Interconnection Networks:

Flow Control

Prof. Natalie Enright Jerger
Switching/Flow Control Overview

• Topology: determines connectivity of network

• Routing: determines paths through network

• Flow Control: determine allocation of resources to messages as they traverse network
  – Buffers and links
  – Significant impact on throughput and latency of network
Flow Control

- Control state records
  - allocation of channels and buffers to packets
  - current state of packet traversing node
- Channel bandwidth advances flits from this node to next
- Buffers hold flits waiting for channel bandwidth
Packets

• Messages: composed of one or more packets
  – If message size is <= maximum packet size only one packet created

• Packets: composed of one or more flits

• Flit: flow control digit

• Phit: physical digit
  – Subdivides flit into chunks = to link width
• Off-chip: channel width limited by **pins**
  – Requires phits
• On-chip: **abundant** wiring means phit size == flit size
• Packet contains destination/route information
  – Flits may not → all flits of a packet must take same route
Switching

• Different flow control techniques based on granularity
  
  – **Message-based**: allocation made at message granularity (circuit-switching)
  
  – **Packet-based**: allocation made to whole packets
  
  – **Flit-based**: allocation made on a flit-by-flit basis
Message-Based Flow Control

• Coarsest granularity

• Circuit-switching
  – Pre-allocates resources across multiple hops
    • Source to destination
    • Resources = links
    • Buffers are not necessary
  – Probe sent into network to reserve resources
Circuit Switching

• Once probe sets up circuit
  – Message does not need to perform any routing or allocation at each network hop
  – Good for transferring large amounts of data
    • Can amortize circuit setup cost by sending data with very low per-hop overheads

• No other message can use those resources until transfer is complete
  – Throughput can suffer due setup and hold time for circuits
  – Links are idle until setup is complete
Circuit Switching Example

- Significant latency overhead prior to data transfer
  - Data transfer does not pay per-hop overhead for routing and allocation
• When there is contention
  – Significant wait time
  – Message from 1 → 2 must wait
Time-Space Diagram: Circuit-Switching

Time to setup+ack circuit from 0 to 8

Time setup from 2 to 8 is blocked
Packet-based Flow Control

• Break messages into packets

• Interleave packets on links
  – Better utilization

• Requires per-node **buffering** to store in-flight packets

• Two types of packet-based techniques
Store and Forward

• Links and buffers are allocated to entire packet

• Head flit \textit{waits} at router until entire packet is received before being forwarded to the next hop

• Not suitable for on-chip
  – Requires buffering at each router to hold entire packet
    • Packet cannot traverse link until buffering allocated to entire packet
  – Incurs high latencies (pays \textit{serialization} latency at each hop)
Store and Forward Example

- High per-hop latency
  - Serialization delay paid at each hop
- Larger buffering required

Total delay = 4 cycles per hop x 3 hops = 12 cycles
Time-Space Diagram: Store and Forward
Packet-based: Virtual Cut Through

• Links and Buffers allocated to entire packets

• Flits can proceed to next hop before tail flit has been received by current router
  – But only if next router has enough buffer space for entire packet

• Reduces the latency significantly compared to SAF

• But still requires large buffers
  – Unsuitable for on-chip
Virtual Cut Through Example

- Lower per-hop latency
- Large buffering required

Total delay = 1 cycle per hop × 3 hops + serialization = 6 cycles

Allocate 4 flit-sized buffers before head proceeds

Allocate 4 flit-sized buffers before head proceeds
Time-Space Diagram: VCT

<table>
<thead>
<tr>
<th>Location</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>5</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>H</td>
<td>B</td>
<td>B</td>
<td>B</td>
<td>T</td>
</tr>
<tr>
<td>1</td>
<td>H</td>
<td>B</td>
<td>B</td>
<td>B</td>
<td>T</td>
</tr>
<tr>
<td>2</td>
<td>H</td>
<td>B</td>
<td>B</td>
<td>B</td>
<td>T</td>
</tr>
<tr>
<td>5</td>
<td>H</td>
<td>B</td>
<td>B</td>
<td>B</td>
<td>T</td>
</tr>
<tr>
<td>8</td>
<td>H</td>
<td>B</td>
<td>B</td>
<td>B</td>
<td>T</td>
</tr>
</tbody>
</table>

