^d db \)
- **Bisection bandwidth**: \( 2k^{d-1}b \)
Relation between Fat Tree and Hypercube

- Binary 2-dim fat tree
- Binary 1-cube
Relation between Fat Tree and Hypercube - cont’d

Binary 3-dim fat tree

Binary 2-cube

Binary 2-cube

A. Jantsch, KTH
Relation between Fat Tree and Hypercube - cont’d

binary 4−dim fat tree

binary 3−cube

binary 3−cube

A. Jantsch, KTH
## Topologies of Parallel Computers

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>nCUBE/2</td>
<td>Hypercube</td>
<td>25</td>
<td>1</td>
<td>40</td>
<td>32</td>
</tr>
<tr>
<td>TMC CM-5</td>
<td>Fat tree</td>
<td>25</td>
<td>4</td>
<td>10</td>
<td>4</td>
</tr>
<tr>
<td>IBM SP-2</td>
<td>Banyan</td>
<td>25</td>
<td>8</td>
<td>5</td>
<td>16</td>
</tr>
<tr>
<td>Intel Paragon</td>
<td>2D Mesh</td>
<td>11.5</td>
<td>8</td>
<td>2</td>
<td>16</td>
</tr>
<tr>
<td>Meiko CS-2</td>
<td>Fat tree</td>
<td>20</td>
<td>8</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>Cray T3D</td>
<td>3D Torus</td>
<td>6.67</td>
<td>16</td>
<td>2</td>
<td>16</td>
</tr>
<tr>
<td>DASH</td>
<td>Torus</td>
<td>30</td>
<td>16</td>
<td>2</td>
<td>16</td>
</tr>
<tr>
<td>J-Machine</td>
<td>3D Mesh</td>
<td>31</td>
<td>8</td>
<td>2</td>
<td>8</td>
</tr>
<tr>
<td>Monsoon</td>
<td>Butterfly</td>
<td>20</td>
<td>16</td>
<td>2</td>
<td>16</td>
</tr>
<tr>
<td>SGI Origin</td>
<td>Hypercube</td>
<td>2.5</td>
<td>20</td>
<td>16</td>
<td>160</td>
</tr>
<tr>
<td>Myricom</td>
<td>Arbitrary</td>
<td>6.25</td>
<td>16</td>
<td>50</td>
<td>16</td>
</tr>
</tbody>
</table>

A. Jantsch, KTH
Trade-offs in Topology Design for the $k$-ary $n$-Cube

- Unloaded Latency
- Latency under Load
Network Scaling for Unloaded Latency

Latency($n$) = Admission + ChannelOccupancy + RoutingDelay + ContentionDelay

RoutingDelay $T_{ct}(n, h) = \frac{n}{b} + h\Delta$

RoutingDistance $h = \frac{1}{2}d(k - 1) = \frac{1}{2}(k - 1) \log_k N = \frac{1}{2}(d\sqrt{N} - 1)$
Unloaded Latency for Small Networks and Local Traffic

Network scalability wrt latency (m=128)

- k=2
- d=5
- d=4
- d=3
- d=2

Network scalability wrt latency (m=128; h=dk/5)

- k=2
- d=5
- d=4
- d=3
- d=2

A. Jantsch, KTH
**Unloaded Latency under a Free-Wire Cost Model**

**Free-wire** cost model: Wires are free and can be added without penalty.
Unloaded Latency under a Fixed-Wire Cost Models

**Fixed-wire** cost model: The number of wires is constant per node:

128 wires per node: \( w(d) = \left\lfloor \frac{64}{d} \right\rfloor \).

<table>
<thead>
<tr>
<th>d</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>w(d)</td>
<td>32</td>
<td>21</td>
<td>16</td>
<td>12</td>
<td>10</td>
<td>9</td>
<td>8</td>
<td>7</td>
<td>6</td>
</tr>
</tbody>
</table>

Latency wrt dimension under fixed−wire cost model (m=32;b=64/d)

Latency wrt dimension under fixed−wire cost model (m=128;b=64/d)
Unloaded Latency under a Fixed-Bisection Cost Models

**Fixed-bisection** cost model: The number of wires across the bisection is constant:

\[
bisection = 1024 \text{ wires: } w(d) = \frac{k}{2} = \frac{d\sqrt{N}}{2}.
\]

Example: \( N=1024 \):

\[
\begin{array}{cccccccccc}
d & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\
w(d) & 512 & 16 & 5 & 3 & 2 & 2 & 1 & 1 & 1
\end{array}
\]

Latency wrt dimension under fixed-bisection cost model (\( m=32\text{B}; b=k/2 \))

Latency wrt dimension under fixed-bisection cost model (\( m=128\text{B}; b=k/2 \))
Unloaded Latency under a Logarithmic Wire Delay Cost Models

**Fixed-bisection Logarithmic Wire Delay** cost model: The number of wires across the bisection is constant and the delay on wires increases logarithmically with the length [Dally, 1990]:

Length of long wires: \( l = k^n/2 - 1 \)

\[
T_c \propto 1 + \log l = 1 + \left( \frac{d}{2} - 1 \right) \log k
\]
Unloaded Latency under a Linear Wire Delay Cost Models

