



NOC Flow Control

Flow Control

- *Flow Control* determines how the resources of a network, such as **channel bandwidth** and **buffer capacity** are allocated to packets traversing a network
- Goal is to use resources as efficient as possible to allow a high throughput
- An efficient flow control is a prerequisite to achieve a good network performance

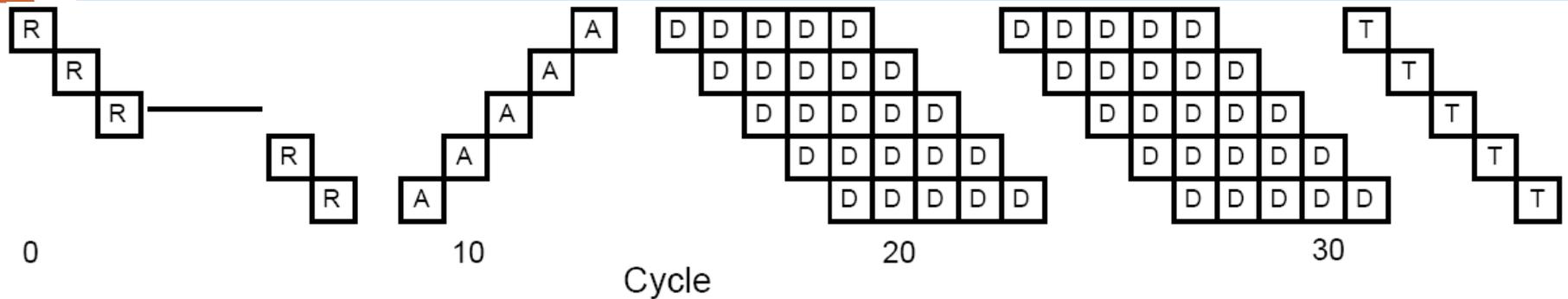
Flow Control

- Flow Control can be viewed as a problem of
 - Resource allocation
 - Contention resolution
- **Resources** in form of channels, buffers and state must be allocated to each packet
- If two packets **compete** for the same channel flow control can only assign the channel to one packet, but must also deal with the other packet

Flow Control

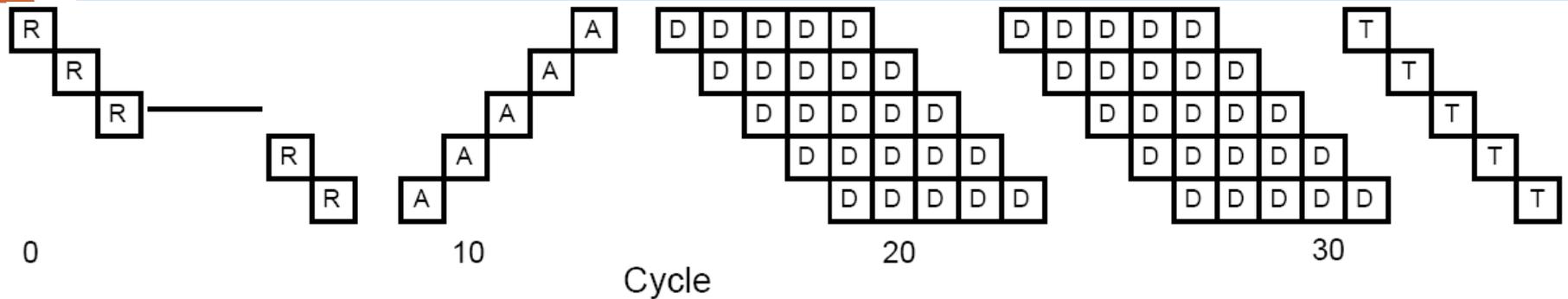
- Flow Control can be divided into
 - Bufferless flow control
 - ✓ Packets are either **dropped** or **misrouted**
 - Buffered flow control
 - ✓ Packets that cannot be routed via the desired channel are stored in buffers

Circuit Switching



- **Circuit-Switching** is a bufferless flow control, where several channels are reserved to form a circuit
- A request (*R*) propagates from source to destination, which is answered by an acknowledgement (*A*)
- Then data is sent (here two five flit packets (*D*)) and a tail flit (*T*) is sent to deallocate the channels

Circuit Switching



- Circuit-switching does not suffer from dropping or misrouting packets
- However there are two weaknesses
 - High latency: $T_0 = 3 H t_r + L/b$ (ignoring wire latency)
 - Low throughput, since channel is used to a large fraction of time for signaling and not for delivery of the payload

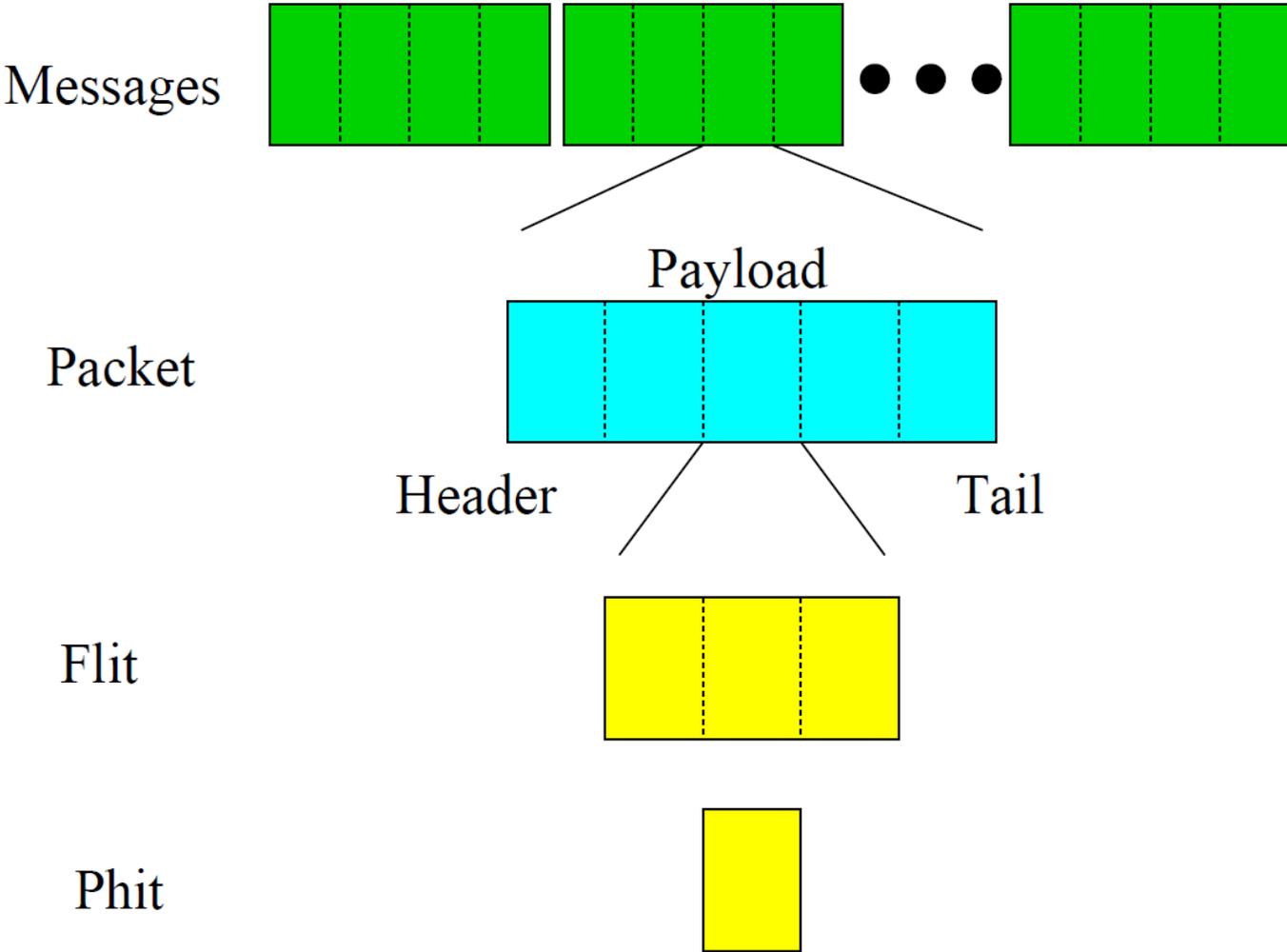
Buffered Flow Control

- More efficient flow control can be achieved by adding buffers
 - With sufficient buffers packets do not need to be misrouted or dropped, since packets can wait for the outgoing channel to be ready

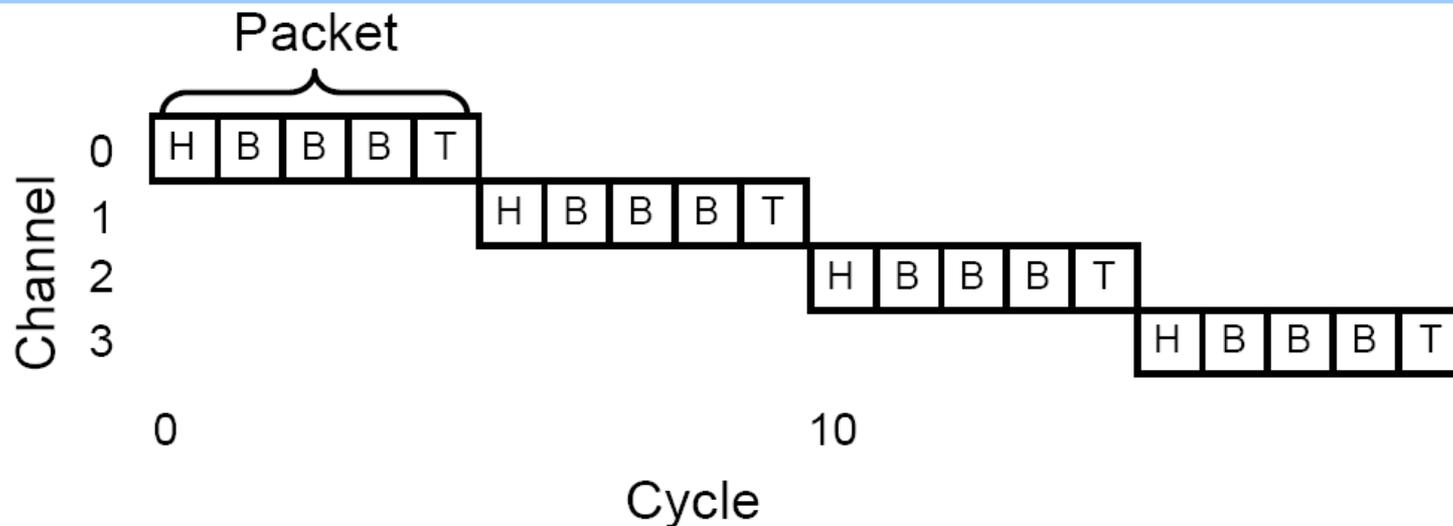
Buffered Flow Control

- Two main approaches
 - Packet-Buffer Flow Control
 - ✓ Store-And-Forward
 - ✓ Cut-Through
 - Flit-Buffer Flow Control
 - ✓ Wormhole Flow Control
 - ✓ Virtual Channel Flow Control

Data Units



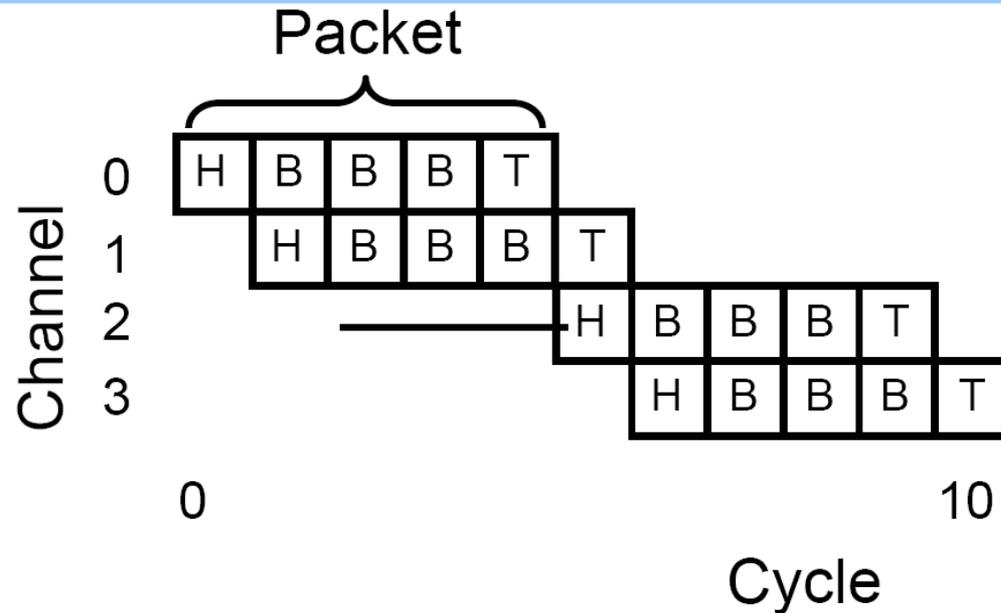
Store and Forward Flow Control



- 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

$$T_0 = H (t_r + L/b)$$

Cut-Through Flow Control



- Transmission on the next channel starts directly when the new header flit is received (otherwise it behaves like Store-Forward)