Time

0 1 2 3 4 5 6 7 8
Virtual Cut Through

- Throughput suffers from inefficient buffer allocation

Cannot proceed because only 2 flit buffers available
Time-Space Diagram: VCT (2)

- Location
- Time

Insufficient Buffers

Time - Space Diagram: VCT (2)
Flit-Level Flow Control

• Help routers meet tight area/power constraints

• Flit can proceed to next router when there is buffer space available for that flit
  – Improves over SAF and VCT by allocating buffers on a flit-by-flit basis
Wormhole Flow Control

• Pros
  – More efficient buffer utilization (good for on-chip)
  – Low latency

• Cons
  – Poor link utilization: if head flit becomes blocked, all links spanning length of packet are idle
    • Cannot be re-allocated to different packet
    • Suffers from head of line (HOL) blocking
Wormhole Example

- 6 flit buffers/input port

Red holds this channel: channel remains idle until read proceeds
Channel idle but red packet blocked behind blue
Buffer full: blue cannot proceed
Blocked by other packets
Time-Space Diagram: Wormhole

Contention

Location

Time
Virtual Channels

• First proposed for deadlock avoidance
  – We’ll come back to this

• Can be applied to any flow control
  – First proposed with wormhole
Virtual Channel Flow Control

• Virtual channels used to combat **HOL** blocking in wormhole

• Virtual channels: **multiple** flit queues per input port
 —Share **same** physical link (channel)

• Link utilization improved
 — Flits on different VC can pass blocked packet
Virtual Channel Flow Control (2)
Virtual Channel Flow Control (3)

In1

<table>
<thead>
<tr>
<th>1</th>
<th>1</th>
<th>2</th>
<th>2</th>
<th>3</th>
<th>3</th>
<th>3</th>
<th>3</th>
<th>2</th>
<th>2</th>
<th>1</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>AH</td>
<td>A1</td>
<td>A2</td>
<td>A3</td>
<td>A4</td>
<td>A5</td>
<td>AT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

In2

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>2</th>
<th>3</th>
<th>3</th>
<th>3</th>
<th>3</th>
<th>3</th>
<th>2</th>
<th>2</th>
<th>1</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>BH</td>
<td>B1</td>
<td>B2</td>
<td>B3</td>
<td>B4</td>
<td>B5</td>
<td>BT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Out

| AH | BH | A1 | B1 | A2 | B2 | A3 | B3 | A4 | B4 | A5 | B5 | AT | BT |

A Downstream

| AH | A1 | A2 | A3 | A4 | A5 | AT |

B Downstream

| BH | B1 | B2 | B3 | B4 | B5 | BT |
Virtual Channel Flow Control (3)

• Packets compete for VC on flit by flit basis

• In example: on downstream links, flits of each packet are available every other cycle

• Upstream links throttle because of limited buffers

• Does not mean links are idle
  – May be used by packet allocated to other VCs
Virtual Channel Example

- 6 flit buffers/input port
- 3 flit buffers/VC

Buffer full: blue cannot proceed
Blocked by other packets
## Summary of techniques

<table>
<thead>
<tr>
<th>Links</th>
<th>Buffers</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Circuit-Switching</td>
<td>Messages</td>
<td>N/A (buffer-less)</td>
</tr>
<tr>
<td>Store and Forward</td>
<td>Packet</td>
<td>Packet</td>
</tr>
<tr>
<td>Virtual Cut Through</td>
<td>Packet</td>
<td>Packet</td>
</tr>
<tr>
<td>Wormhole</td>
<td>Packet</td>
<td>Flit</td>
</tr>
<tr>
<td>Virtual Channel</td>
<td>Flit</td>
<td>Flit</td>
</tr>
</tbody>
</table>
Deadlock

• Using flow control to guarantee deadlock freedom give more flexible routing
  – Recall: routing restrictions needed for deadlock freedom

