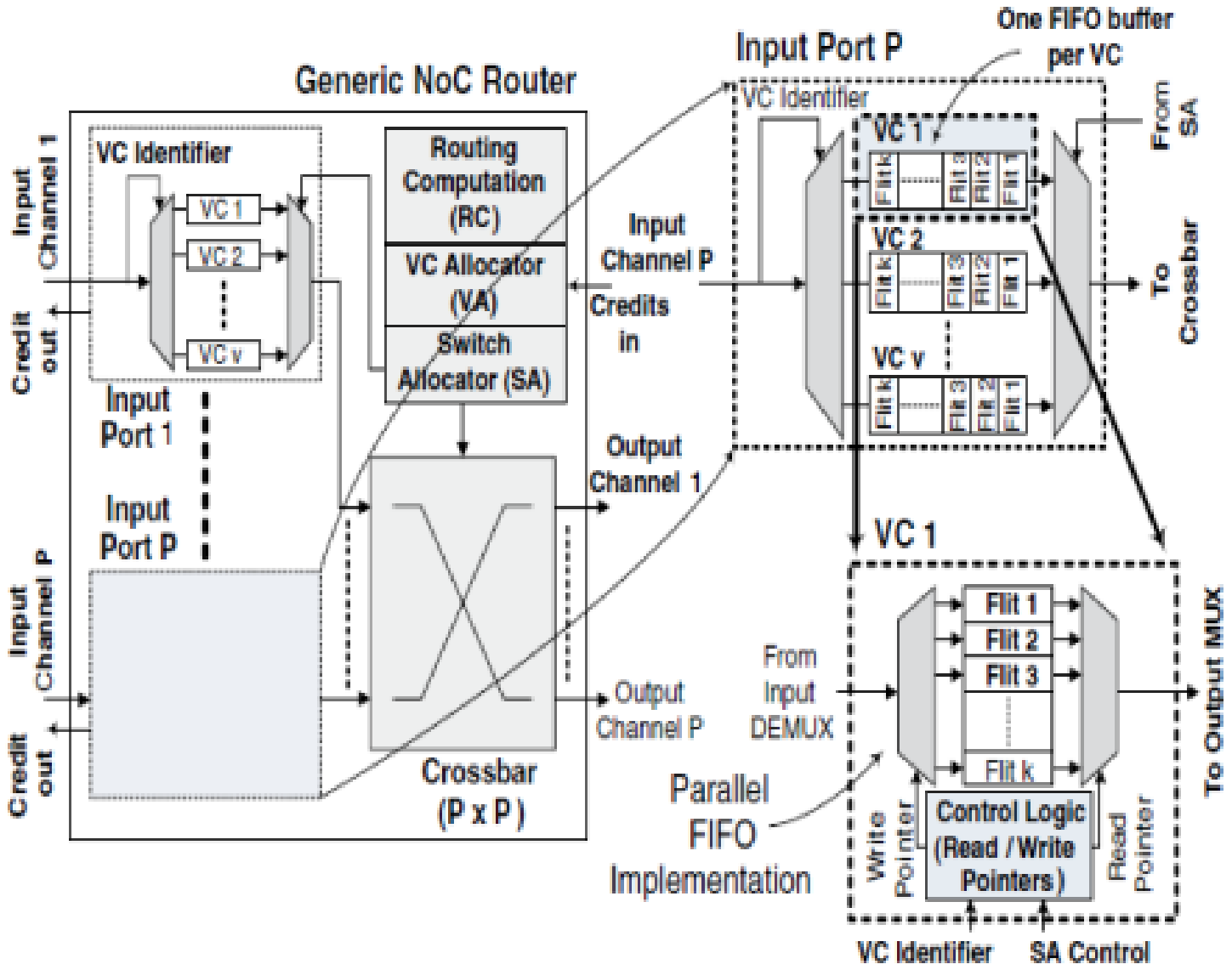
Introduction to Network on Chip (2)

Kun-Chih (Jimmy) Chen 陳坤志

kcchen@nycu.edu.tw

Institute of Electronics,
National Yang Ming Chiao Tung University

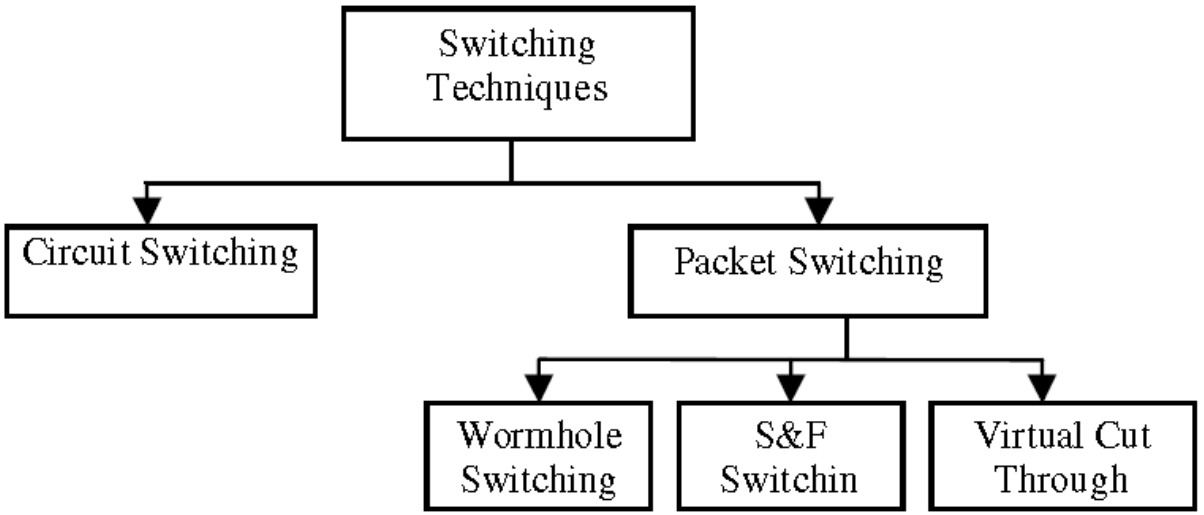
Generic NoC Router Architecture



Switching

NoC Switching

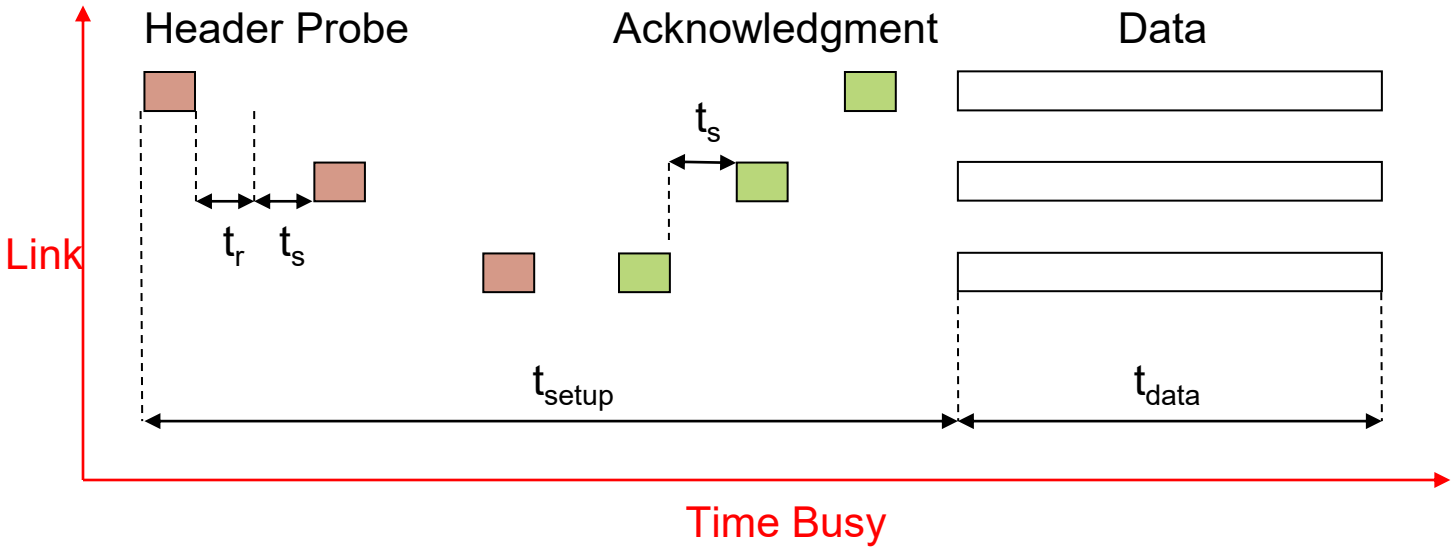
- Switching techniques define the way and time of connections between input and output ports **inside a switch**.
- Circuit switched networks reserve a physical path before transmitting the data packets
- Packet switched networks transmit the packets without reserving the entire path.



Circuit Switching

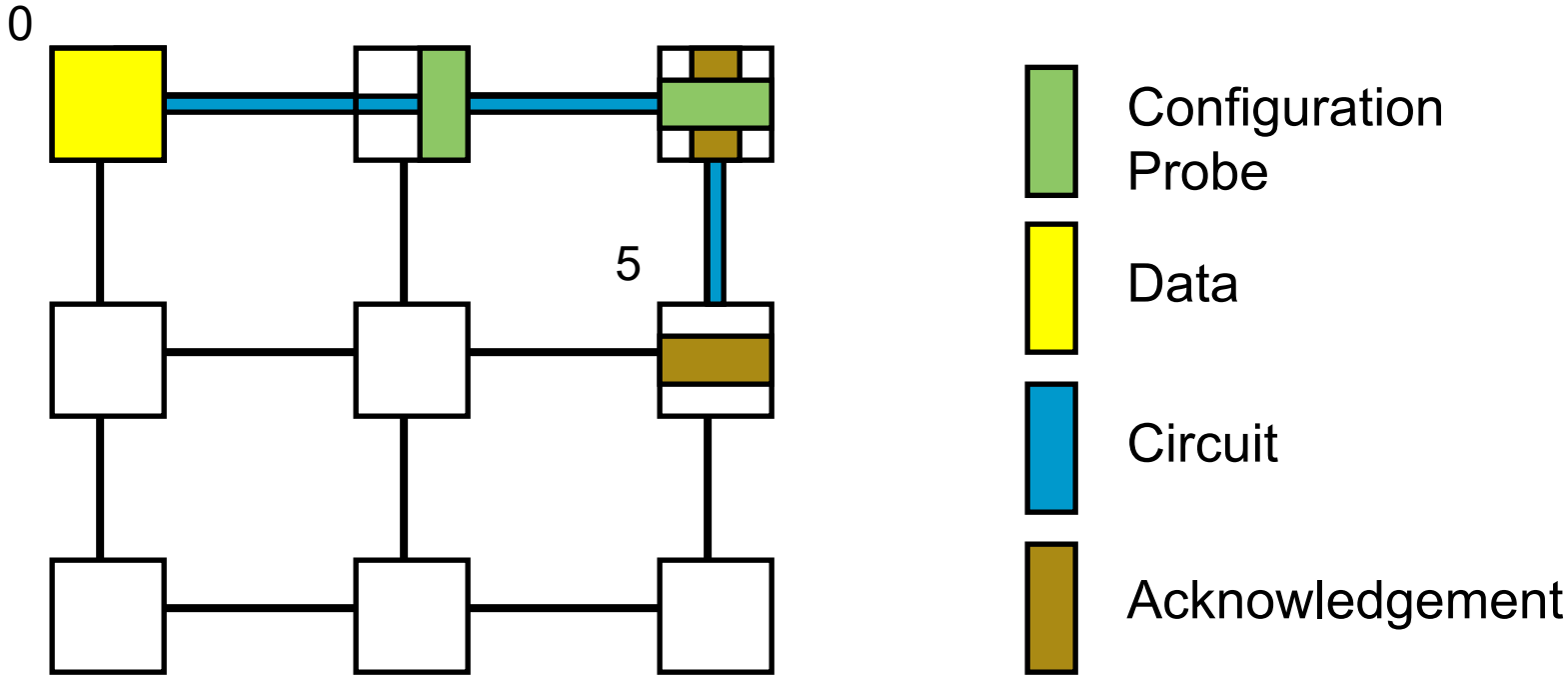
- All resources (from source to destination) are allocated to the message prior to transport
  - Probe sent into network to reserve resources
- Once probe sets up circuit
  - Message does not need to perform any routing or allocation at each network hop
  - Good for transferring large amounts of data
    - 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

Circuit Switching



- Hardware path setup by a *routing header* or *probe*
- End-to-end acknowledgment initiates transfer at full hardware bandwidth

Circuit Switching Example



- Significant latency overhead prior to data transfer
- Other requests forced to wait for resources

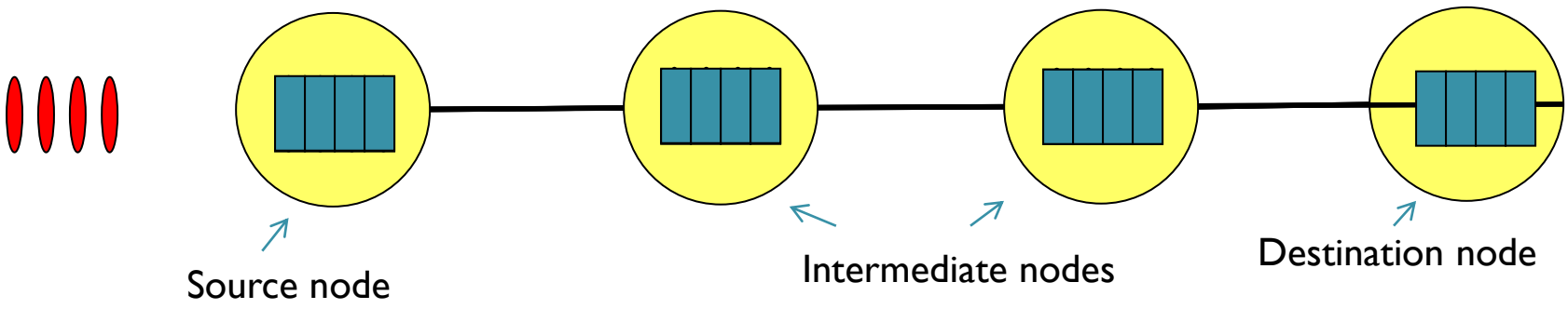
Circuit Switching Delay

$$T = 3\,H\,tr + L/b$$

H : time required to set up the channel and delivers the head flit
tr: serialization latency
L: size of the packet
b: bandwidth (contention time)

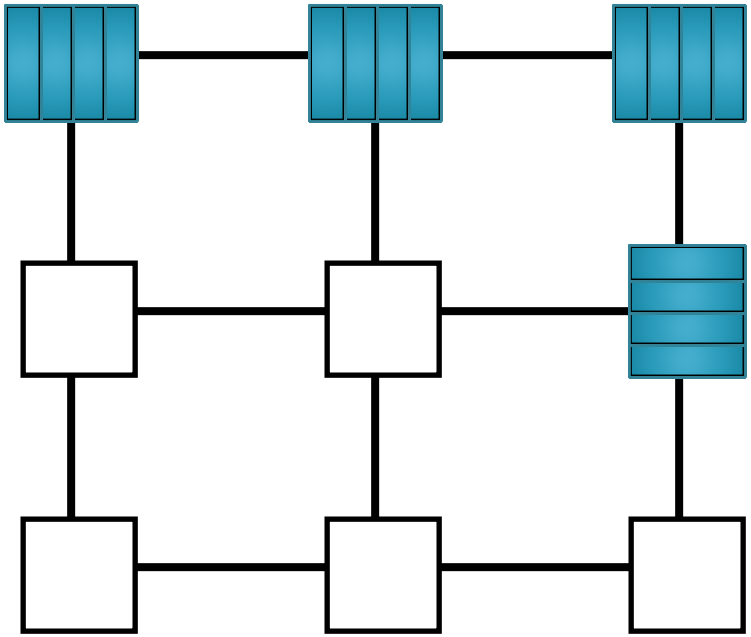
Note: 3 x header latency because the path from source to destination must be traversed 3 times to deliver the packet: Once in each direction to set up the circuit and then again to deliver the first flit

Store & Forward Switching



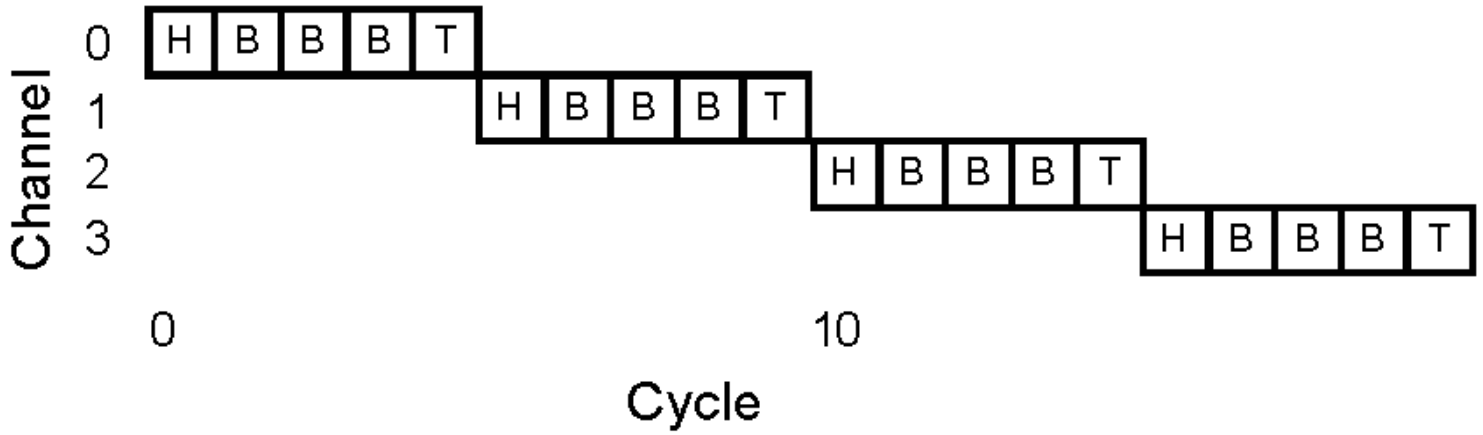
- Each node along a route waits until a packet is completely received (stored) and then the packet is forwarded to the next node
- Two resources are needed
  - Packet-sized buffer in the switch
  - Exclusive use of the outgoing channel

Store & Forward Switching Example



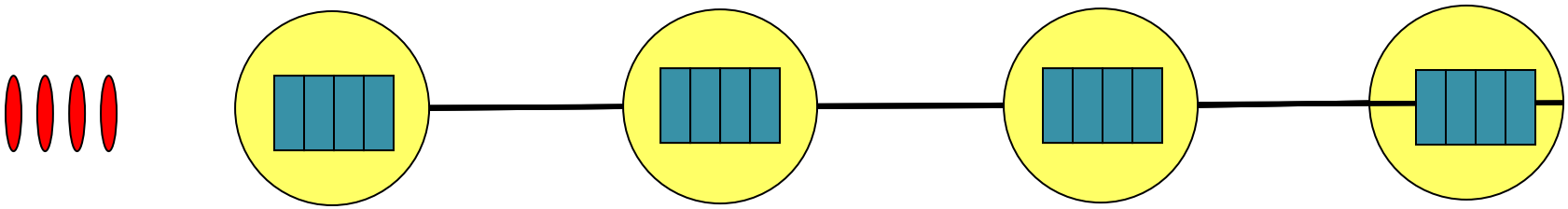
- High per-hop latency
- Larger buffering required

Store & Forward Switching

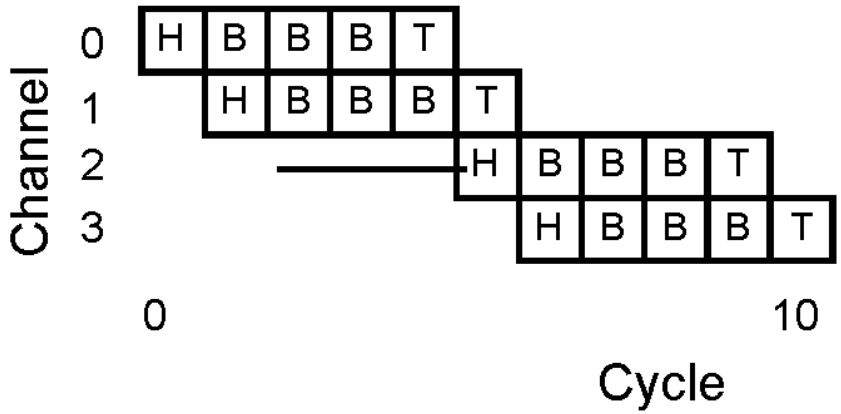


- Advantage
  - While waiting to acquire resources, no channels are being held idle
- Disadvantage
  - Requires a large amount of buffer space at each node
  - Very long latency

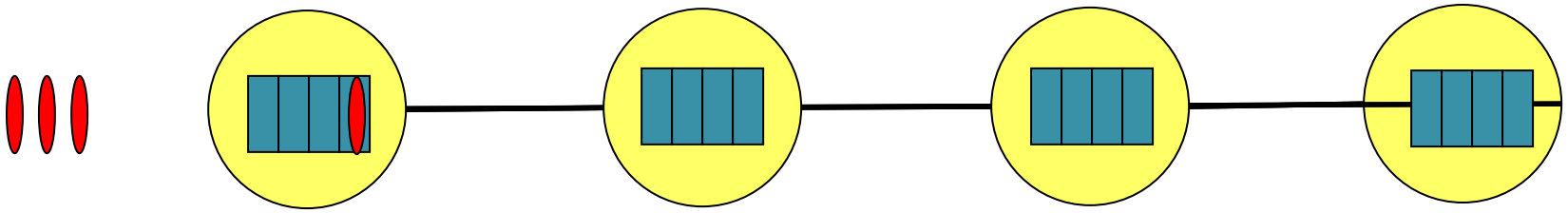
Virtual Cut-through Switching



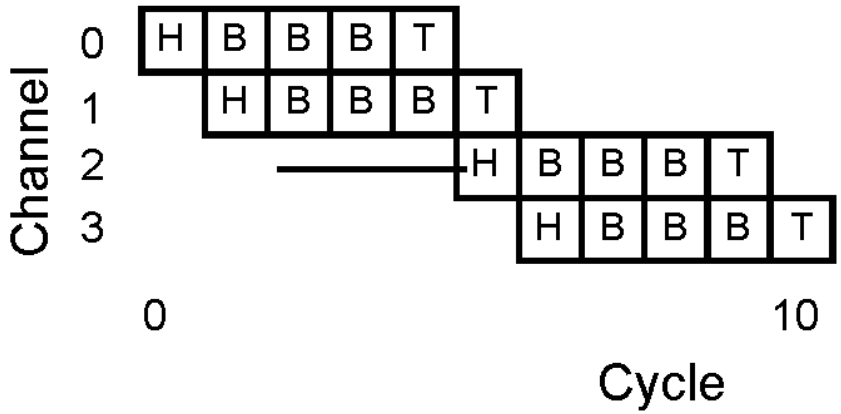
- Transmission on the next channel starts directly when the new header flit is received
- Channel is released after tail flit



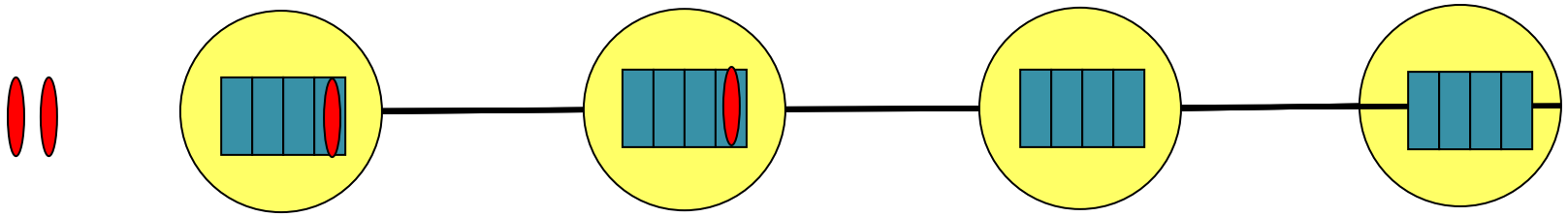
Virtual Cut-through Switching



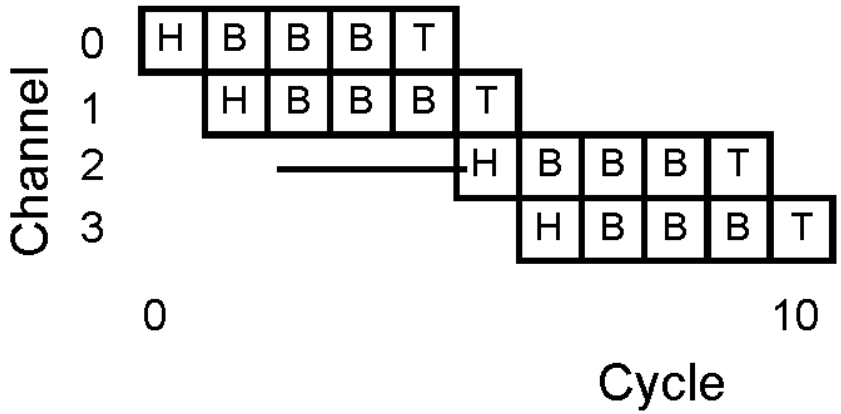
- Transmission on the next channel starts directly when the new header flit is received
- Channel is released after tail flit



