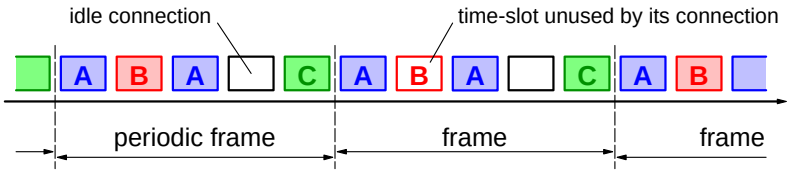
2.2 Circuit Switching, Time-Division Multiplexing (TDM), Time Switching, Cut-through

- Circuit Switching versus Packet Switching
- Digital Telephony, Time-Division Multiplexing (TDM)
- Time Switching, Time-Slot Interchange (TSI)
- Switching and Computers: 1st and 2nd Generations
- Cut-through

 1

Circuit Switching

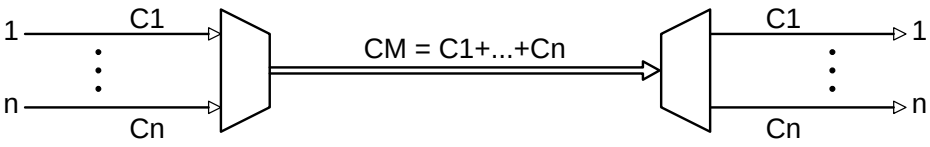


- Data are transmitted inside fixed, periodic frames; each circuit (connection) is allocated a fixed subset of the time-slots in each frame; connectionID and routing information is provided implicitely by the time-slotID in which a datum is transmitted.
- The transmission capacity of a link is partitioned into a fixed number of circuits, each of them having a fixed rate; unused capacity in one circuit cannot be used by other circuits.
- Advantage: simple.
- Disadvantage: wasteful in transmission capacity, especially when actual rate of connections varies widely with time.

 2

Multiplexing - Demultiplexing

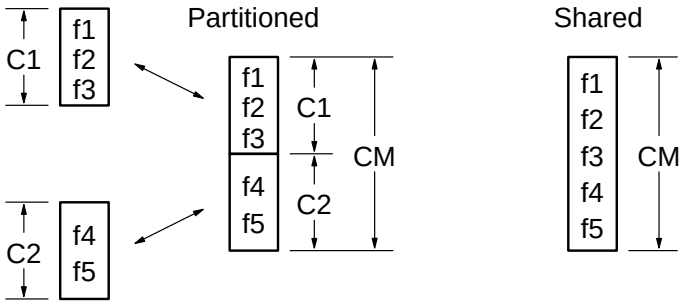
at fixed aggregate capacity (circuit-switching style)



Examples:
- circuit switching: frames & time-slots
- wide (bit-parallel) buses inside switch elements

Minimal buffering requirements:
one time-slot-worth of data per mux'ed/demux'ed link

Partitioned versus Shared Link Capacity



Resource Partitioning leads to Underutilization:

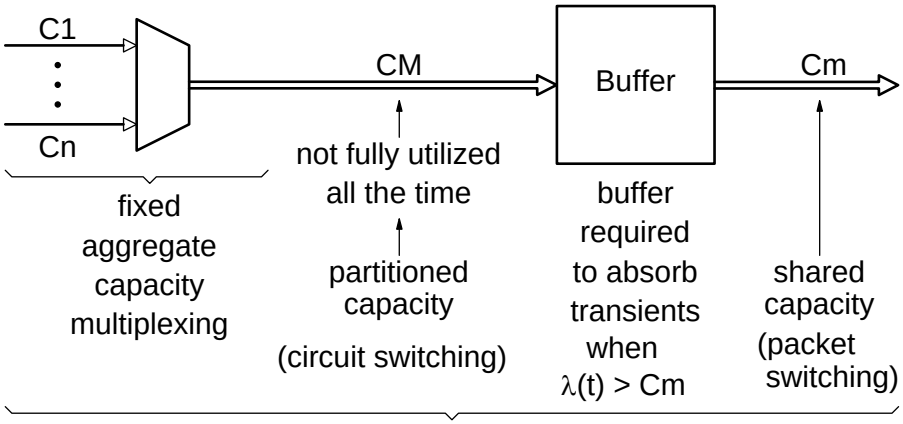
In a link carrying multiplexed traffic of fixed aggregate capacity type, the flows in one partition may lack capacity, while other partitions may have excess capacity.
This is the disadvantage of circuit switching.

