Computer Architecture
ELE 475 / COS 475
Slide Deck 14: Interconnection Networks
David Wentzlaaff
Department of Electrical Engineering
Princeton University
Overview of Interconnection Networks: Buses
Overview of Interconnection Networks: Buses

Diagram of interconnection networks with four core nodes connected by a bus.
Overview of Interconnection Networks: Buses
Overview of Interconnection Networks: Point-to-point / Switched
Overview of Interconnection Networks: Point-to-point / Switched
Explicit Message Passing (Programming)

- Send(Destination, *Data)
- Receive(&Data)
- Receive(Source, &Data)

- Unicast (one-to-one)
- Multicast (one-to-multiple)
- Broadcast (one-to-all)
Message Passing Interface (MPI)

```c
#include <stdio.h>
#include <assert.h>
#include <mpi.h>
int main (int argc, char **argv) {
    int myid, numprocs, x, y;
    int tag = 475;
    MPI_Status status;
    MPI_Init(&argc,&argv);
    MPI_Comm_size(MPI_COMM_WORLD,&numprocs);
    MPI_Comm_rank(MPI_COMM_WORLD,&myid);
    assert(numprocs == 2);
    if(myid==0) {
        x = 475;
        MPI_Send(&x, 1, MPI_INT, 1, tag, MPI_COMM_WORLD);
        MPI_Recv(&y, 1, MPI_INT, 1, tag, MPI_COMM_WORLD, &status);
        printf("received number: ELE %d\n", y);
    }
    else {
        MPI_Recv(&y, 1, MPI_INT, 0, tag, MPI_COMM_WORLD, &status);
        y += 100;
        MPI_Send(&y, 1, MPI_INT, 0, tag, MPI_COMM_WORLD);
    }
    MPI_Finalize();
    exit(0);
}
```
Message Passing vs. Shared Memory

• **Message Passing**
  – Memory is private
  – Explicit send/receive to communicate
  – Message contains data and synchronization
  – Need to know Destination on generation of data (send)
  – Easy for Producer-Consumer

• **Shared Memory**
  – Memory is shared
  – Implicit communication via loads and stores
  – Implicit synchronization needed via Fences, Locks, and Flags
  – No need to know Destination on generation of data (can store in memory and user of data can pick up later)
  – Easy for multiple threads accessing a shared table
  – Needs Locks and critical sections to synchronize access
Shared Memory Tunneled over Messaging

• Software
  – Turn loads and stores into sends and receives

• Hardware
  – Replace bus communications with messages sent between cores and between cores and memory
Shared Memory Tunneled over Messaging

• Software
  – Turn loads and stores into sends and receives

• Hardware
  – Replace bus communications with messages sent between cores and between cores and memory
Messaging Tunneled over Shared Memory

• Use software queues (FIFOs) with locks to transmit data directly between cores by loads and stores to memory
Interconnect Design

• Switching
• Topology
• Routing
• Flow Control
Anatomy of a Message

- **Flit**: flow control digit (Basic unit of flow control)
- **Phit**: physical transfer digit (Basic unit of data transferred in one clock)
Switching

• Circuit Switched
• Store and Forward
• Cut-through
• Wormhole
Topology

Bus

1  2  3  4

Pipelined Bus / Segmented Bus

1  2  3  4
Topology

Ring / 1D Torus
Topology

2D Mesh

2D Torus
Topology

Hyper Cube
Topology

Star / Fully connected crossbar
Topology

Omega Network
Topology

Fat Tree
Topology

- **K-ary**
  - Nodes in each dimension

- **N-cube**
  - N dimensional grid

**3-ary 3-cube mesh**
Topology Parameters

- **Routing Distance**: Number of links between two points
- **Diameter**: Maximum routing distance between any two points
- **Average Distance**
- **Minimum Bisection Bandwidth (Bisection Bandwidth)**: The bandwidth of a minimal cut though the network such that the network is divided into two sets of nodes
- **Degree of a Router**
Topology Parameters

Diameter: \( 2\sqrt{N} - 2 \)

Bisection Bandwidth: \( 2\sqrt{N} \)

Degree of a Router: 5
Topology Influenced by Packaging

- Wiring grows as N-1
- Physically hard to pack into 3-space (pack in sphere?)