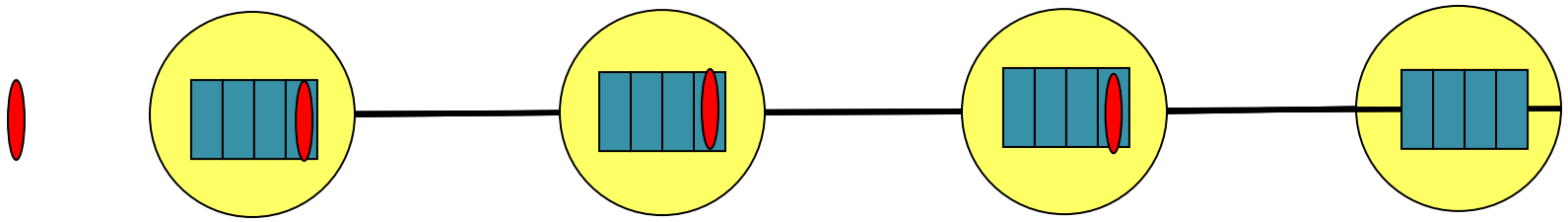
Virtual Cut-through Switching



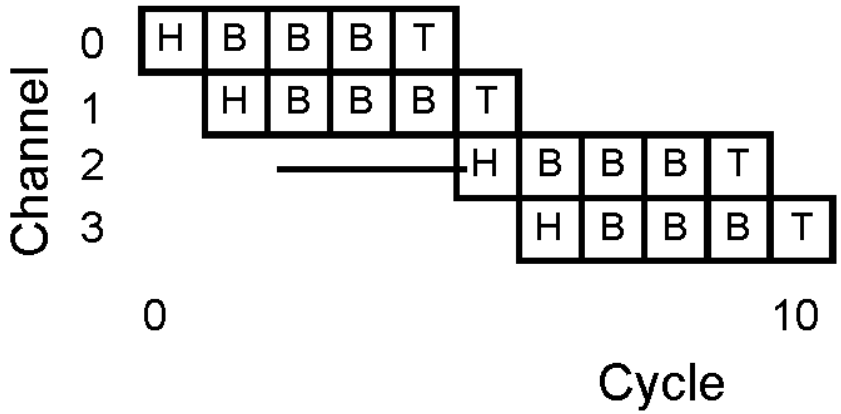
- Transmission on the next channel starts directly when the new header flit is received
- Channel is released after tail flit



Virtual Cut-through Switching



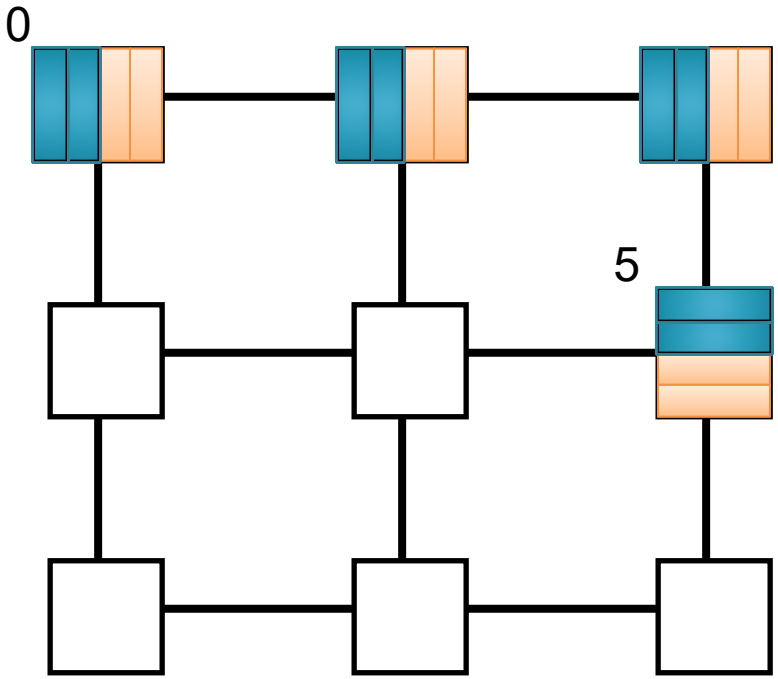
- Transmission on the next channel starts directly when the new header flit is received
- Channel is released after tail flit



Virtual Cut-through Switching

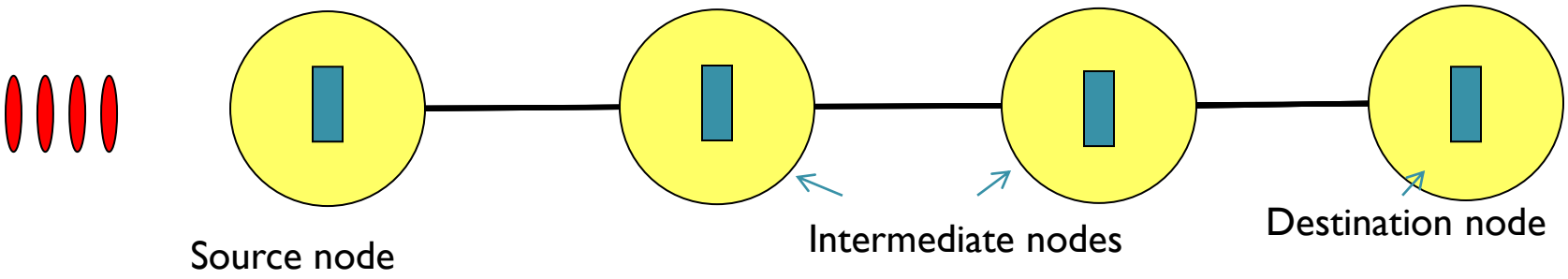
- Packet-based: similar to Store and Forward
- Links and Buffers allocated to entire packets
- Flits can proceed to next hop before tail flit has been received by current router
  - But only if next router has enough buffer space for entire packet
- Reduces the latency significantly compared to SAF
- But still requires large buffers
  - Unsuitable for hardware implementation

Virtual Cut-through Switching Example



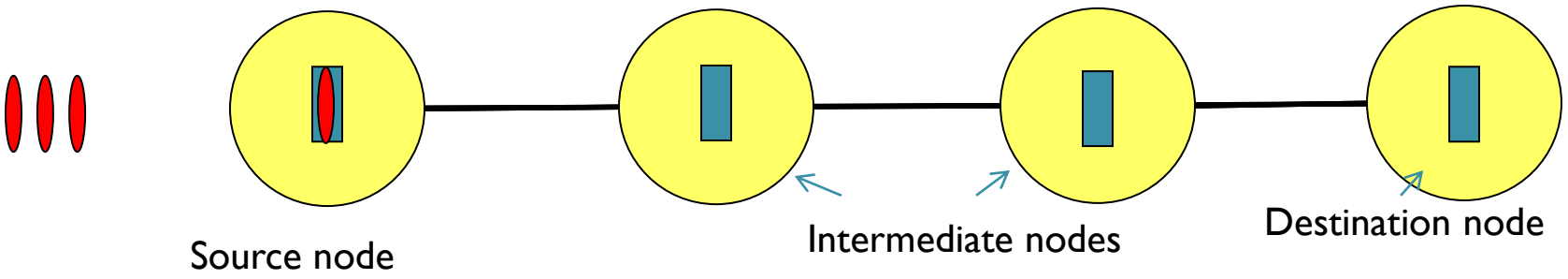
- Shorter per-hop latency
- Larger buffering required

Wormhole Switching



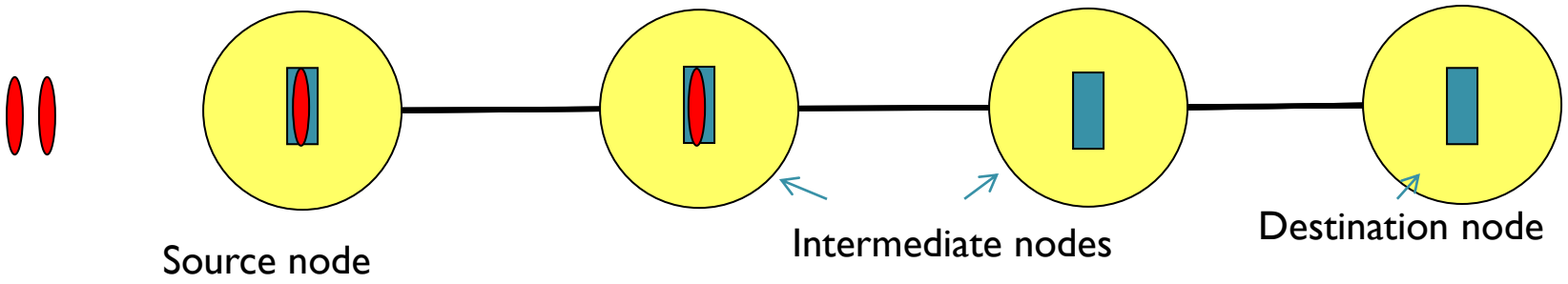
- Large packets are divided into small flits
- An entire packet need not be buffered to move on to the next node, increasing throughput.
- More efficient use of buffers than virtual cut-through
- Bandwidth and Channel allocation are decoupled

Wormhole Switching



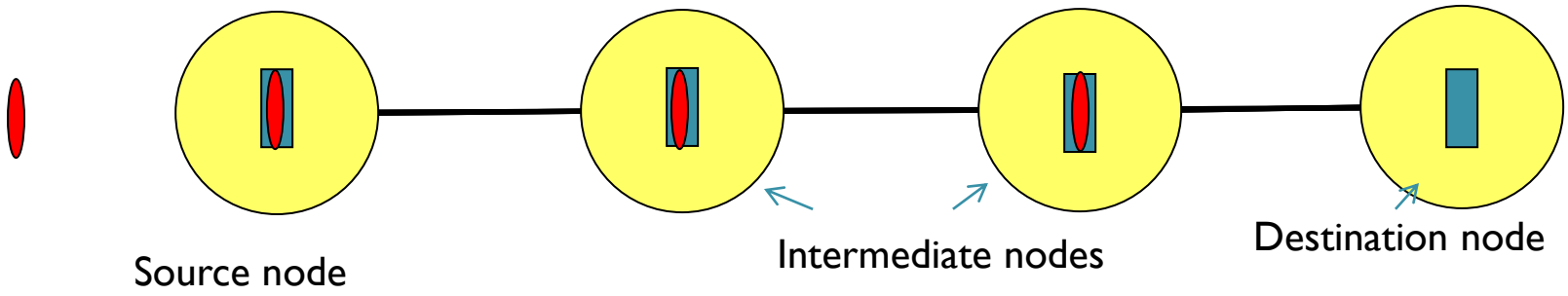
- Large packets are divided into small flits
- An entire packet need not be buffered to move on to the next node, increasing throughput.
- More efficient use of buffers than virtual cut-through
- Bandwidth and Channel allocation are decoupled

Wormhole Switching



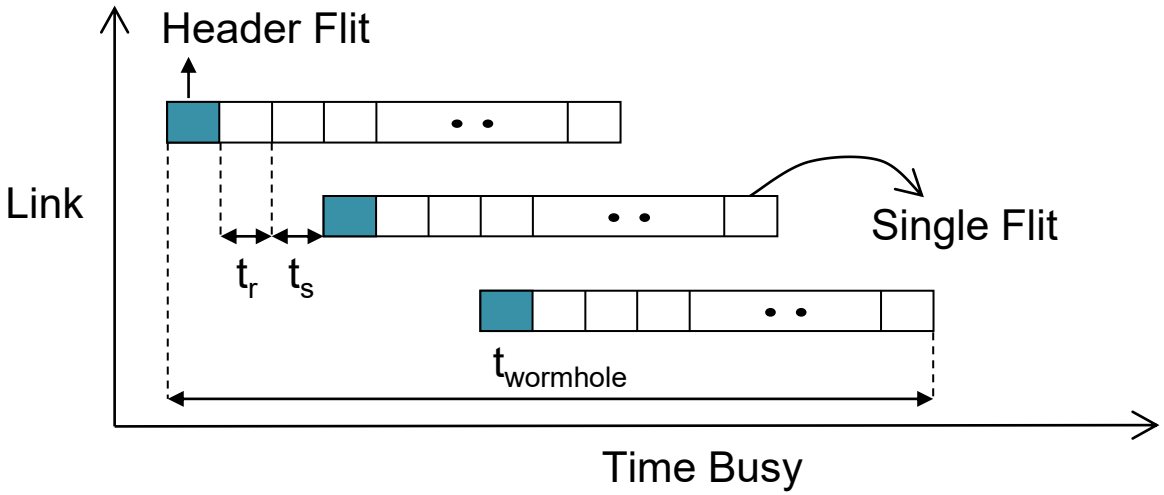
- Large packets are divided into small flits
- An entire packet need not be buffered to move on to the next node, increasing throughput.
- More efficient use of buffers than virtual cut-through
- Bandwidth and Channel allocation are decoupled

Wormhole Switching



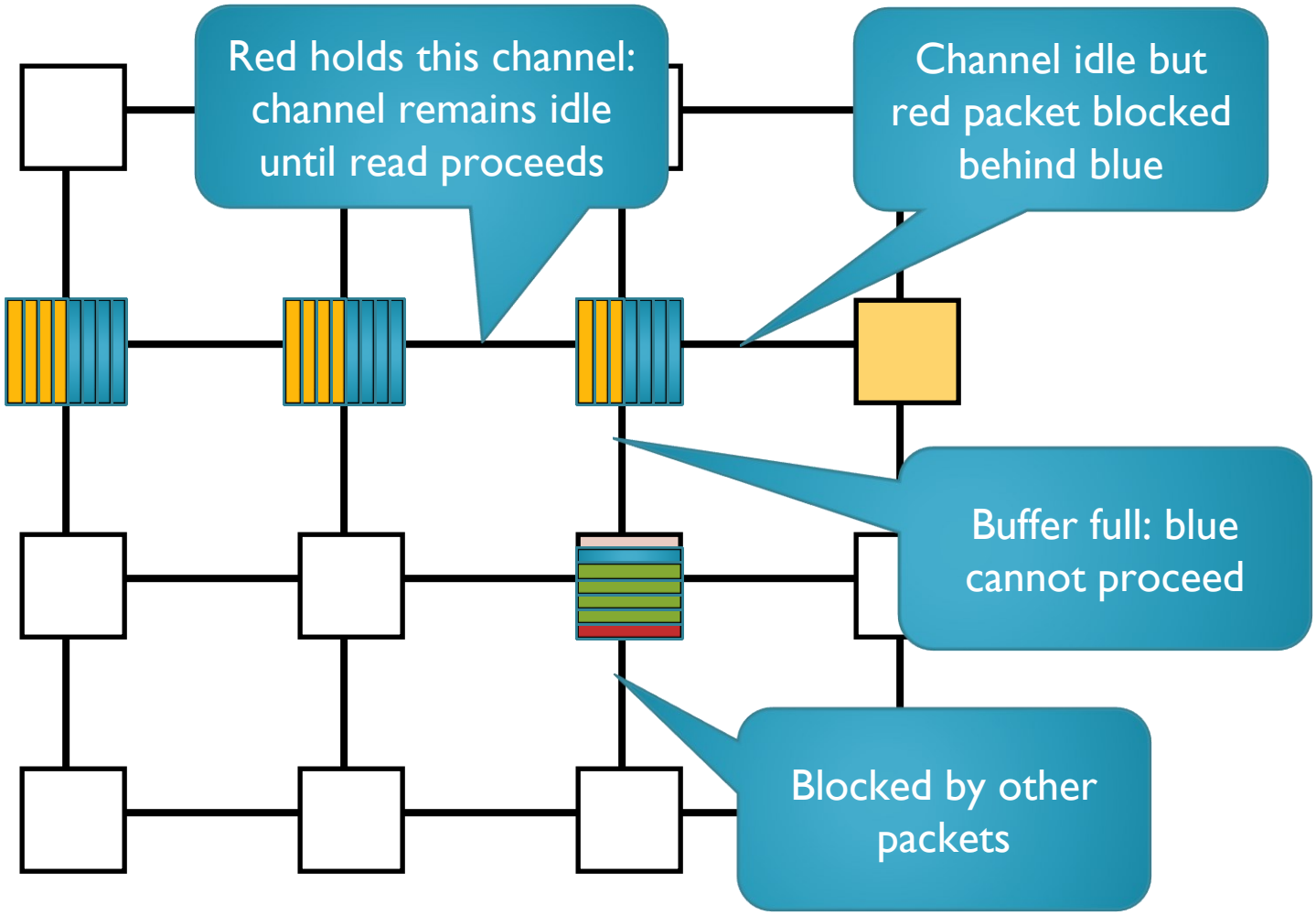
- Large packets are divided into small flits
- An entire packet need not be buffered to move on to the next node, increasing throughput.
- More efficient use of buffers than virtual cut-through
- Bandwidth and Channel allocation are decoupled

Wormhole Switching



- Message are pipelined, but buffer space is on the order of a few flits
- Small buffers + message pipelining → small compact switches/routers
- Messages cannot be interleaved over a channel: routing information is only associated with the header

Head-of-Line (HoL) Blocking in Wormhole

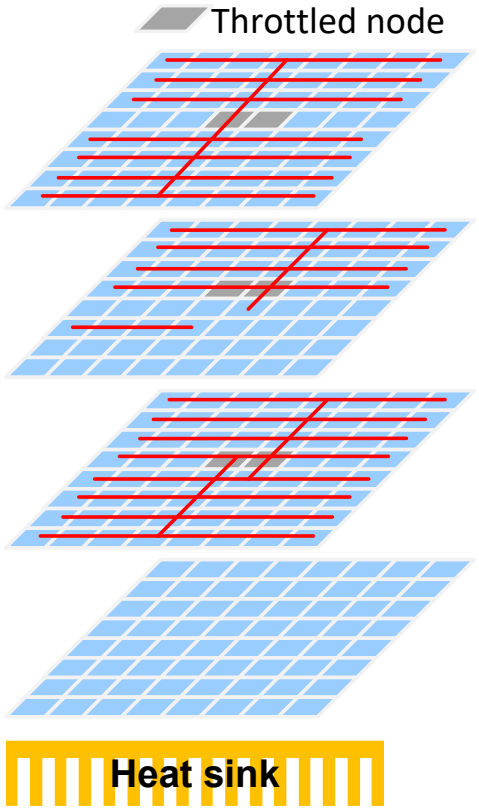


- 6 flit buffers/input port

Problem of Blocked Channels

Problem Formulation

- Blocked channels make congestion tree grow early
  - Network will be blocked more easily
  - Network blocking become severer with more prescient prediction scheme



Preliminary Solution

- Adaptive routing is used to detour the packets
  - Both consider Effective Service Rate and Current Buffer Length
  - Routing direction depends on Waiting Time

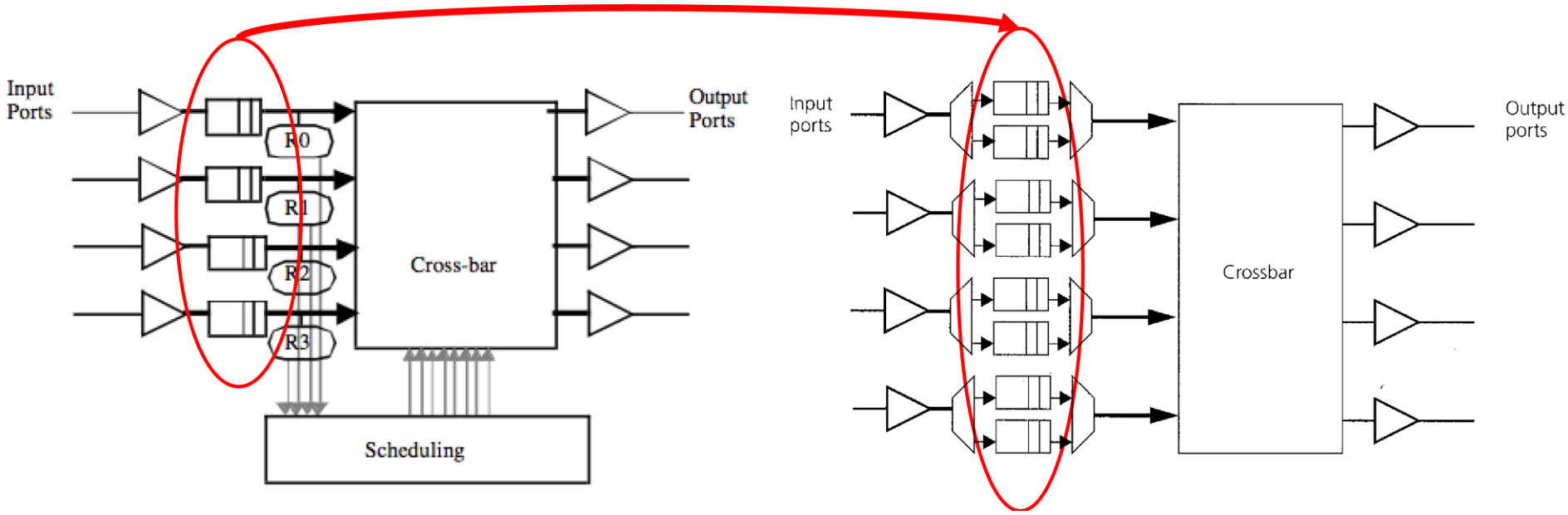
↓

Waiting Time = **Effective Service Time** x **Current Buffer Length**

Effective service time is increased by Contention occurrence

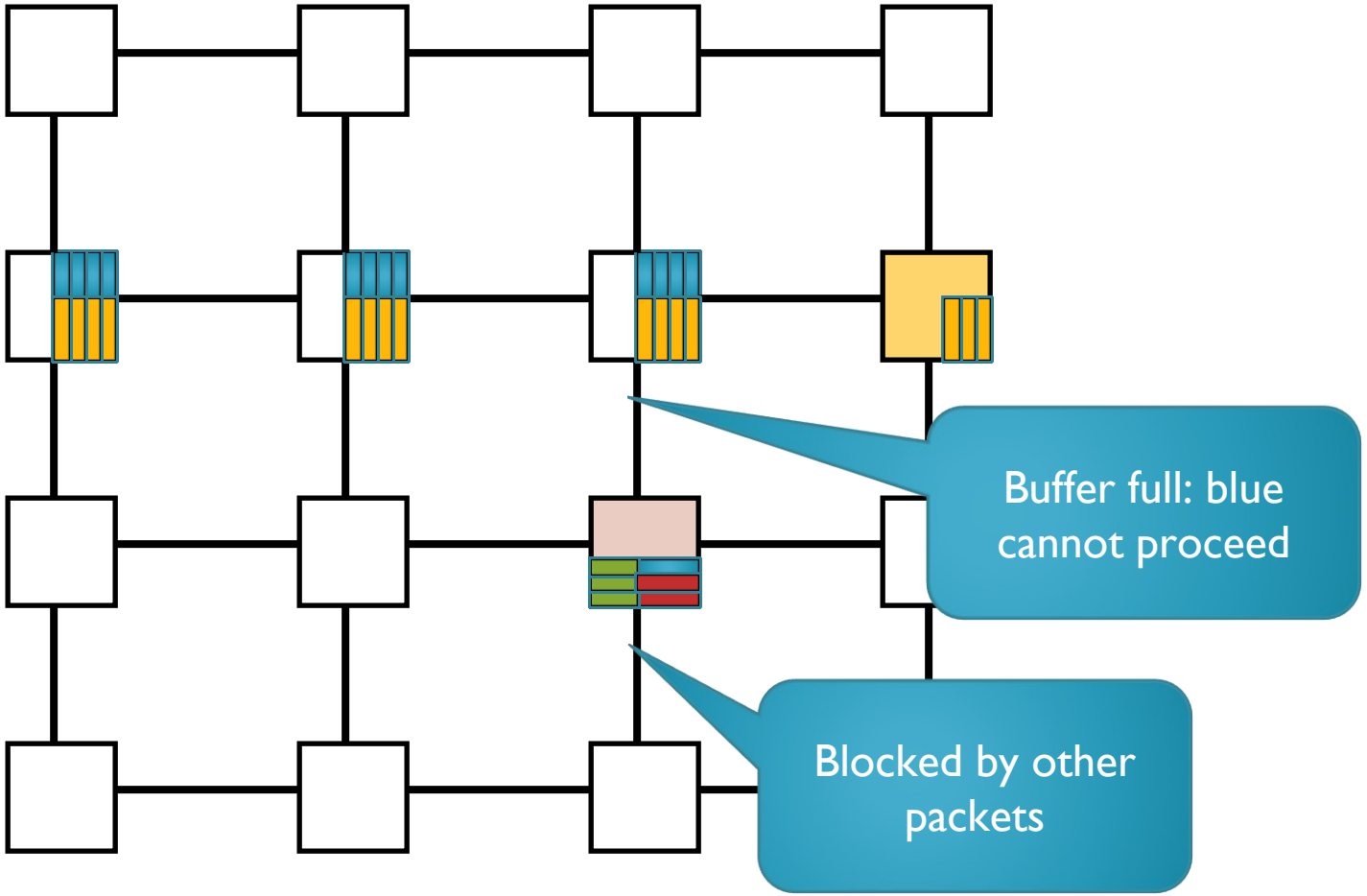
Virtual Channel Flow Control

- Idea: Multiplex multiple channels over one physical channel
- Reduces head-of-line blocking
- Divide up the input buffer into multiple buffers sharing a single physical channel



Dally, "Virtual Channel Flow Control," ISCA 1990

Virtual Channel Example



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

Can blue use another channel?

