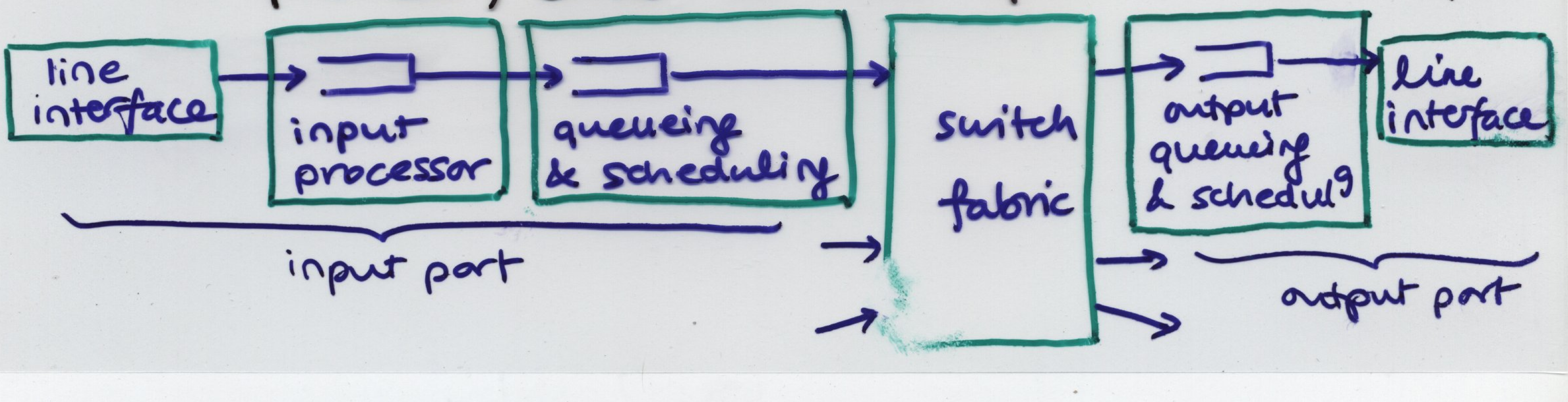
Components of a packet switch

- a line interface extracts the packet from the input link (identifies bits & packet boundaries)
- an input processor extracts the header, which is used by the associated forwarding engine to determine the appropriate output link(s) by looking up the routing table / switch table
  - in a multiservice network, the service type of the packet is also determined at this stage
  - in large networks, this processing can be relatively complex & take a variable amount of time, thus requiring a queue
- the switch fabric moves packets from inputs to outputs
  - output contention occurs when two or more packets arriving on different links want to leave from the same output link at the same time
  - all but one of the contending packets must be queued, either at the input or the output



- the <u>control processor</u> performs control & management functions, eg routing protocols to update routing tables
  - it is connected to the ports either directly or via the switch fabric
  - typically, local copies of the forwarding tables are stored at the input ports & updated by the control processor as needed

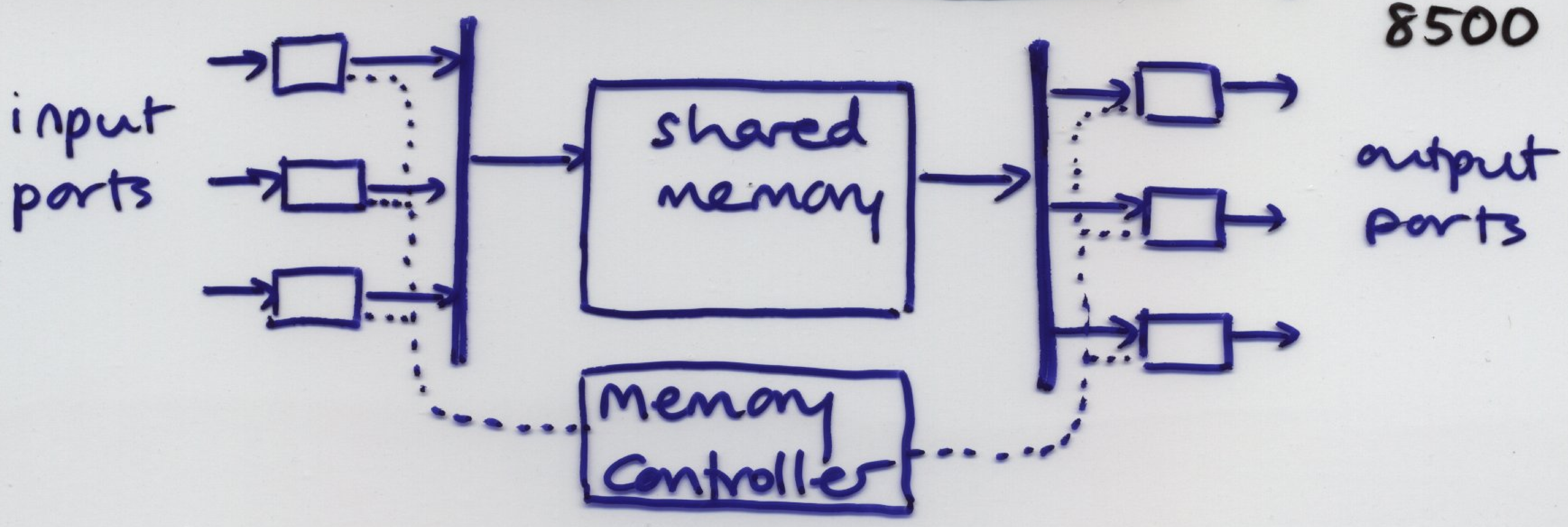
Architectural & performance issues in packet switches