→ Channel is released after tail flit

$$T_0 = H t_r + L/b$$

Cut-Through Flow Control

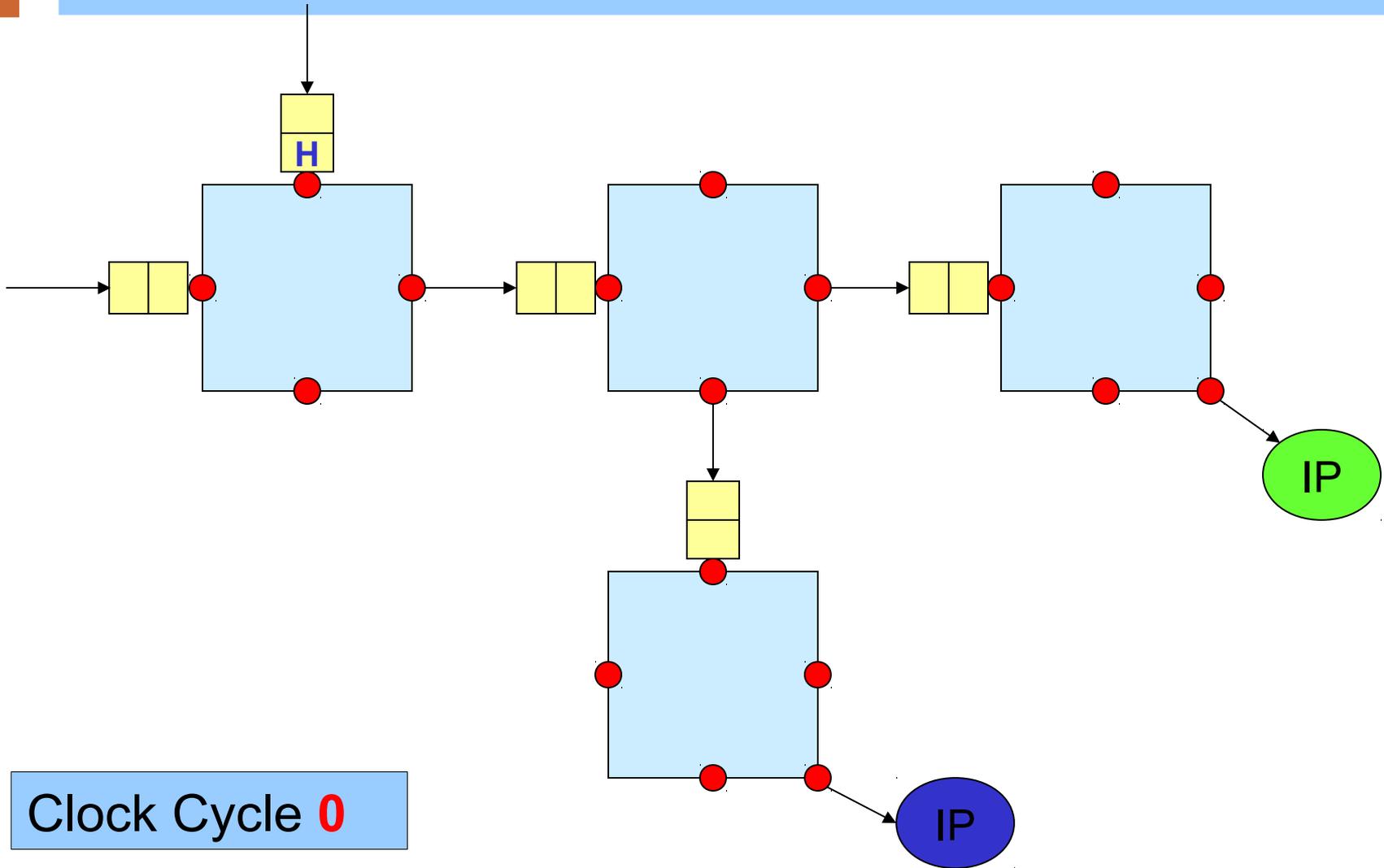
■ Shortcomings

- Very inefficient use of buffer space
 - ✓ As buffers are allocated in units of packets
 - ✓ Often we need multiple independent buffer sets to reduce blocking or provide deadlock avoidance
- By allocating buffers in units of packets → contention latency is increased
 - ✓ E.g., High-priority packet colliding with a low-priority packet
 - Must wait the entire low-priority packet to be transmitted before it can acquire the channel

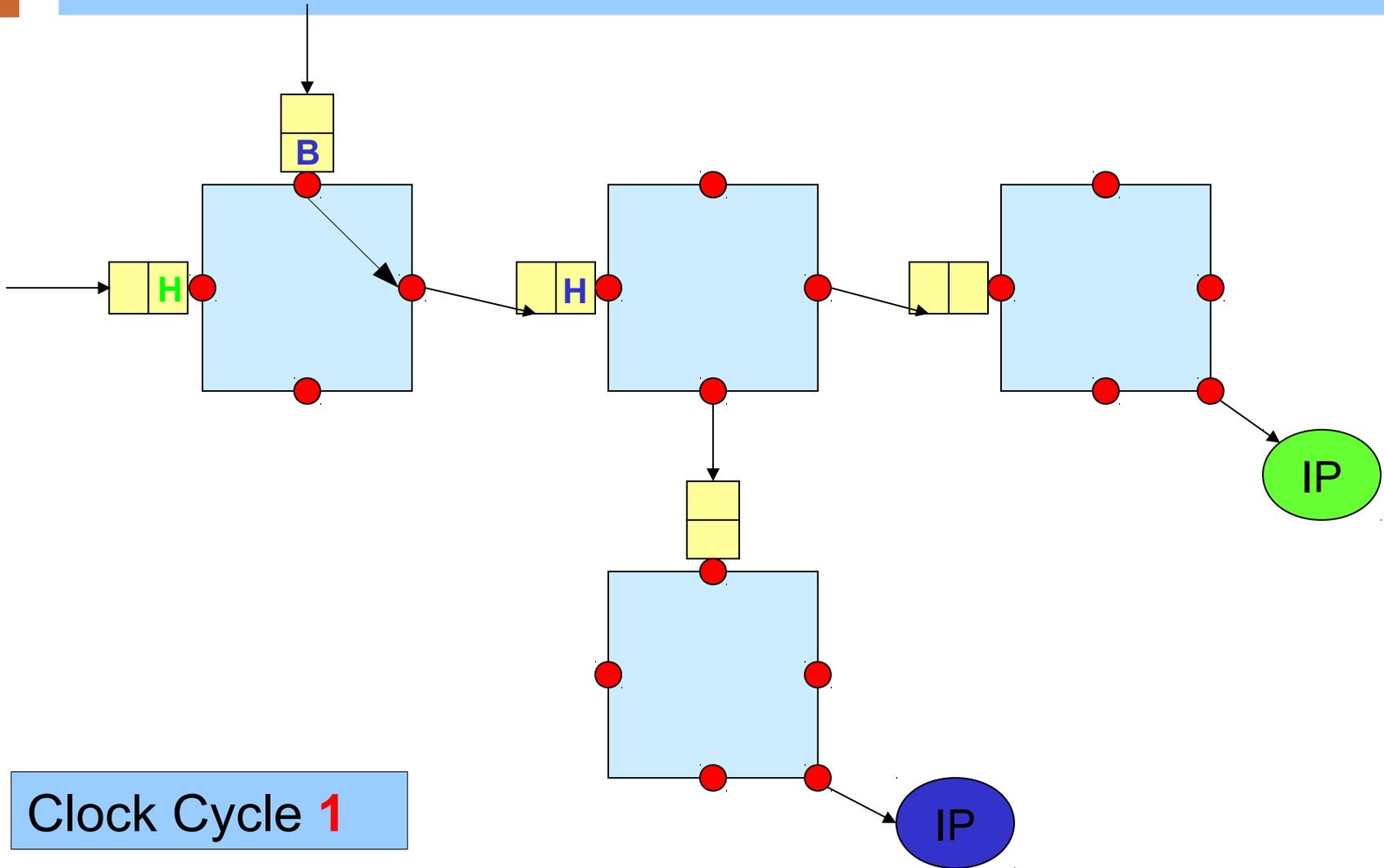
Wormhole Flow Control

- **Wormhole** flow control operates like cut-through, but with channel and buffers allocated to flits rather than packets
- Three resources are needed
 - A virtual channel for the packet
 - ✓ Body flits of a packet use the VC acquired by the head flit
 - One flit buffer
 - One flit channel bandwidth

