Shared Memory switches

Masoud Sabaei

Associate professor

Department of Computer Engineering,
Amirkabir University of Technology

Shared Memory Switches

- Introduction
  - A common shared memory for all inputs and outputs
  - Every time slot:
    - Input ports store incoming cells
    - Output ports retrieve outgoing cells
  - Work as output buffered switch
    - Optimal throughput
    - Optimal delay performance
  - A shared buffer → Centralized memory management → Switch size limitation:

Cell Slot

$$\text{memory access cycle} \le \frac{\text{cell length}}{2 \cdot N \cdot \text{link speed}}$$

Shared-Memory Switches

For instance, with a

cell time slot of 2.83 μs,

(53-byte cells at the line rate of 149.76 Mbit/s, or 155.52 Mbit/s × 26/27)

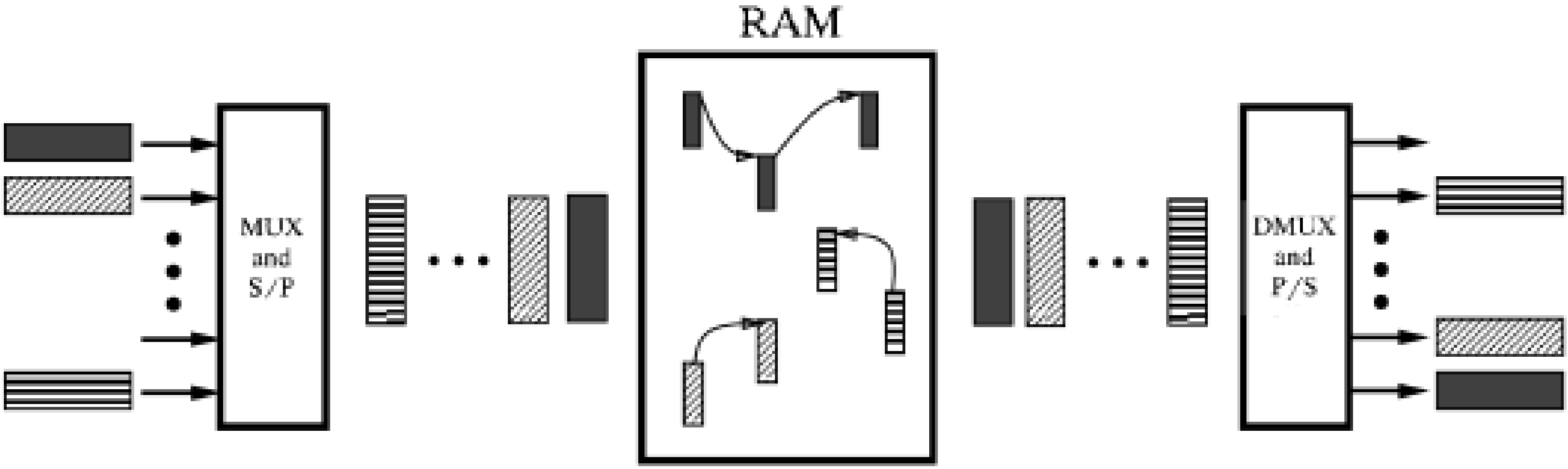
and with a

memory cycle time of 10 ns,

the **switch size** is limited to **141**

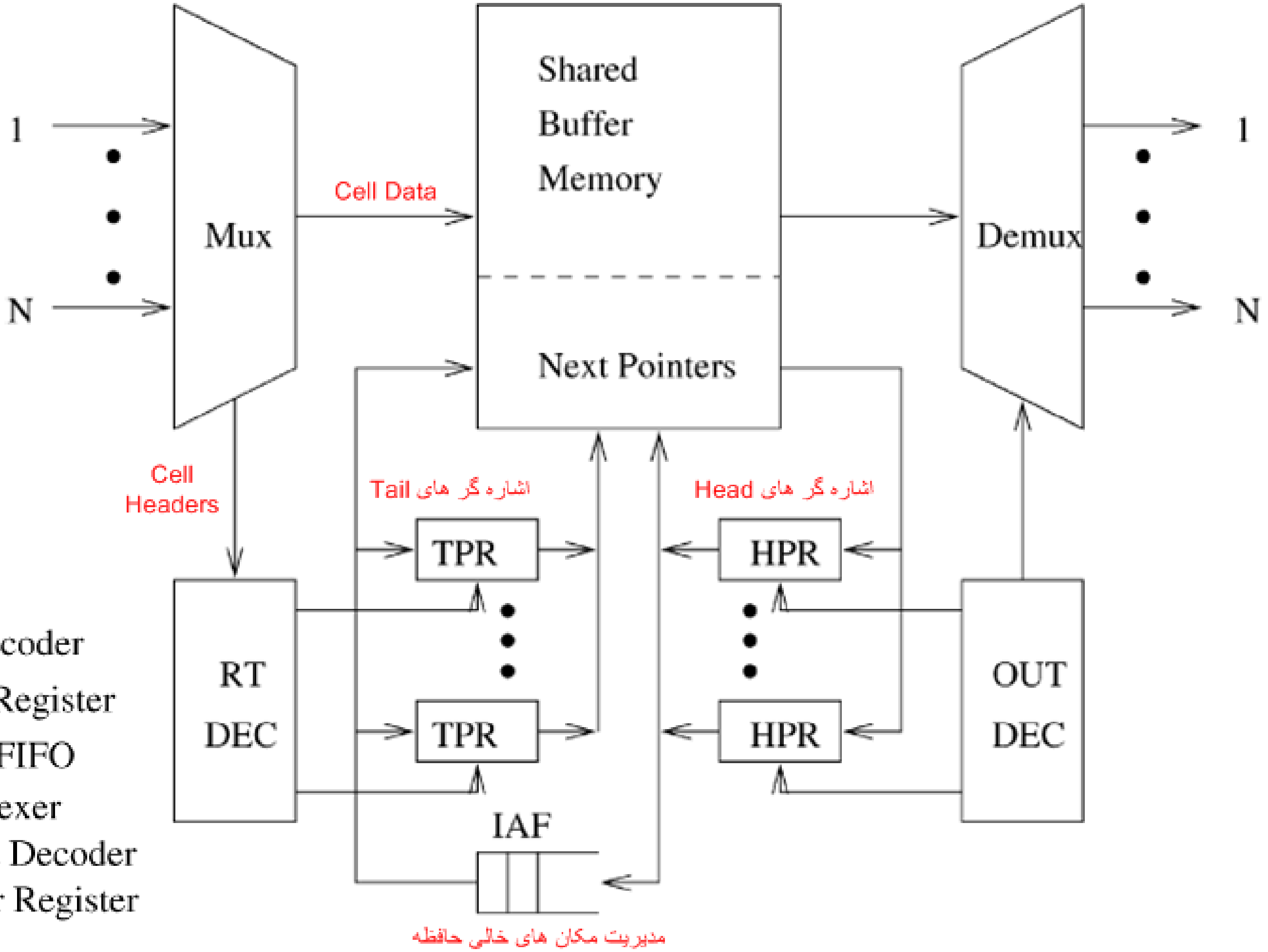
Using Linked Lists

Logical queues in a shared-memory switches



Using Linked Lists

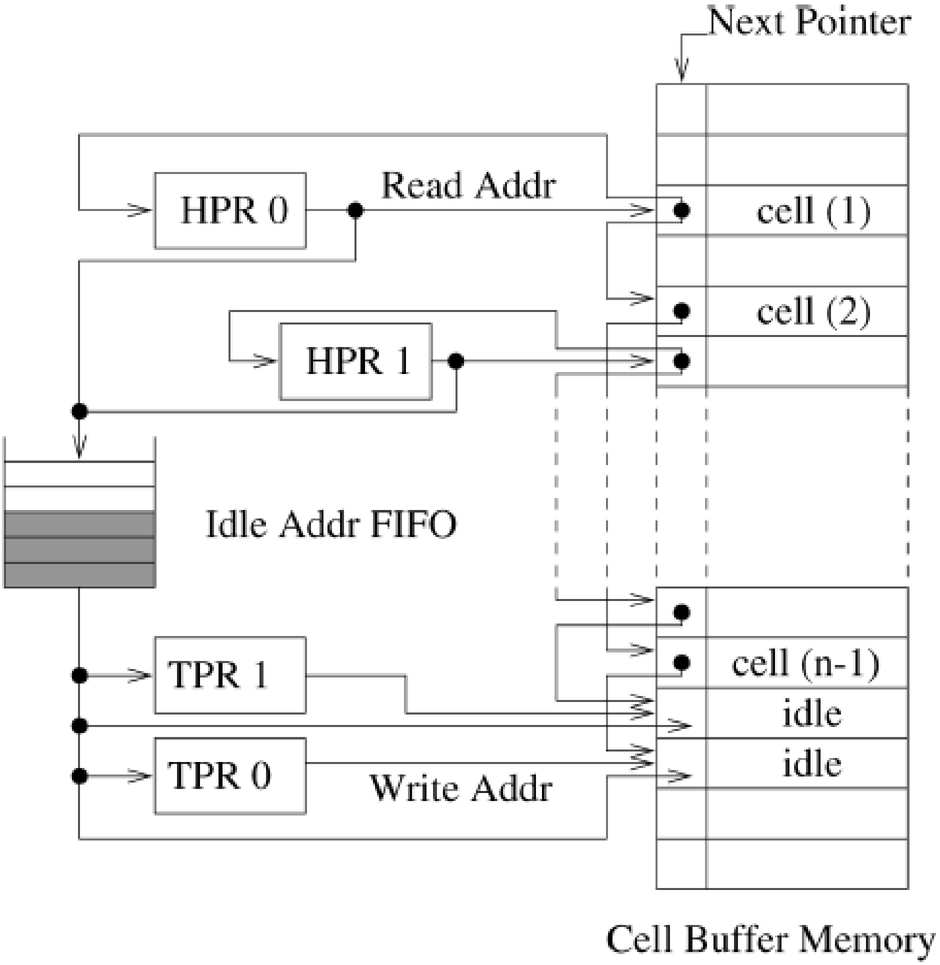
Basic structure of a linked-list-based shared-memory switches



Mux: Multiplexer
RT DEC: Route decoder
TPR: Tail Pointer Register