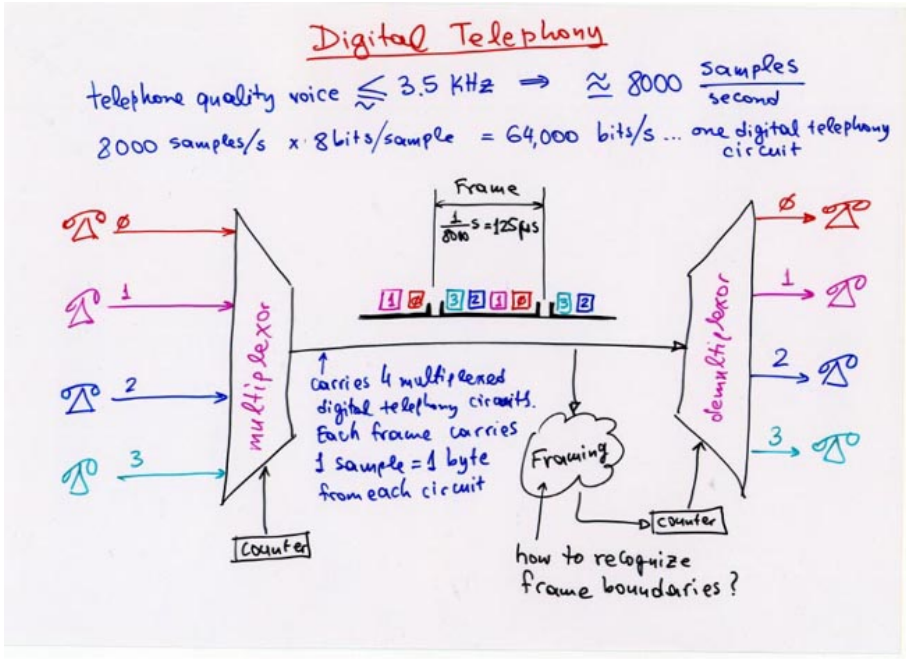
Packet Switching

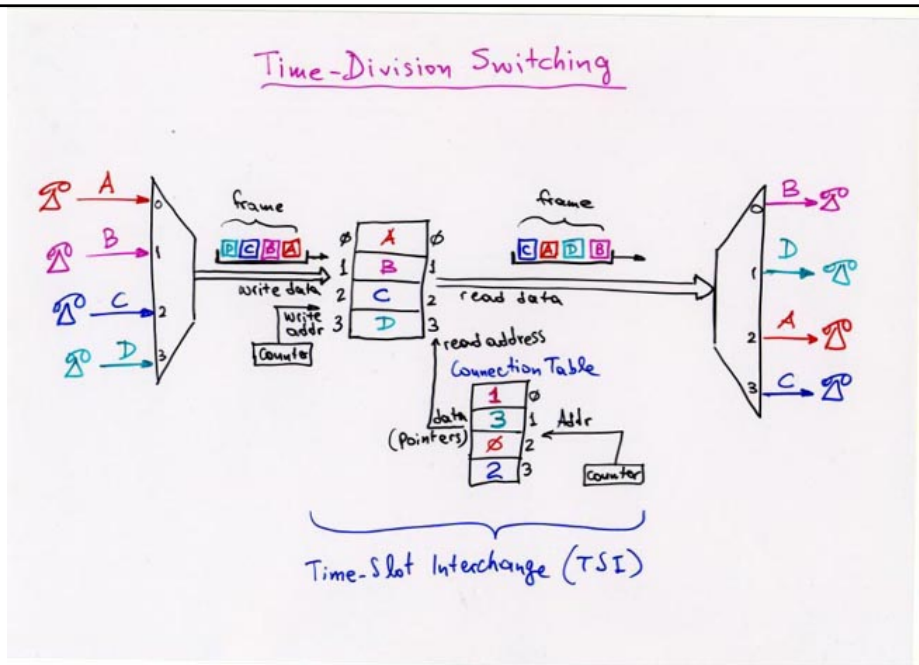
- Non-periodic multiplexing of packets, on a demand basis; each packet carries its own source and destination (connection) ID, and can be stored and forwarded at any later time.
- The transmission capacity of a link is shared among all flows (connections) that pass through it, on a demand basis; any capacity that is not used by one flow can be used by another.
- Advantage: no waste of transmission capacity.
- Challenges:
  dynamic control (per packet), rather than static (at conn. set-up);
  unpredictability of traffic, leading to contention for resources.