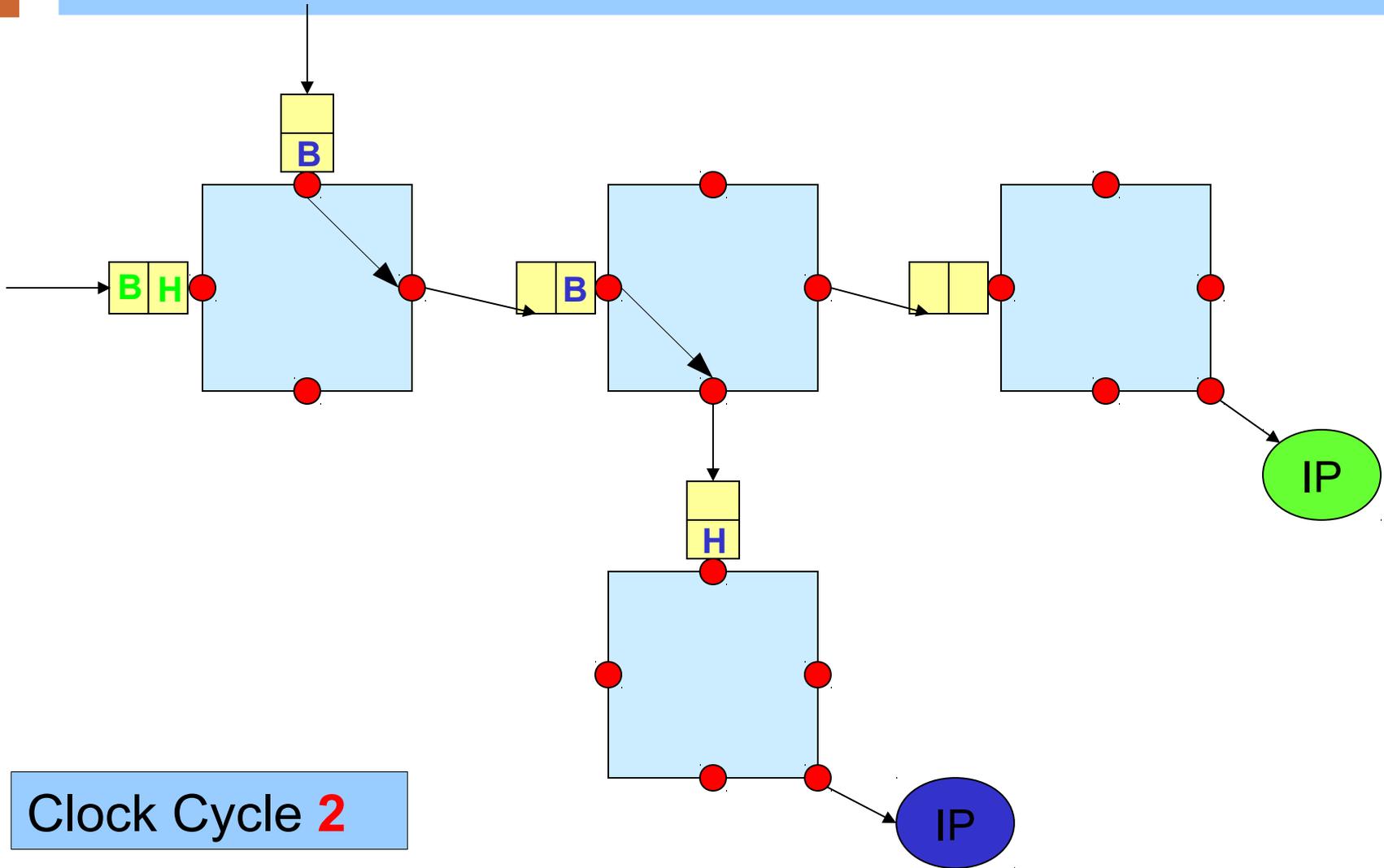
Wormhole - Example



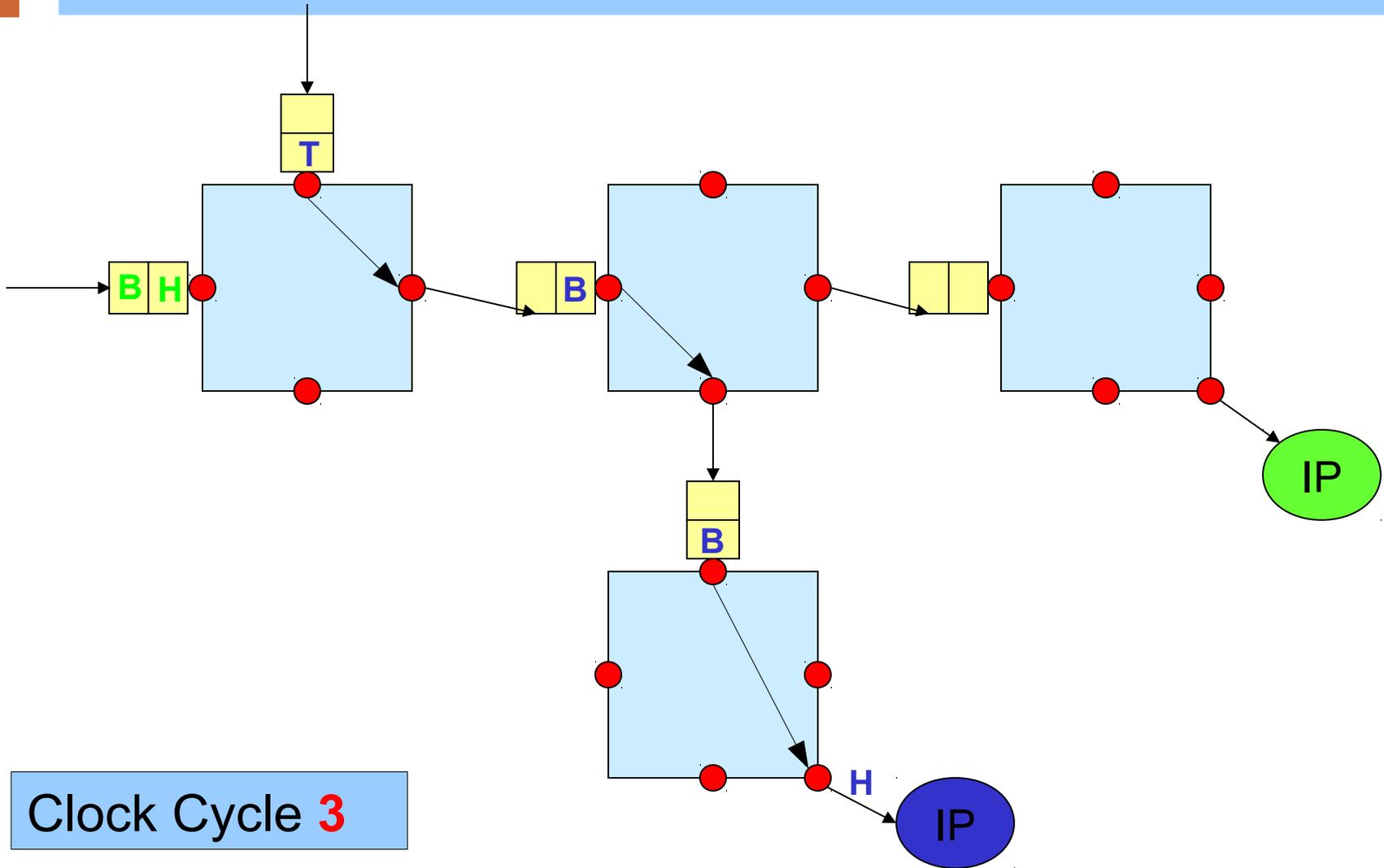
Wormhole - Example



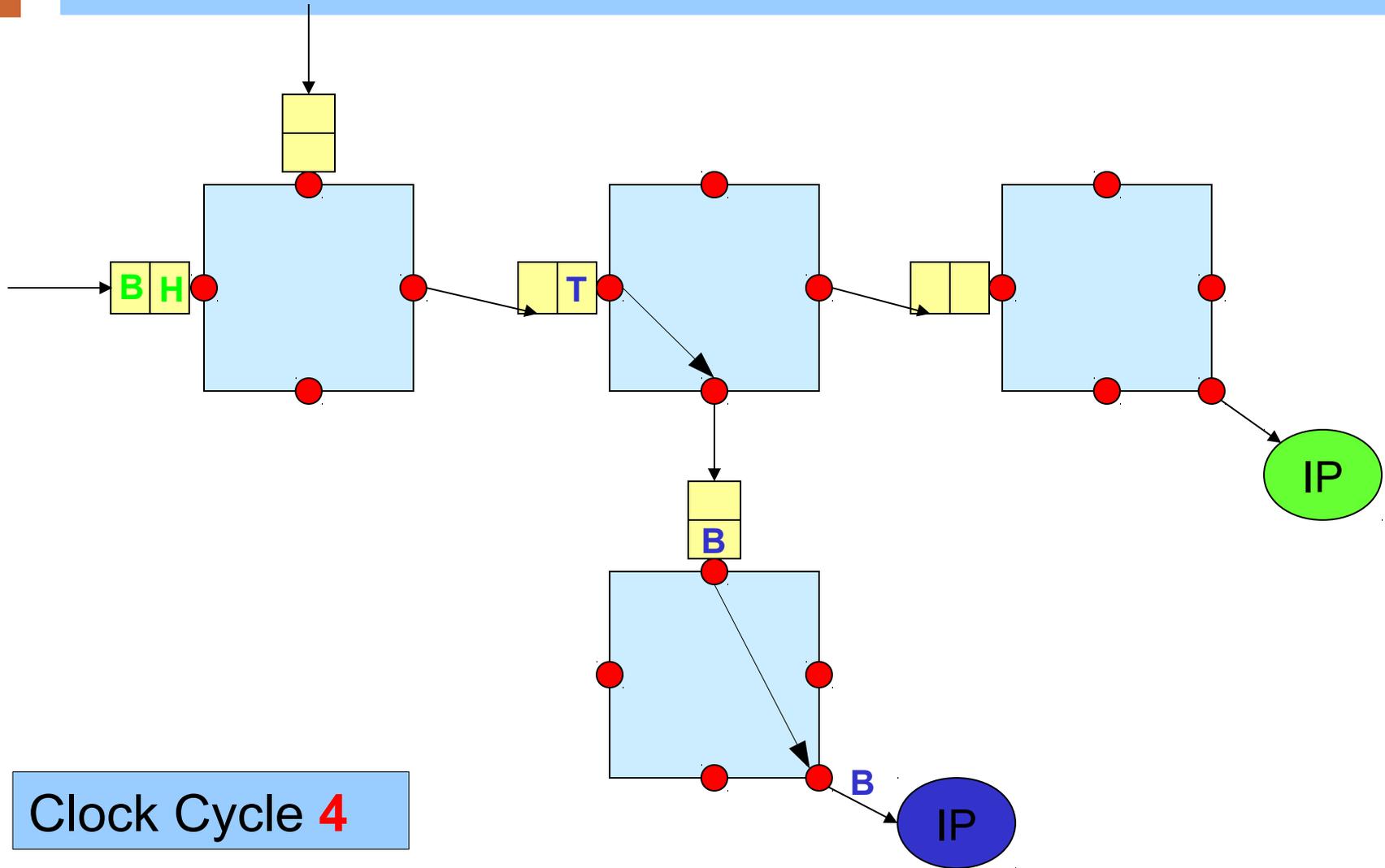
Wormhole - Example



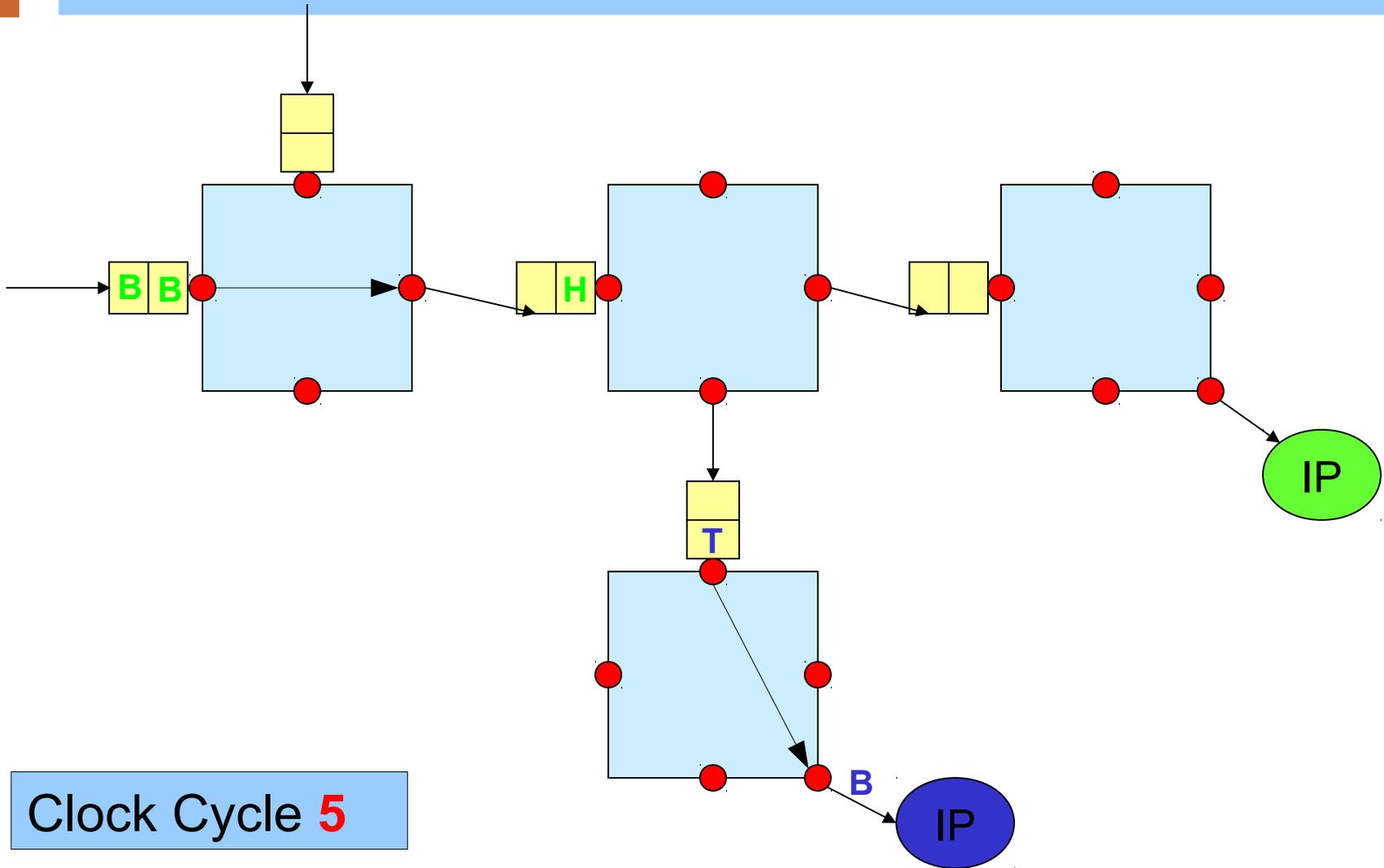
Wormhole - Example



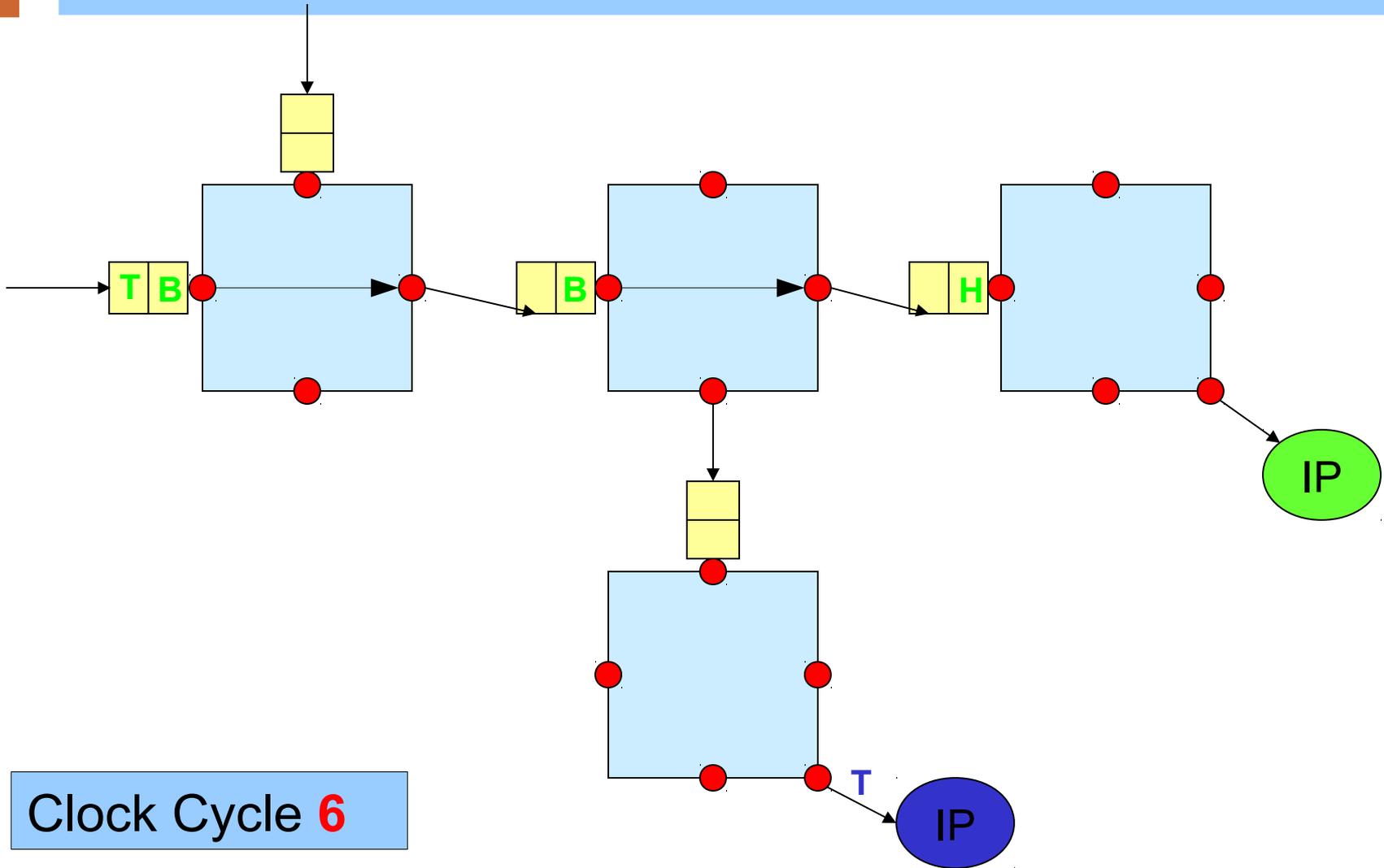
Wormhole - Example



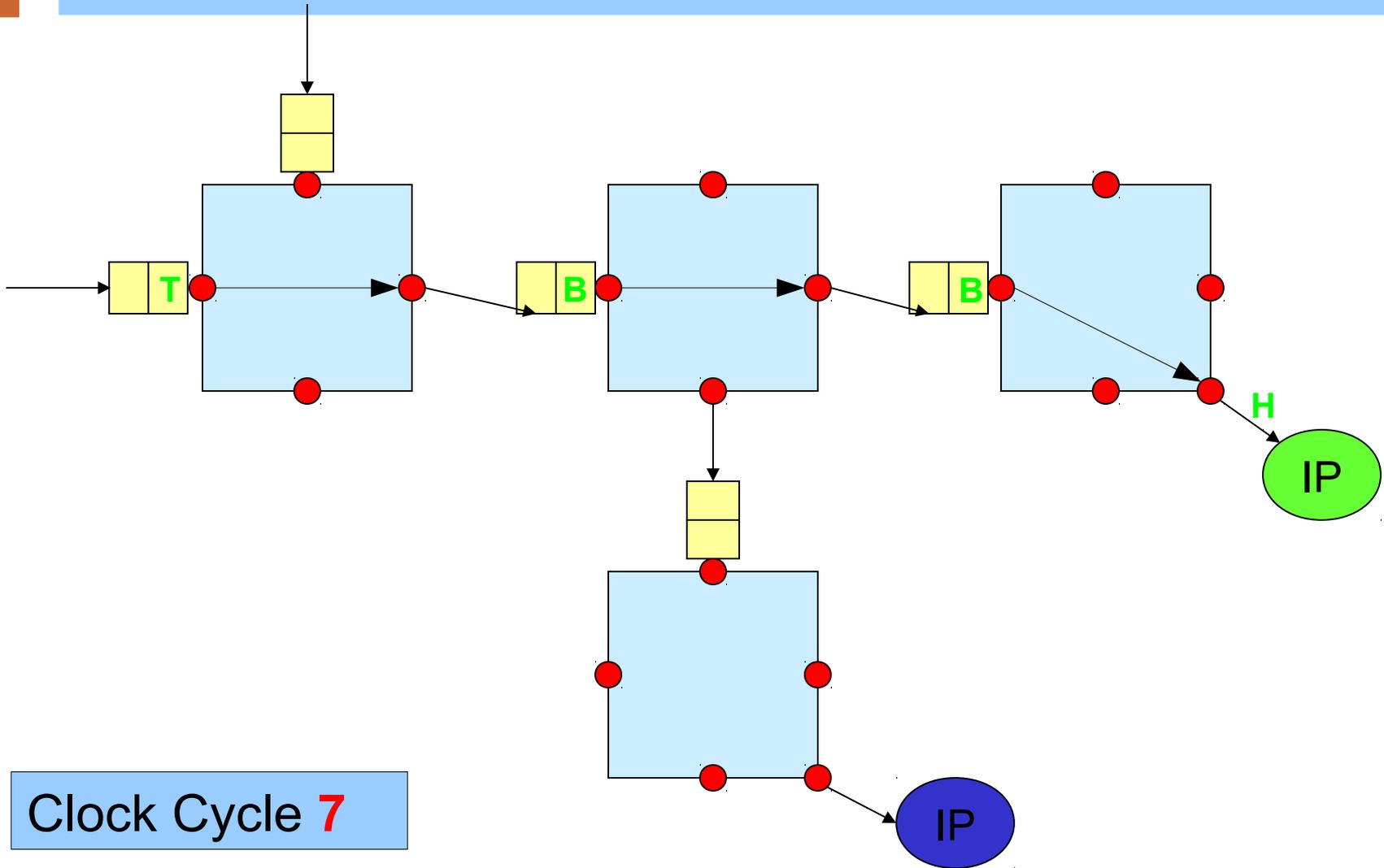
Wormhole - Example



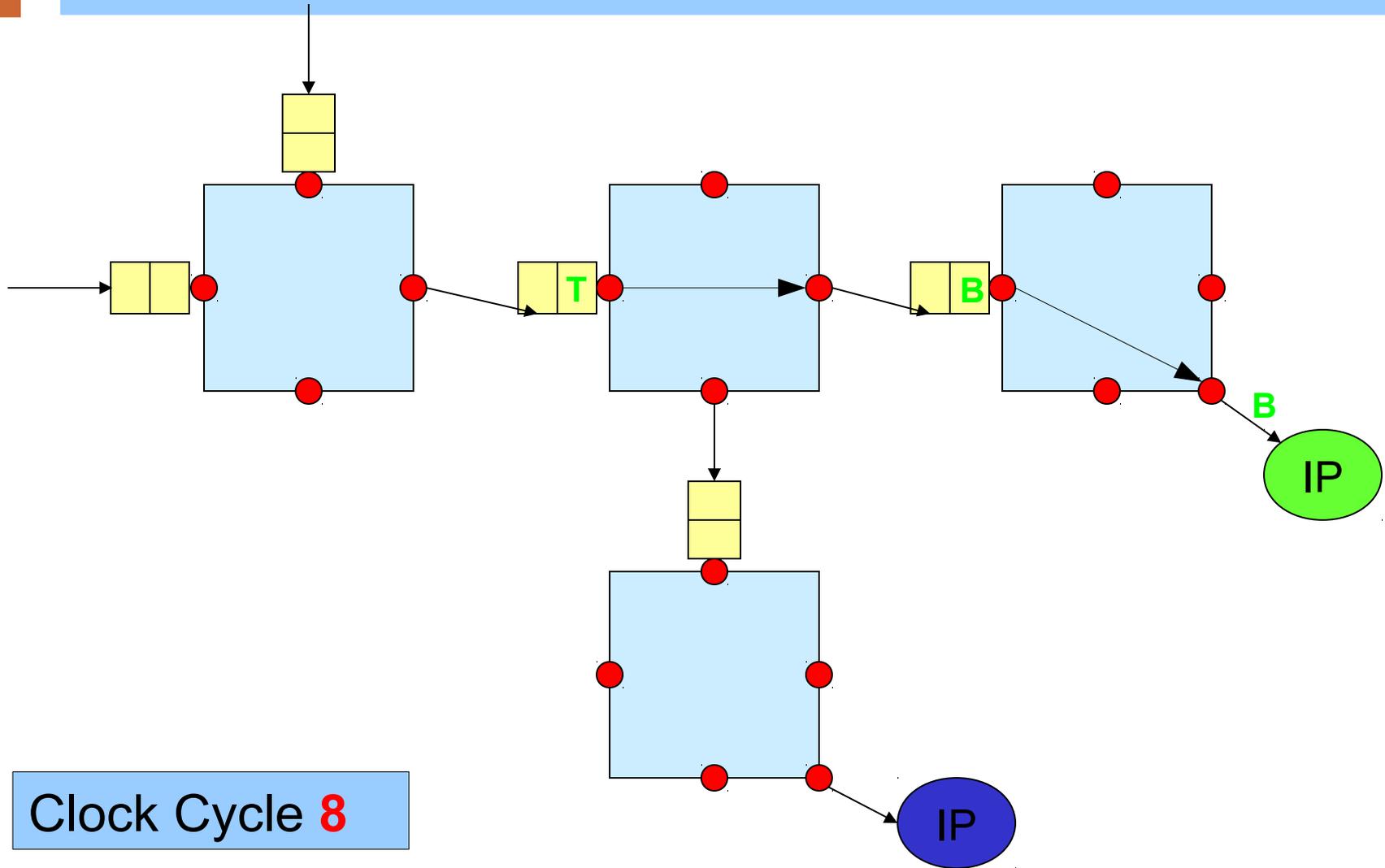
Wormhole - Example



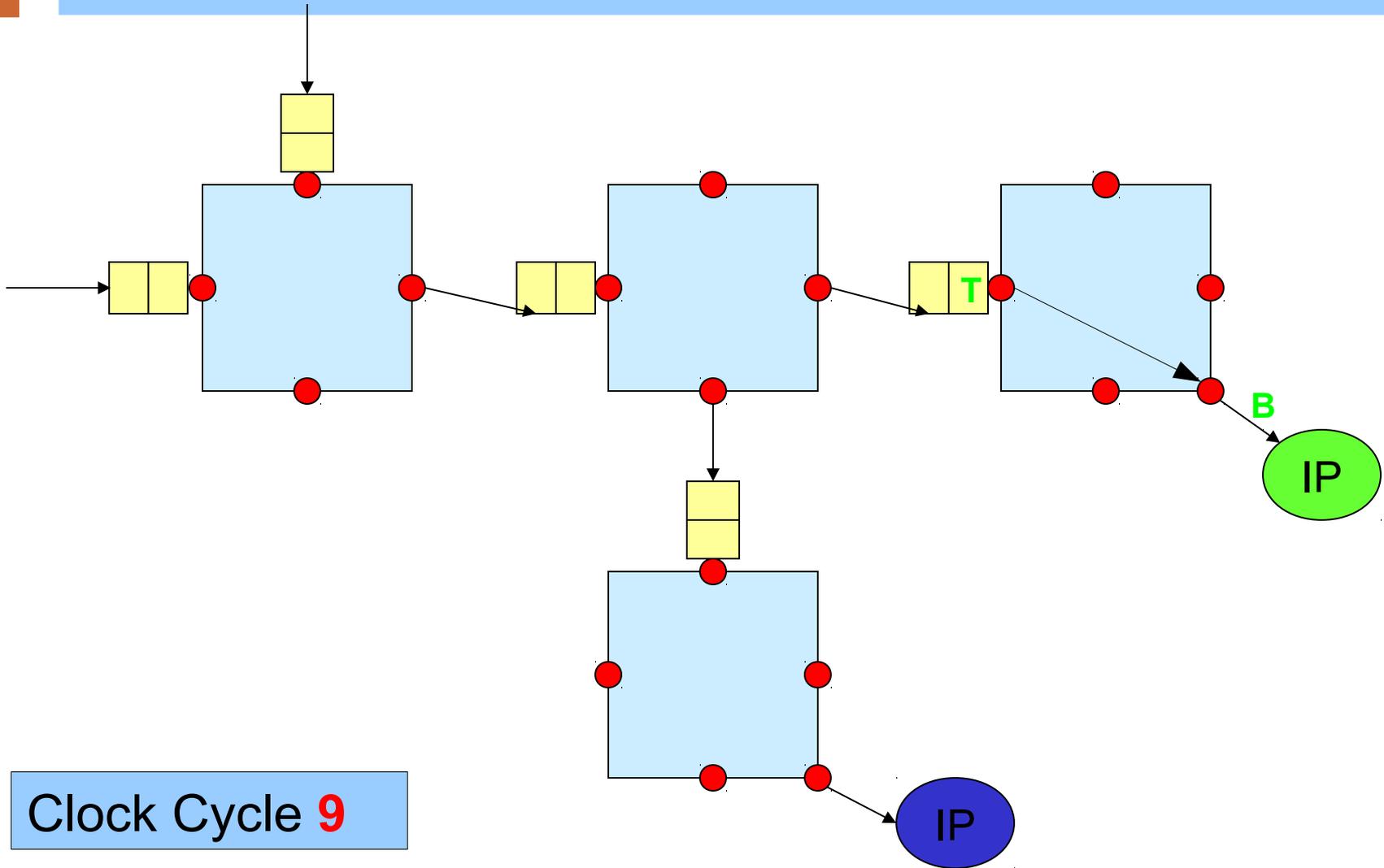
Wormhole - Example



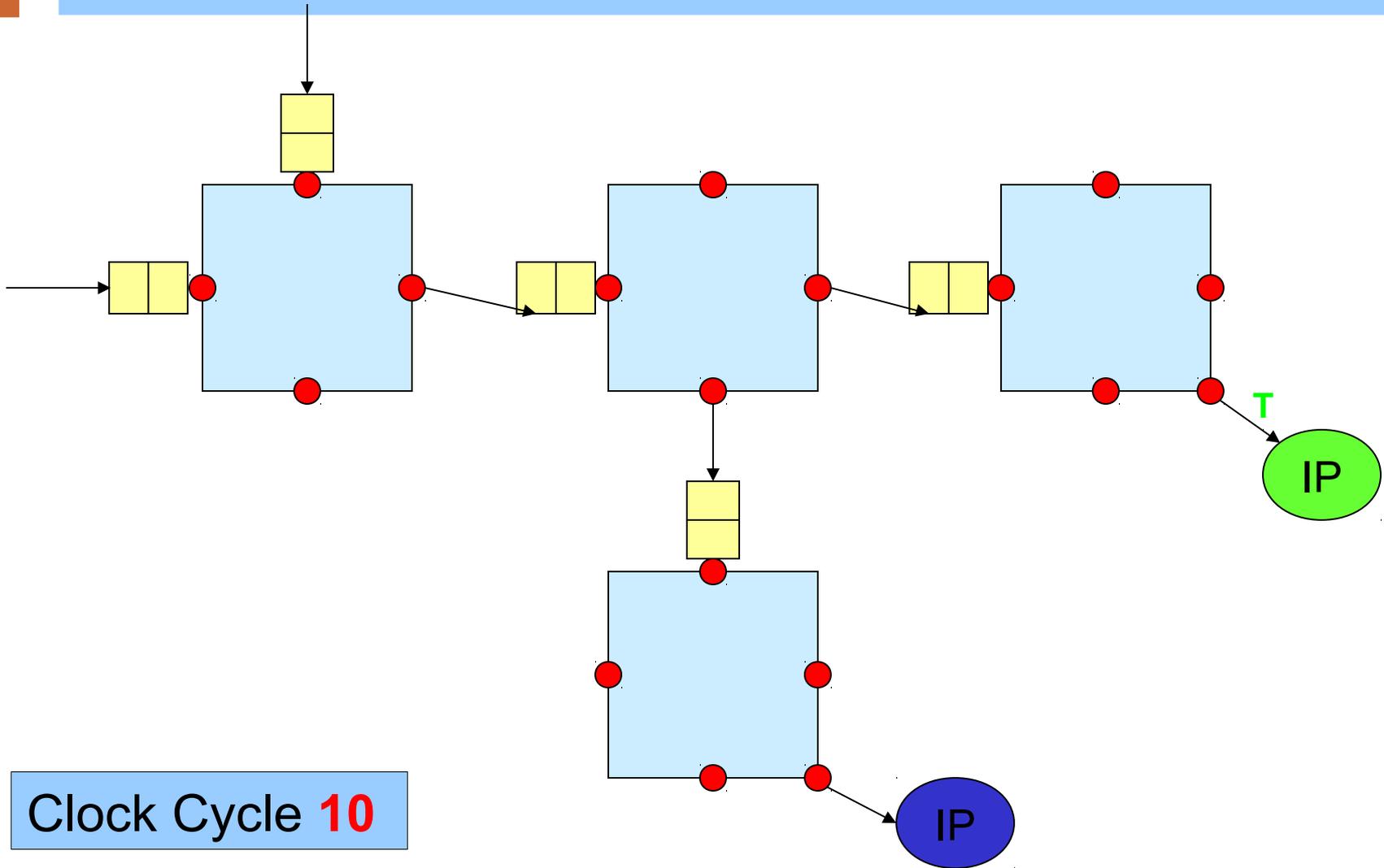
Wormhole - Example



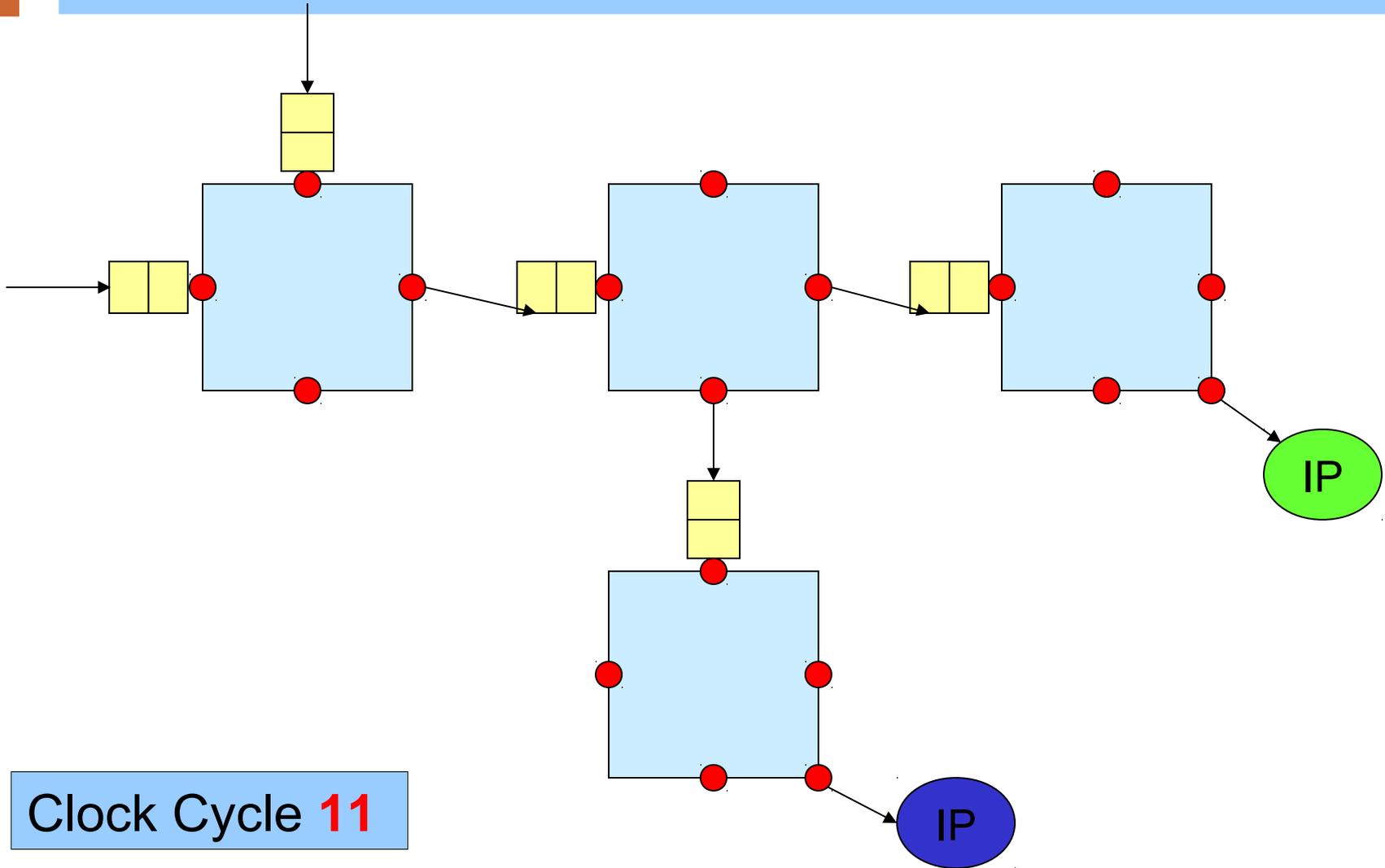
Wormhole - Example



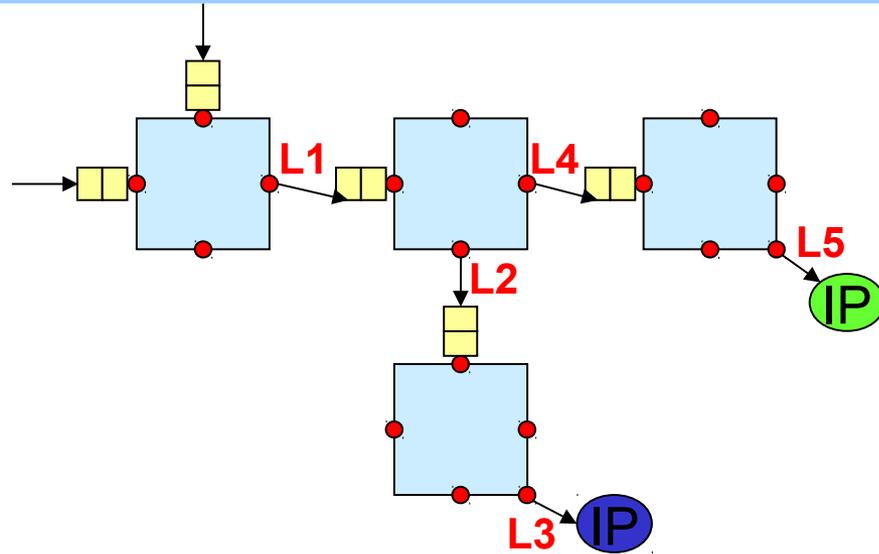