• If routing algorithm is not deadlock free
  – VCs can break resource cycle

• Each VC is time-multiplexed onto physical link
  – Holding VC implies holding associated buffer queue
  – Not tying up physical link resource

• Enforce order on VCs
Deadlock: Enforce Order

- All messages sent through VC 0 until cross dateline.
- After dateline, assigned to VC 1.
  - Cannot be allocated to VC 0 again.
Deadlock: Escape VCs

• Enforcing order lowers VC utilization
  – Previous example: VC 1 underutilized

• Escape Virtual Channels
  – Provide an escape path for every packet in a potential cycle
  – Have 1 VC that uses a deadlock free routing subfunction
  – Example: VC 0 uses DOR, other VCs use arbitrary routing function
  – Access to VCs arbitrated fairly: packet always has chance of landing on escape VC
    • Once in escape VC, must remain there (routing restrictions)

• Assign different message classes to different VCs to prevent protocol level deadlock
  – Prevent req-ack message cycles
Virtual Channel Reallocation

• When can a VC be reallocated to a new packet?
• Two options: conservative and aggressive
  – Conservative (atomic): only reallocate empty VC
  – Aggressive (non-atomic): reallocate once tail flit has been received
• Implications for deadlock
VCT + non-atomic allocation

• Typically uses non-atomic VC allocation
• VC allocated only if there is enough free buffer space to hold entire packet
  – Guarantees that whole packet can be received by VC
    • Will entirely vacate upstream VC
    • New packet at head of VC can bid for resources
VC Reallocation: Wormhole + Aggressive

- Turn-model and deterministic routing
  - No cyclic channel dependence
  - Can use aggressive reallocation

- High VC utilization but low adaptivity

P1 can be forwarded immediately
VC Reallocation: Wormhole + Adaptive Routing

- Adaptive routing
  - Cyclic channel dependence
- With aggressive reallocation
  - Packet spans 2 VCs and head flit not at head of one VC
    - Cannot reach escape VC
    - Leads to deadlock
- Requires conservative reallocation
  - Can only reallocate an empty VC
  - Low VC utilization
Buffer Backpressure

• Need mechanism to prevent buffer overflow
  – Avoid dropping packets
  – Upstream nodes need to know buffer availability at downstream routers

• Significant impact on throughput achieved by flow control

• Two common mechanisms
  – Credits
  – On-off
Credit-Based Flow Control

• Upstream router stores credit counts for each downstream VC

• Upstream router
  – When flit forwarded
    • Decrement credit count
  – Count == 0, buffer full, stop sending

• Downstream router
  – When flit forwarded and buffer freed
    • Send credit to upstream router
    • Upstream increments credit count
Credit Timeline

- Round-trip credit delay:
  - Time between when buffer empties and when next flit can be processed from that buffer entry
  - If only single entry buffer, would result in significant throughput degradation
  - Important to size buffers to tolerate credit turn-around
On-Off Flow Control

- Credit: requires upstream signaling for every flit
- On-off: decreases upstream signaling

On signal
  - Sent when number of free buffers rises above threshold $F_{on}$

Off signal
  - Sent when number of free buffers falls below threshold $F_{off}$
• Less signaling but more buffering
  – On-chip buffers more expensive than wires
Buffer Sizing

• Prevent backpressure from limiting throughput
  – Buffers must hold flits $\geq$ turnaround time

• Assume:
  – 1 cycle propagation delay for data and credits
  – 1 cycle credit processing delay
  – 3 cycle router pipeline

• At least 6 flit buffers
Actual Buffer Usage & Turnaround Delay

Flit arrives at node 1 and uses buffer

Credit propagation delay

Flit leaves node 1 and credit is sent to node 0

Credit pipeline delay

Node 0 receives credit, freed buffer reallocated to new flit

Flit pipeline delay

New flit leaves Node 0 for Node 1

Node 0 processes credit

New flit arrives at Node 1 and reuses buffer
Flow Control Summary

• On-chip networks require techniques with lower buffering requirements
  – Wormhole or Virtual Channel flow control

• Avoid dropping packets in on-chip environment
  – Requires buffer backpressure mechanism

• Complexity of flow control impacts router microarchitecture (next)