IAF: Idle Address FIFO
Demux: Demultiplexer
OUT DEC: Output Decoder
HPR: Head Pointer Register

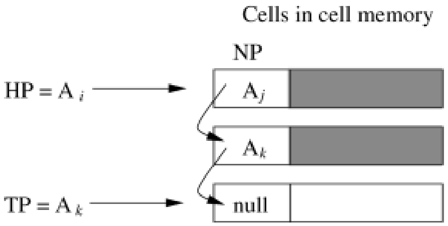
Using Linked Lists

- Linked list structure



Using Linked Lists

- Insertion/Deletion
  - Two different ways:

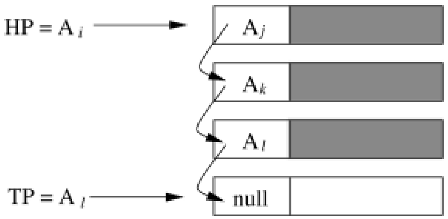


HP : Head Pointer

TP : Tail Pointer

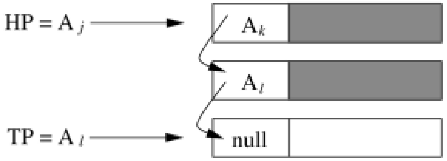
NP : Next Pointer

(a) Original logical queue



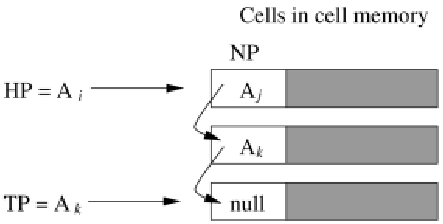
1. Access TP to get A_k
2. Store the arrived cell at A_k
3. Access IAF to get A_l
4. Update NP pointed by TP with A_l
5. Update TP with A_l

(b) Add a cell to the logical queue in (a)



1. Access HP to get A_i
2. Read cell at A_i
3. Store A_i to IAF
4. Access NP pointed by HP to get A_j
5. Update HP with A_j

(c) Delete HOL cell from the logical queue in (b)

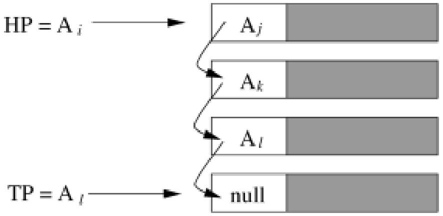


HP : Head Pointer

TP : Tail Pointer

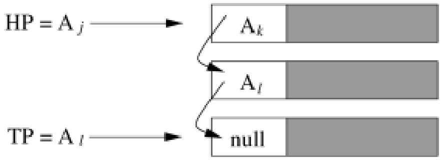
NP : Next Pointer

(a) Original logical queue



1. Access TP to get A_k
2. Access IAF to get A_l
3. Store the arrived cell at A_l
4. Update NP of cell at TP=A_k with A_l
5. Update TP with A_l

(b) Add a cell to the logical queue in (a)