Packet Switching: Statistical Multiplexing

C1 + ... + Cn = CM > Cm

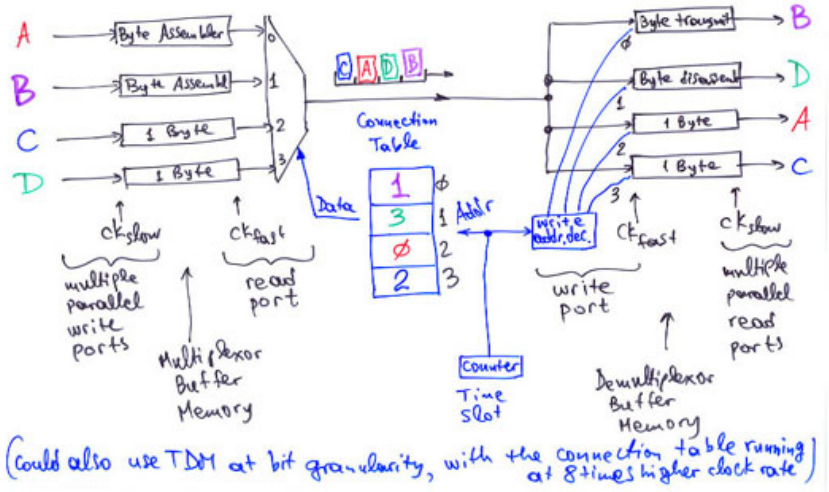



Digital Telephony
telephone quality voice ≲ 3.5 KHz ⇒ ≈ 8000 samples/second
8000 samples/s x 8 bits/sample = 64,000 bits/s ... one digital telephony circuit
Frame
1/8000 s = 125 μs
multiplexor
demultiplexor
carries 4 multiplexed digital telephony circuits. Each frame carries 1 sample = 1 byte from each circuit
Framing
counter
counter
how to recognize frame boundaries?


Time-Division Switching
frame
write data
write addr
counter
frame
read data
read address
Connection Table
data (pointers)
Addr
counter
Time-Slot Interchange (TSI)

Time-Division Switching:

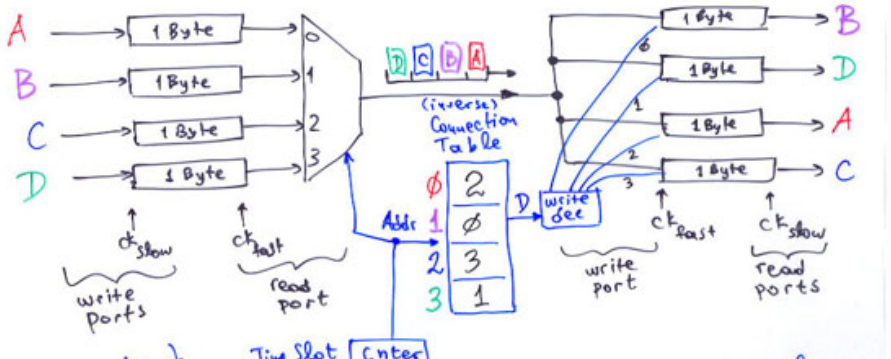
When each external link carries a single connection, the time-slot interchange can be merged with the multiplexor (or with...



(Could also use TDM at bit granularity, with the connection table running at 8 times higher clock rate)

Time-Division Switching with external links carrying a single connection each:

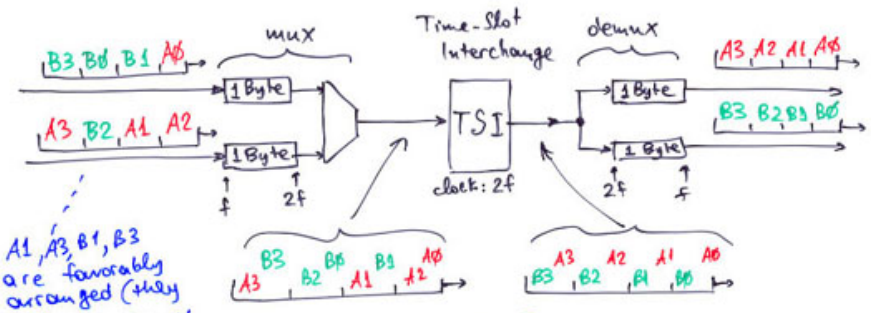
Alternatively, the time-slot interchange can be merged with the demultiplexor



Note: for (time) switching to be possible, it is essential to have a single, shared line going from all inputs to all outputs. Every single quantum of information goes through that single common line. The throughput of that line = Σ inp.throup. = Σ out.throup. In this case, $f_{fastclock} = n \cdot f_{slowclock}$ (n = # ext. lines).