Wormhole - Example



Wormhole - Example



Wormhole - Example



CC	0	1	2	3	4	5	6	7	8	9	10	11
Blue	Injected	L1	L1, L2	L1, L2, L3	L1, L2, L3	L2, L3	L3	Drained				
Green		Injected				L1	L1, L4	L1, L4, L5	L1, L4, L5	L4, L5	L5	Drained

■ Blue packet

- Injected at CC 0
- Delivered at CC 7
- Latency 7 clock cycles

■ Green packet

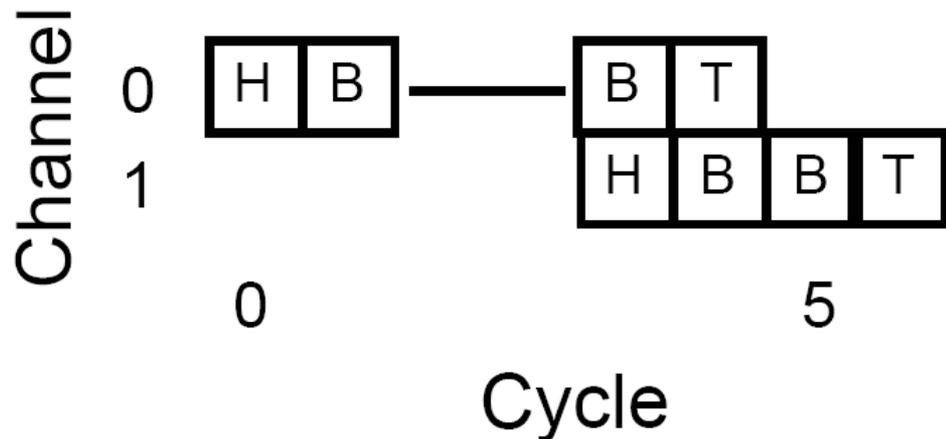
- Injected at CC 1
- Delivered at CC 11
- Latency 10 clock cycles

Wormhole Flow Control

■ Comparison to cut-through

- Wormhole flow control makes far more efficient use of buffer space
- Throughput maybe less, since wormhole flow control may block a channels mid-packets

Wormhole Flow Control