Star/Fully connected crossbar
Topology Influenced by Packaging

- Packing $N$ dimensions in $N-1$ space leads to long wires
- Packing $N$ dimensions in $N-2$ space leads to really long wires
Network Performance

• Bandwidth: The rate of data that can be transmitted over the network (network link) in a given time
• Latency: The time taken for a message to be sent from sender to receiver

• Bandwidth can affect latency
  – Reduce congestion
  – Messages take fewer Flits and Phits

• Latency can affect Bandwidth
  – Round trip communication can be limited by latency
  – Round trip flow-control can be limited by latency
Packet 1

<table>
<thead>
<tr>
<th>Head Phit</th>
<th>Body Phit</th>
<th>Body Phit</th>
<th>Tail Phit</th>
</tr>
</thead>
</table>

Serialization Latency L/b
Channel Latency tc
Router Latency tR

Router Pipeline: R0 R1 R2
Link traversal: L0 C1

29
Anatomy of Message Latency

\[ T = T_{\text{head}} + \frac{L}{b} \]

\( T_{\text{head}} \): Head Phit Latency, includes \( t_C \), \( t_R \), hop count, and contention

Unloaded Latency:

\[ T_0 = H_R \times t_R + H_C \times t_C + \frac{L}{b} \]
Anatomy of Message Latency

Packet 1 Head Phy 

Body Phy 

Tail Phy 

Serialization Latency L/b 

Channel Latency t_C 

Router Pipeline Latency t_R 

Unloaded Latency:

\[ T_0 = H_R \times t_R + H_C \times t_C + L/b \] 

Shorter routes Faster routes Faster channels Wider channels or shorter messages
Interconnection Network Performance

- Latency (seconds)
- Offered Bandwidth (bits/second)
- Ideal Throughput
- Flow Control
- Routing
- Topology

Zero-Load Latency
Routing

• Oblivious (routing path independent of state of network)
  – Deterministic
  – Non-Deterministic

• Adaptive (routing path depends on state of network)
Flow Control

- Local (Link or hop based) Flow Control
- End-to-end (Long distance)
Deadlock

• Deadlock can occur if cycle possible in “Waits-for” graph
Deadlock Example (Waits-for and Holds analysis)
Deadlock Avoidance vs. Deadlock Recovery

• Deadlock Avoidance
  – Protocol designed to never deadlock

• Deadlock Recovery
  – Allow Deadlock to occur and then resolve deadlock usually through use of more buffering
Acknowledgements

- These slides contain material developed and copyright by:
  - Arvind (MIT)
  - Krste Asanovic (MIT/UCB)
  - Joel Emer (Intel/MIT)
  - James Hoe (CMU)
  - John Kubiatowicz (UCB)
  - David Patterson (UCB)
  - Christopher Batten (Cornell)

- MIT material derived from course 6.823
- UCB material derived from course CS252 & CS152
- Cornell material derived from course ECE 4750
On/off with Combinational Stall Signal

Packet

0  A B C D
1  A B C D
2  A B C
3  A B C C
4  A B B B B B B C
5  A A A A A A B C D
6  A A A A A
7  A A A A A A B C D

Stall D

Unstall D
On/Off with Pipelined Stall Signal

Diagram with labels and connections for pipeline stages and output signals.
On/Off with Partial Pipelined Stall Signal

A B C D
A B C D
A B C D D D D D D D
A B C D
A B C D
A B B B B B C D
A A A A A B C D
A B C D
Credit-Based Flow Control

- Decrement counter on send packet
- Increment counter on credit received