Time-Division Switching: more complex case: multiple connections per external line

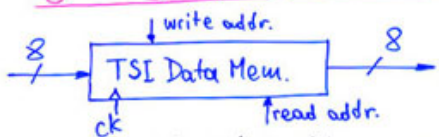
- Mux and Demux need less buffer memory than full frame
- Internal TSI needed with 1 full frame of buf. memory
- Internal TSI cannot be merged w. mux or demux
- Worst-case delay = 1 frame time, again.



A1, A3, B1, B3 are favorably arranged (they arrive on correct time slot); however, not so for A0, A2, B0, B2, because A0, B0 arrive on same link ⇒ cannot be placed both in correct timeslot.

Discussion: Can I freely rearrange positions of connections inside inp. & output frames?

Byte-by-Byte Time Switching: Throughput Limit ?



Assume 300 MHz 2-port SRAM
⇒ Peak Throughput = 300 MBytes/s = 2.4 Gbits/s = 37,500 × 64Kb/s

(if making an 8×8 switch ⇒ each link up to 300 Mb/s ... quite low @ today's standards)

Can we Increase Throughput by Widening the TSI Memory?

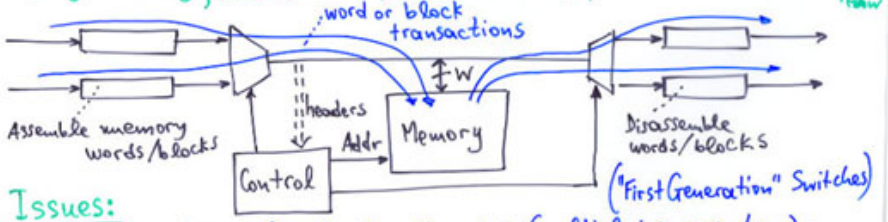
128 bits = 16 Bytes
128 b × 300 MHz = 38.4 Gbits/s

⇒ What is the Multiplexing Quantum ???

- 16 Bytes belonging to a same 64 Kbps channel ?:
  - → must wait 16 frames = 16 × 125 μs = 2 ms to collect all these bytes!
  - → buffer size for collection = 2 ms × 38.4 Gb/s = 76.8 Mbits
- 16 Bytes belonging to 16 "adjacent" 64 Kbps channels ?:
  - → must switch all 16 channels together: where one of them goes, all 16 of them must go!

Time Division Switching: From Circuit Switching to Packet Switching

...from statically, off-line scheduled, fixed-throughput/channel to dynamically, demand-driven, on-line scheduled, variable-throughput/flow