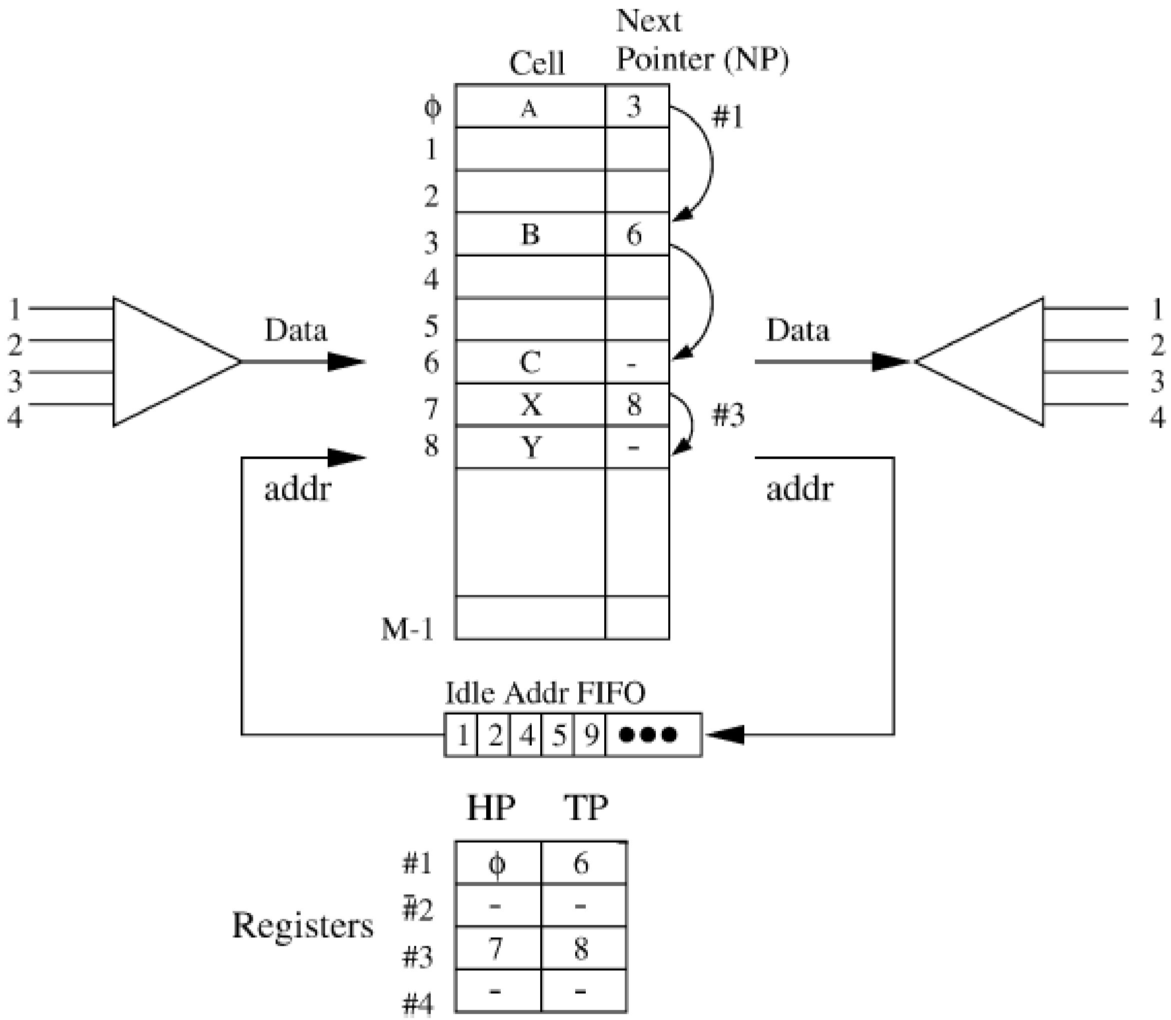


1. Access HP to get A_i
2. Read cell at A_i
3. Store A_i to IAF
4. Access NP pointed by HP to get A_j
5. Update HP with A_j

(c) Delete HOL cell from the logical queue in (b)