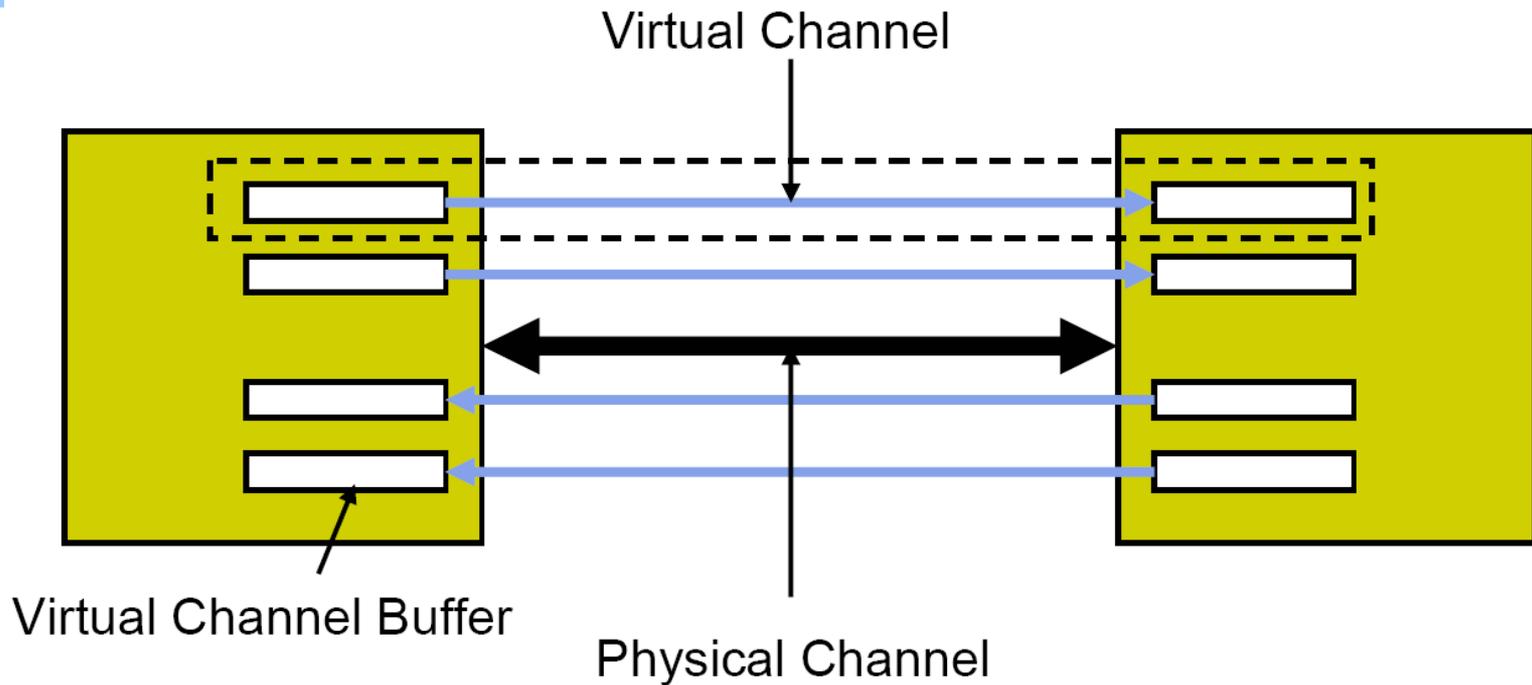


- The main advantage of wormhole to cut-through is that buffers in the routers do not need to be able to hold full packets, but only need to store a number of flits
- This allows to use smaller and faster routers

Virtual Channel Flow Control

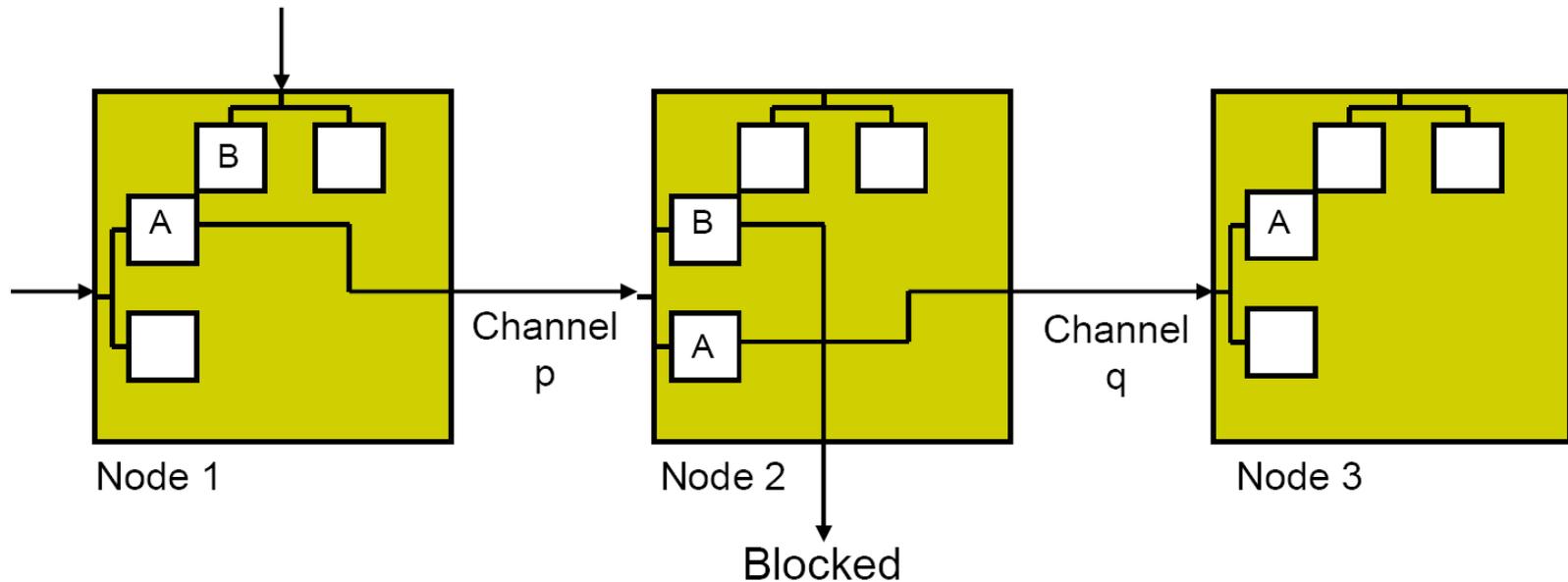
- In virtual channel flow-control several channels are associated with a single physical channel
- This allows to use the bandwidth that otherwise is left idle when a packet blocks the channel
- Unlike wormhole flow control subsequent flits are not guaranteed bandwidth, since they have to compete for bandwidth with other flits