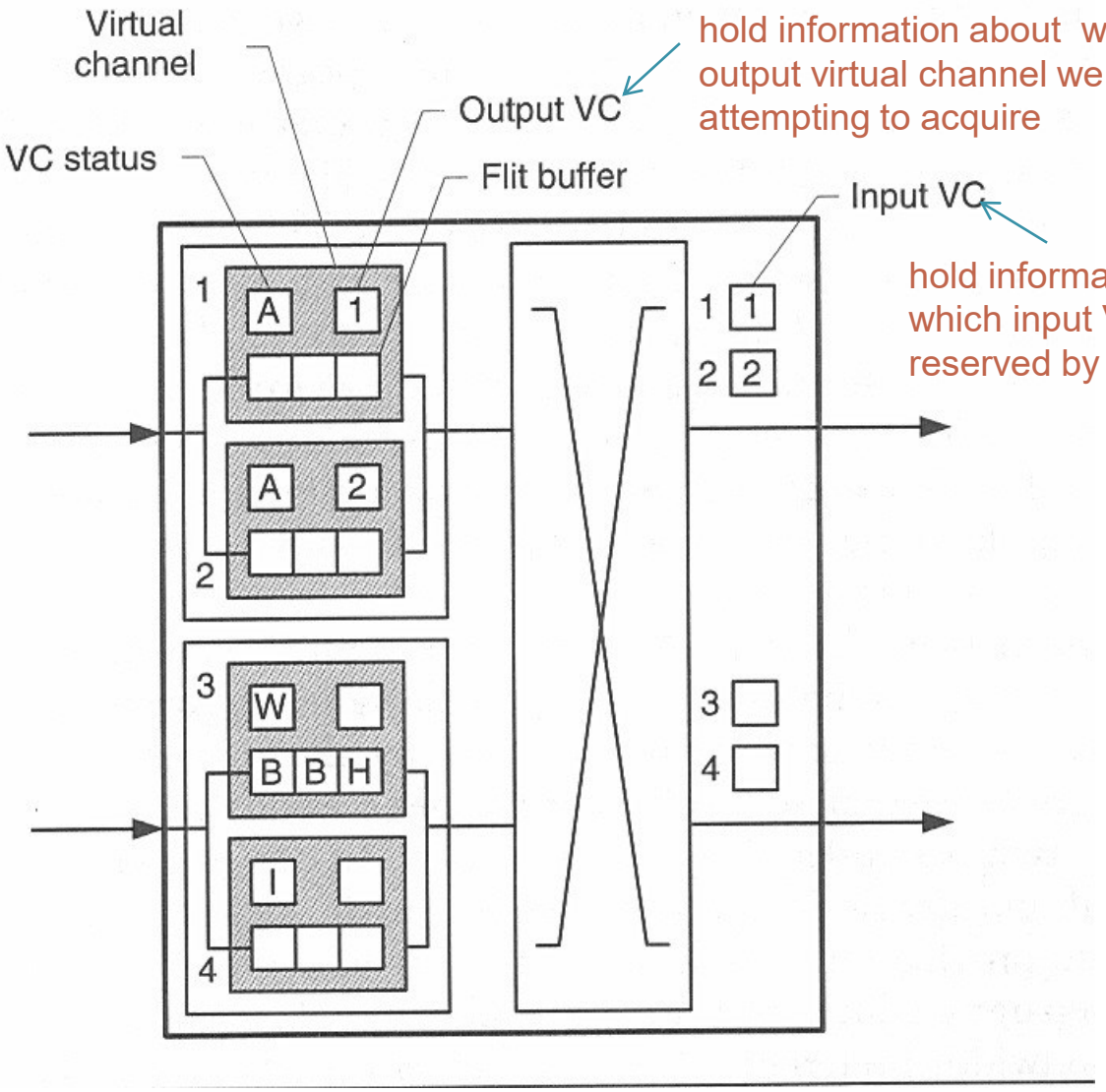
Other Uses of Virtual Channels

- **Deadlock avoidance**
  - Enforcing switching to a different set of virtual channels on some “turns” can break the cyclic dependency of resources
  - Escape VCs: Have at least one VC that uses deadlock-free routing. Ensure each flit has fair access to that VC.
  - Protocol level deadlock: Ensure request and response packets use different VCs → prevent cycles due to intermixing of different packet classes

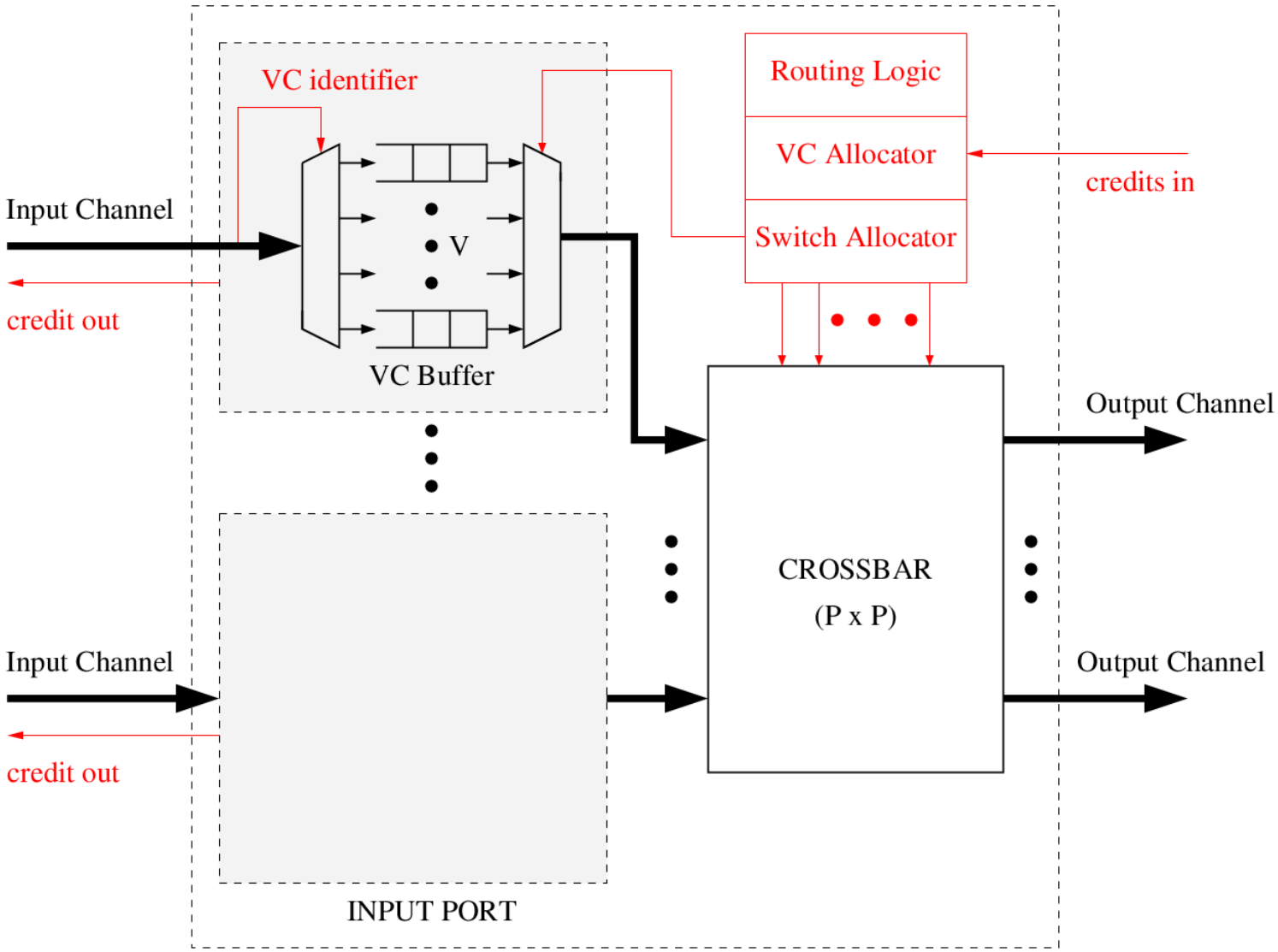
- **Prioritization of traffic classes**
  - Some virtual channels can have higher priority than others

Virtual Channel Example

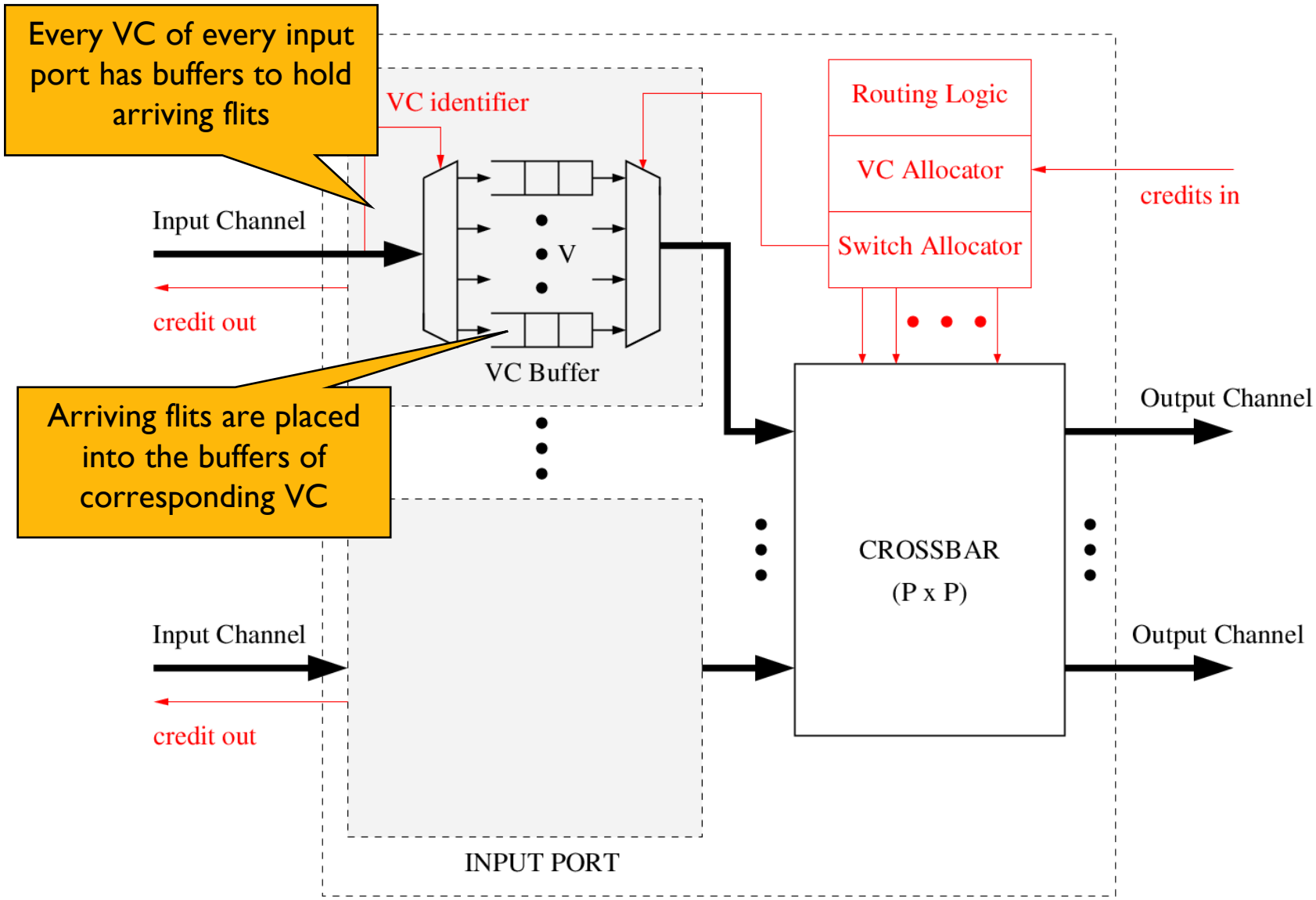
A: active
W: waiting
I: idle



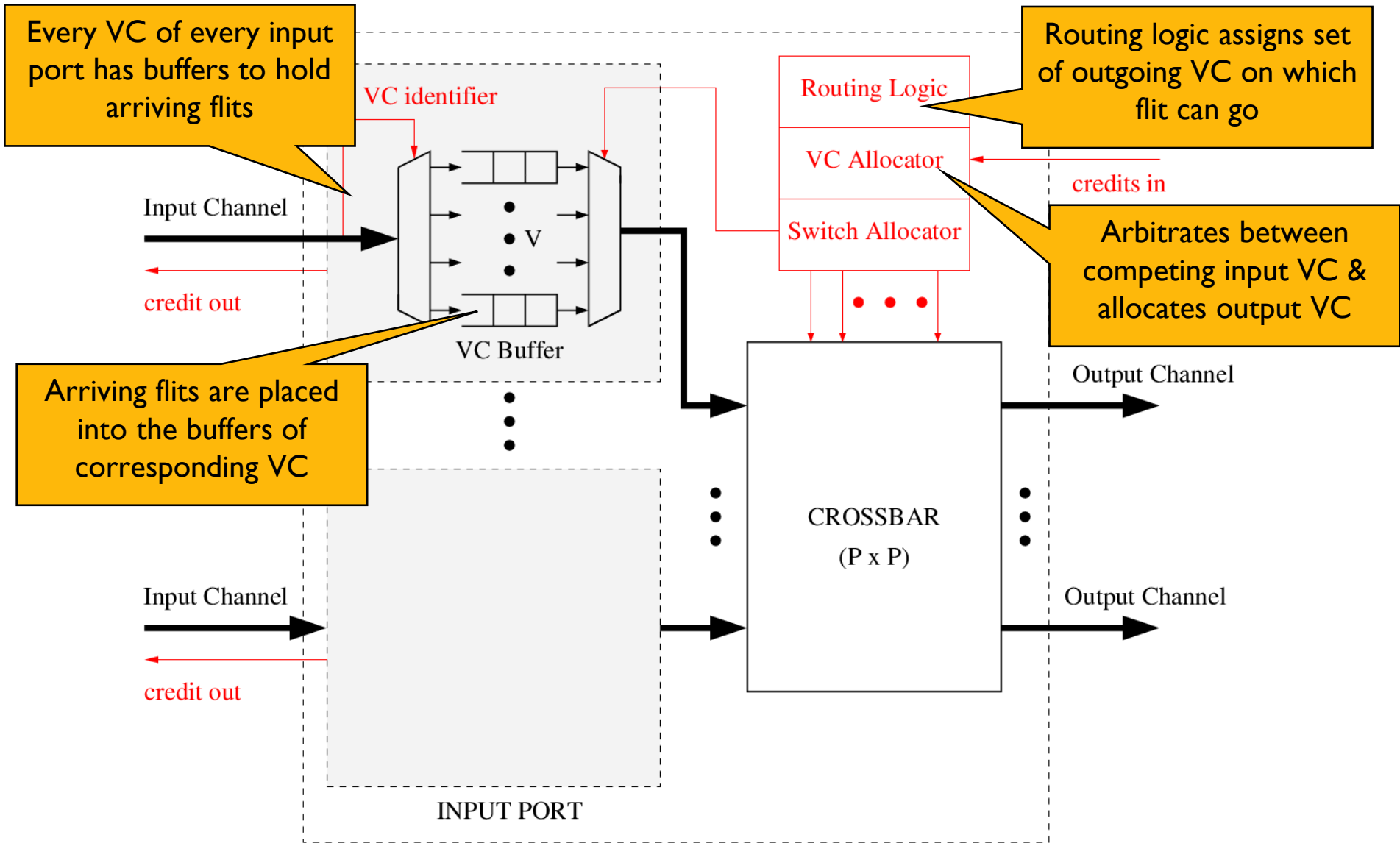
A Virtual Channel Router



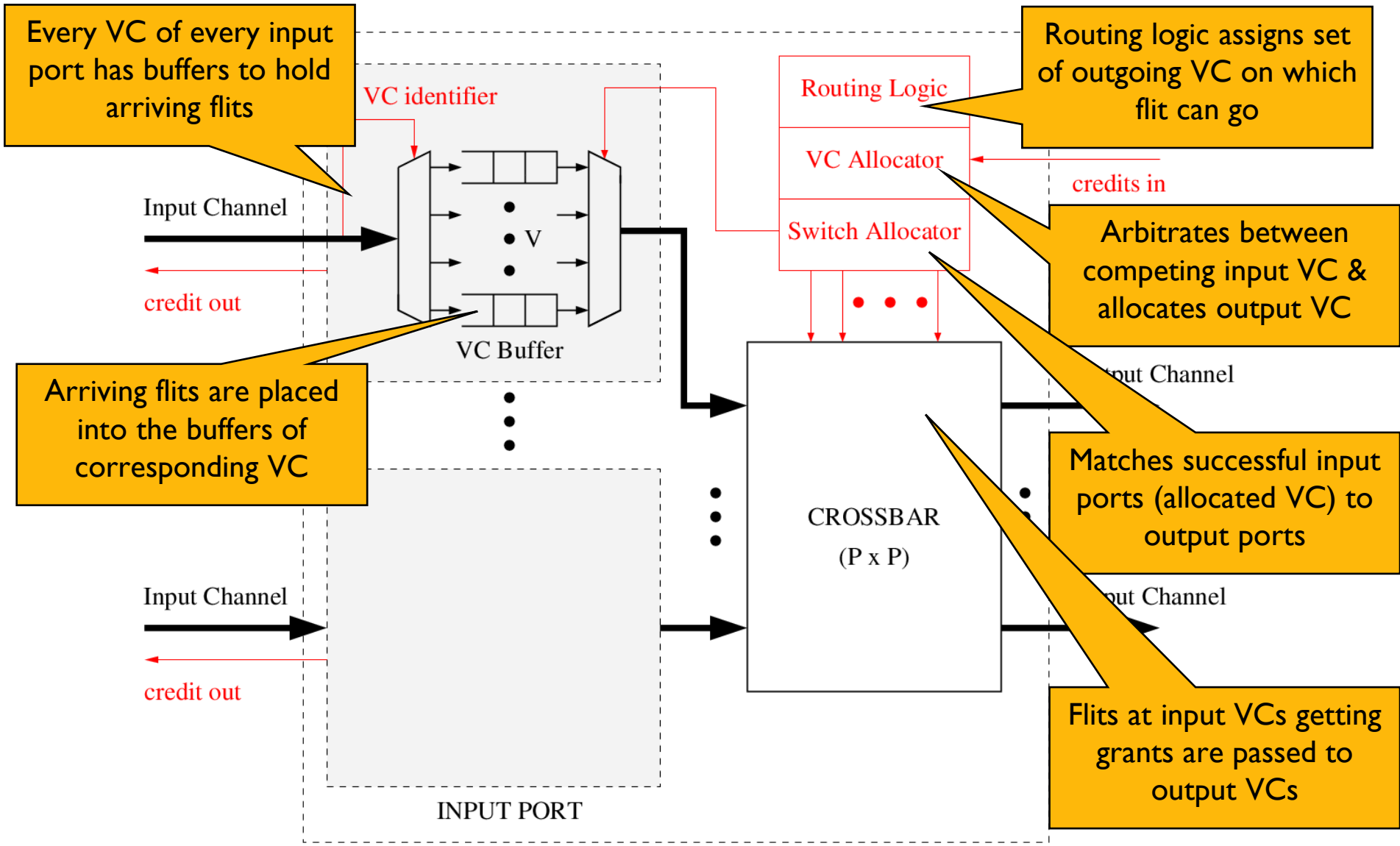
A Virtual Channel Router



A Virtual Channel Router

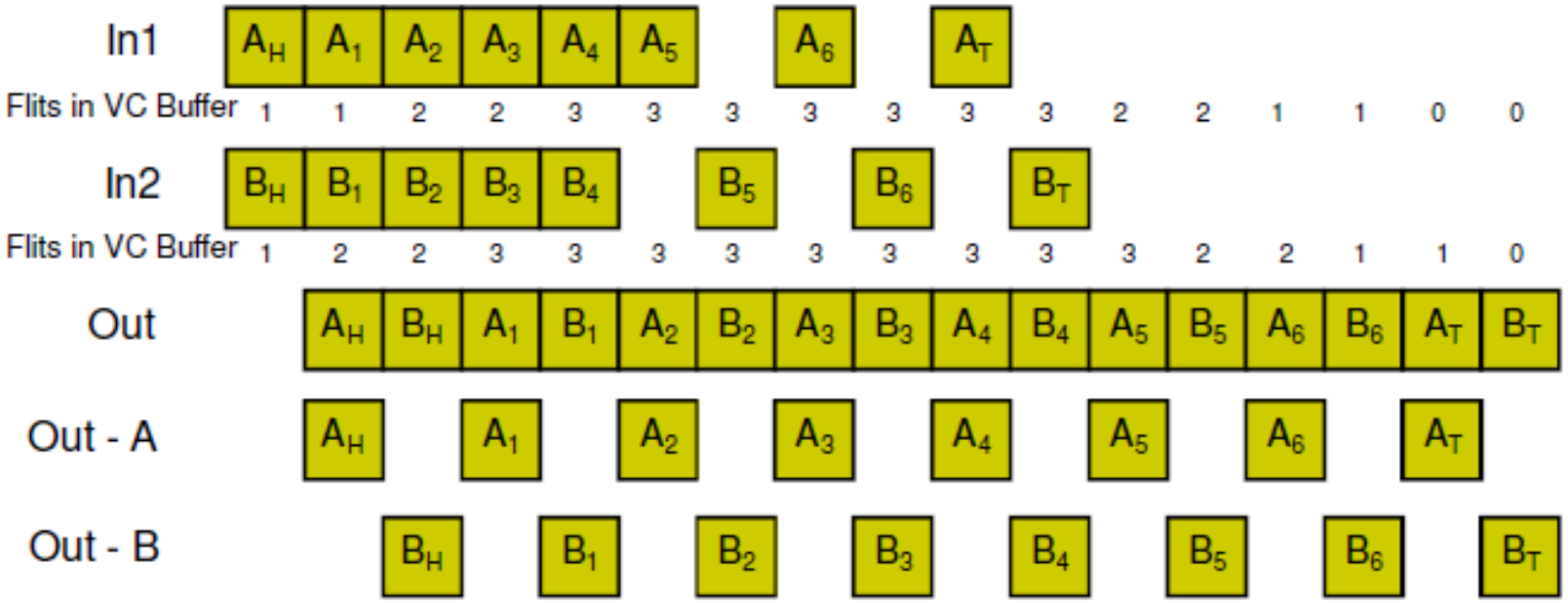


A Virtual Channel Router



VC Arbitration: Fair Bandwidth

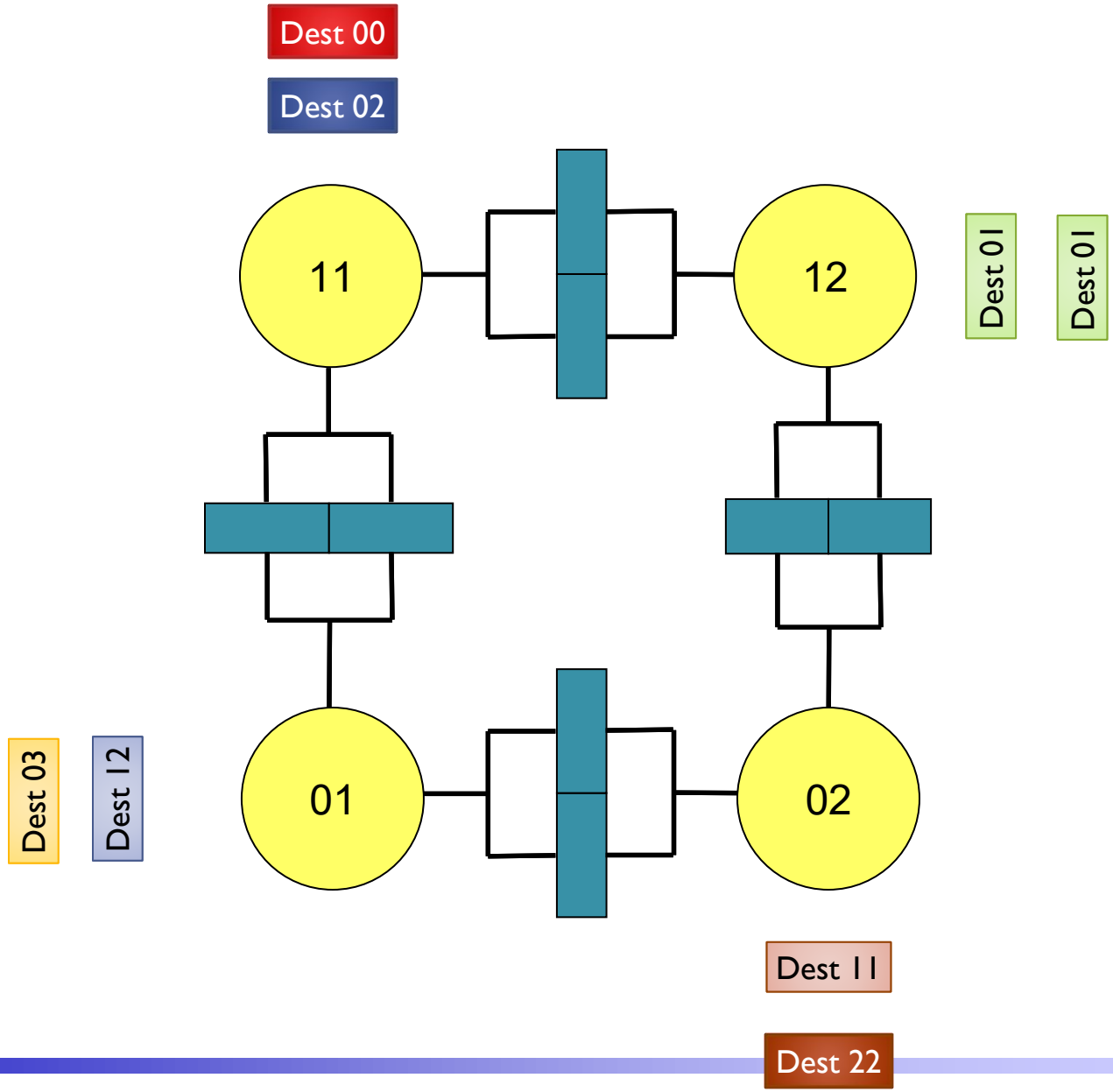
- The virtual channels interleave their flits
- This results in a high average latency



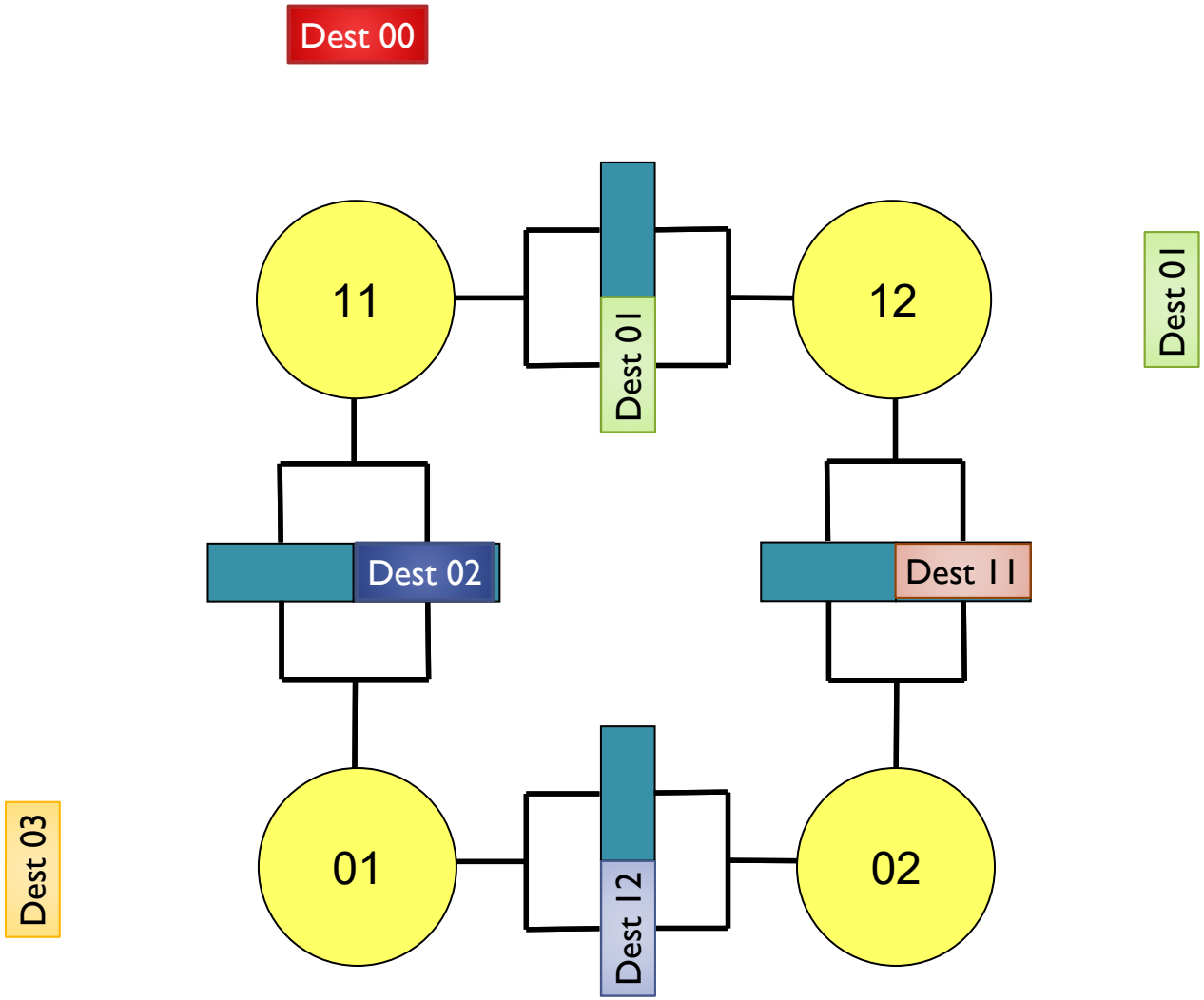
VC Arbitration: Winner-Take-All

- A winner-take all arbitration reduces the average latency with no throughput penalty