Using Linked Lists

- Example: a 4*4 switch



	Cell	Next Pointer (NP)
ϕ	A	3
1		
2		
3	B	6
4		
5		
6	C	-
7	X	8
8	Y	-
M-1		

Idle Addr FIFO

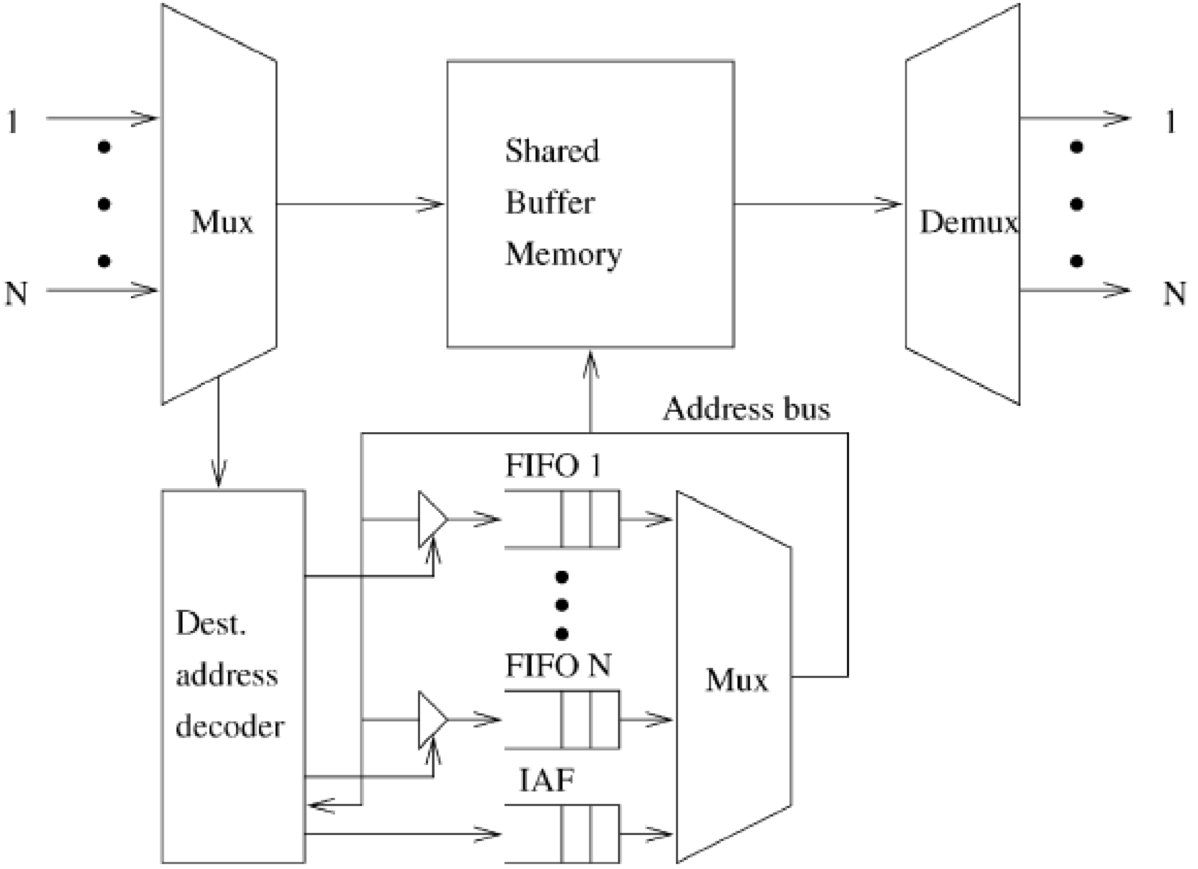
1	2	4	5	9	●●●

Registers

	HP	TP
#1	ϕ	6
#2	-	-
#3	7	8
#4	-	-

Using Linked Lists

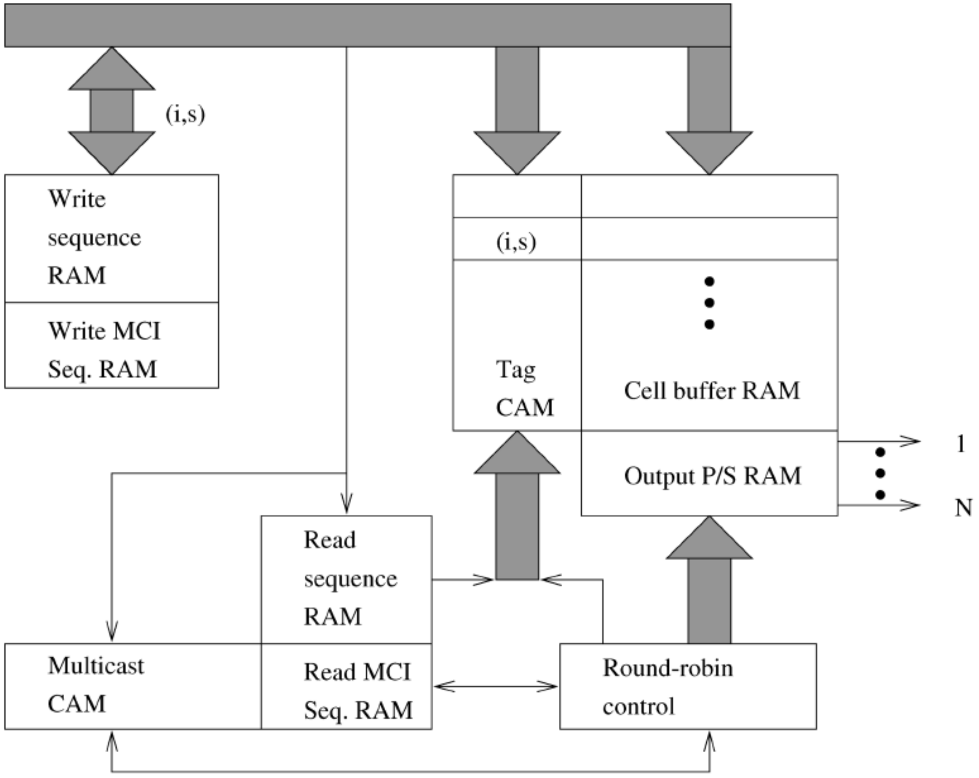
- Using Dedicated FIFOs



Mux: Multiplexer
Demux: Demultiplexer
IAF: Idle address FIFO

Using CAM

- Using CAM
  - RAM stores the cell
  - CAM stores a tag
- Unique tag for each cell: (i,s)
  - i: output port number
  - s: sequence number
- The switch architecture:



Tag: (output address, sequence number), e.g., (i,s)

MCI: Multicast connection identifier

Using CAM

For a write:

1. Read the write sequence number WS[i] from the write sequence RAM (WSRAM) (corresponding to the destination port i), and use this value (s) for the cell's tag {i, s}.
2. Search the tag CAM for the first empty location, emp.
3. Write the cell into the buffer B[emp]=cell, and {i, s} into the associated tag.
4. Increment the sequence number s by one, and update WS[i] with s+1.

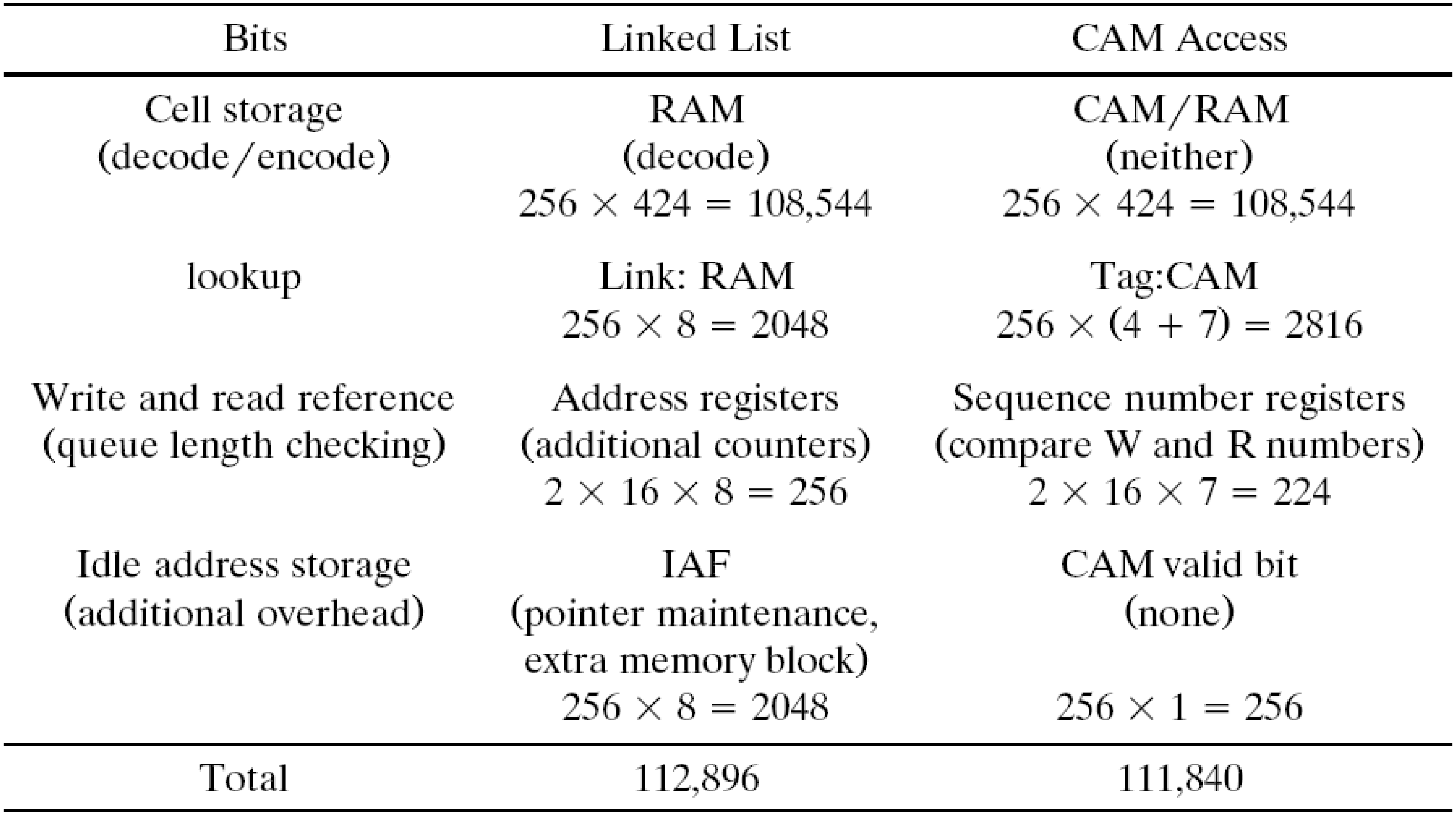
Using CAM

For a read:

1. **Read the read sequence number RS[j] from the read sequence RAM (RSRAM) (corresponding to the destination port j) , say t.**
2. **Search for the tag with the value {j, t}.**
3. **Read the cell in the buffer associated with the tag with the value {j, t}.**
4. **Increment the sequence number t by one and and update RS[i] with t+1.**

Using CAM

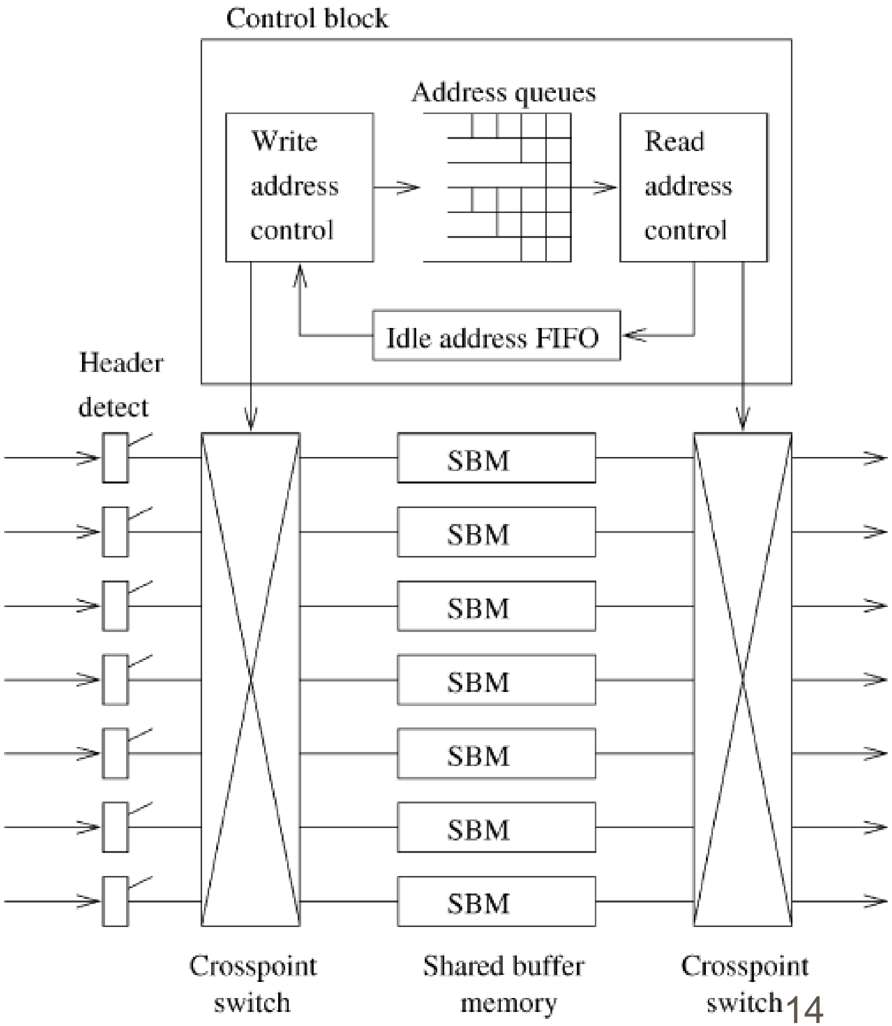
- Comparison with linked list



Bits	Linked List	CAM Access
Cell storage (decode/encode)	RAM (decode) 256 × 424 = 108,544	CAM/RAM (neither) 256 × 424 = 108,544
lookup	Link: RAM 256 × 8 = 2048	Tag:CAM 256 × (4 + 7) = 2816
Write and read reference (queue length checking)	Address registers (additional counters) 2 × 16 × 8 = 256	Sequence number registers (compare W and R numbers) 2 × 16 × 7 = 224
Idle address storage (additional overhead)	IAF (pointer maintenance, extra memory block) 256 × 8 = 2048	CAM valid bit (none) 256 × 1 = 256
Total	112,896	111,840

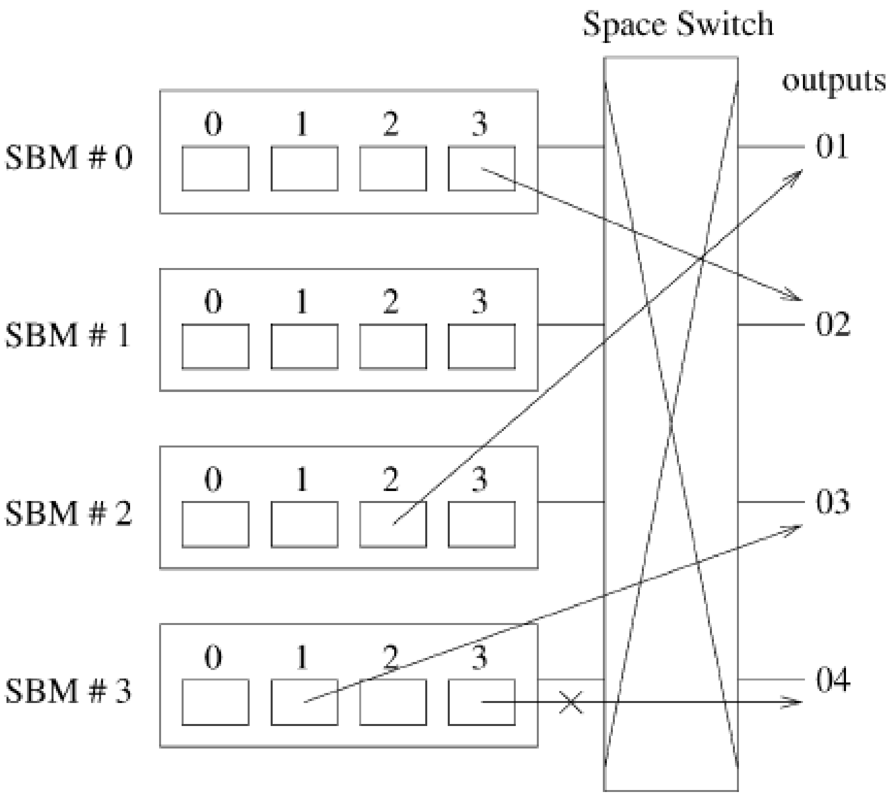
Space-time-space approach

- Space-time-space approach
  - Shared memory is partitioned into separate memories (SBMs)
  - Two crosspoint space division switches used to distribute access to SBMs
  - Crosspoint switching instead of time division MUX → Speedup