Concept of Virtual Channels



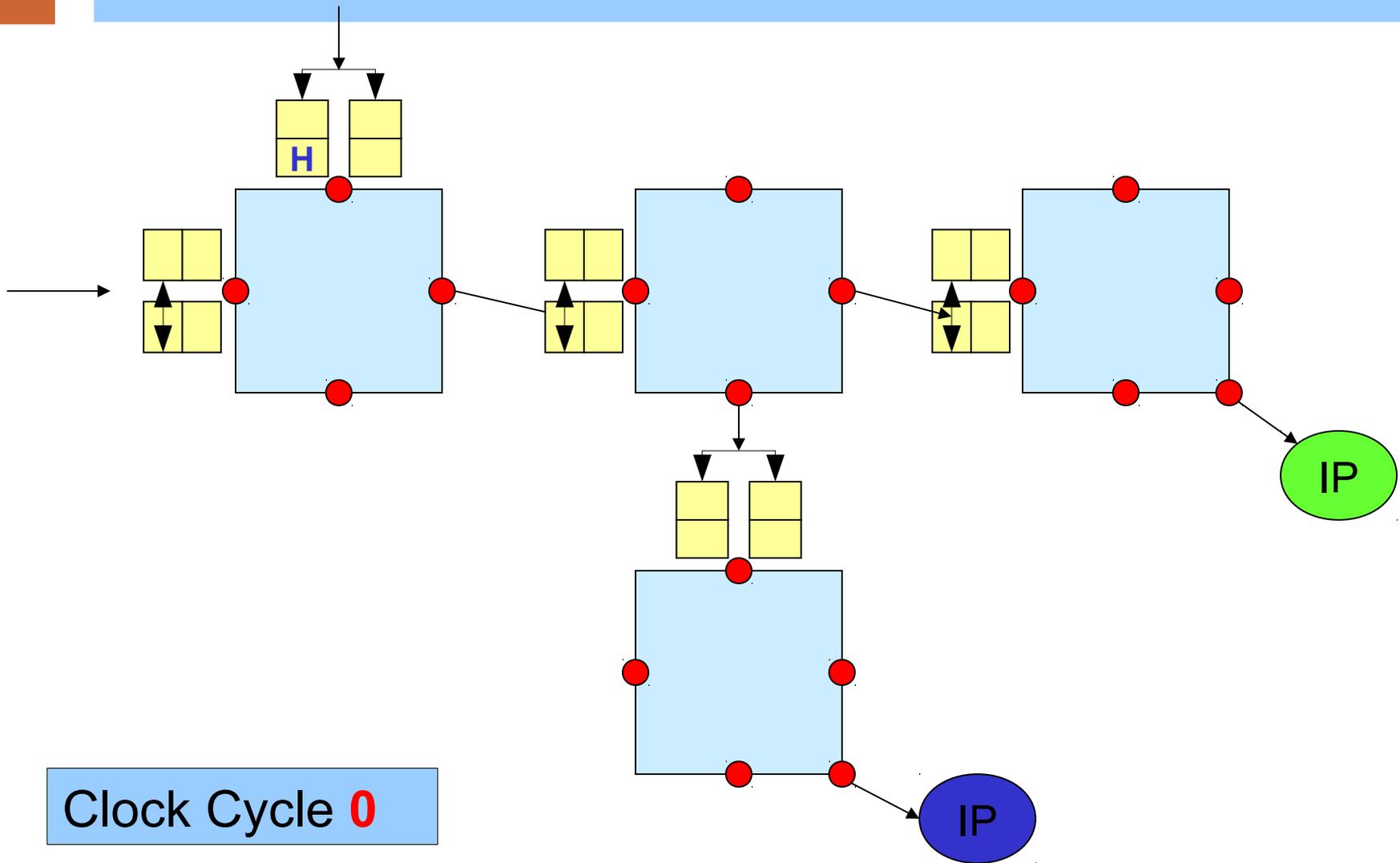
- A physical channel is shared by several virtual channels
- Naturally the speed of each virtual channel connection is reduced

Virtual Channel Flow Control

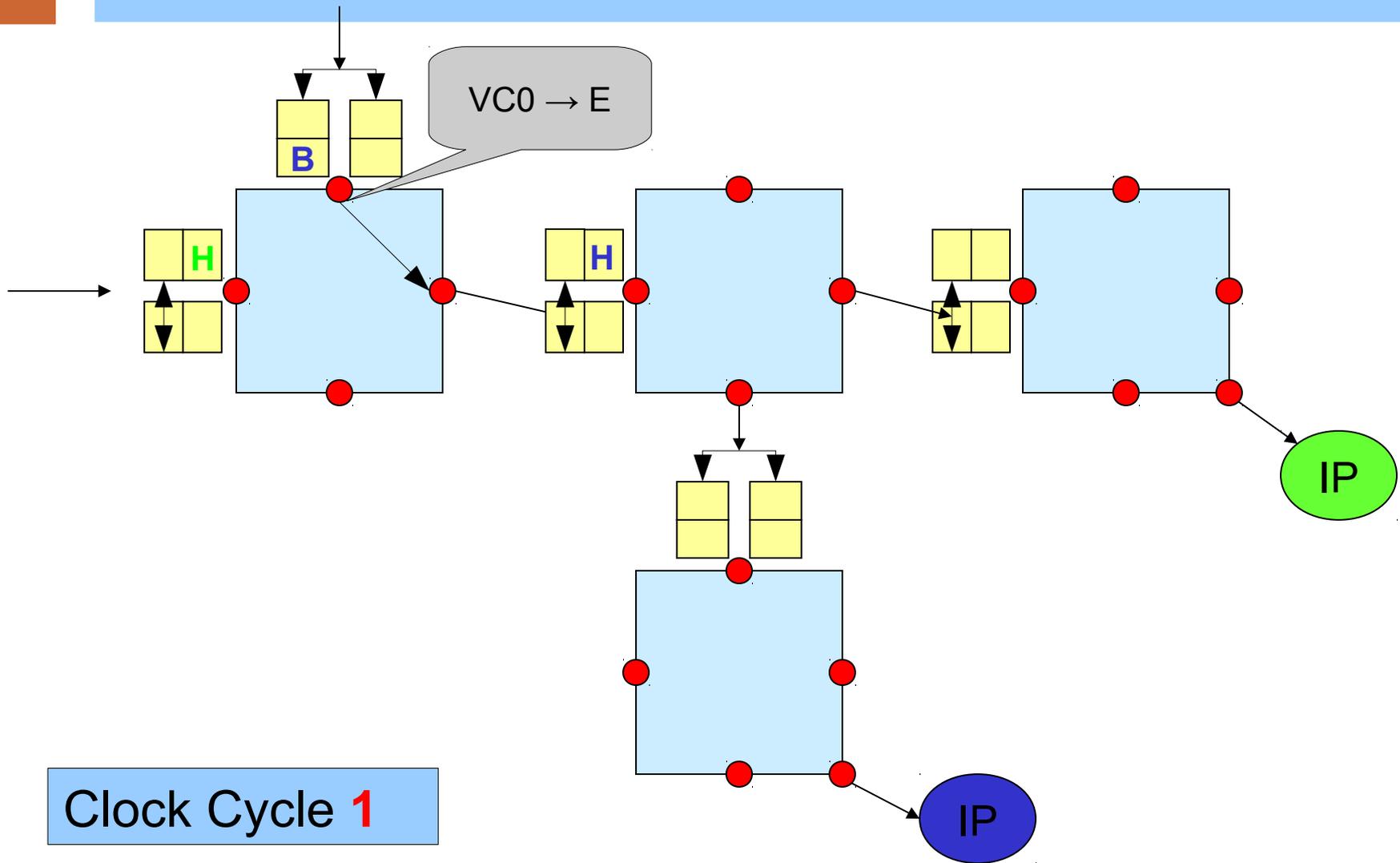


- There are several virtual channels for each physical channel
- Packet *A* can use a second virtual channel and thus proceed over channel *p* and *q*