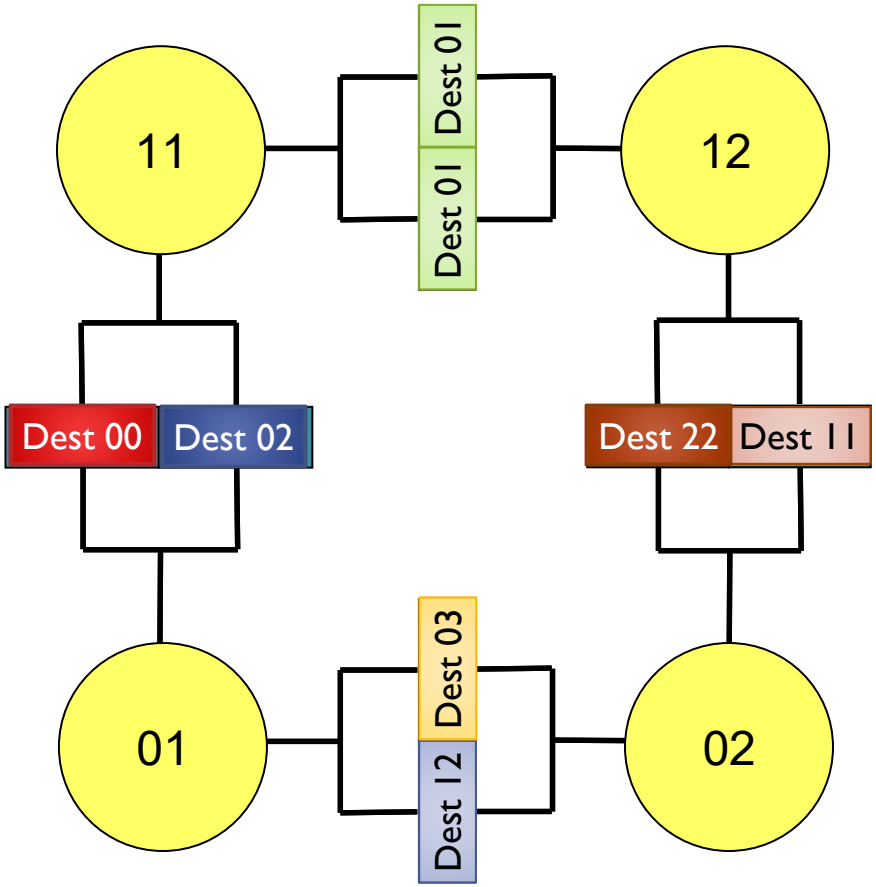
Virtual Channels



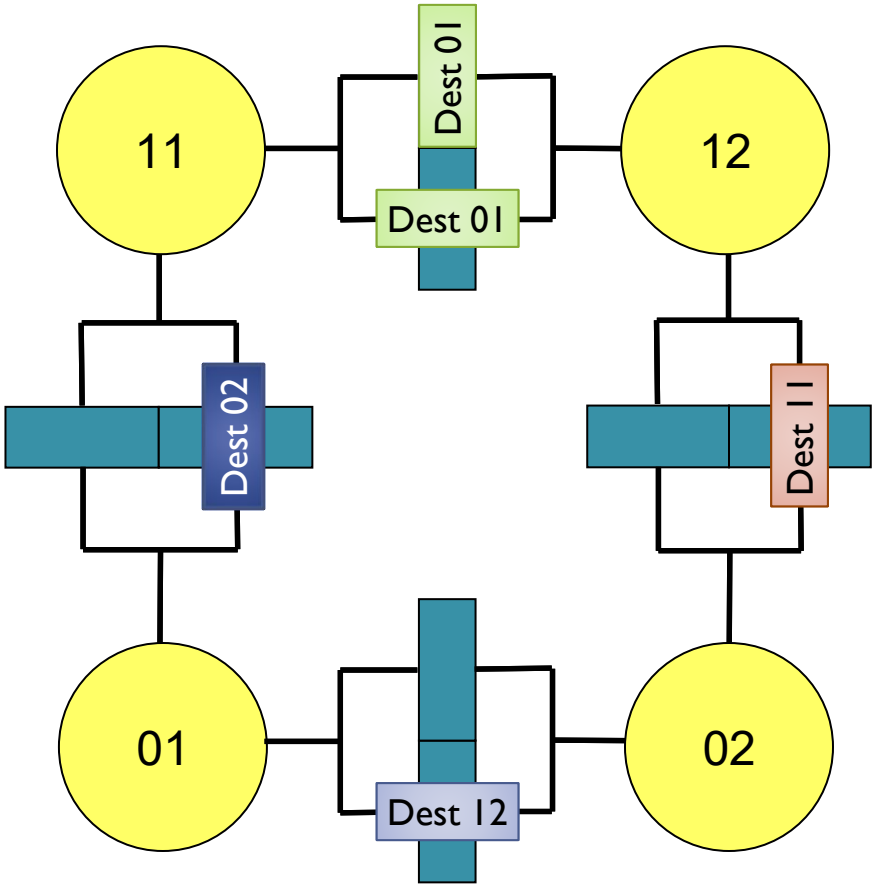
Virtual Channels



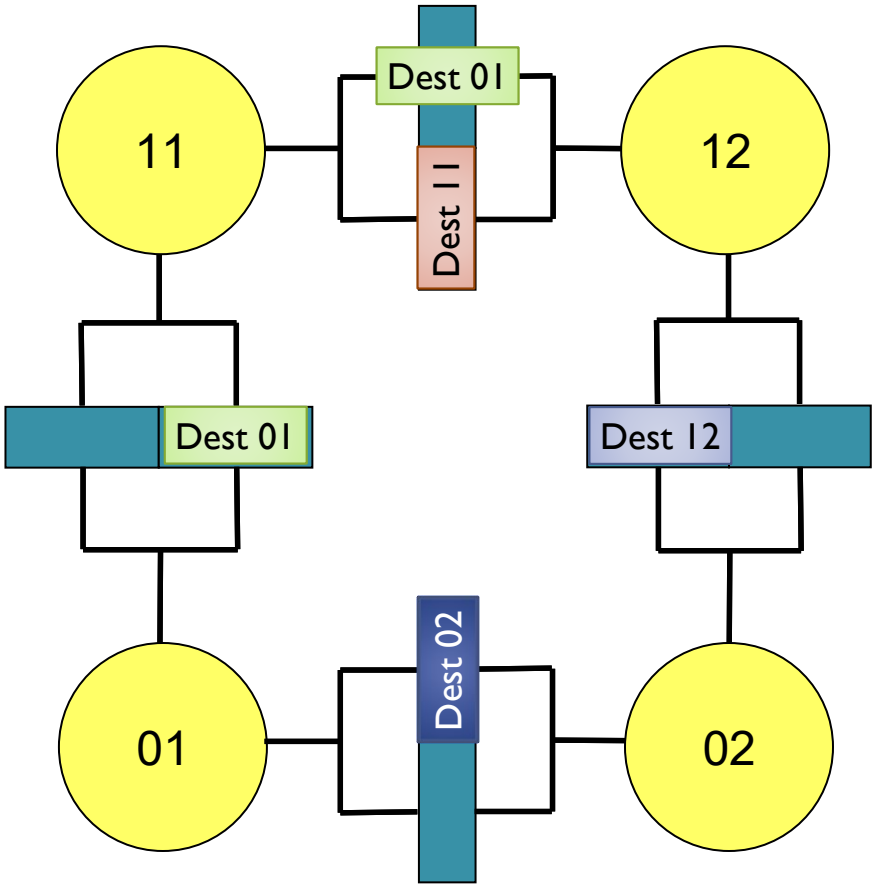
Virtual Channels



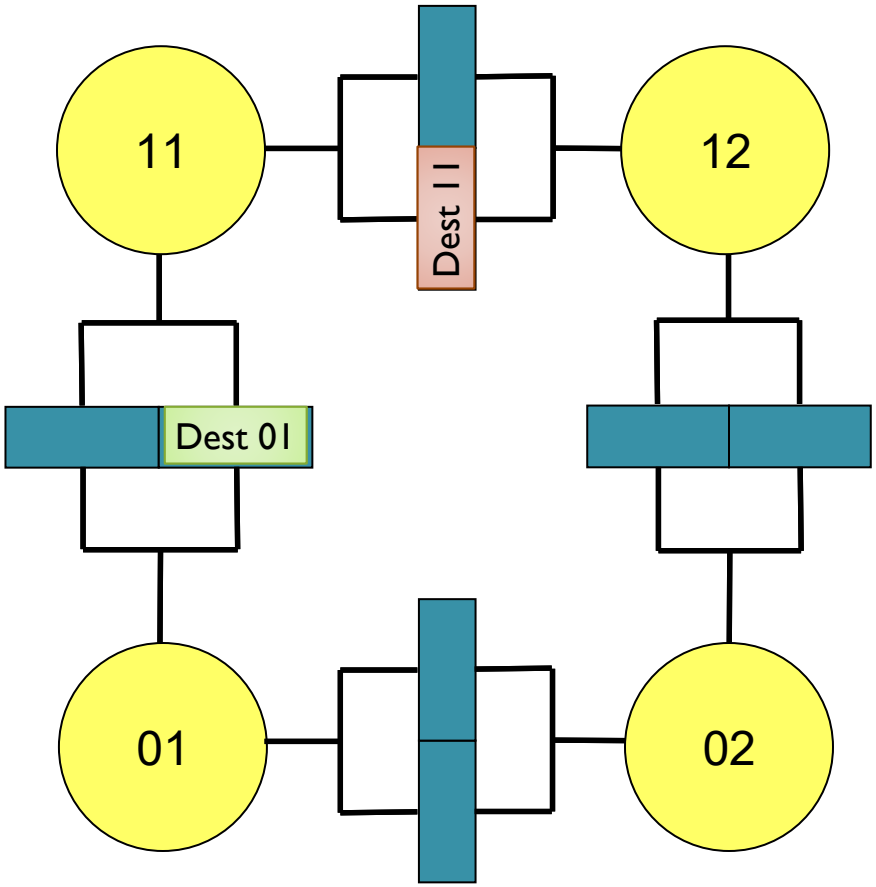
Virtual Channels



Virtual Channels



Virtual Channels



Summary of Switching Techniques

Switching Technique	Communication Entity	Path Reservation	Buffer Size	Resource Utilization
Circuit Switching	Flit/Packet	Yes	Small/Large	Poor
SAF Switching	Packet	No	Large	Good
VCT Switching	Packet	No	Large	Good
Wormhole Switching	Flit	No	Small	Moderate

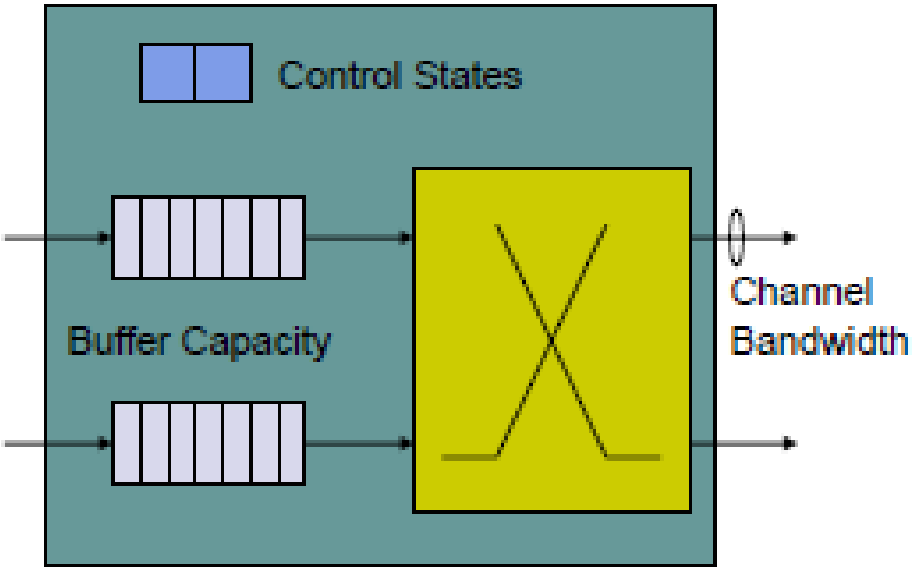
Flow Control

Flow Control (FC)

- FC determines (1) how resources (Buffers and channel bandwidth) are allocated and (2) how packet collisions over resources are resolved.
- Goal is to use resources as efficient as possible to allow a high throughput
- A resource collision occurs when a packet P is unable to proceed because some resource it needs is held by another packet.

Node Resources

- **Control State**
  - Tracks the resources allocated to the packet in the node and the state of the packet
- **Buffer**
  - Packet is stored in a buffer before it is sent to next node
- **Bandwidth**
  - To travel to the next node, bandwidth has to be allocated for the packet



Flow Control

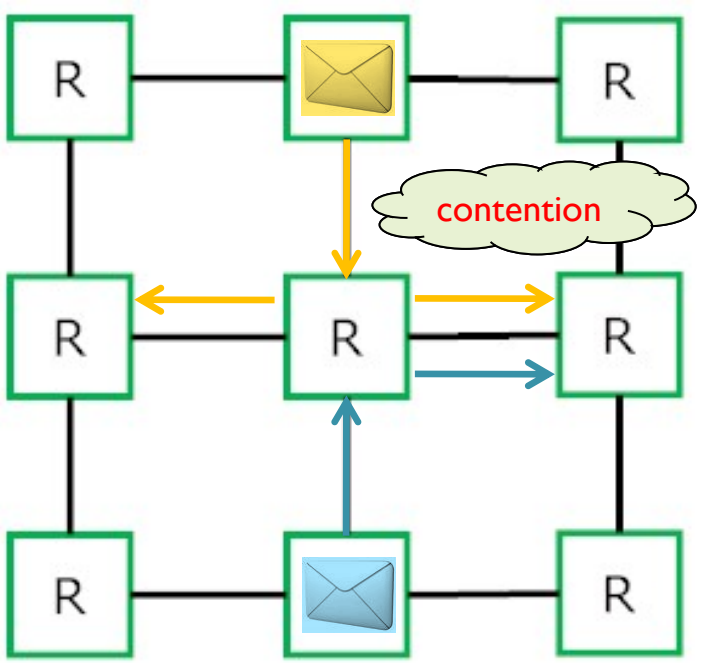
NoC Flow Control can be divided into:

1. Bufferless flow control
   - Packets are either *dropped* or *misrouted*

2. Buffered flow control
   - Packets that cannot be routed via the desired channel are stored in buffers. A certain protocol should be involved
     - Stop-Go (or On-off or Stall-go),
     - ACK/NACK,
     - Credit-Based

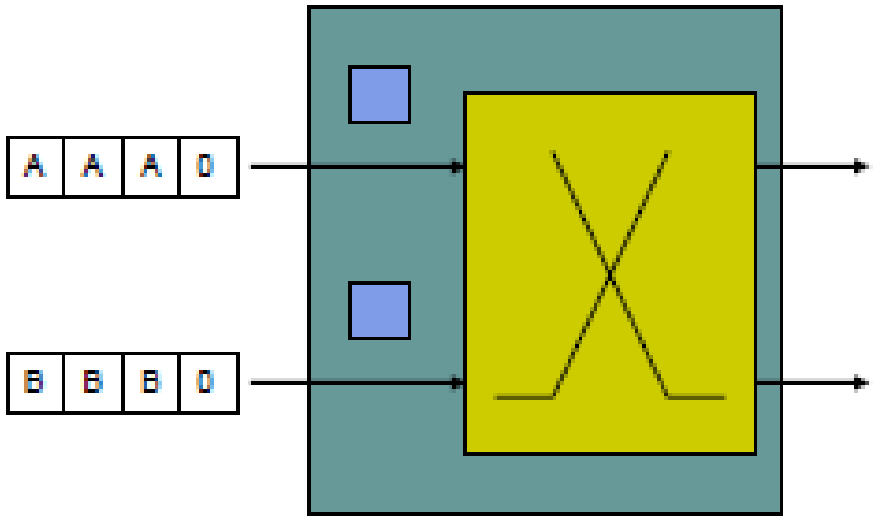
Bufferless flow Control

- Flits can't wait in routers.
- Contention is handled by:
  - Dropping and retransmitting from the source.
  - Deflecting to a free output.



Bufferless Flow Control

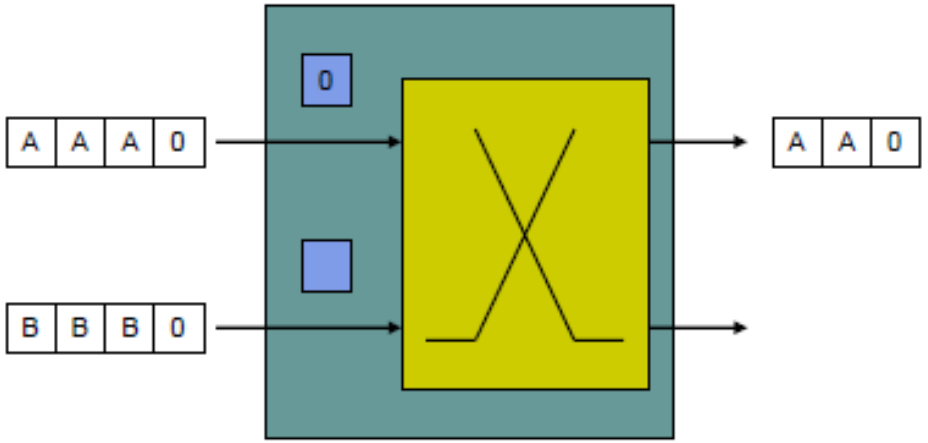
- No buffers mean **less implementation cost**
- If more than one packet shall be routed to the same output, one has to be
  - Misrouted or
  - Dropped



Example: 2 packets A and B (consisting of several flits) arrive at a network node

Bufferless Flow Control

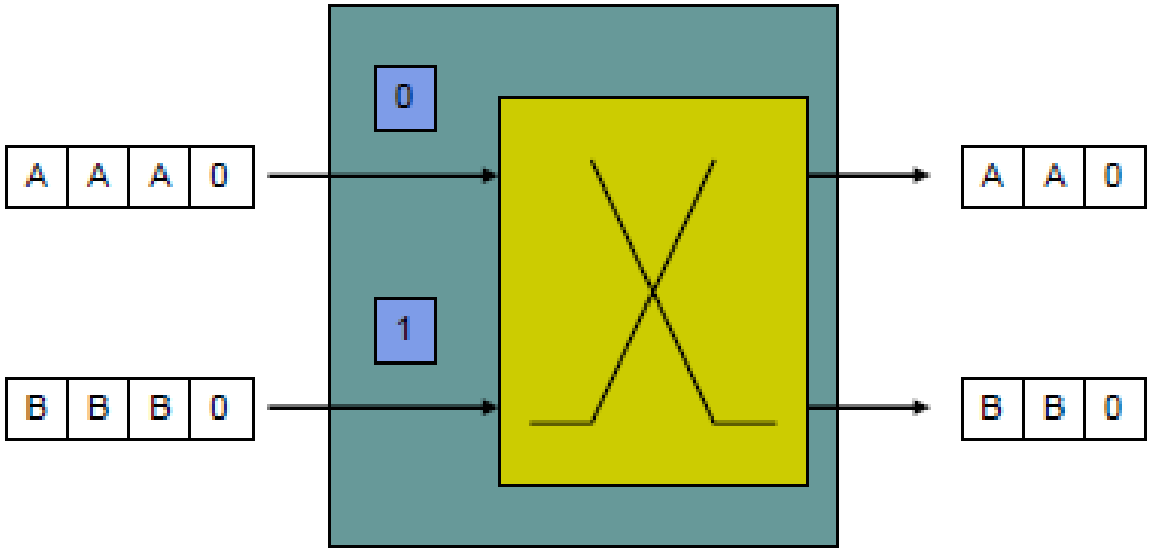
- Packet B is **dropped** and must be resended



- But, there must be a **protocol** that informs the sending node that the packet has been dropped
  - E.g., Resend after no ACK has been received within a given time

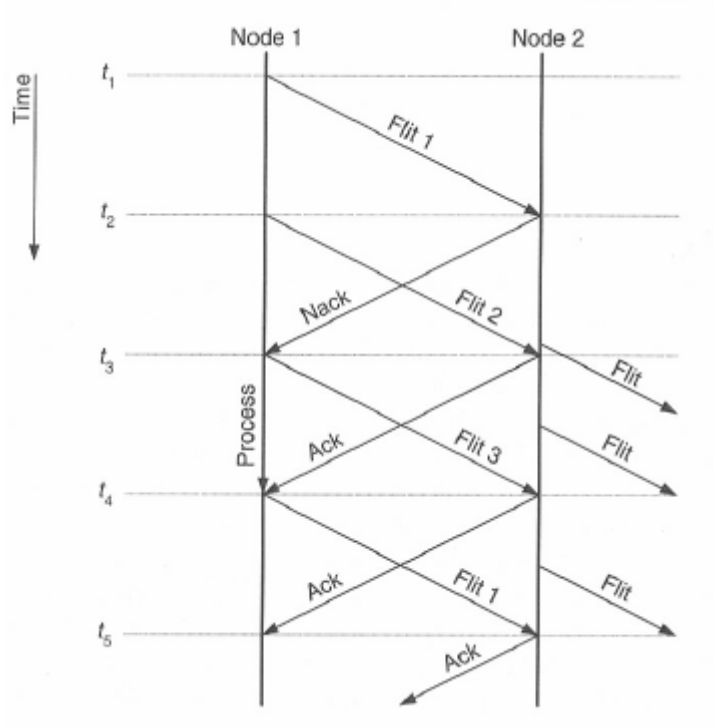
Bufferless Flow Control

- Packet B is **misrouted**



- No further action is required here, but at the receiving node packets have to be **sorted** into original order

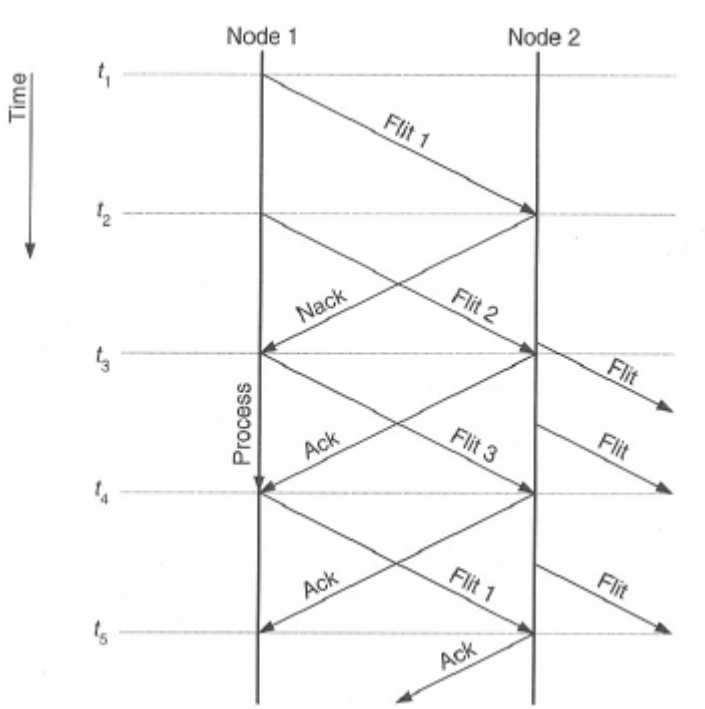
ACK/NACK Protocol (1/3)



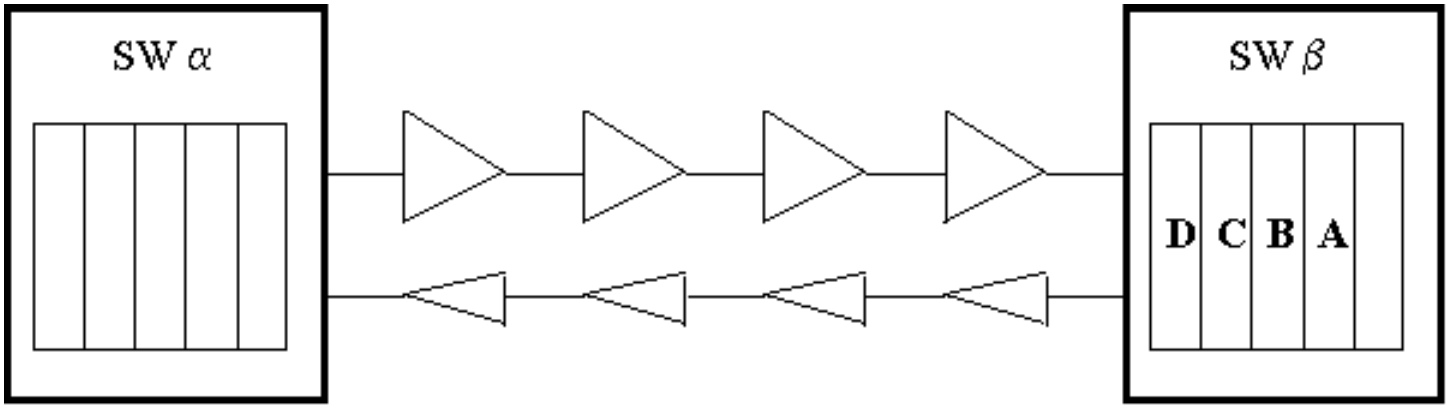
- Upstream node sends packets without knowing, if there are free buffers in the downstream node.