Space-time-space approach

- No blocking at inputs while SBMs are not full
- Blocking at outputs:

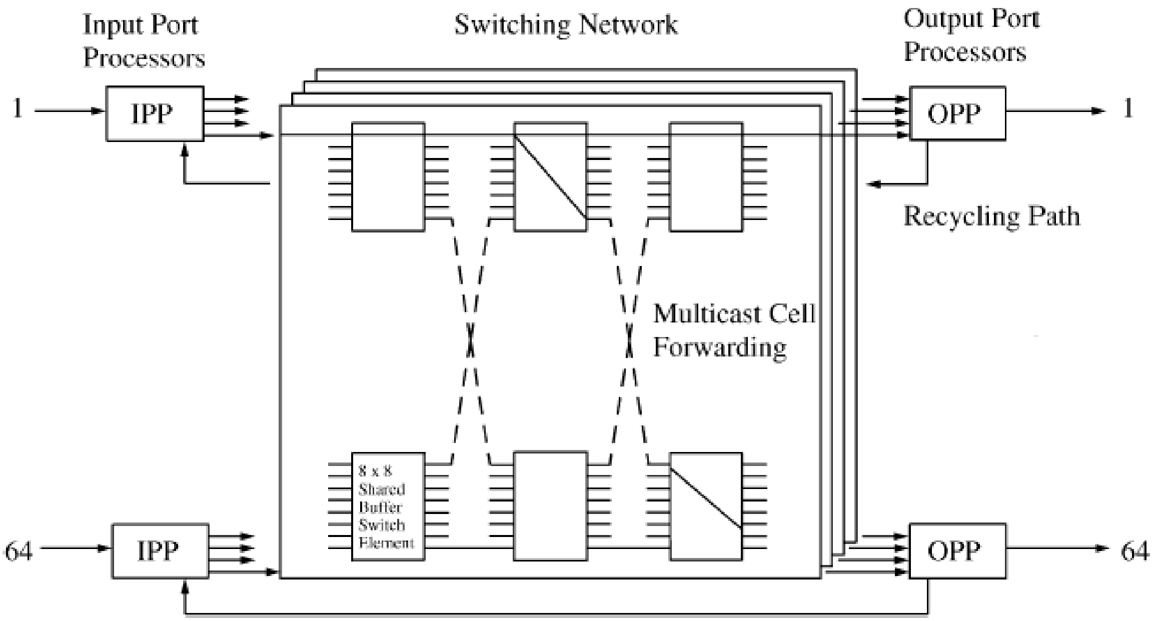


Multistage shared memory switches

- Small shared memory modules +
- Multistage network
- Examples
  - Washington university gigabit switch (WUGS) [16]
  - Concentrator based growable switch [4]
  - Multinet switch [8]
  - Siemens switch [5]
  - Alcatel switch [2]

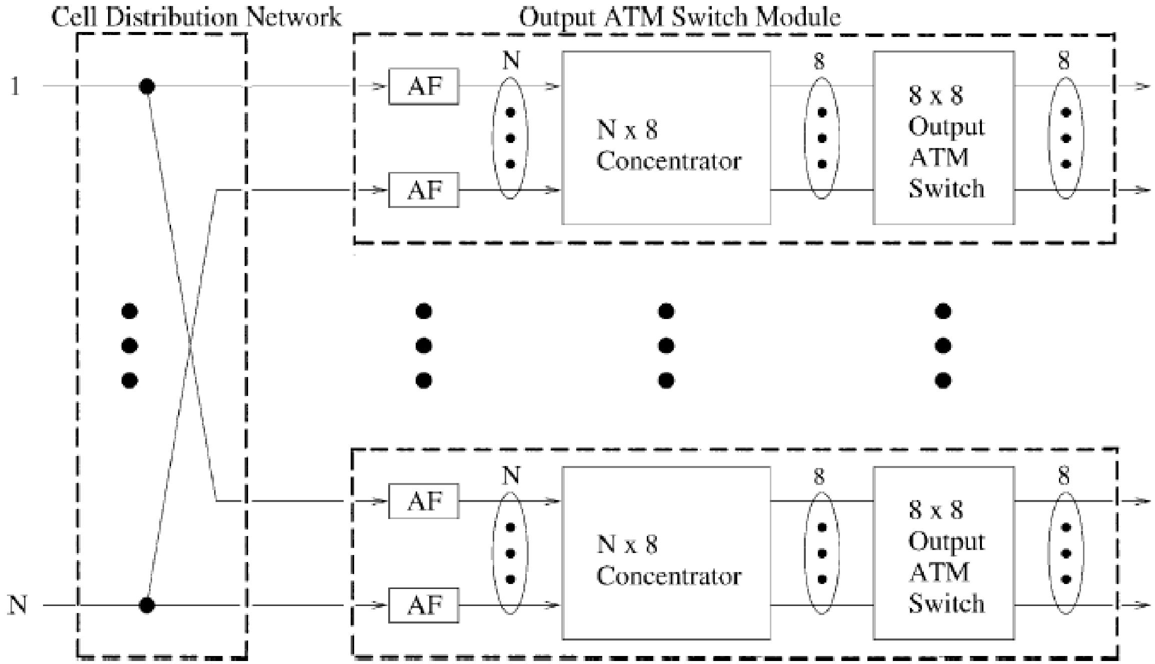
Multistage shared memory switches

- Washington university gigabit switch (WUGS)
  - Consists of 3 parts
    - Input port processors (IPP)
    - Central switching network
    - Output port processors (OPP)
  - IPP tasks
    - Buffering input cells
    - Virtual-path-circuit translation
  - OPP tasks
    - Resequencing cells
    - Recycling multicast cells to corresponding IPP
  - Central switching network
    - Benes topology
    - Good scalability due to recursive expansion
    - Good load balancing



Multistage shared memory switches

- Concentrator based growable switch
- From left to right:
  - Front-end broadcast network
  - Address filters
  - N*8 concentrators
  - 8*8 shared memory ATM swithes