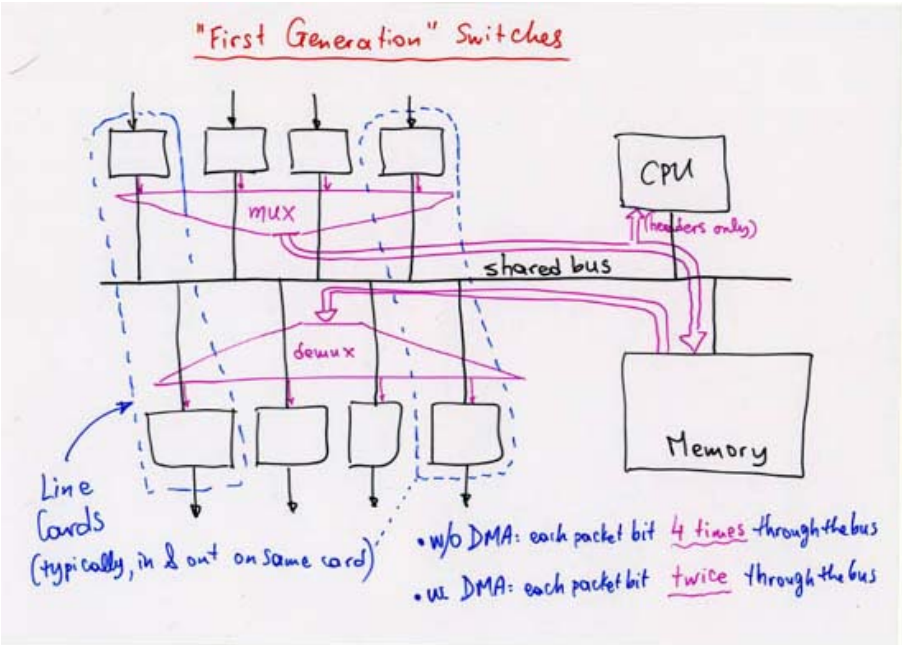


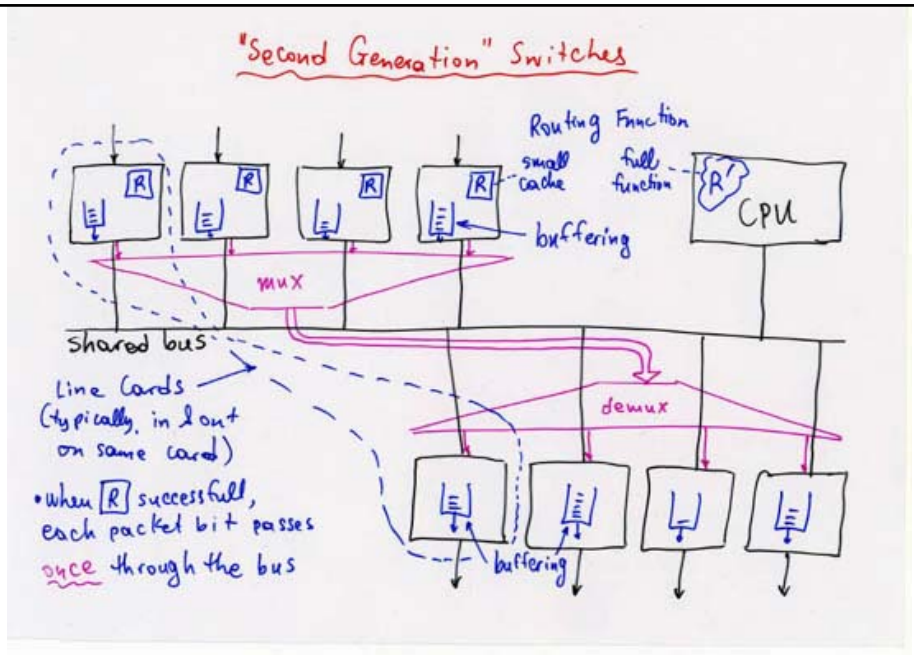
("First Generation" Switches)

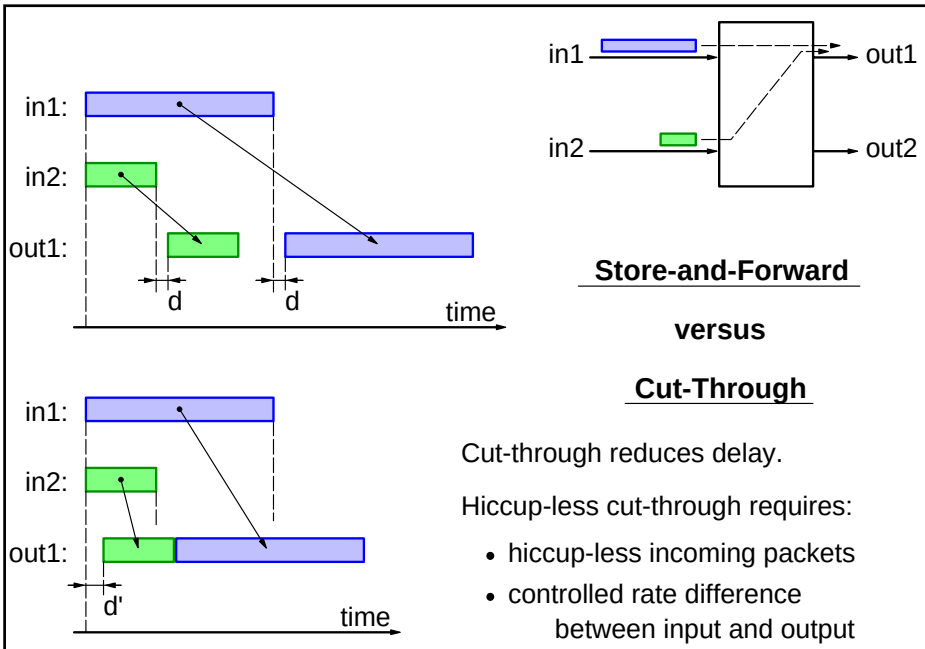
Issues:

- Granularity of transfers In→Mem, Mem→Out (multiplexing quantum):
  - fine grain... (narrow word/block): ⊕ small buffers in I/O; ⊕ small packets OK!
  - ⊖ narrow mem ⇒ small throughput; ⊖ access rate/bus turn-around bound
- Control structure & operation:
  - where to store the words or blocks of each packet? Contiguous? scattered?
    can I mix multiple packets in one block?
  - where to store the packets going to a certain output link?
    (or from a certain input link?) (or of a same QoS class?)

"First Generation" Switches







Store-and-Forward

versus

Cut-Through

Cut-through reduces delay.

Hiccup-less cut-through requires:

- hiccup-less incoming packets
- controlled rate difference between input and output