ACK/NACK Protocol (2/3)

- If there is no buffer available:
  - the downstream node sends ***Nack*** and drops the flit
  - the flit must be resent
  - flits must be reordered at the downstream node
- If there is a buffer available:
  - the downstream node sends ***Ack*** and stores the flit in a buffer



ACK/NACK Protocol (3/3)



Transmission

ACK and buffering

NACK

ACK/NACK propagation

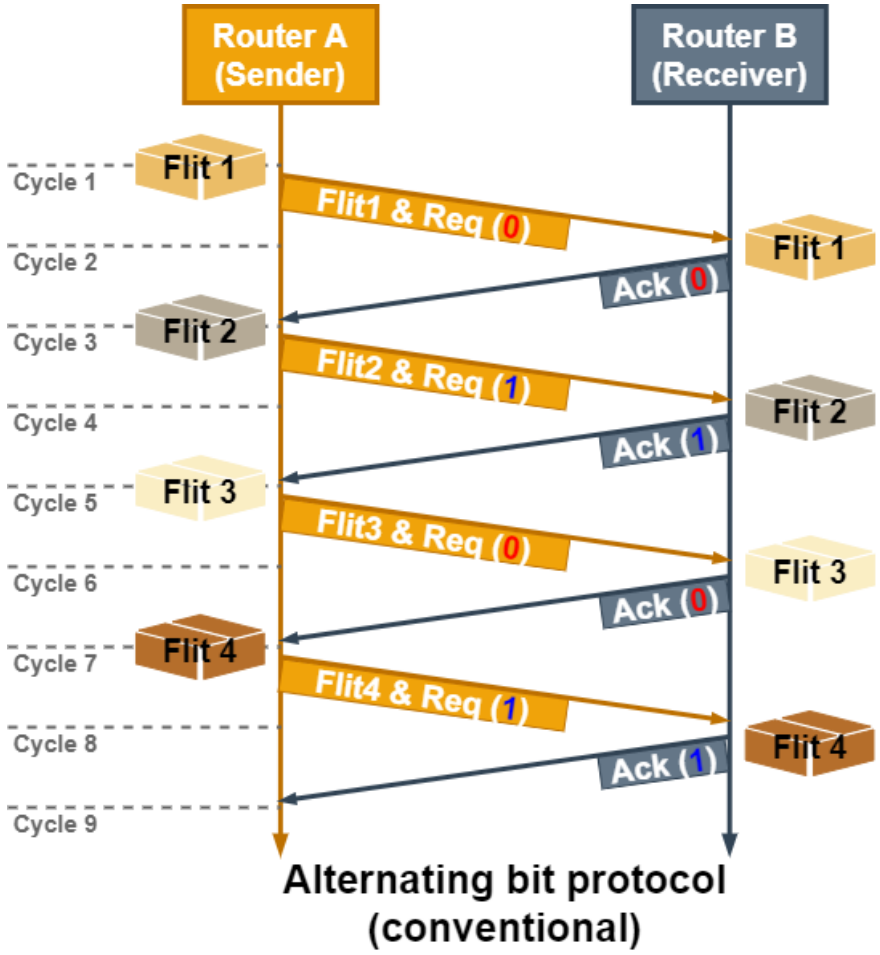
Memory deallocation

Retransmission

Go-back-N

ACK/NACK in NoC

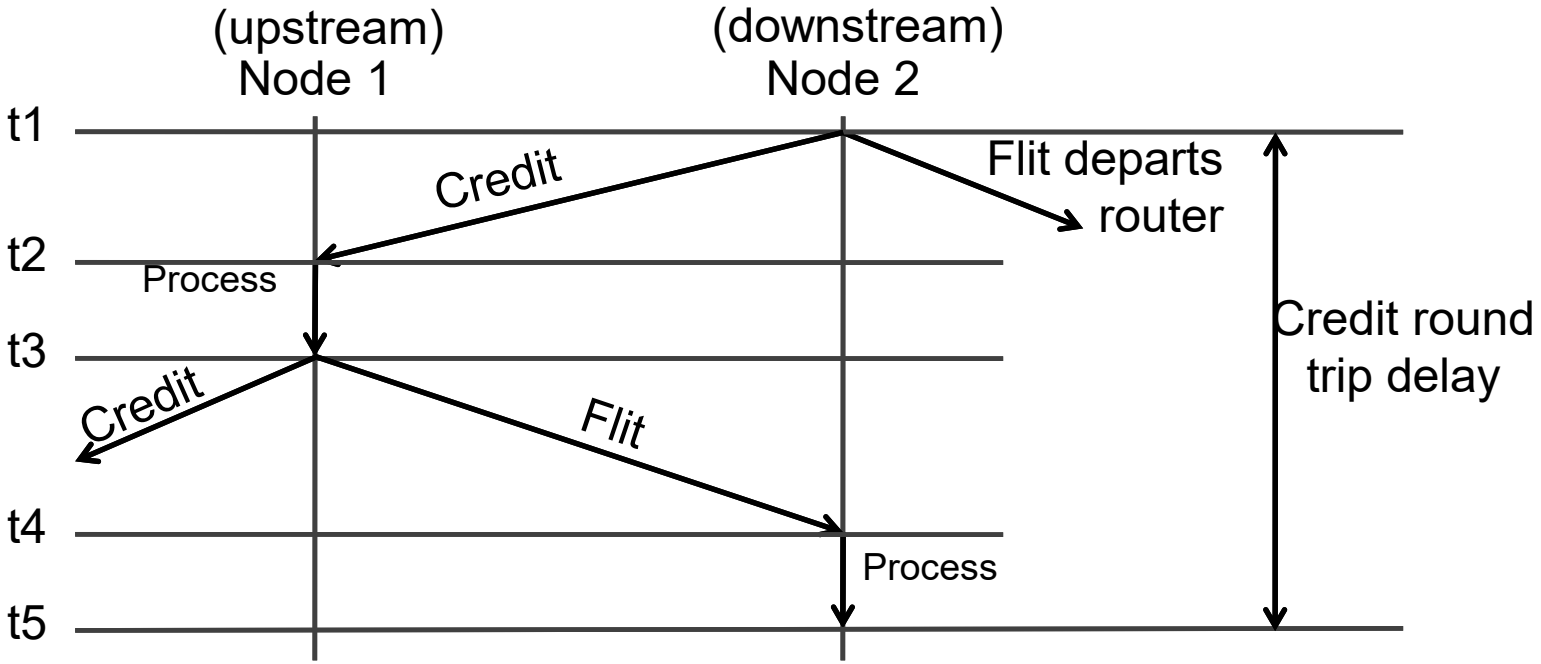
- A.K.A. Alternating Bit Protocol (ABP)
  - In ABP, the sender needs to wait for the Ack signal after sending a data and Req signal.
  - Valid data will transfer when both sender and receiver complete a bit alternate.
  - Since each flit transfer needs to wait for the receiver's Ack signal, it will cause high transfer latency.



Credit-Based Protocol

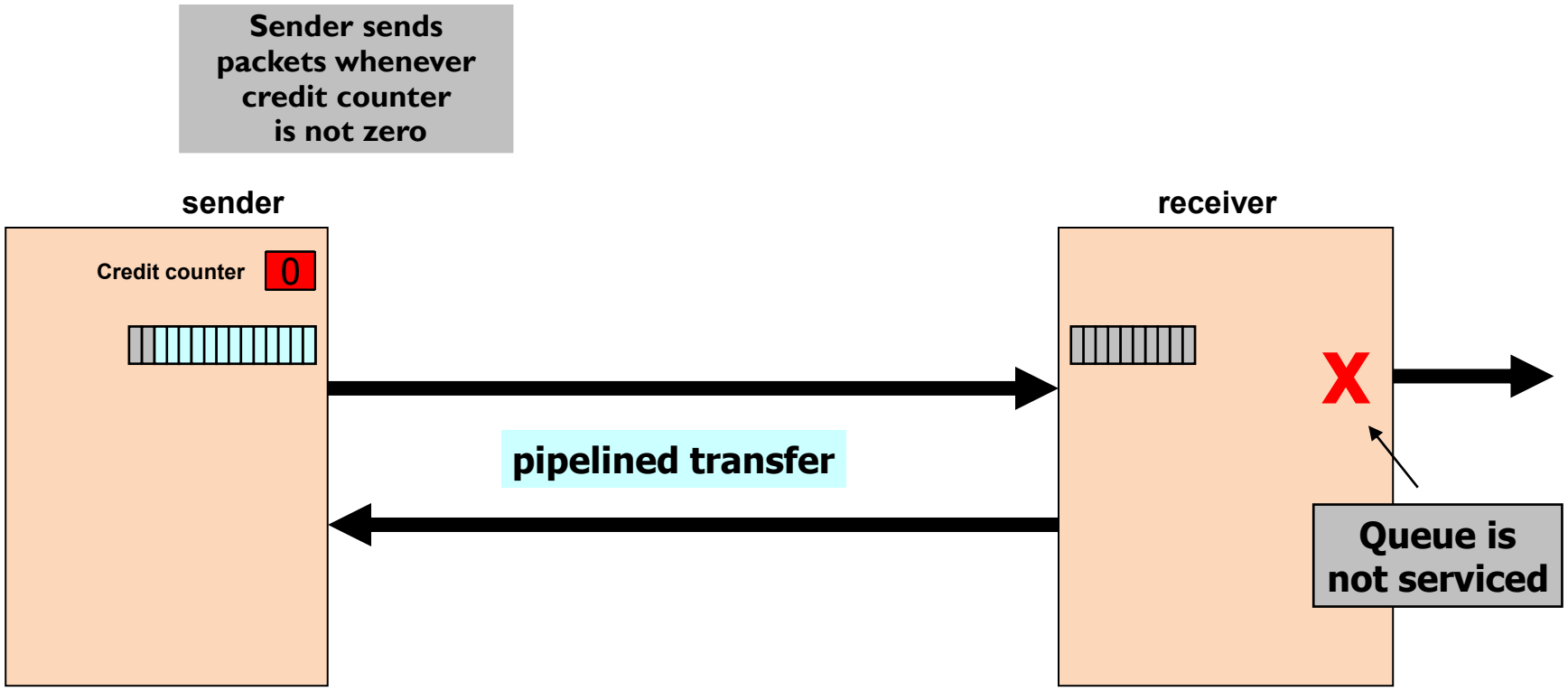
- Upstream router stores credit counts for each downstream VC
- Upstream router
  - When flit forwarded
    - Decrement credit count
  - Count == 0, buffer full, stop sending
- Downstream router
  - When flit forwarded and buffer freed
    - Send credit to upstream router
    - Upstream increments credit count

Timeline of Credit-based Protocol

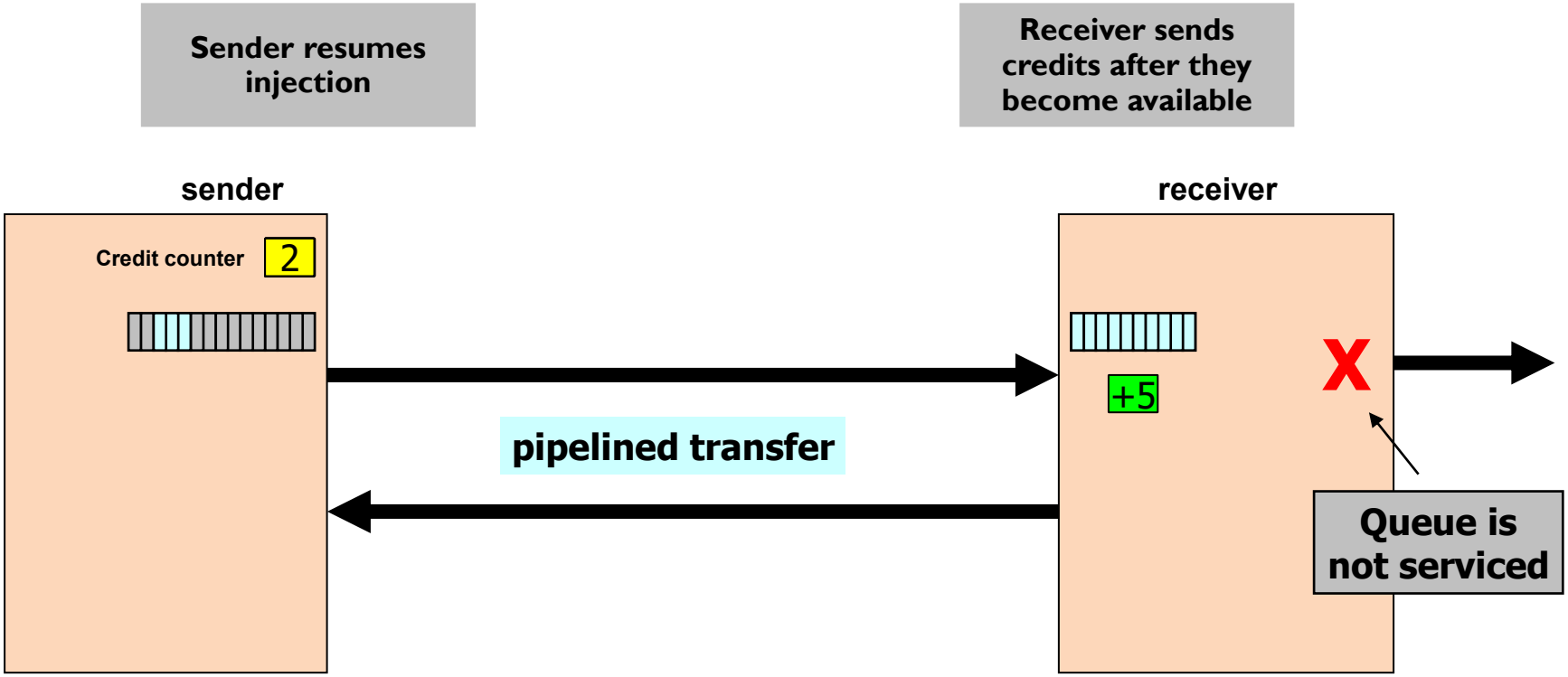


- Round-trip credit delay:
  - Time between when buffer empties and when next flit can be processed from that buffer entry
  - If only single entry buffer, would result in significant throughput degradation
  - Important to size buffers to tolerate credit turn-around

Credit-Based Protocol in action



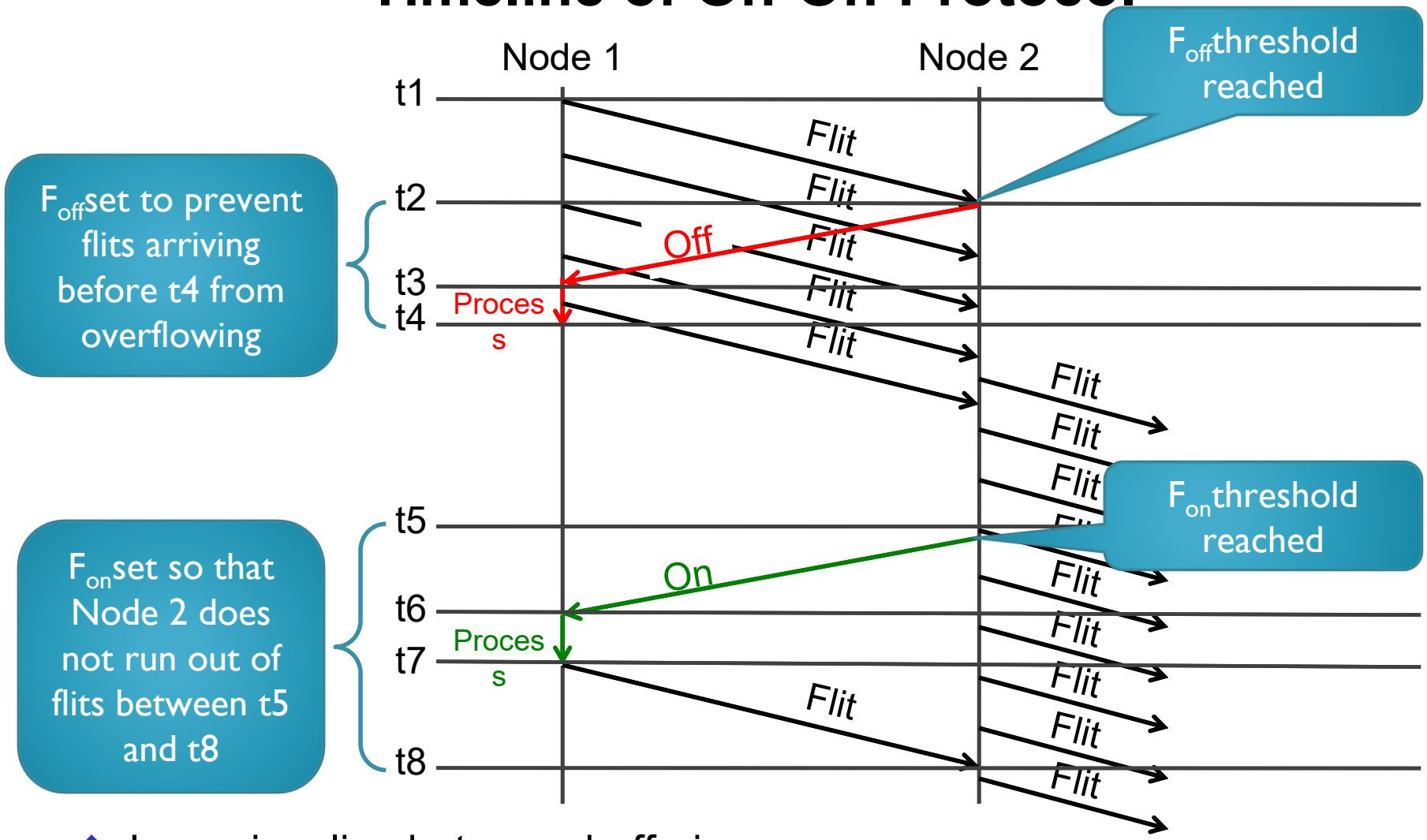
Credit-Based Protocol in action



On-Off (stall-go or stop-go) Protocol

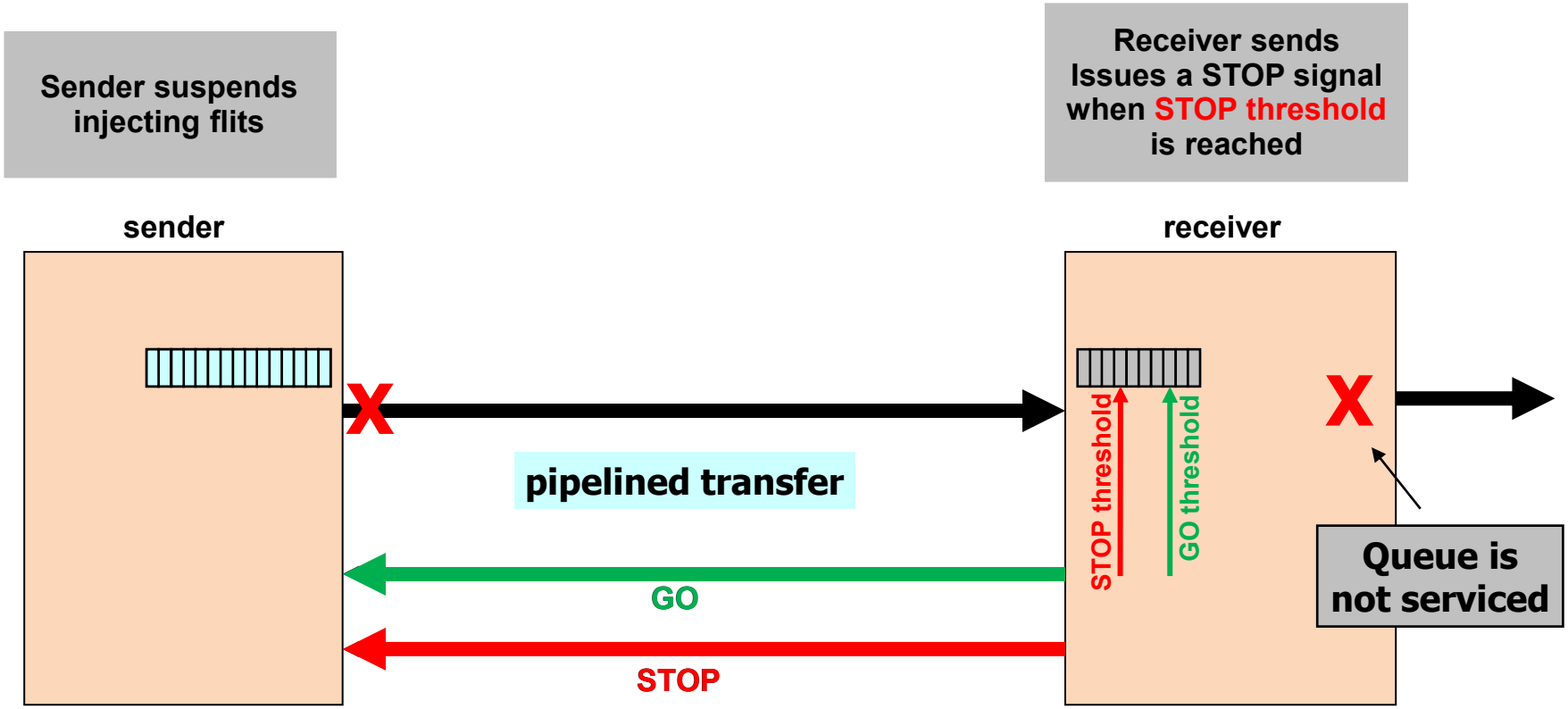
- Credit: requires upstream signaling for every flit
- On-off: decreases upstream signaling
- Off signal
  - Sent when number of free buffers falls below threshold F_{off}
- On signal
  - Send when number of free buffers rises above threshold F_{on}

Timeline of On-Off Protocol

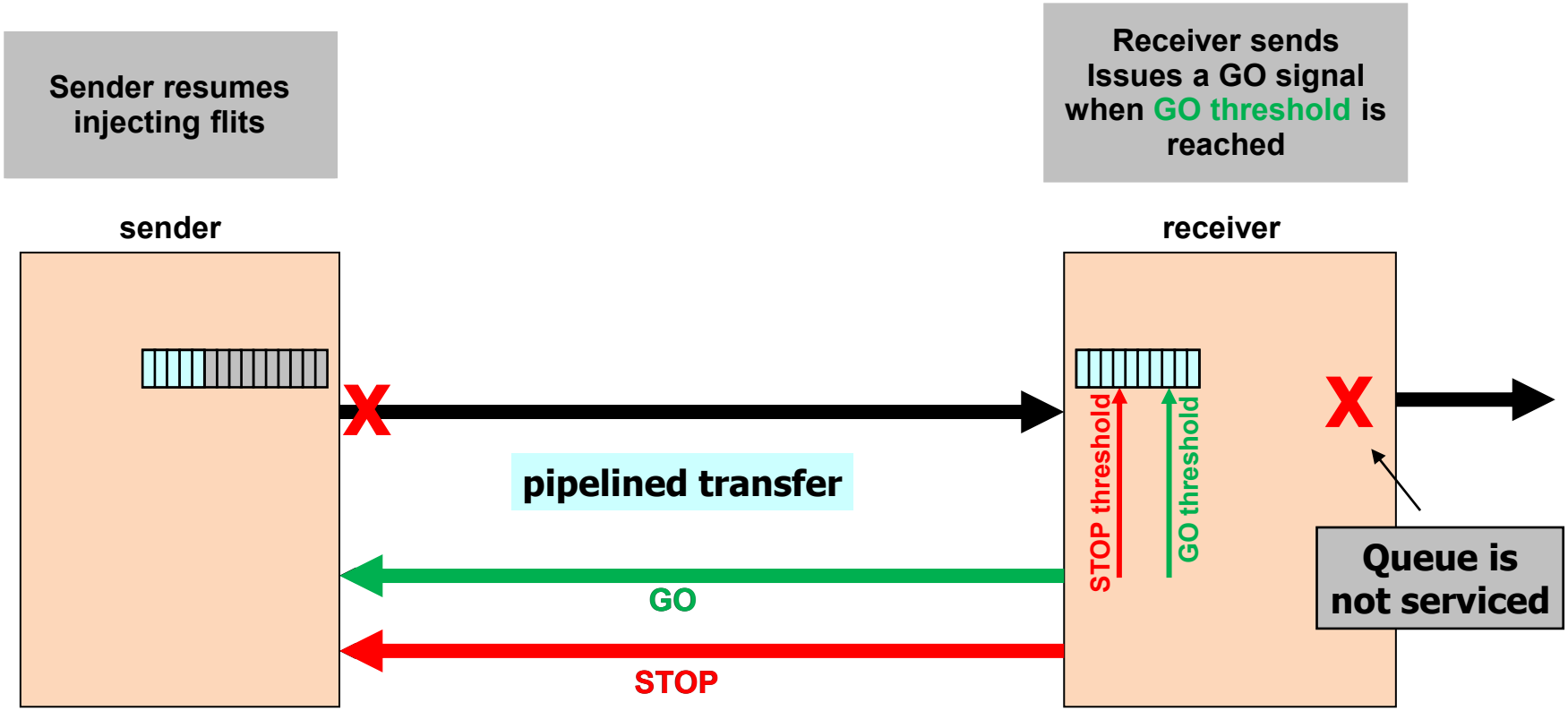


- Less signaling but more buffering
  - On-chip buffers are more expensive than wires