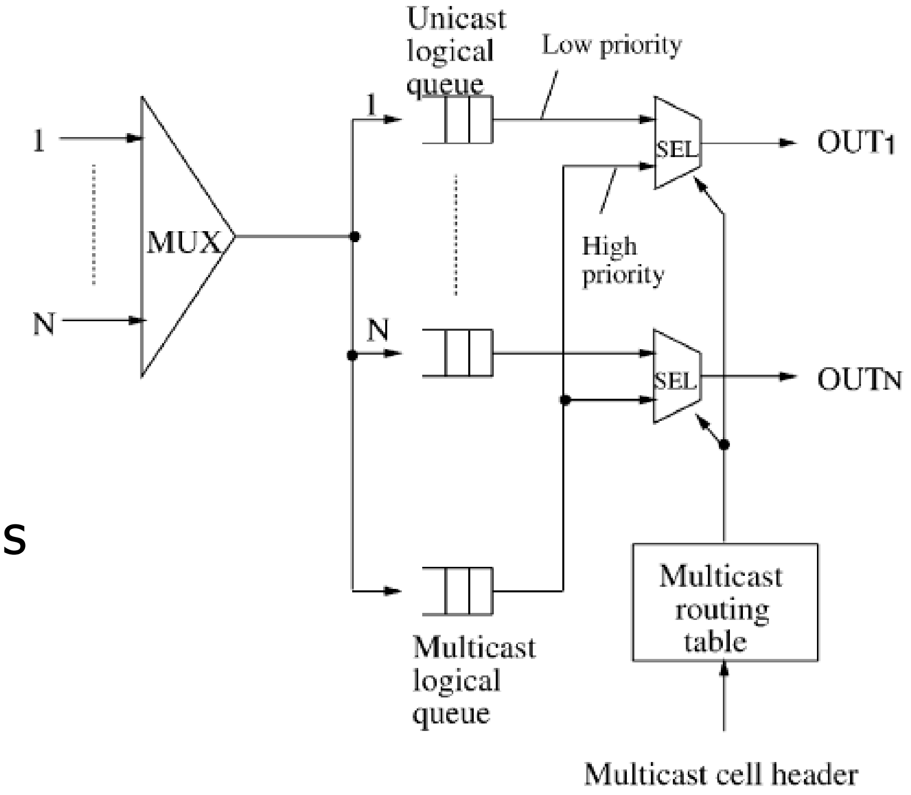


Multicast shared memory switches

- Multicast shared memory switches
  - 3 methods
    - Multicast logical queue
    - Cell copy
    - Address copy

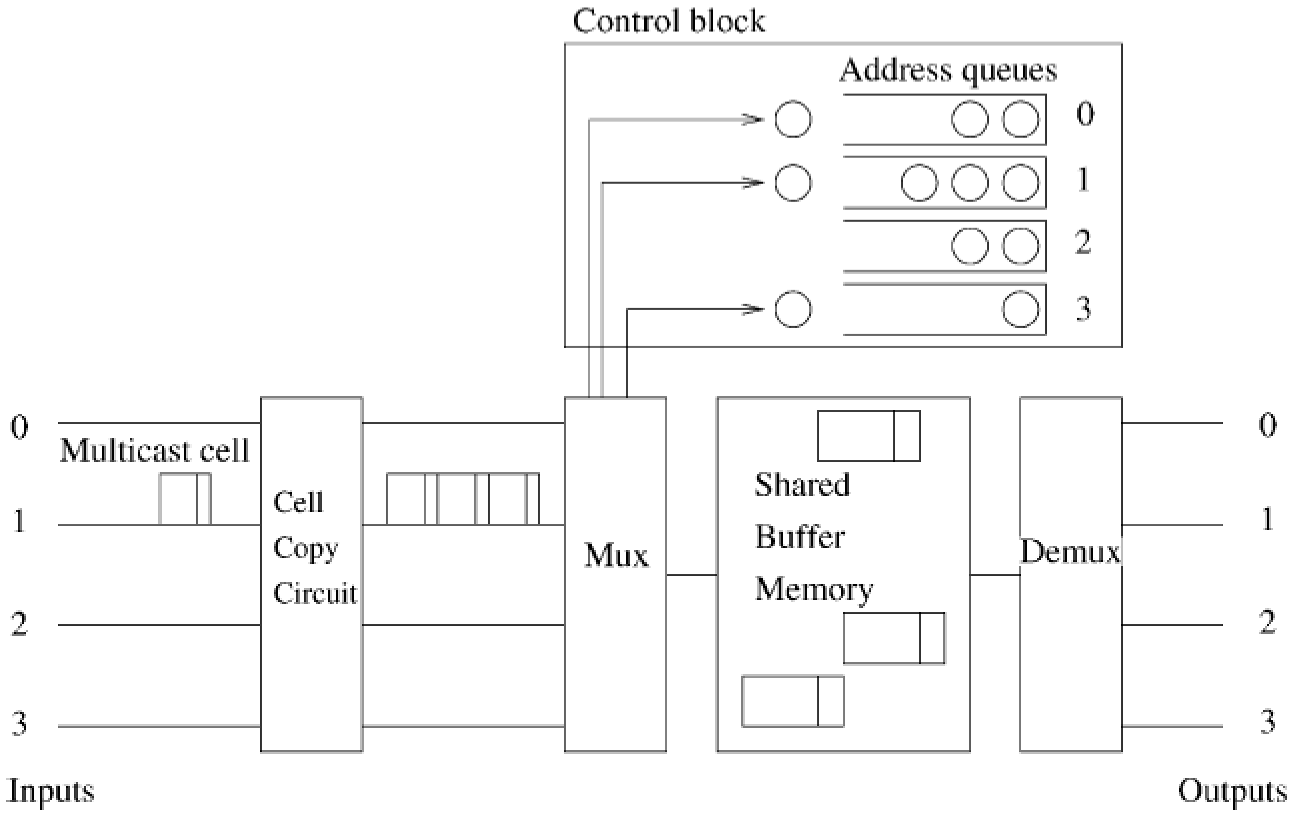
Multicast shared memory switches

- Multicast logical queue
  - An additional logical queue for multicast cells
  - Advantages
    - Simplicity
    - Minimized queue update operations per time slot
  - Service policy
    - Strict priority for multicast cells
    - Round robin
    - Weighted round robin
  - Disadvantage
    - HOL blocking for multicast cells when strict priority is not used