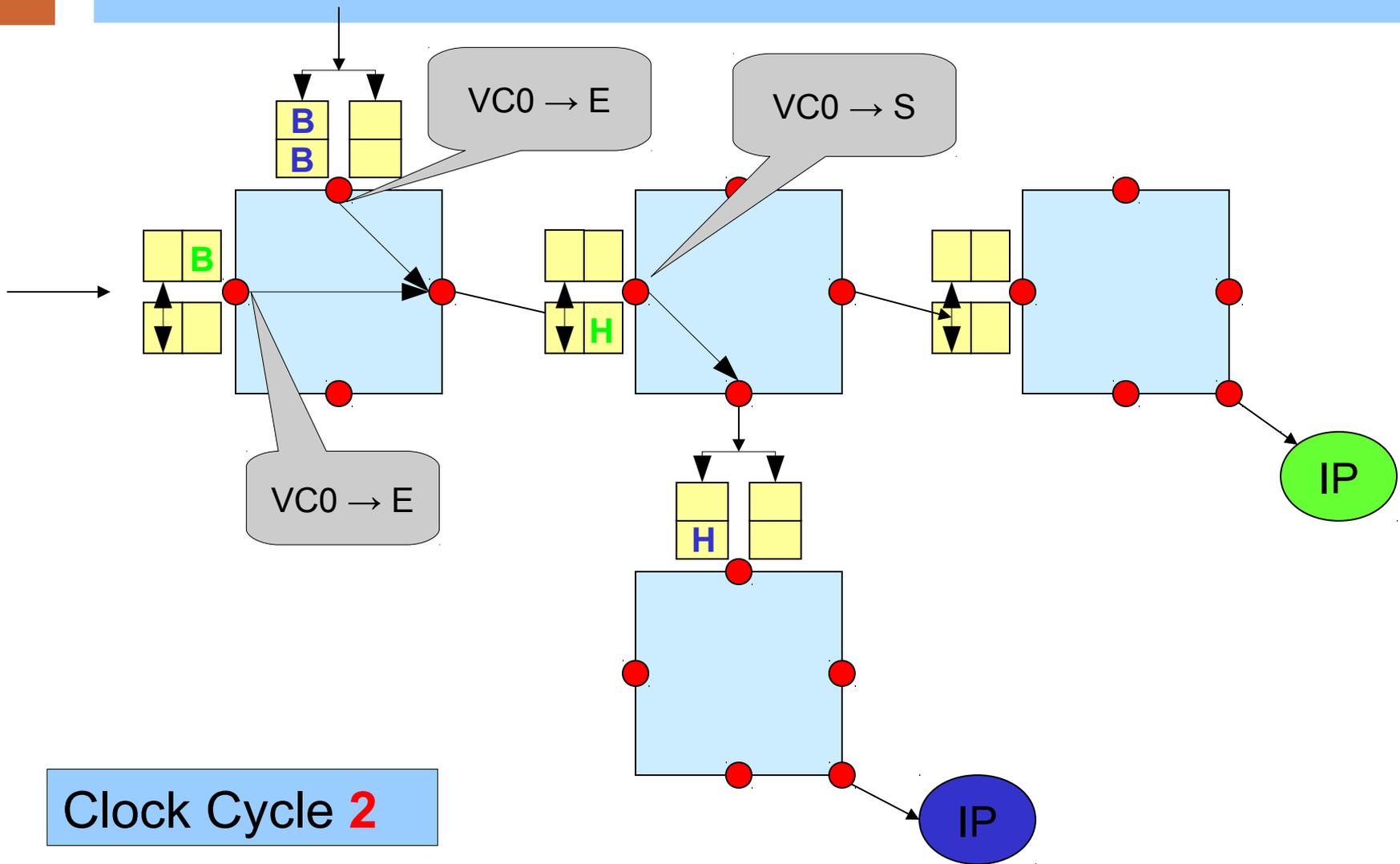
Virtual Channels - Example



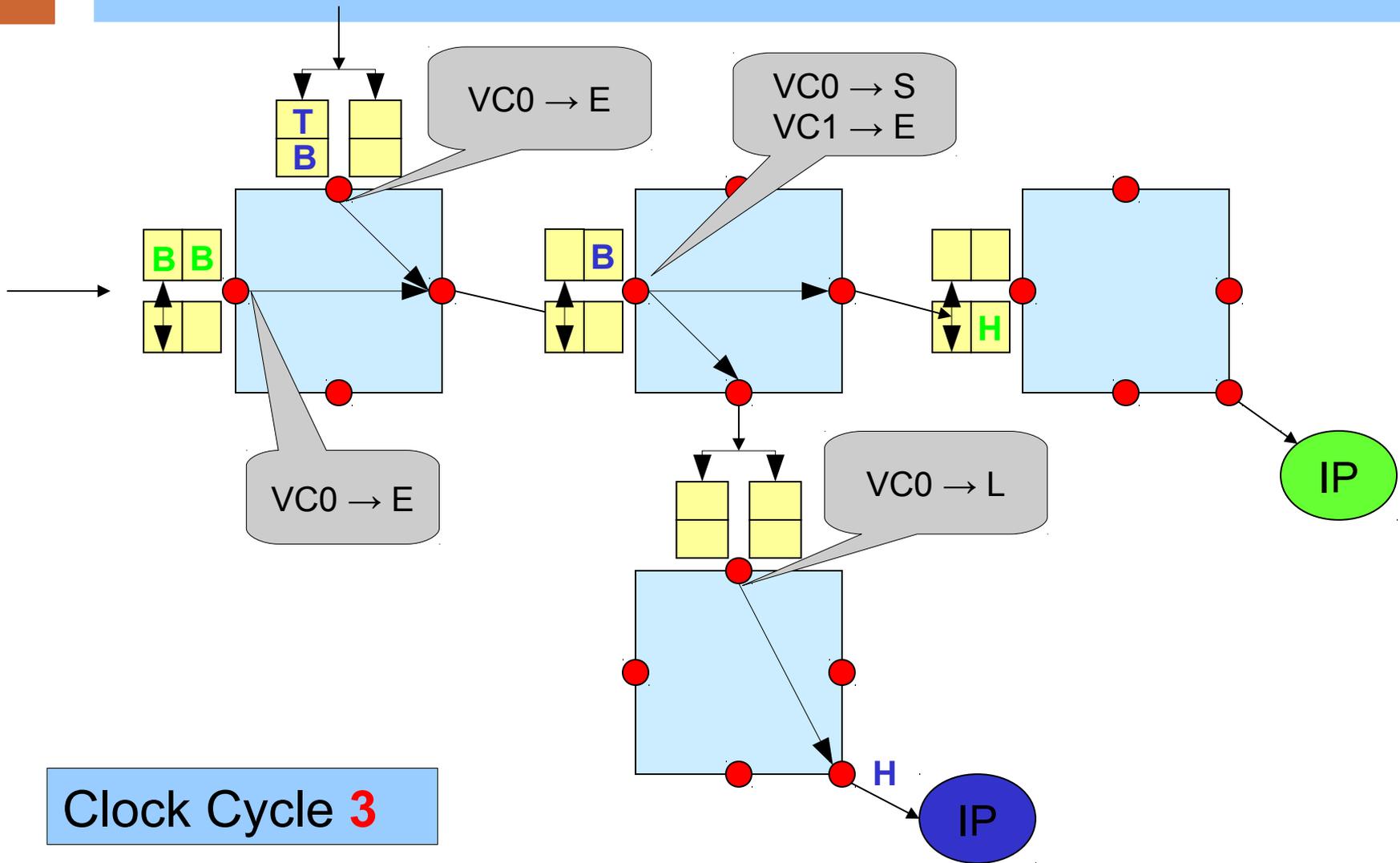
Virtual Channels - Example



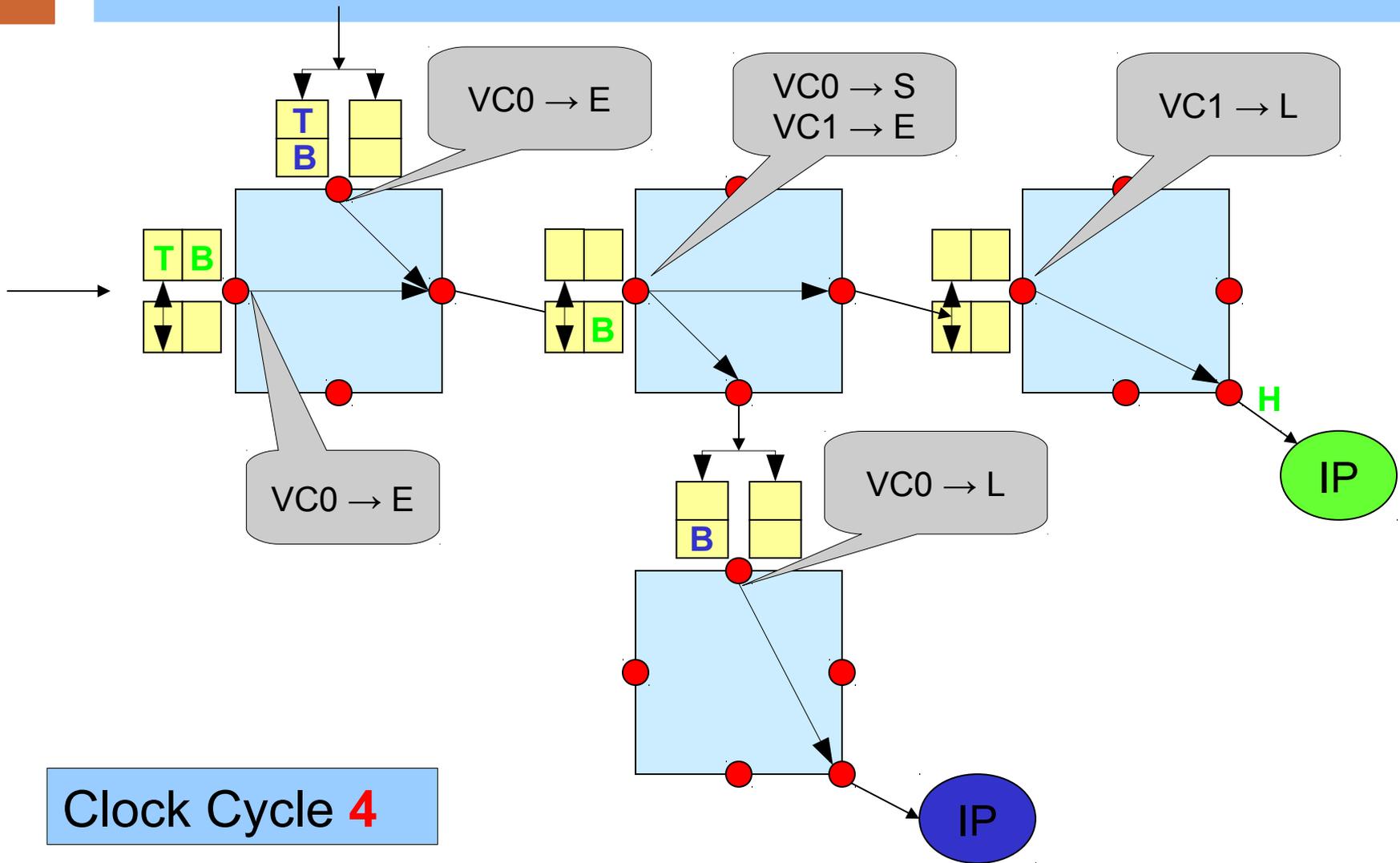
Virtual Channels - Example



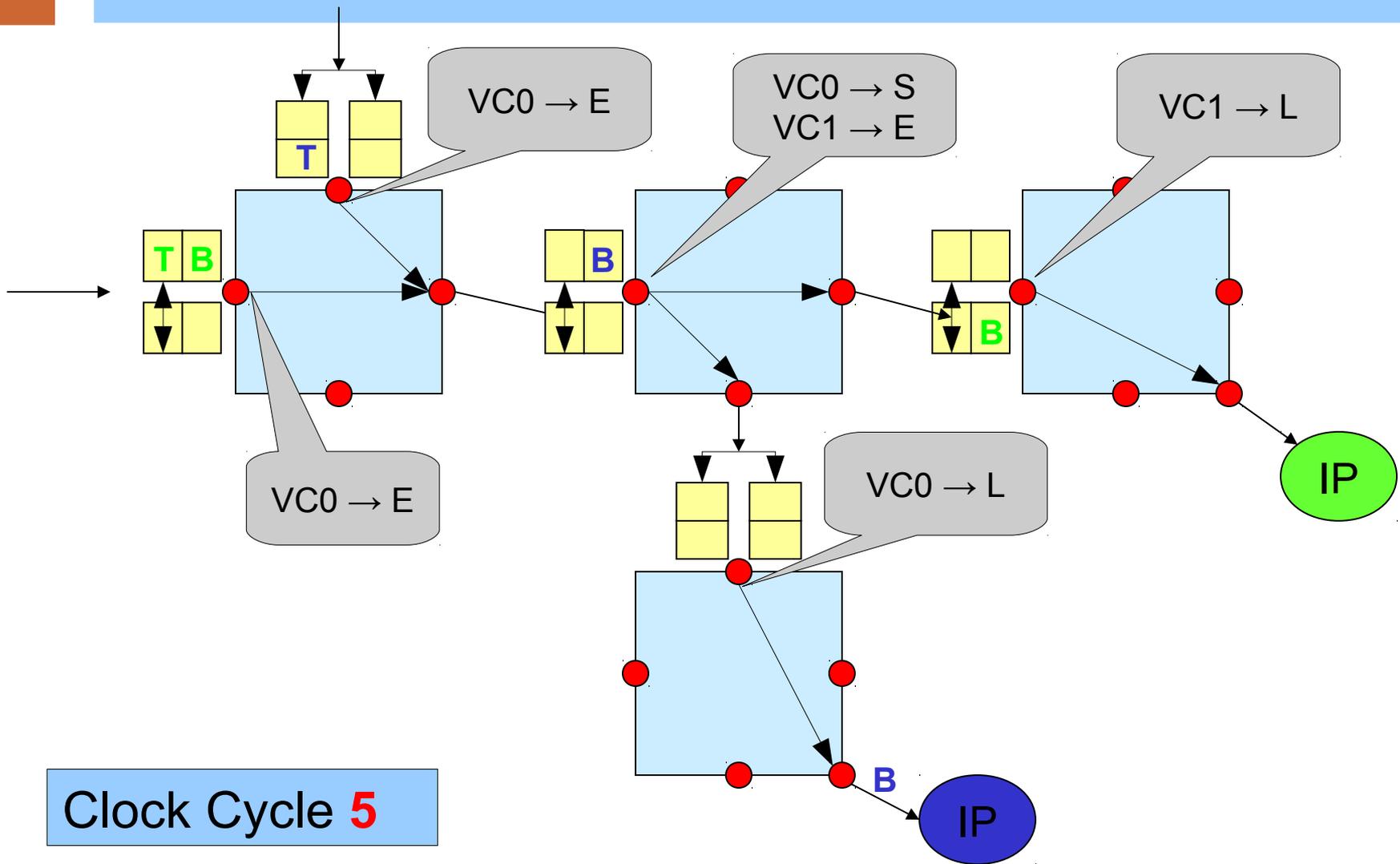
Virtual Channels - Example



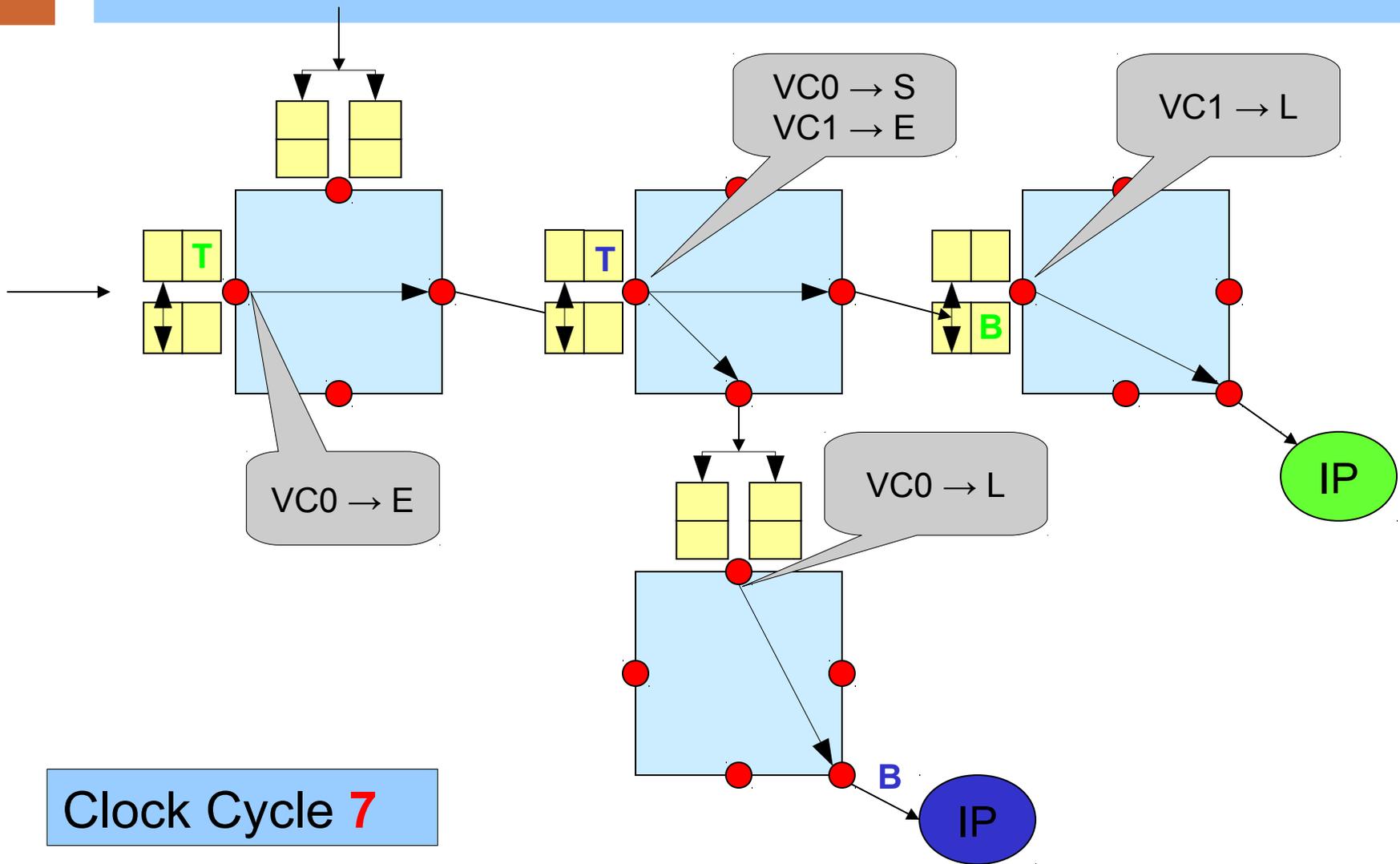
Virtual Channels - Example



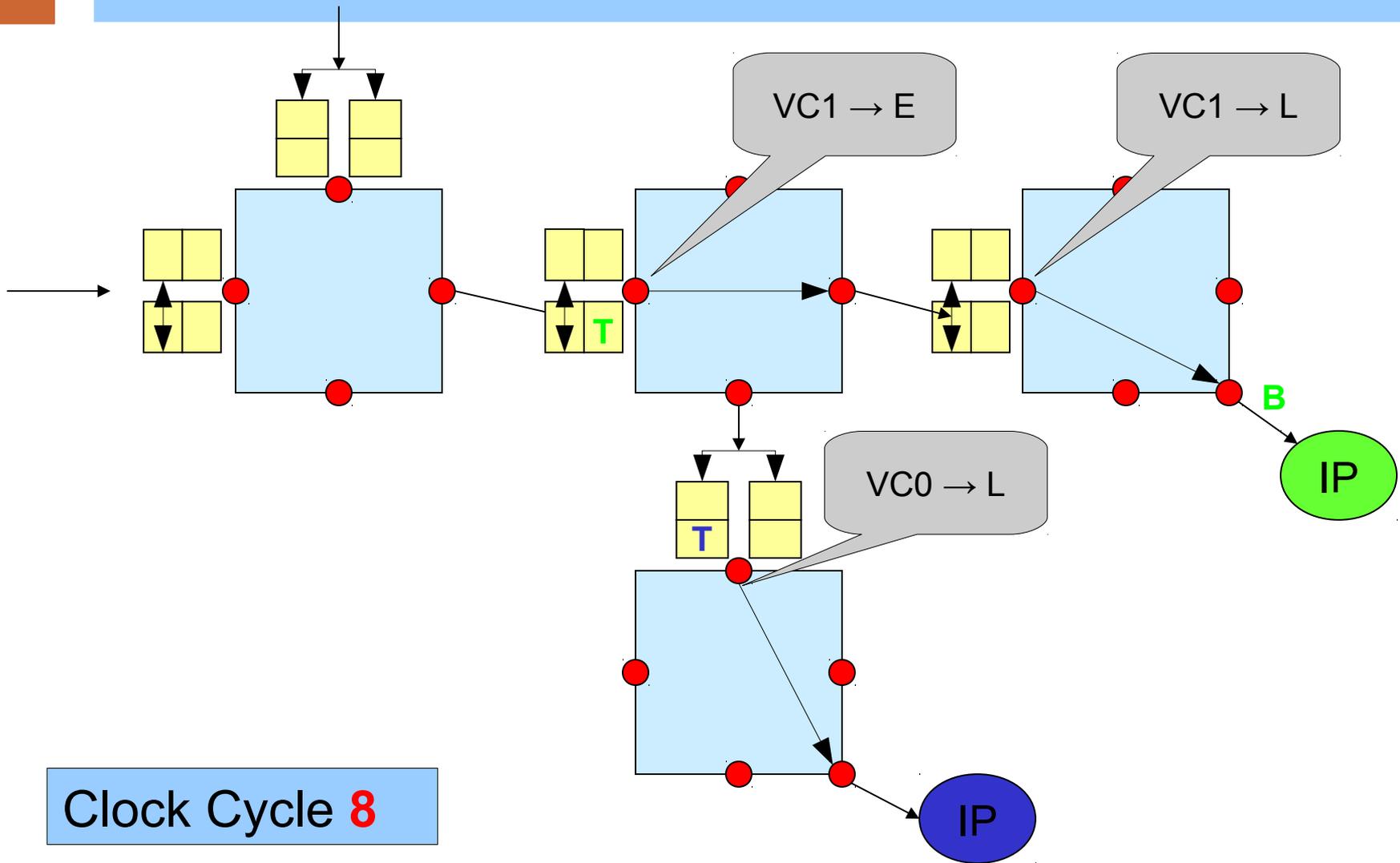
Virtual Channels - Example



Virtual Channels - Example

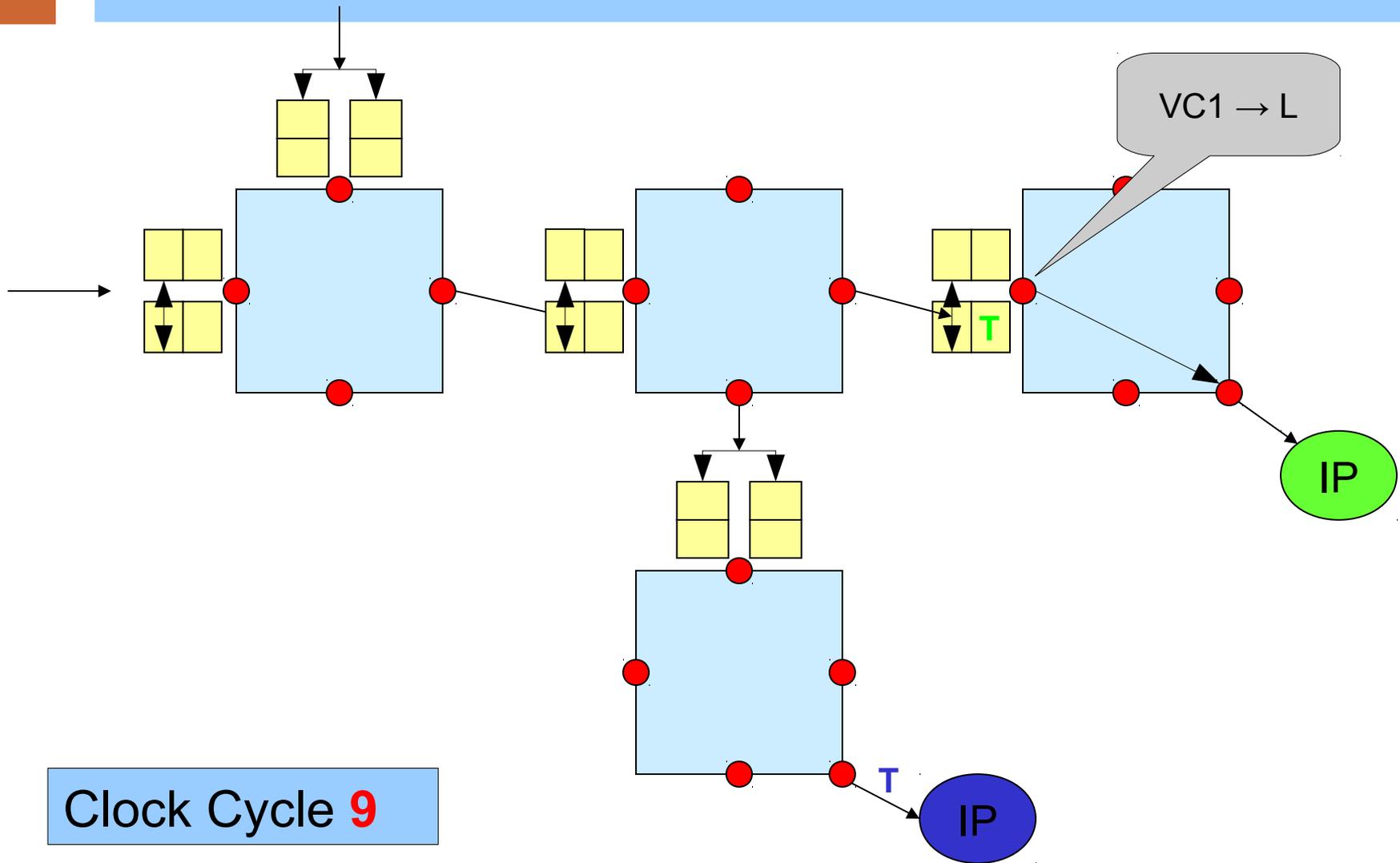


Virtual Channels - Example

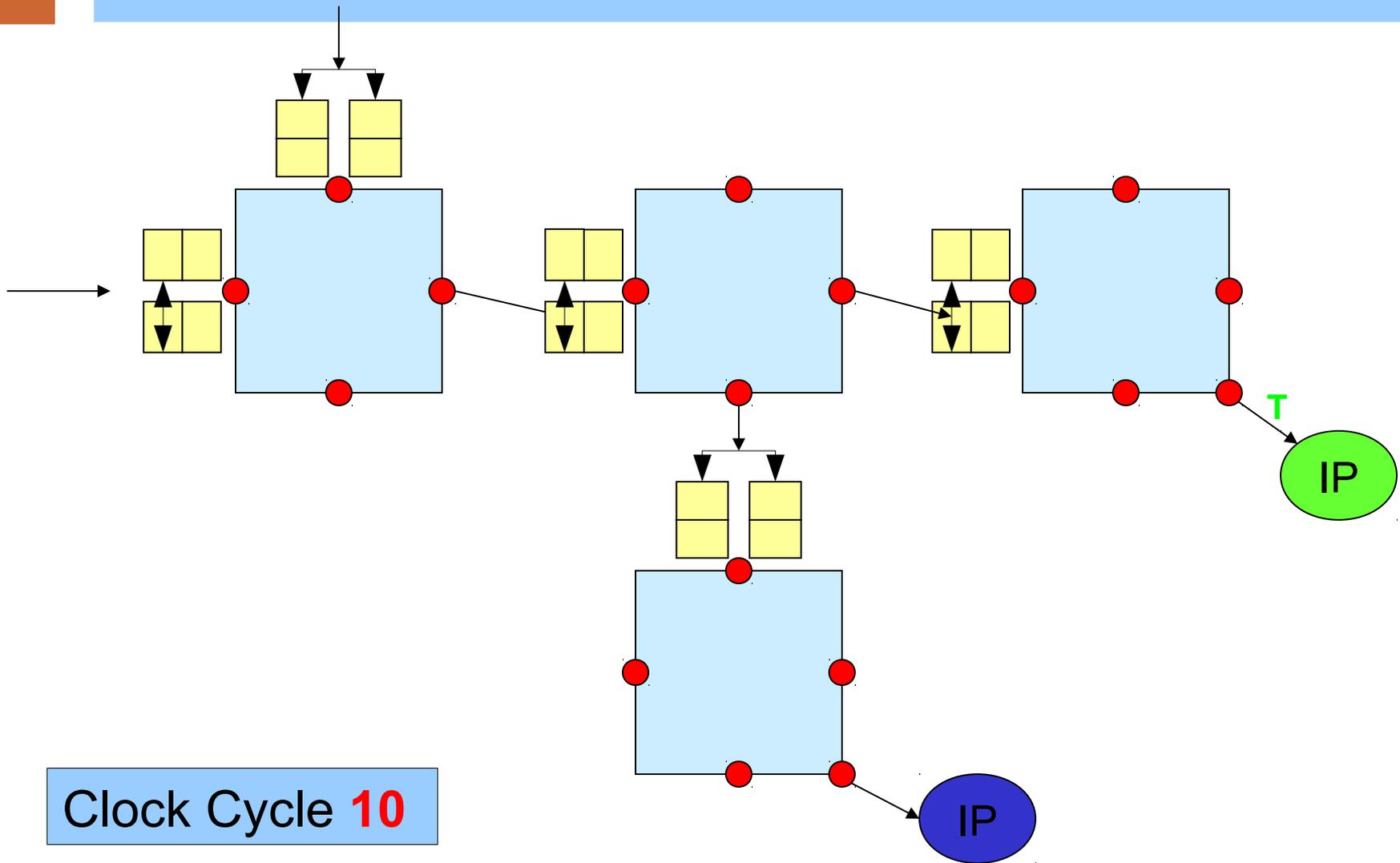