On-Off (stall-go or stop-go) Protocol

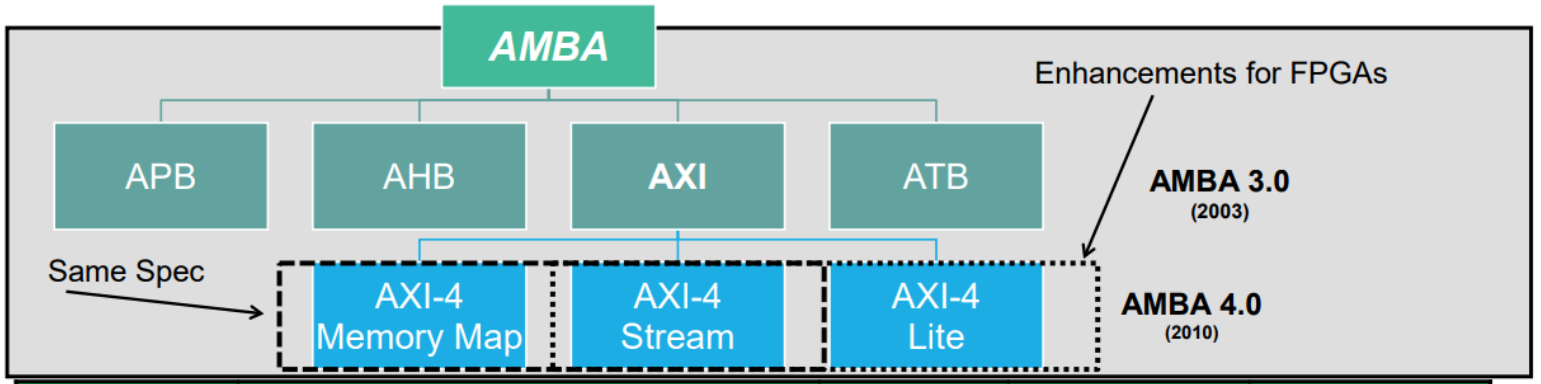


On-Off Protocol



Other Modern Protocols

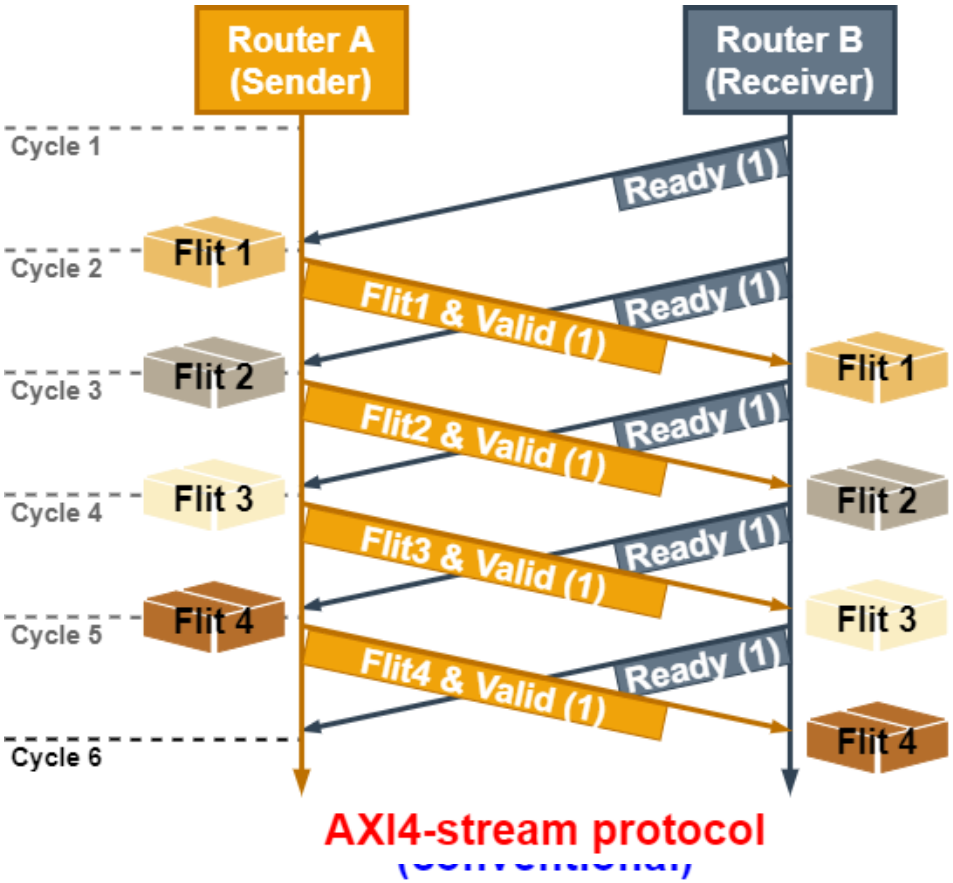
- **Advanced Microcontroller Bus Architecture (AMBA):**
  - An open-standard, on-chip interconnect specification for the connection and management of functional blocks in system-on-a-chip (SoC) designs.
  - First version introduced by ARM in 1996.
- **AMBA Advanced eXtensible Interface 4 (AXI4):**
  - Introduced by ARM in 2010.



Interface	Features	Burst	Data Width	Applications
AXI4	Traditional Address/Data Burst (single address, multiple data)	Up to 256	32 to 1024 bits	Embedded, Memory
AXI4-Stream	Data-Only, Burst	Unlimited	Any Number	DSP, Video, Communications
AXI4-Lite	Traditional Address/Data—No Burst (single address, single data)	1	32 or 64 bits	Small Control Logic, FSM

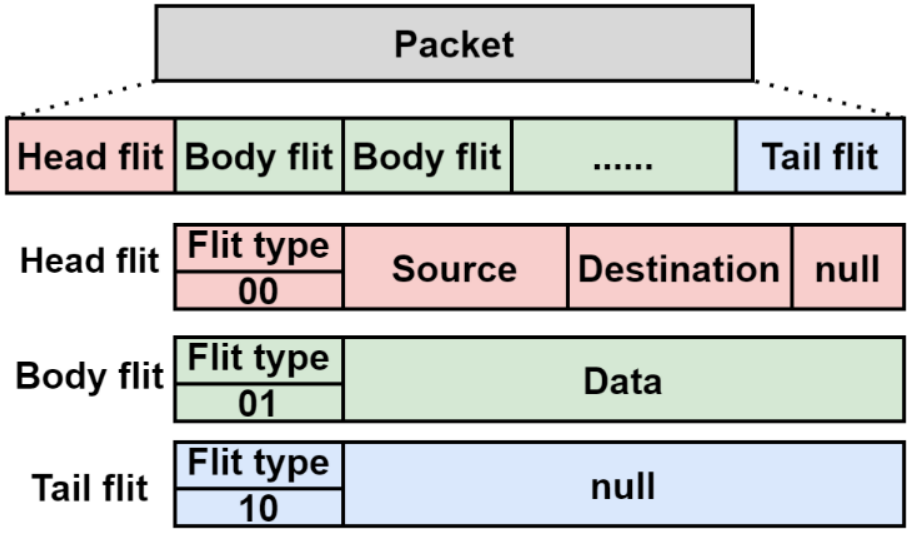
AXI4-Stream Protocol in NoC

- We changed the original signals of the router to AXI4-stream protocol
- Handshake process
  - In AXI4-stream, for a transfer to occur both the TVALID and TREADY signals must be asserted.
  - A receiver is permitted to wait for TVALID to be asserted before asserting the corresponding TREADY.

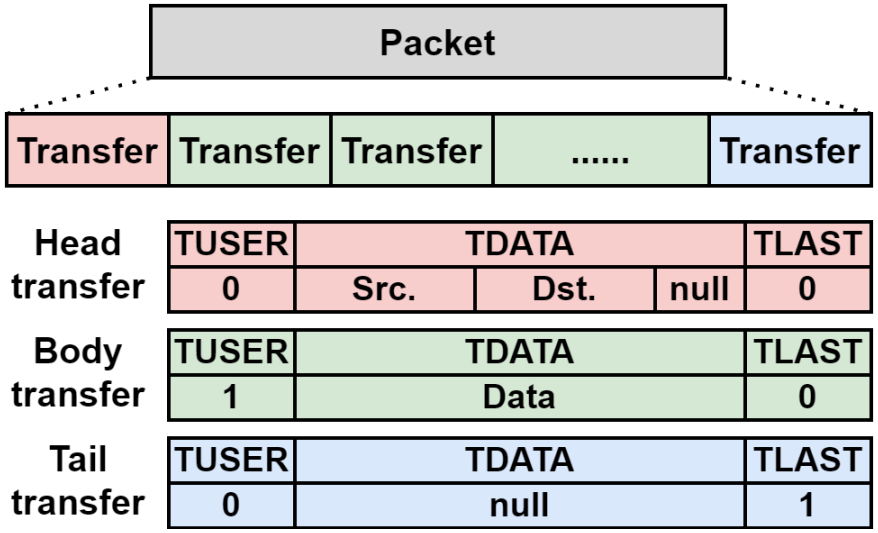


Packet Format in Different Protocols

- We use the content composed of TLAST and TUSER to distinguish multiple transfers in the AXI4-stream packet.
  - The router architecture won't change much.



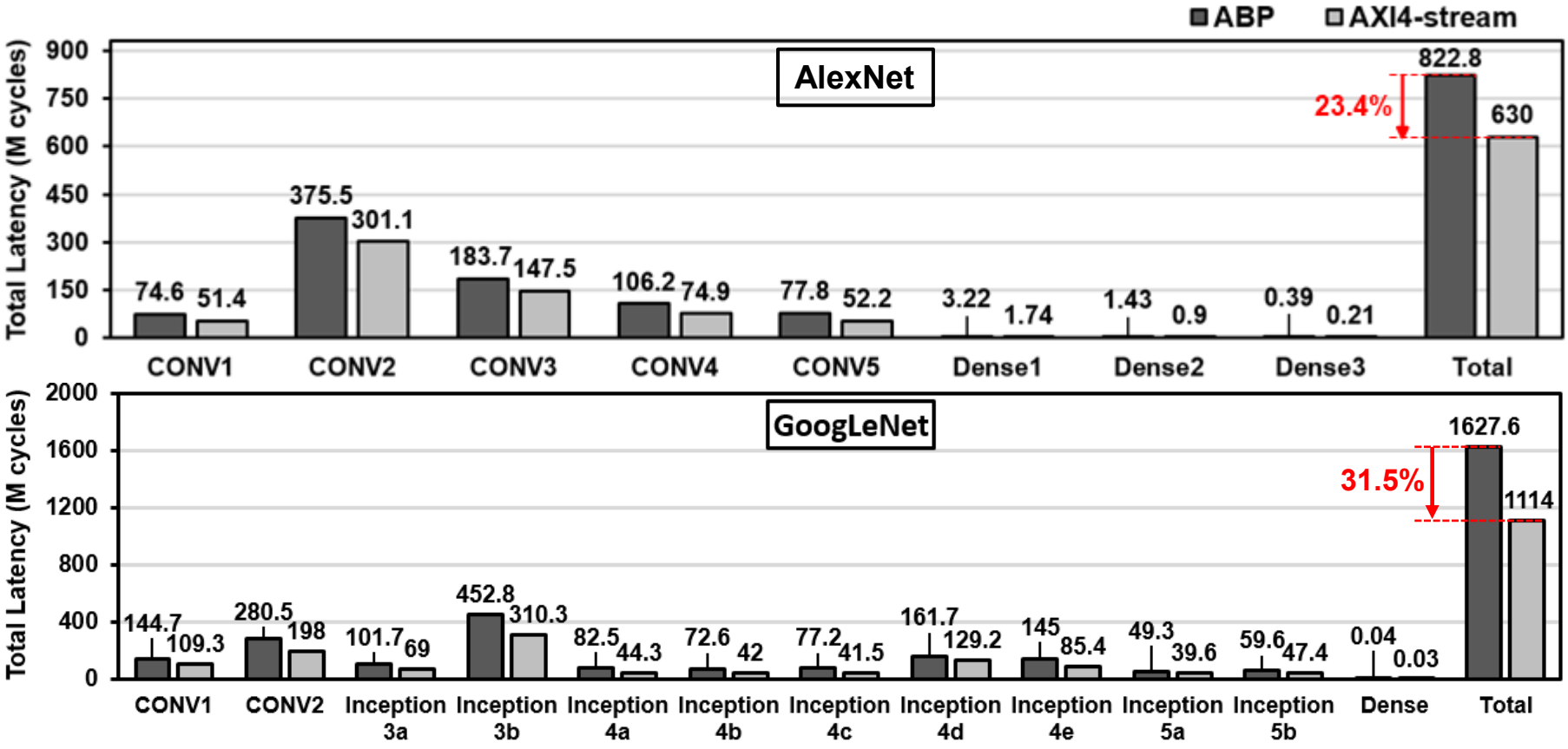
Conventional protocol



AXI4-stream protocol

ABP vs. AXI4-stream

- Analysis between ABP-based router and AXI4-stream-based router.
  - The AXI4-stream-based router can reduce latency efficiently.



Summary of FC

- ACK/NACK: is rarely used because of its buffer and bandwidth inefficiency
  - Upstream optimistically sends downstream
  - Buffer cannot be deallocated until ACK/NACK received
  - Inefficiently utilizes buffer space
- Credit-based: Used in systems with small numbers of buffers.
  - Upstream knows how many buffers are downstream
  - Downstream passes back credits to upstream
  - Significant upstream signaling (esp. for small flits)
- On/Off : Used in systems that have large numbers of flit buffers.
  - Downstream has on/off signal to upstream
- AXI: a popular on-chip communication protocol in modern product
  - Can reduce the on-chip communication latency compared with ABP

Router Architecture

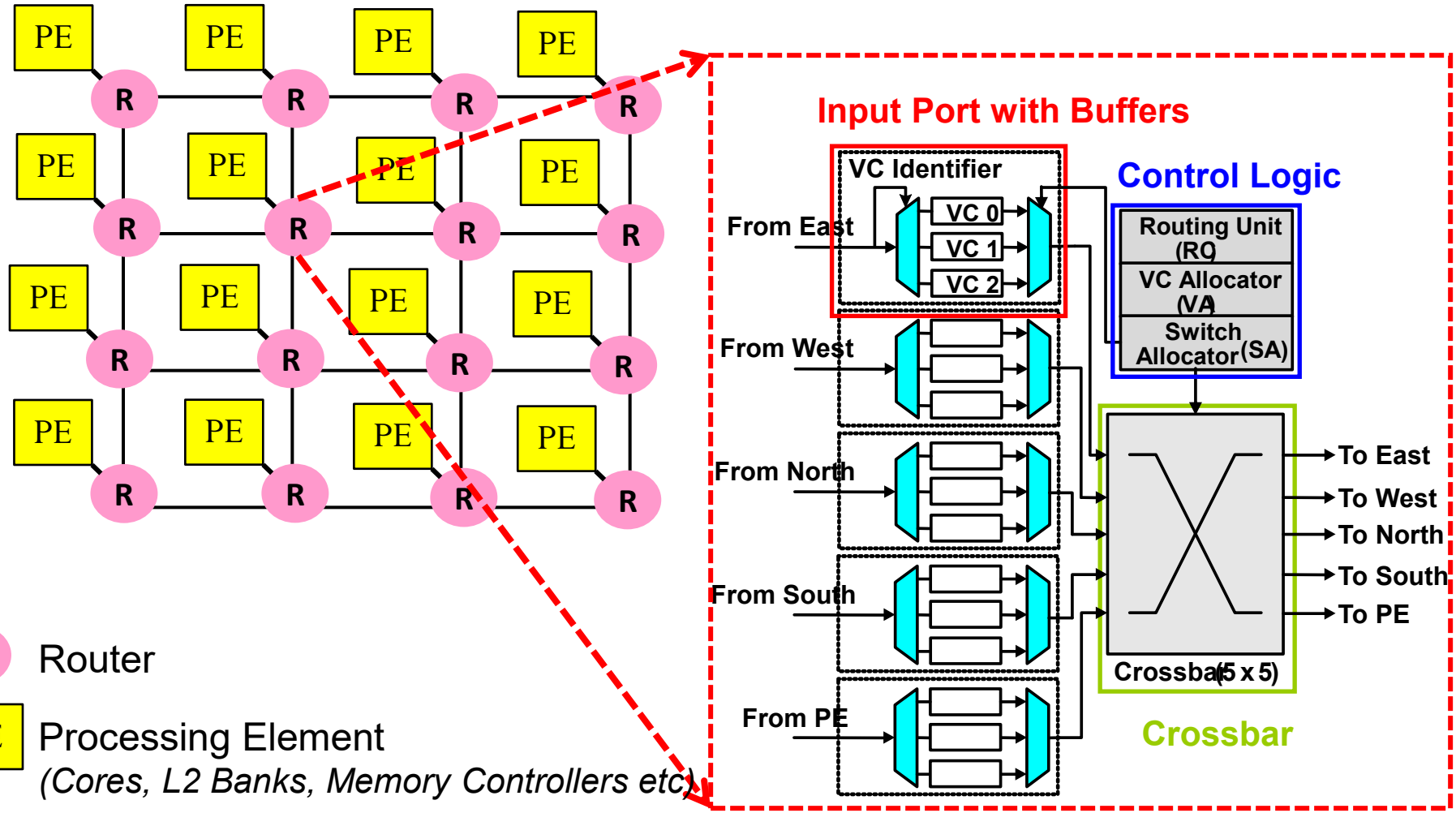
Typical Virtual Channel Router

A router functional blocks can be divided into:

1. Datapath: handles storage and movement of a packet payload
   - ❖ Input buffers , Switch, Output buffers

2. Control path: coordinating the movements of the packets through the resources of the datapath
   - ❖ Route Computation, VC Allocator, Switch Allocator

Example of NoC architecture



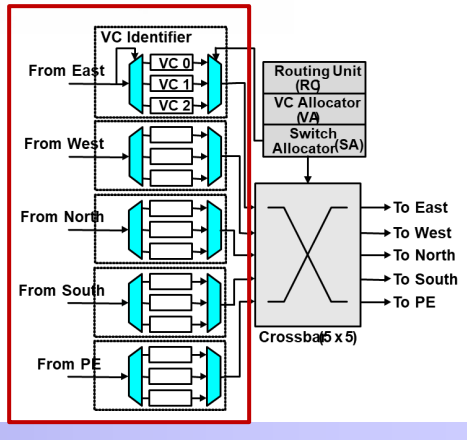
Router Design: Functions of a Router

- Buffering (of flits)
- Route computation
- Arbitration of flits (i.e. prioritization) when contention
  - Called packet scheduling
- Switching
  - From input port to output port
- Power management
  - Scale link/router frequency

State Fields of Input Virtual Channel

Virtual channel state fields, represented by a 5-vector: GROPC.

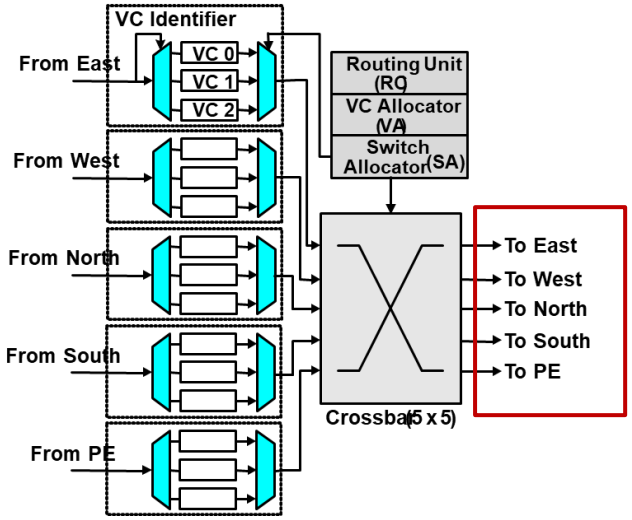
Field	*Name*	*Description*
G	Global state	Either idle (I), routing (R), waiting for an output VC (V), active (A), or waiting for credits (C).
R	Route	After routing is completed for a packet, this field holds the output port selected for the packet.
O	Output VC	After virtual-channel allocation is completed for a packet, this field holds the output virtual channel of port R assigned to the packet.
P	Pointers	Flit head and tail pointers into the input buffer. From these pointers, we can also get an implicit count on the number of flits in the buffer for this virtual channel.
C	Credit count	The number of credits (available downstream flit buffers) for output virtual channel O on output port R.



State Fields of Output Virtual Channel

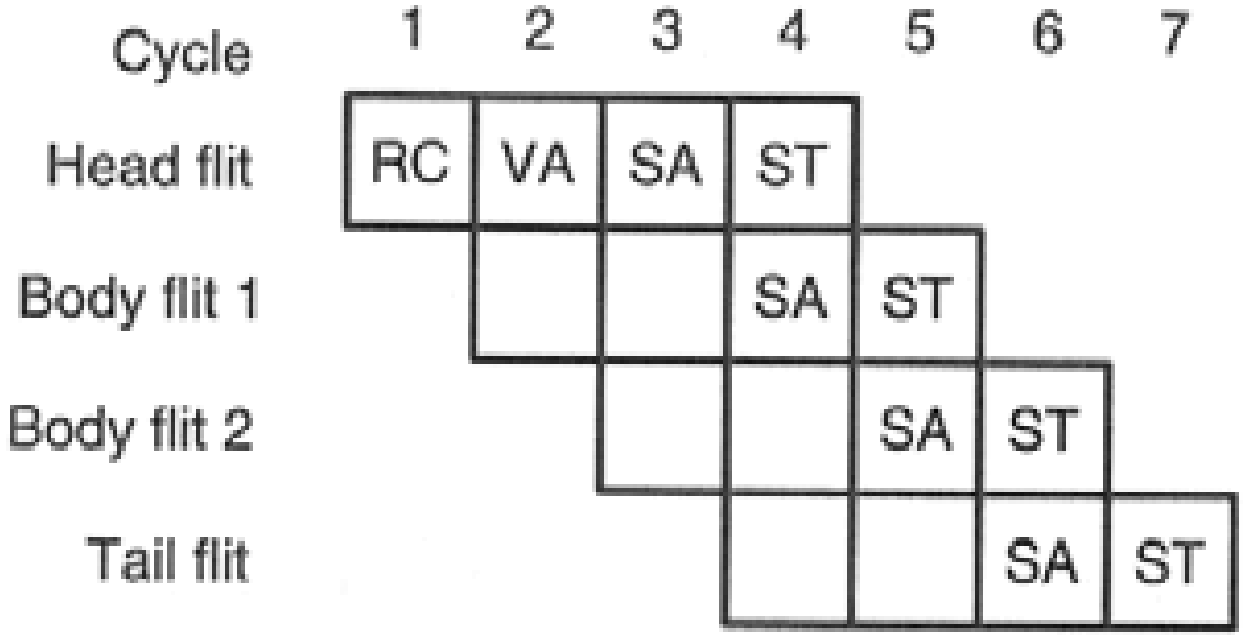
Output virtual channel state fields, represented by a 3-vector: GIC.

Field	*Name*	*Description*
G	Global state	Either idle (I), active (A), or waiting for credits (C).
I	Input VC	Input port and virtual channel that are forwarding flits to this output virtual channel.
C	Credit count	Number of free buffers available to hold flits from this virtual channel at the downstream node.



The Router Pipeline

- A typical router pipeline includes the following stages:
  - **RC** (Routing Computation)
  - **VA** (Virtual Channel Allocation)
  - **SA** (Switch Allocation)
  - **ST** (Switch Traversal)

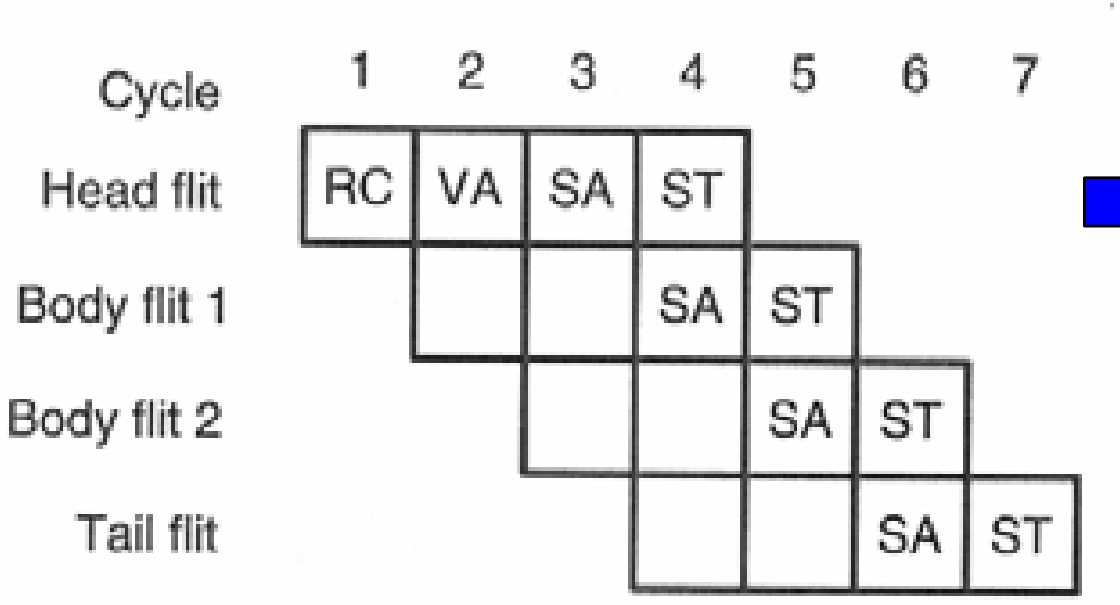


Why no RC and VA in Body/ Tail flit transmission?