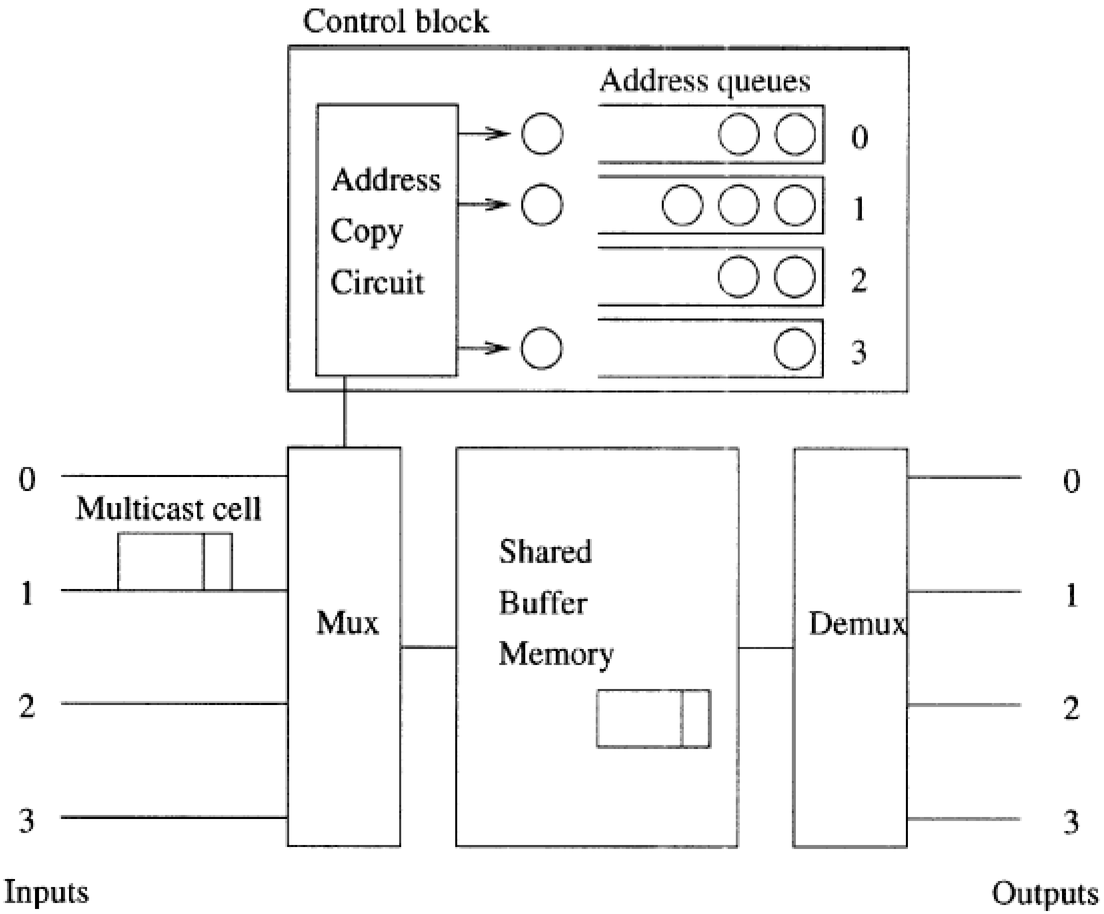
Multicast shared memory switches

- Cell copy method
  - Multicast cells are replicated
  - Disadvantages
    - Replication storage of O(N^2)
    - Need for cell copy circuit



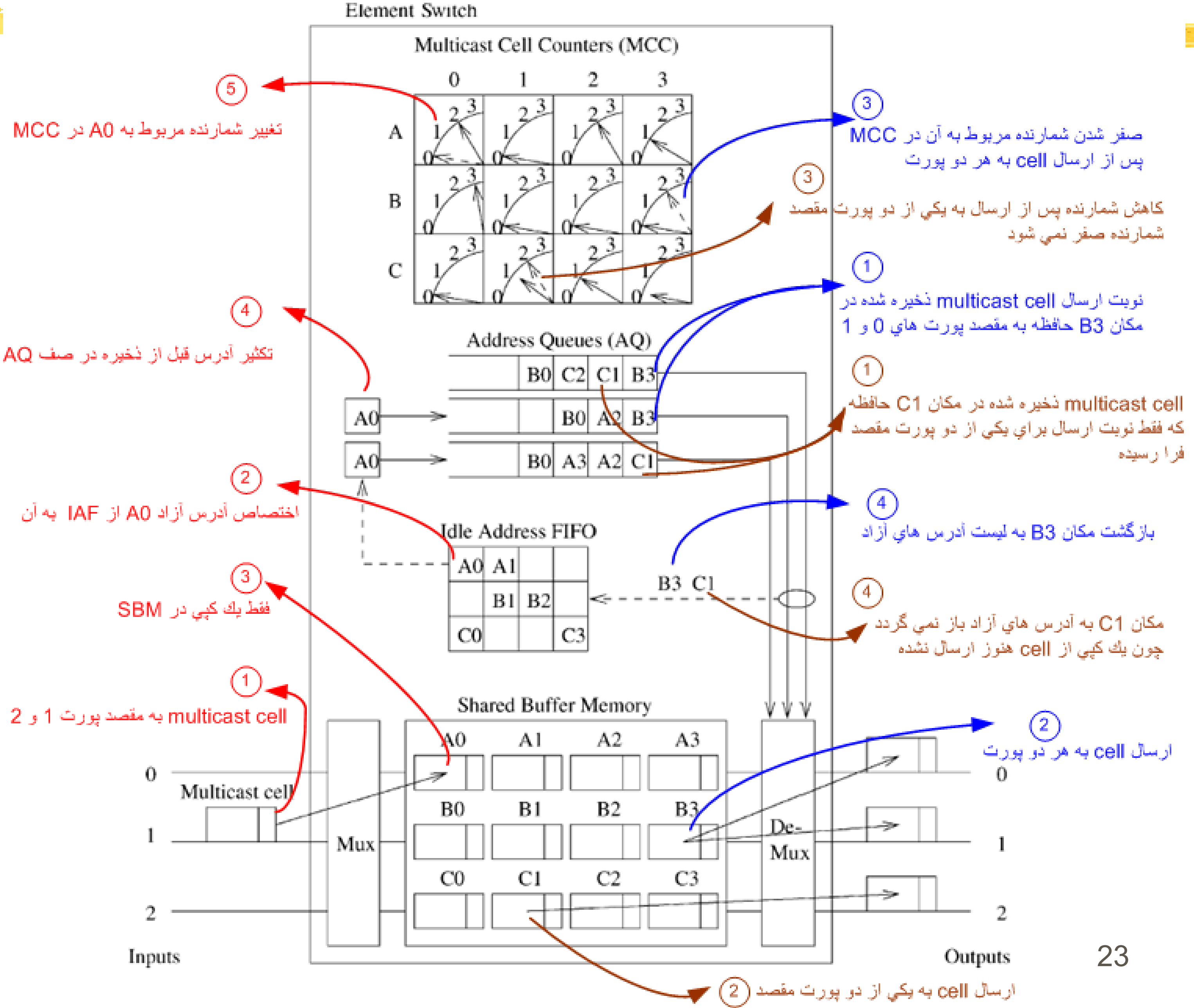
Multicast shared memory switches

- Address copy method
  - Address in SBM is replicated instead of the cell itself
  - Less memory usage than cell copy method
  - The need for multicast cell counters (MCC)



Multicast shared memory switches

- Address copy method
  - Detailed example



5 تغيير شمارنده مربوط به A0 در MCC

4 تكثير آدرس قبل از ذخيره در صف AQ

2 اختصاص آدرس آزاد A0 از IAF به آن

3 فقط يك كپي در SBM

1 multicast cell به مقصد پورت 1 و 2

3 صفر شدن شمارنده مربوط به آن در MCC
پس از ارسال cell به هر دو پورت

3 كاهش شمارنده پس از ارسال به يكي از دو پورت مقصد
شمارنده صفر نمي شود

1 نوبت ارسال multicast cell ذخيره شده در
مكان B3 حافظه به مقصد پورت هاي 0 و 1

1 multicast cell ذخيره شده در مكان C1 حافظه
كه فقط نوبت ارسال براي يكي از دو پورت مقصد
فرا رسيده

4 بازگشت مكان B3 به ليست آدرس هاي آزاد

4 مكان C1 به آدرس هاي آزاد باز نمي گردد
چون يك كپي از cell هنوز ارسال نشده

2 ارسال cell به هر دو پورت

2 ارسال cell به يكي از دو پورت مقصد