3. **Time Switching, Multi-Queue Memories, Shared Buffers, Output Queueing Family**

3.1 TDM, Time Switching, Cut-Through

3.2 Wide Memories for High Thruput, Segm’tn Ovrhd

3.3 Multiple Queues within a Buffer Memory

3.4 Queueing for Multicast Traffic

3.5 Shared Buffering and the Output Q’ing Family

*Manolis Katevenis*

CS-534 – Univ. of Crete and FORTH, Greece

http://archvlsi.ics.forth.gr/~kateveni/534
3.2 Wide Memories for High Throughput

Table of Contents:

• 3.2.1 Wide Memories for High Throughput
  – Shared Buffer Switch using a Wide Memory
  – Pipelined Memory: Optimized Shared Buffer Switching
  – Generalization: Interleaved Memory Banks

• 3.2.2 Variable Size Packet Segmentation Overhead
  – Number of memory accesses or crossbar cell-times versus packet-time on the line
3.2.1 Shared Buffer Switch using a Wide Memory

- Example: 4×4 switch @ 10 Gbps/link = 320 MHz × 32 bits
- Memory serves each I/O register for one clock cycle out of every eight
- Note hidden input and output crossbars; note required cell time alignment

Wide (256 bits) Memory Shared Buffer
3.2.1 Wide Memory: Double Buffering Needed

- Input latches need double buffering for the case when multiple cells complete at the same time.
- If cut-through is desired, separate paths and a crossbar are needed.
3.2.1 Optimized Implementation: Pipelined Memory

- Monolithic wide memory replaced by multiple banks
- Concurrent rd/wr access to entire width replaced by a “wave” of same-address accesses moving from left to right
  - Benefit 1: no double buffering needed at inputs, single latch at outputs
  - Benefit 2: cut-through occurs automatically – no paths, no control needed
Pipelined Memory Operation Illustrated


3.2 - U. Crete - M. Katevenis - CS-534
3.2.1 Generalization: Interleaved Memory Banks

- Can we scale arbitrarily wide-memory throughput by increasing its width?
- Not past individual packet width – then it becomes something else; see ↓
Interleaved Mem. Sh.Buf. – PPS, Birkhoff-vonNeumann

• Wide memory width versus packet size
  – Wide (or pipelined) memory owes its simplicity to the controller generating a single address per cycle for all “banks” (entire width)
  – To have multiple packets per line (e.g. red & blue in figure): need independent address controllers per bank (per effective xbar port)
  – Note: infeasible to pack together into the same line packets that both arrive and depart consecutively in time
  – Scheduling interleaved memory accesses against bank conflicts is equivalent to crossbar scheduling under input queueing…

• Scalable Shared-Buffer Switches through Interleaved Banks:
  – Complexity: maintain distributed queues, forward cells in-order
3.2.2 Variable Size Packet Segmentation Overhead

- Most buffer memories, queueing structures, & crossbars operate on fixed-size "segments", "blocks", or "cells".
- Most appl'ns use variable size packets.

- Packets segmented into blocks/cells upon entry from line to switch.
- Throughput of buffer memories & crossbars, measured in terms of cells, must be higher than line thruput, to compensate for segment’n overhead.
Overspeed req’d to compensate for Segmentation

- Example:
  - min. pck. size = 40 B
  - segment (cell) = 64 B
  - line overhead = 16 B

- Rules of thumb:
  - For line overhead = 0 ⇒ overspeed = often 2.0
  - … but check min.pck.sz.: if minimum packet is too small (smaller than about half a cell), then higher overspeed is required
  - for line overhead > 0 ⇒ overspeed < 2.0, but check minimum packets
  - for very large line ovrhd ⇒ check larger pck sizes