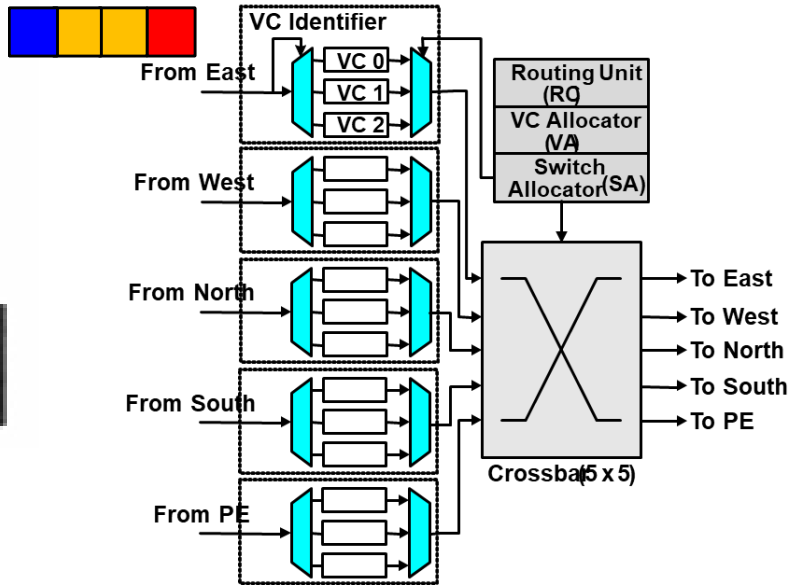
No pipeline stalls

The Router Pipeline

- Cycle 0
  - Head flit arrives and the packet is directed to a virtual channel of the input port (G = I)

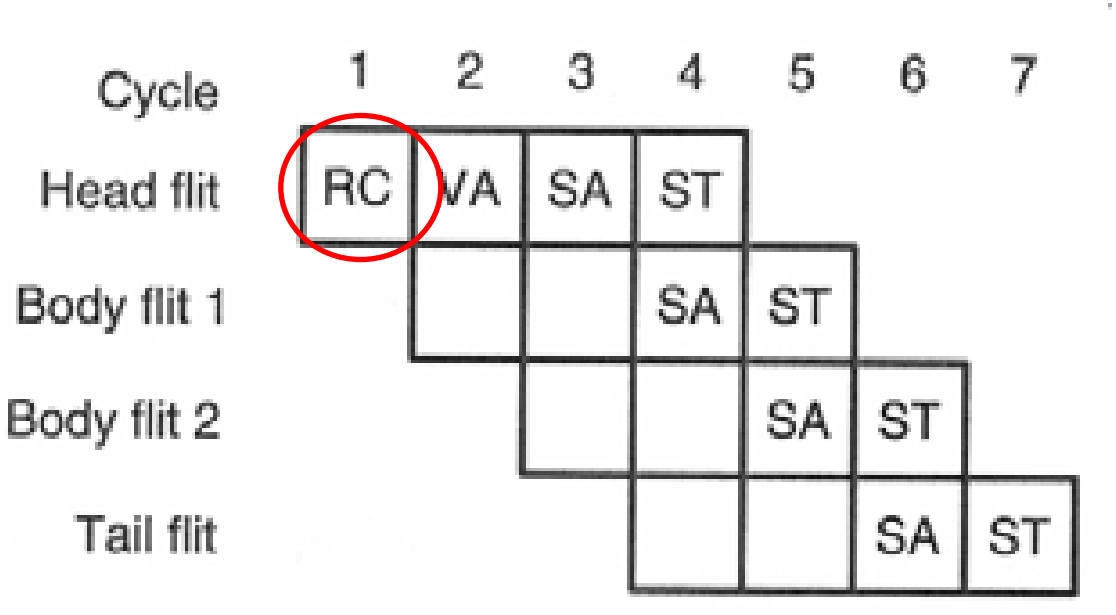


no pipeline stalls

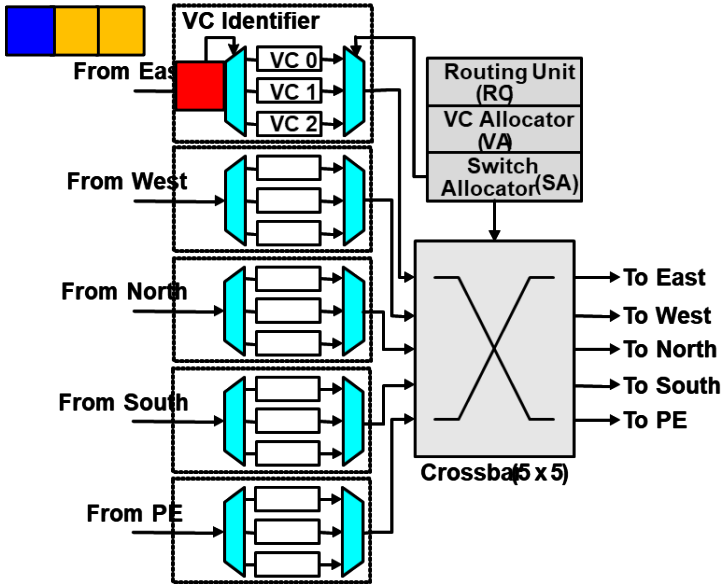


The Router Pipeline

- Cycle 1
  - Routing computation
  - Virtual channel state changes to routing (G = R)
  - Head flit enters RC-stage
  - First body flit arrives at router

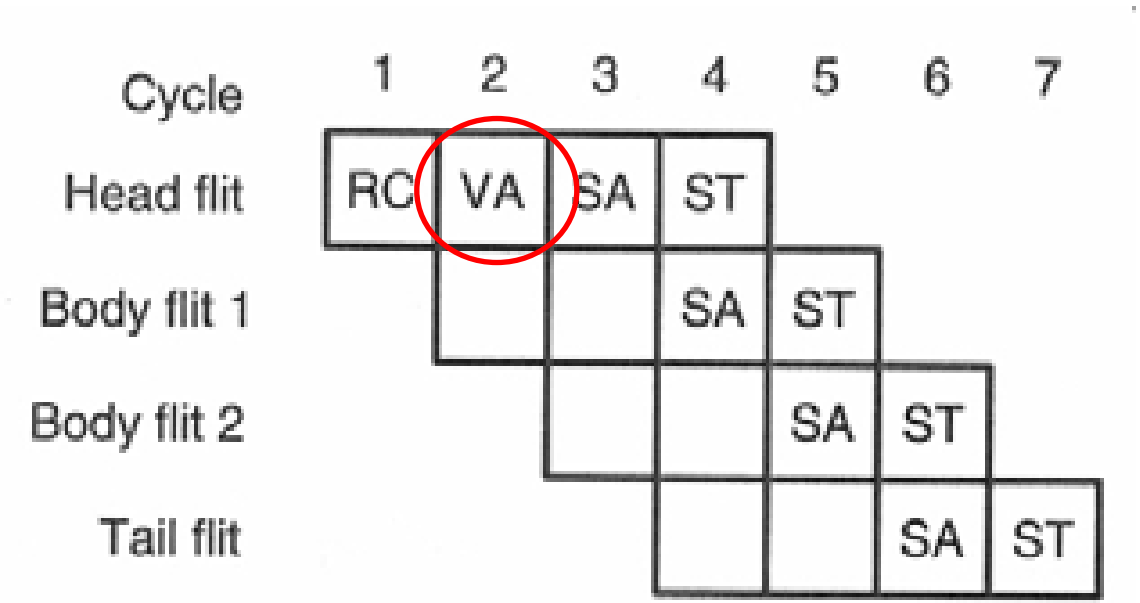


no pipeline stalls

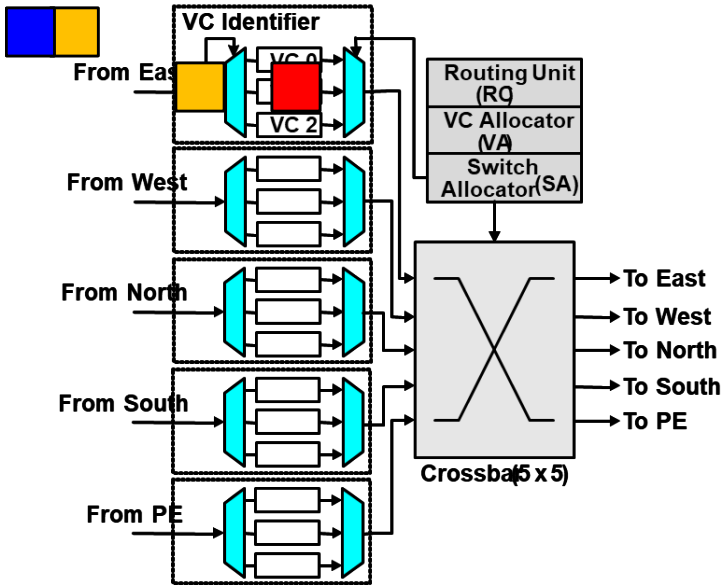


The Router Pipeline

- Cycle 2: Virtual Channel Allocation
  - Route field (R) of virtual channel is updated
  - Virtual channel state is set to “waiting for output virtual channel” (G = V)
  - Head flit enters VA state
  - First body flit enters RC stage (but do nothing)
  - Second body flit arrives at router

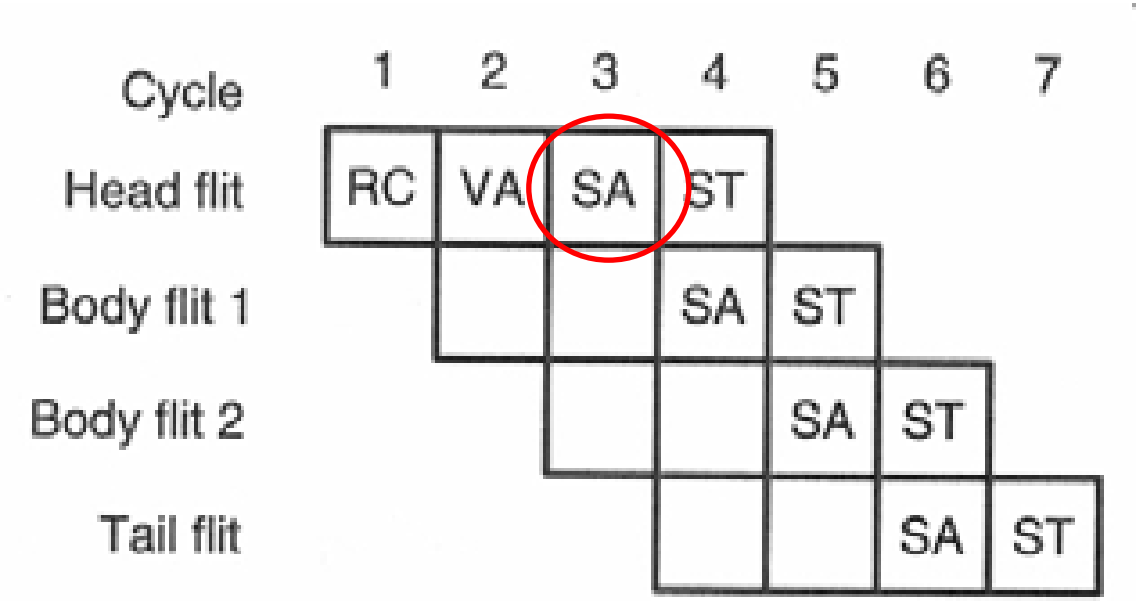


no pipeline stalls

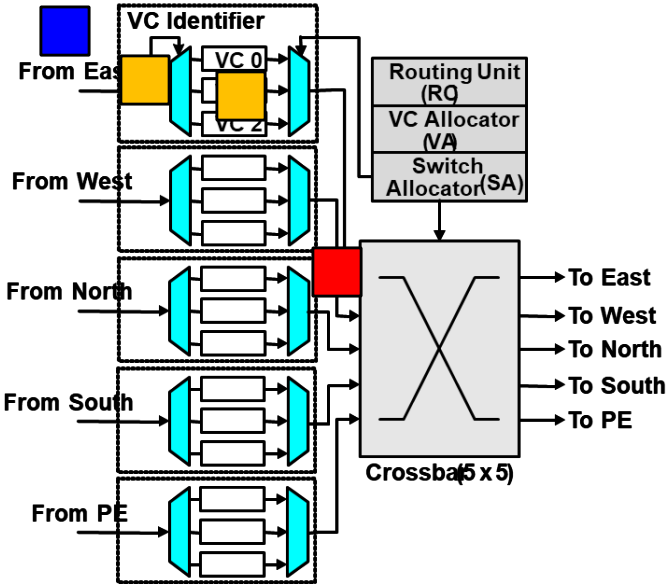


The Router Pipeline

- Cycle 3: Switch Allocation
  - All further processing is done on a flit base
  - Head flit enters SA stage
  - Any active VA (G = A) that contains buffered flits (indicated by P) and has downstream buffers available (C > 0) bids for a single-flit time slot through the switch from its input VC to the output VC
  - If successful, pointer field is updated
  - Credit field is decremented

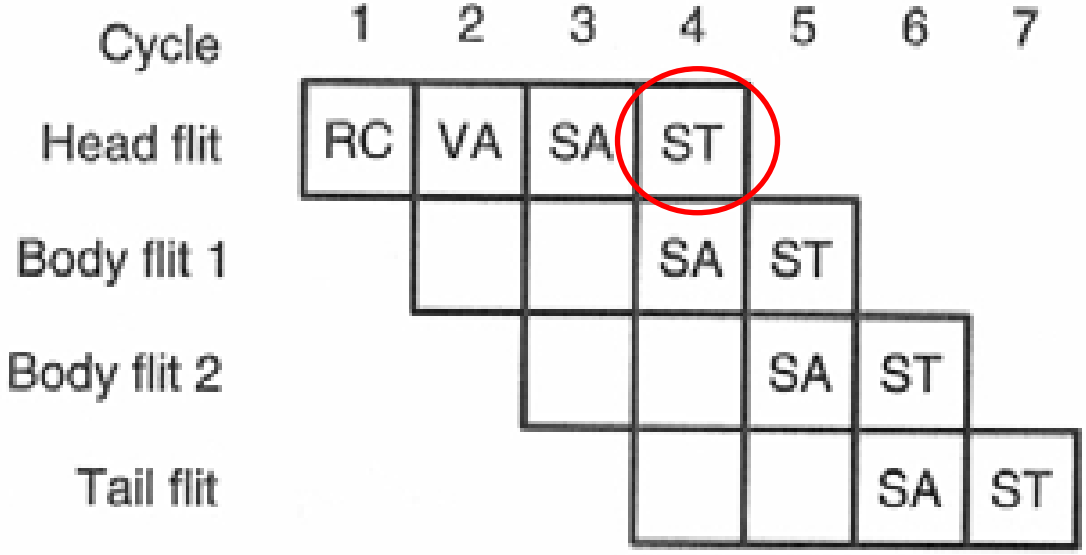


no pipeline stalls

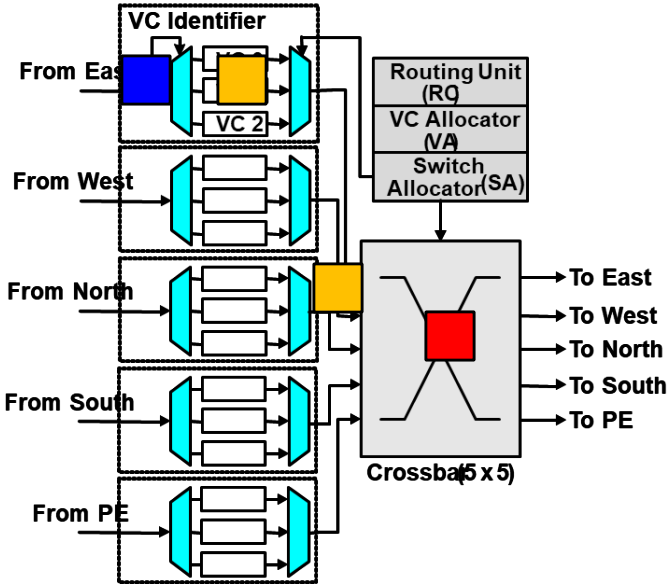


The Router Pipeline

- Cycle 4: Switch Traversal
  - Head flit traverses the switch
- Cycle 5:
  - Head flit starts traversing the channel to the next router



no pipeline stalls

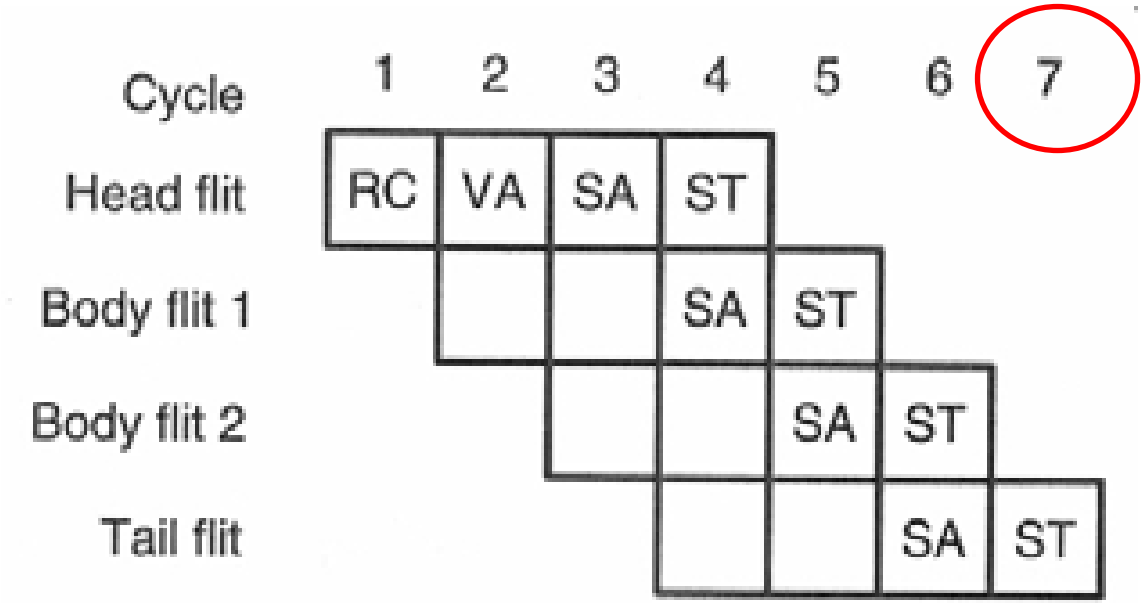


The Router Pipeline

- Cycle 7:
  - Tail traverses the switch
  - Output VC set to idle
  - Input VC set to idle (G = I), if buffer is empty
  - Input VC set to routing (G = R), if another head flit is in the buffer

- Only the head flits enter the RC and VA stages

- The body and tail flits are stored in the flit buffers until they can enter the SA stage



no pipeline stalls

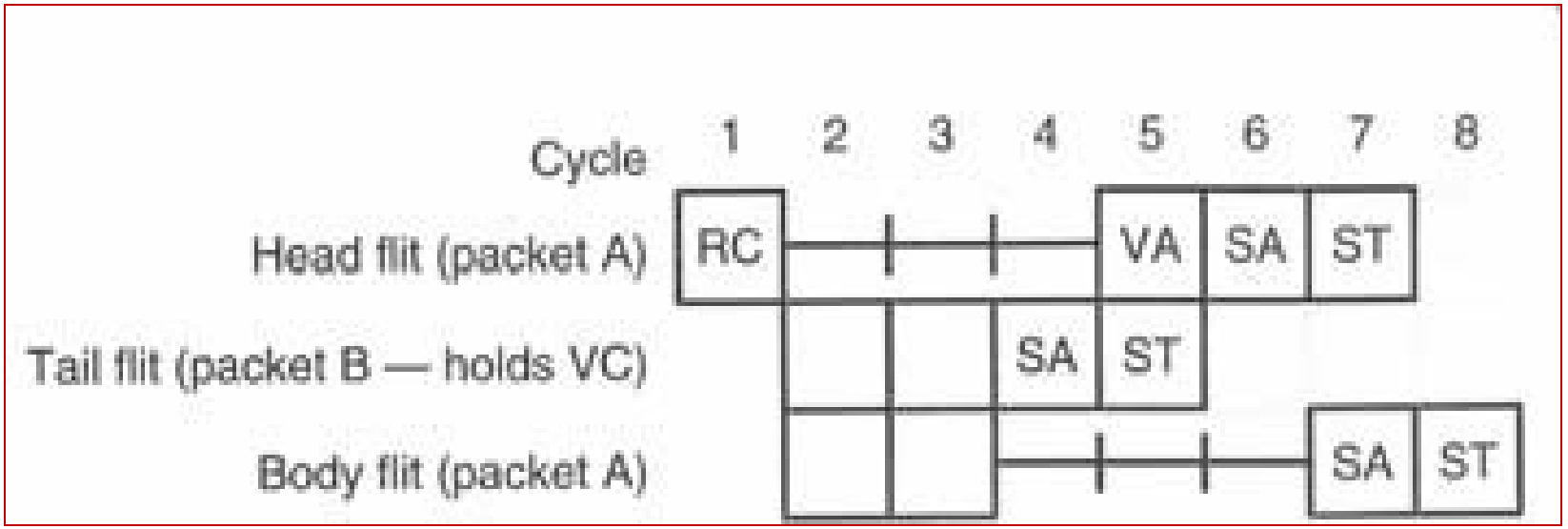
Pipeline Stalls

Pipeline stalls can be divided into:

- **Packet stalls**
  - can occur if the virtual channel cannot advance to its R, V, or A state
- **Flit stalls**
  - If a virtual channel is in active state and the flit cannot successfully complete switch allocation due to
    - Lack of flit, Lack of credit, Losing arbitration for the switch time slot

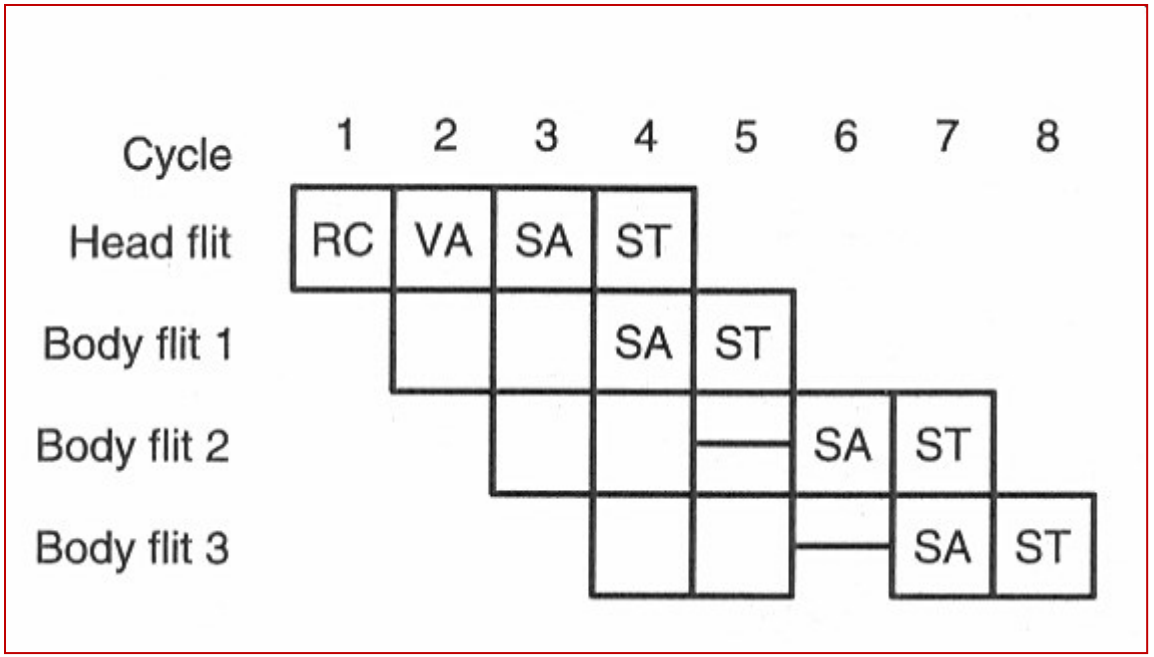
Example for Packet Stall

1. Virtual-channel allocation stall
   - Head flit of A can first enter the VA stage when the tail flit of packet B completes switch allocation and releases the virtual channel