Clock Cycle **8**

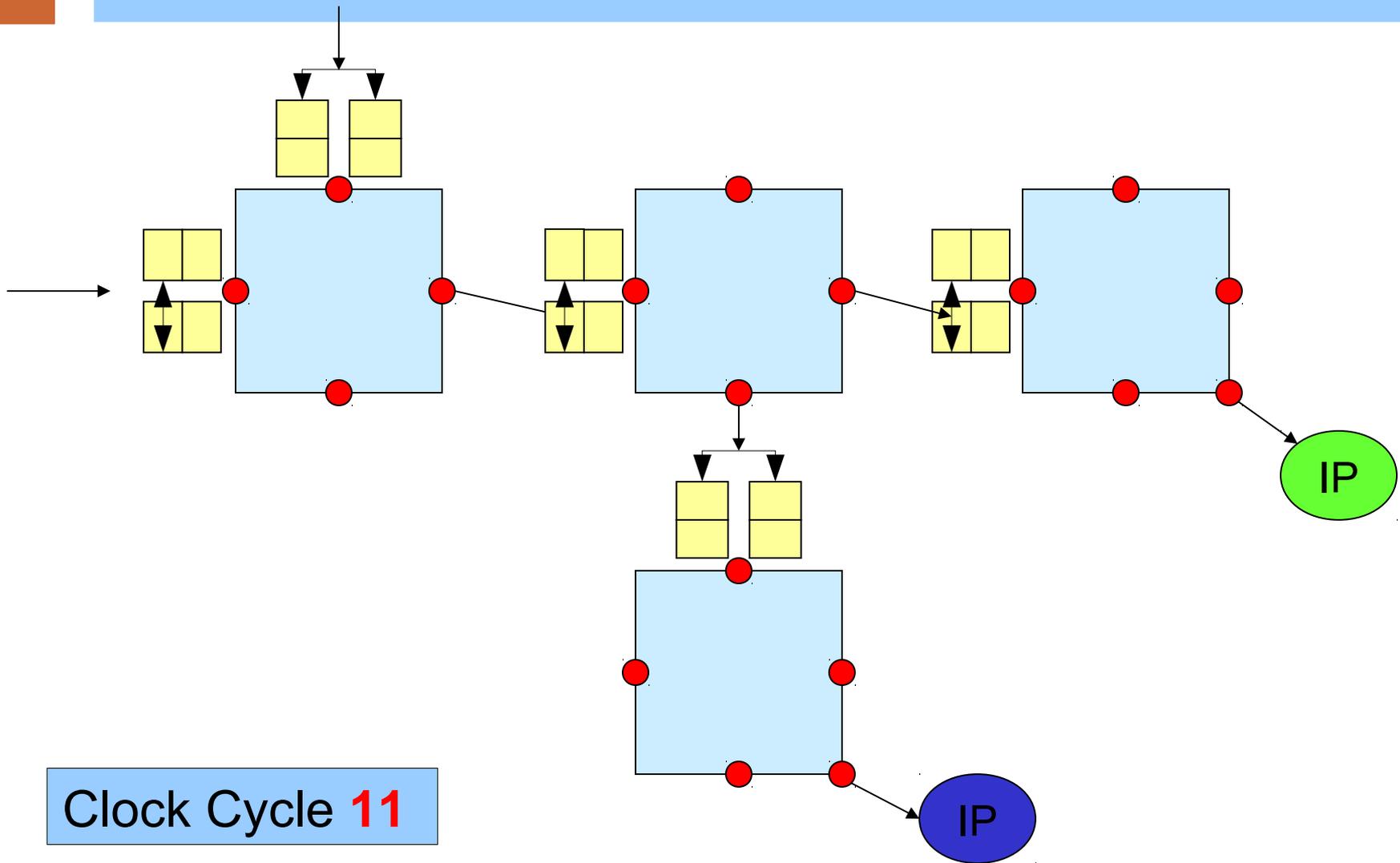
Virtual Channels - Example



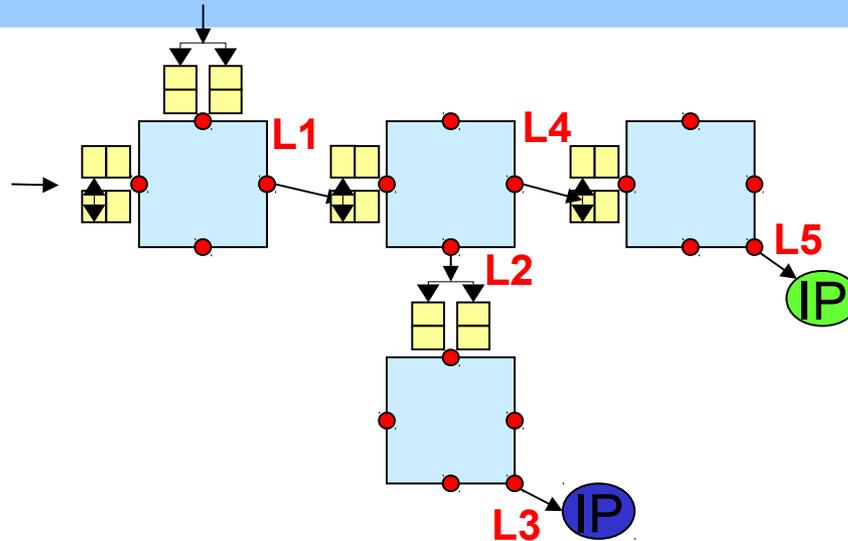
Virtual Channels - Example



Virtual Channels - Example



Virtual Channels - Example



CC	0	1	2	3	4	5	6	7	8	9	10	11
Blue	Injected	L1	L2	L1, L3	L2	L1, L3	L2	L1, L3	L2	L3	Drained	
Green		Injected	L1	L4	L1, L5	L4	L1, L5	L4	L1, L5	L4	L5	Drained

■ Blue packet

- Injected at CC 0
- Delivered at CC 10
- Latency 10 clock cycles

■ Green packet

- Injected at CC 1
- Delivered at CC 11
- Latency 10 clock cycles