**Fixed-bisection Linear Wire Delay** cost model: The number of wires across the bisection is constant and the delay on wires increases linearly with the length [Dally, 1990]:

Length of long wires: \( l = k^{n/2} - 1 \)

\[ T_c \propto l = k^{d/2} - 1 \]
Latency under Load

Assumptions [Agarwal, 1991]:

- $k$-ary $n$-cubes
- random traffic
- dimension-order cut-through routing
- unbounded internal buffers (to ignore flow control and deadlock issues)
Latency under Load - cont’d

Latency($n$) = Admission + ChannelOccupancy + RoutingDelay + ContentionDelay

\[ T(m, k, d, w, \rho) = \text{RoutingDelay} + \text{ContentionDelay} \]

\[ T(m, k, d, w, \rho) = \frac{m}{w} + dh_k(\Delta + W(m, k, d, w, \rho)) \]

\[ W(m, k, d, w, \rho) = \frac{m}{w} \cdot \frac{\rho}{1 - \rho} \cdot \frac{h_k - 1}{h_k^2} \cdot \left( 1 + \frac{1}{d} \right) \]

\[ h = \frac{1}{2}d(k - 1) \]

$m \cdots$ message size

$w \cdots$ bitwidth of link

$\rho \cdots$ aggregate channel utilization

$h_k \cdots$ average distance in each dimension

$\Delta \cdots$ switching time in cycles
Latency vs Channel Load

Latency wrt channel utilization (w=8; delta=1)

- m8,d3,k10
- m8,d2,k32
- m32,d3,k10
- m32,d2,k32
- m128,d3,k10
- m128,d2,k32

A. Jantsch, KTH
Routing

**Deterministic routing** The route is determined solely by source and destination locations.

**Arithmetic routing** The destination address of the incoming packet is compared with the address of the switch and the packet is routed accordingly. (relative or absolute addresses)

**Source based routing** The source determines the route and builds a header with one directive for each switch. The switches strip off the top directive.

**Table-driven routing** Switches have routing tables, which can be configured.

**Adaptive routing** The route can be adapted by the switches to balance the load.

**Minimal routing** allows only shortest paths while non-minimal routing allows even longer paths.
Deadlock

**Deadlock** Two or several packets mutually block each other and wait for resources, which can never be free.

**Livelock** A packet keeps moving through the network but never reaches its destination.

**Starvation** A packet never gets a resource because it always loses the competition for that resource (fairness).
Deadlock Situations

- Head-on deadlock;
- Nodes stop receiving packets;
- Contention for switch buffers can occur with store-and-forward, virtual-cut-through and wormhole routing. Wormhole routing is particularly sensible.
- Cannot occur in butterflies;
- Cannot occur in trees or fat trees if upward and downward channels are independent;
- Dimension order routing is deadlock free on $k$-ary $n$-arrays but not on tori with any $n \geq 1$. 
Deadlock in a 1-dimensional Torus

Message 1 from C→ B, 10 flits
Message 2 from A→ D, 10 flits
Routing is deadlock free if the channel dependence graph has no cycles.
Deadlock-free Routing

- Two main approaches:
  - Restrict the legal routes;
  - Restrict how resources are allocated;
- Number the channel cleverly
- Construct the channel dependence graph
- Prove that all legal routes follow a strictly increasing path in the channel dependence graph.
Virtual Channels

- Virtual channels can be used to break cycles in the dependence graph.
- E.g. all \( n \)-dimensional tori can be made deadlock free under dimension-order routing by assigning all wrap-around paths to a different virtual channel than other links.
Virtual Channels and Deadlocks
Turn-Model Routing

What are the minimal routing restrictions to make routing deadlock free?

- Three minimal routing restriction schemes:
  - North-last
  - West-first
  - Negative-first

- Allow complex, non-minimal adaptive routes.
- Unidirectional $k$-ary $n$-cubes still need virtual channels.

A. Jantsch, KTH
Adaptive Routing

- The switch makes routing decisions based on the load.
- Fully adaptive routing allows all shortest paths.
- Partial adaptive routing allows only a subset of the shortest path.
- Non-minimal adaptive routing allows also non-minimal paths.
- Hot-potato routing is non-minimal adaptive routing without packet buffering.
Summary

- Communication Performance: bandwidth, unloaded latency, loaded latency
- Organizational Structure: NI, switch, link
- Topologies: wire space and delay domination favors low dimension topologies;
- Routing: deterministic vs source based vs adaptive routing; deadlock;
Issues beyond the Scope of this Lecture

- Switch: Buffering; output scheduling; flow control;
- Flow control: Link level and end-to-end control;
- Power
- Clocking
- Faults and reliability
- Memory architecture and I/O
- Application specific communication patterns
- Services offered to applications; Quality of service
NoC Research Projects

- Nostrum at KTH
- Æthereal at Philips Research
- Proteo at Tampere University of Technology
- SPIN at UPMC/LIP6 in Paris
- XPipes at Bologna U
- Octagon at ST and UC San Diego
To Probe Further - Books and Classic Papers