Example for Flit Stalls

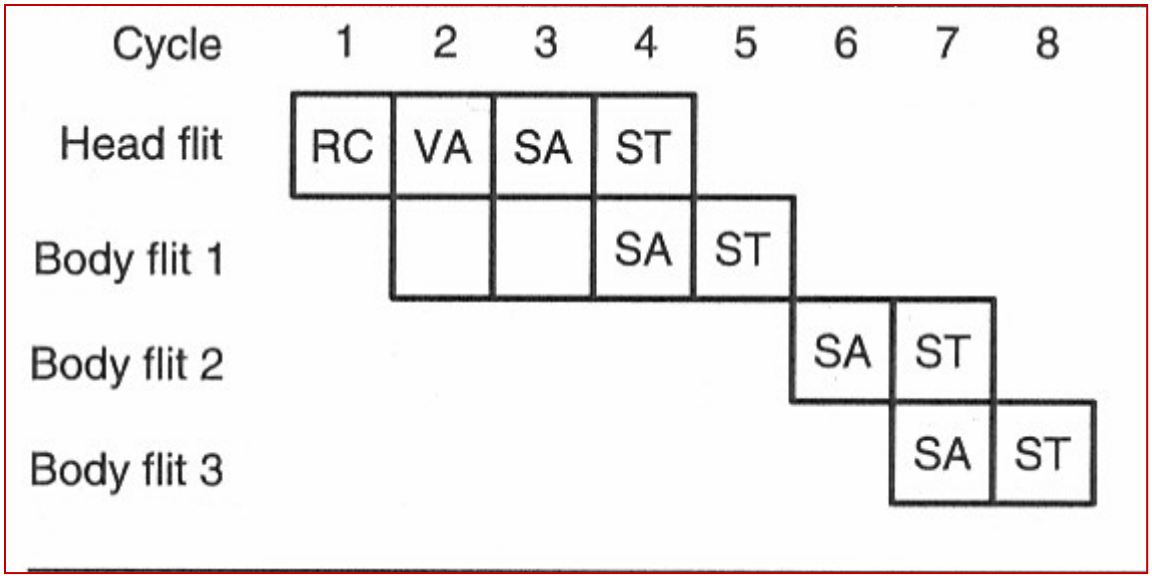
2. Switch allocation stall



Second body flit fails to allocate the requested connection in cycle 5

Example for Flit Stalls

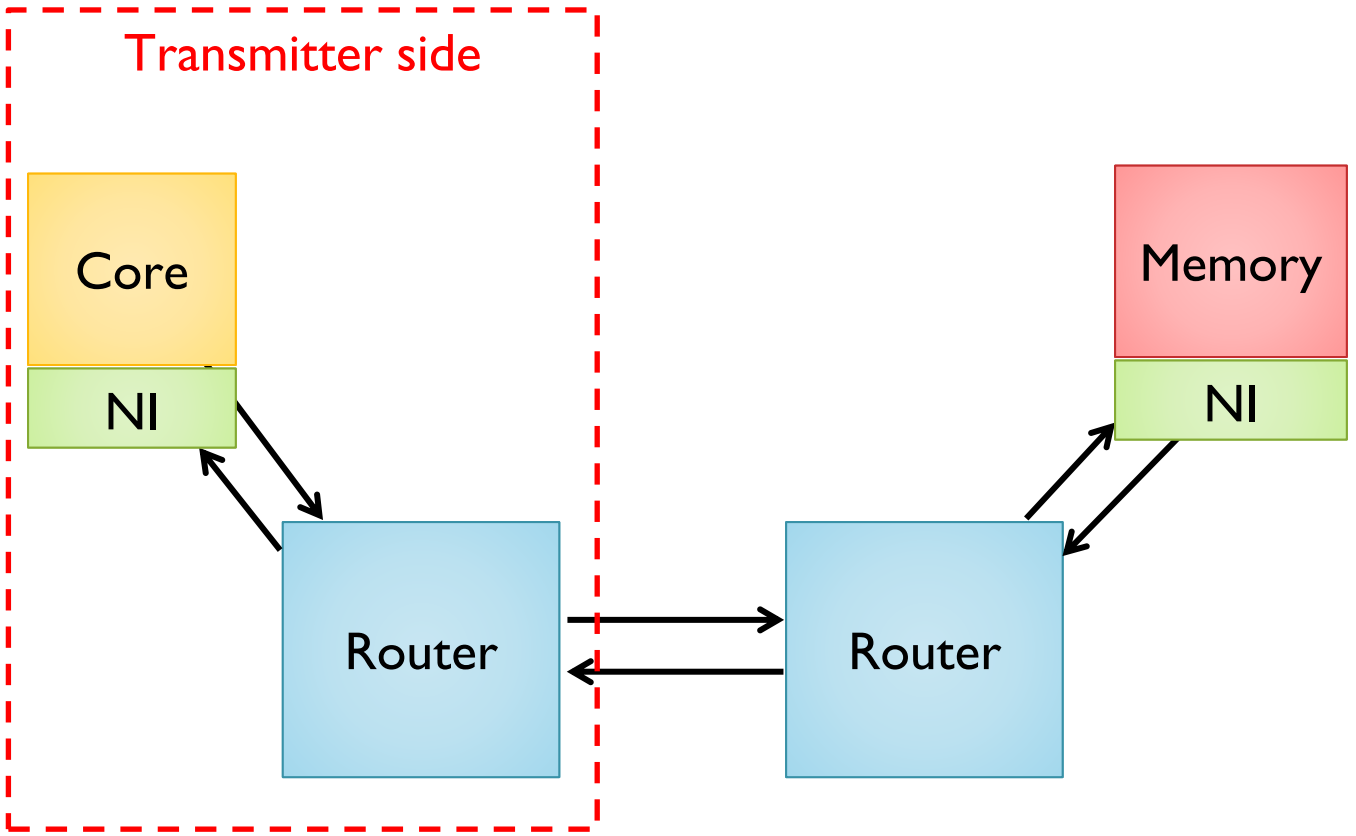
3. Buffer empty stall



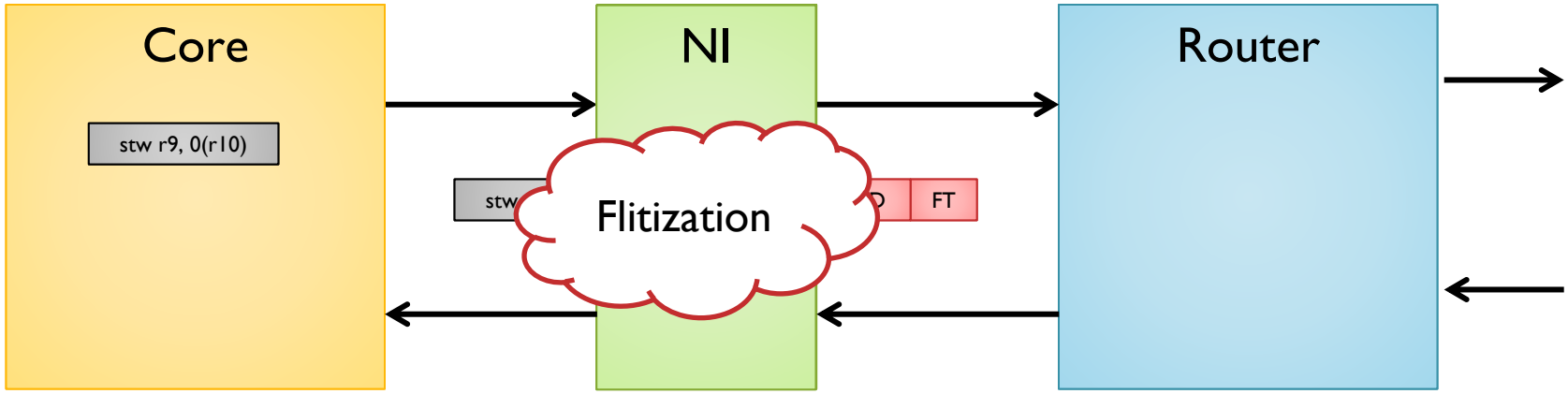
Body flit 2 is delayed three cycles. However, since it does not have to enter the RC and VA stage the output is only delayed one cycle!

Network Interface

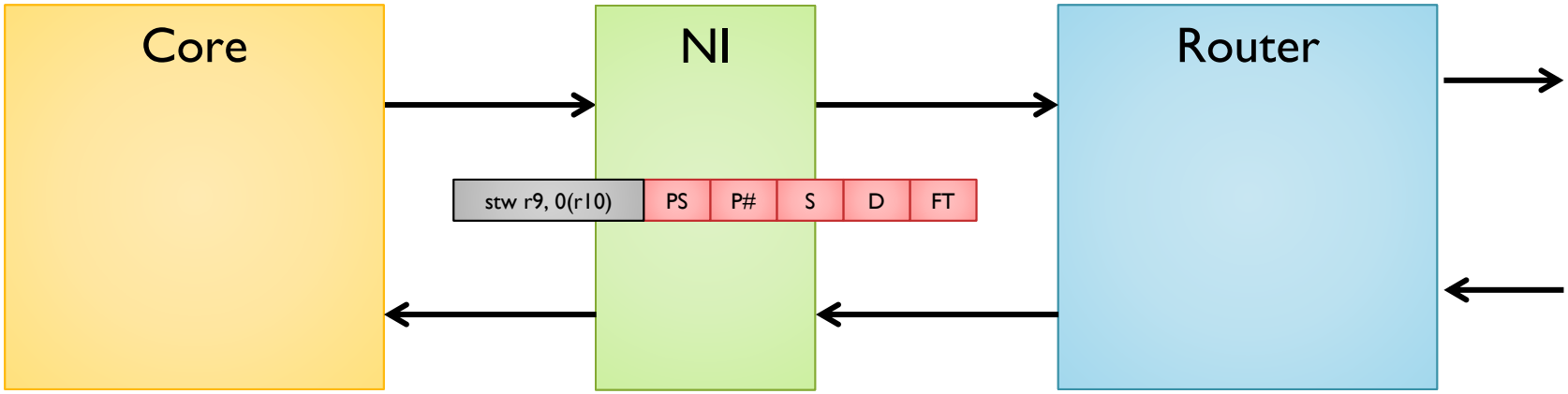
Network Interface



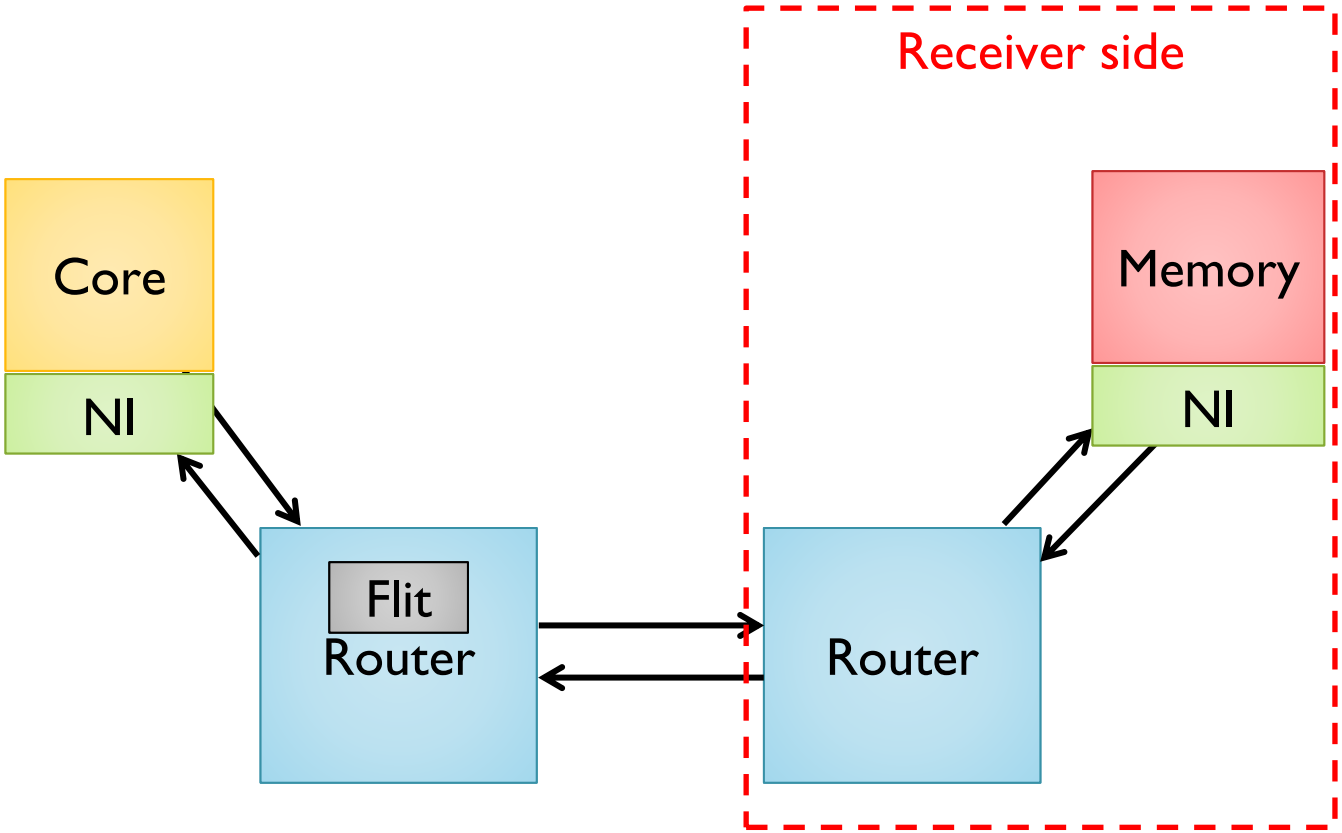
Network Interface



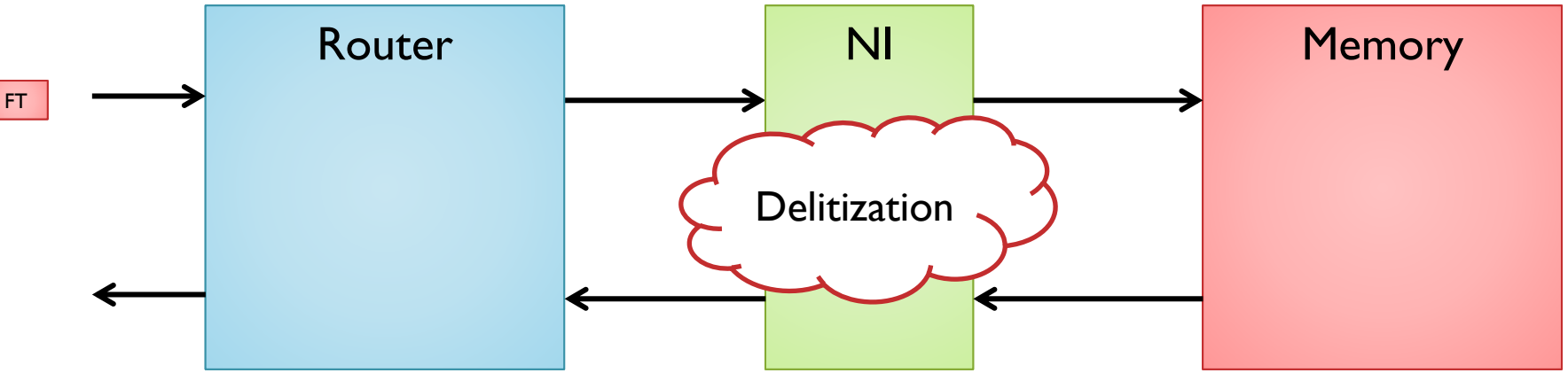
Network Interface



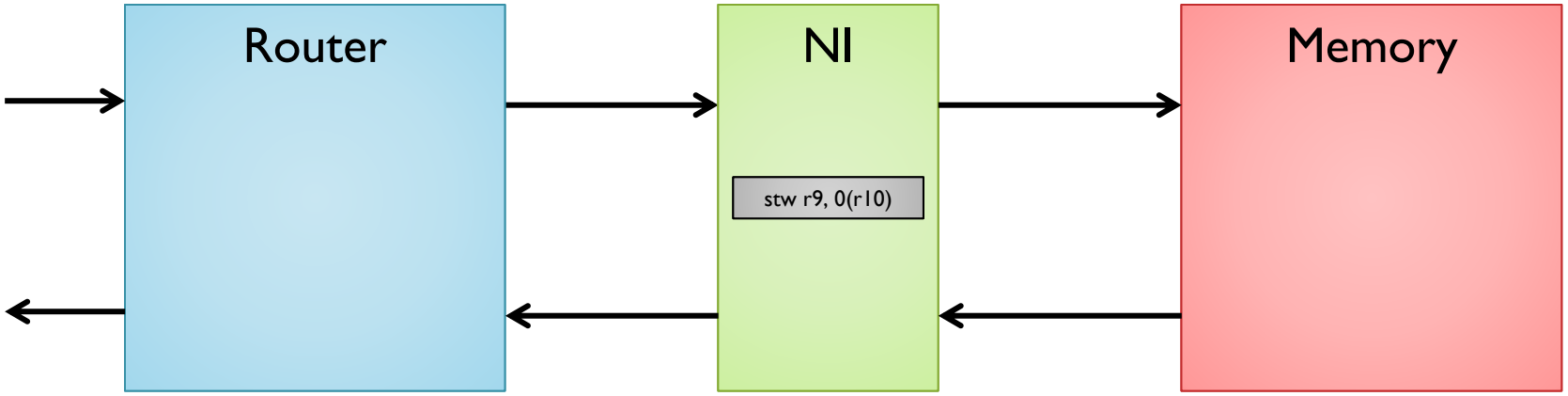
Network Interface



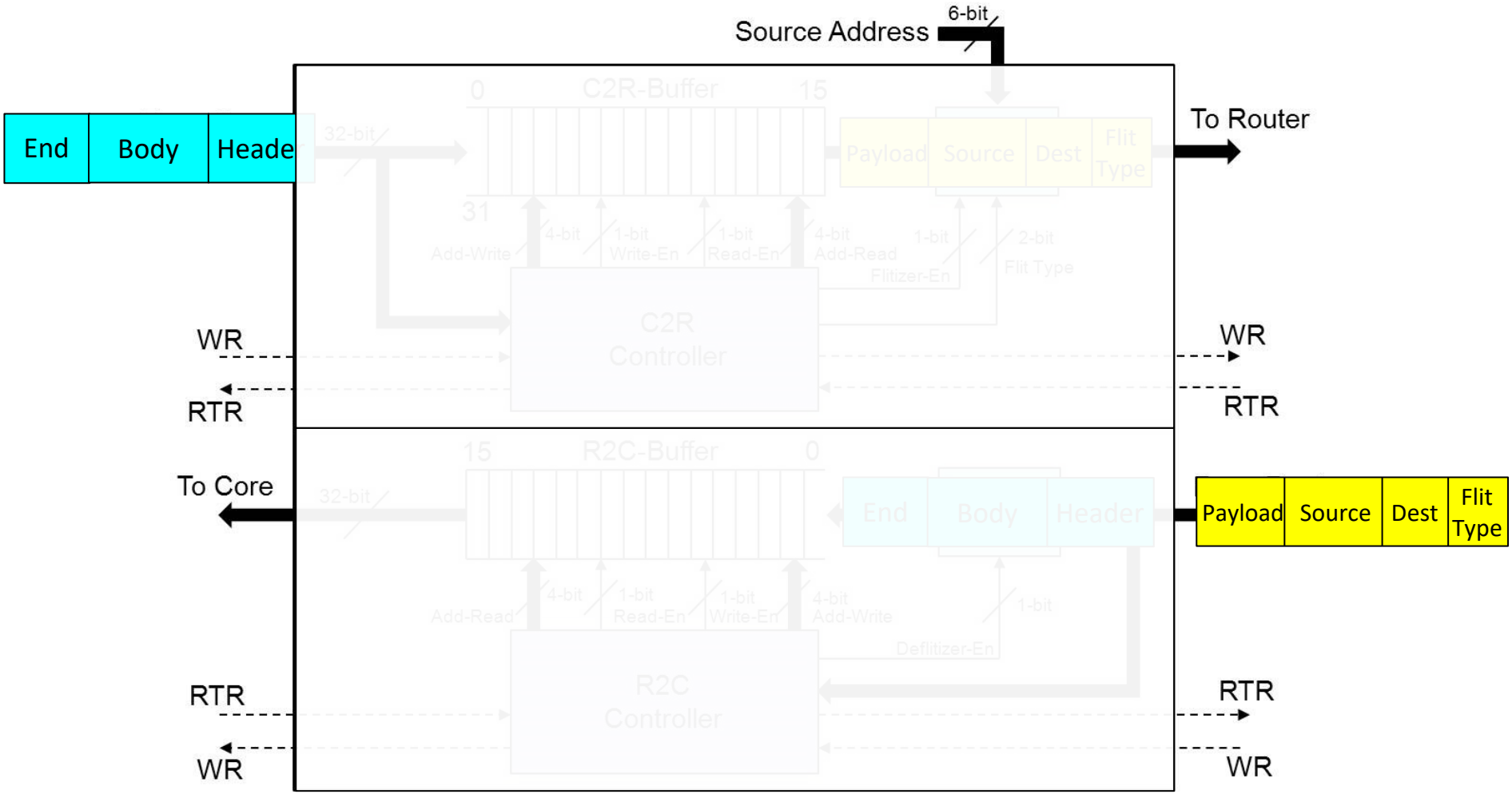
Network Interface



Network Interface



Network Interface



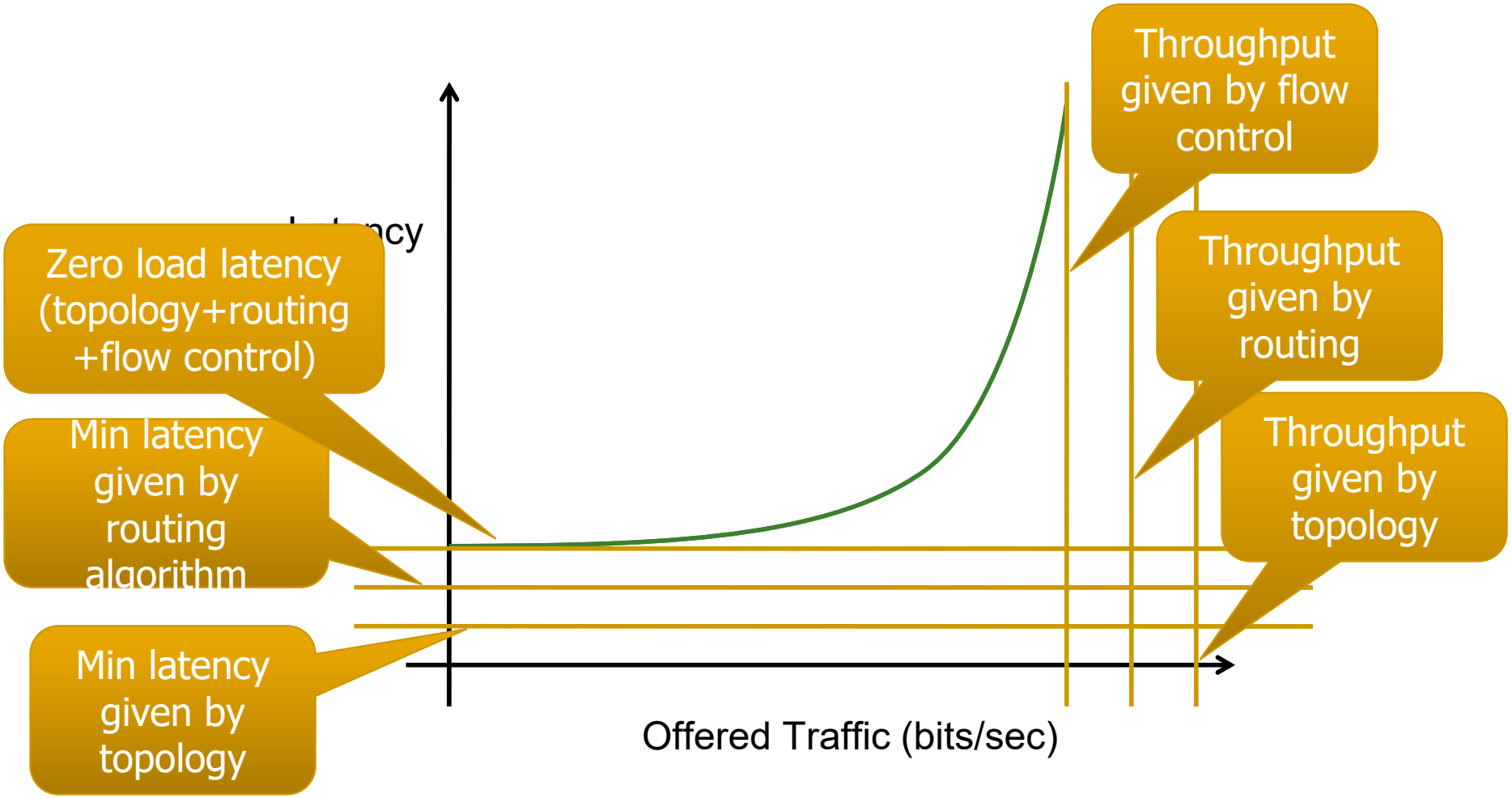
OASIS NoC NI

Network performance metrics

Packet Rate and Flit Rate

- The control of the router operates at two distinct frequencies
  - Packet Rate (performed once per packet)
    - Route computation
    - Virtual-channel allocation
  - Flit Rate (performed once per flit)
    - Switch allocation
    - Pointer and credit count update

Interconnection Network Performance



Ideal Latency

- Ideal latency
  - Solely due to wire delay between source and destination

$$T_{ideal} = \frac{D}{v} + \frac{L}{b}$$

  - D = Manhattan distance
  - L = packet size
  - b = channel bandwidth
  - v = propagation velocity

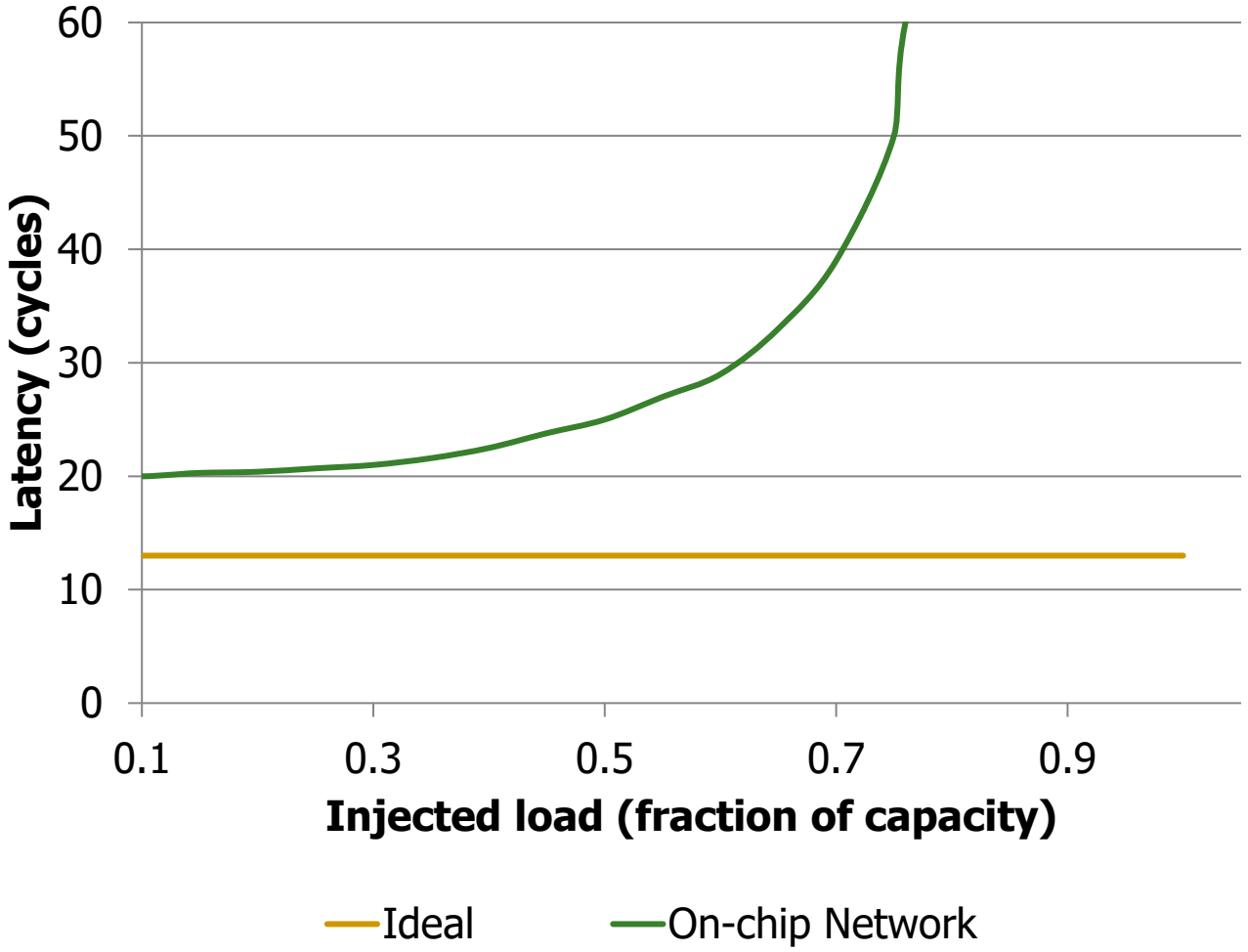
Actual Latency

- Dedicated wiring impractical
  - Long wires segmented with insertion of routers

$$T_{actual} = \frac{D}{v} + \frac{L}{b} + H \cdot T_{router} + T_c$$

  - D = Manhattan distance
  - L = packet size
  - b = channel bandwidth
  - v = propagation velocity
  - H = hops
  - T_{router} = router latency
  - T_c = latency due to contention

Latency and Throughput Curve



Summary

- NoC is a scalable platform for billion-transistor chips.
- Several driving forces behind it.
- Telecommunication devices, embedded and general-purposed (GP) domains are attractive applications for NoC.
- Expected to change the way we structure and model VLSI systems.
- Many open research questions.