- Continuous-time vs cell switching
  - IP packets have variable lengths & arrive at variable intervals
  - simple approach used in many commercial switches is to break packets into fixed-size cells & use an internal cell switch
  - cells are attached to internal switch headers that are used to reassemble them at the fabric output; headers are removed before transmission on the output link
  - if a packet is not equal in length to an integral no. of cells, the last cell of the packet is padded
  - small cell size → large switch header overhead
    large cell size → large padding overhead
  - continuous-time switching: switch operations not synchronized into slots; work directly with variable length packets
    → avoids complexity & overhead associated with segmentation of packets into cells, but scheduling is more complex

• Switch fabrics

- a switch / switch fabric is called <u>nonblocking</u> if for any matching between inputs & outputs (i.e. no output contention), the corresponding connections can be simultaneously achieved
- in large switches, some amount of blocking is tolerated to ↓ implementation complexity
- the switching capacity is the sum of the input link transmission rates that can be supported by the switch
- <u>elementary switching structures</u>
    - can be used to build moderate-capacity switches with relatively few ports, or interconnected to form high-capacity switch fabrics with many ports

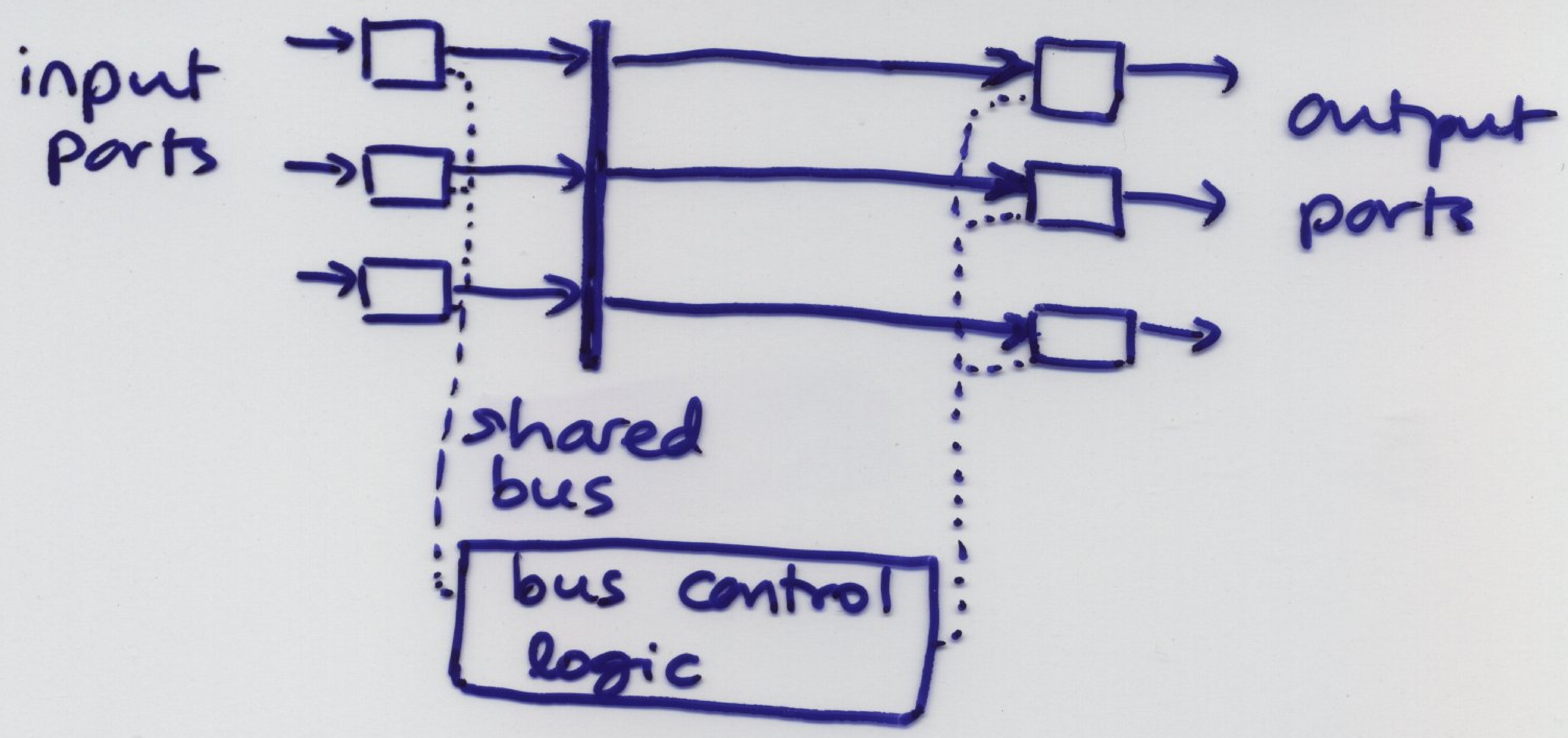
a) <u>shared-memory switch fabric</u> eg. Cisco Catalyst 8500 series



- for each incoming packet, the output port is determined & this information is used by the memory controller to choose the location where the packet is stored in the shared memory

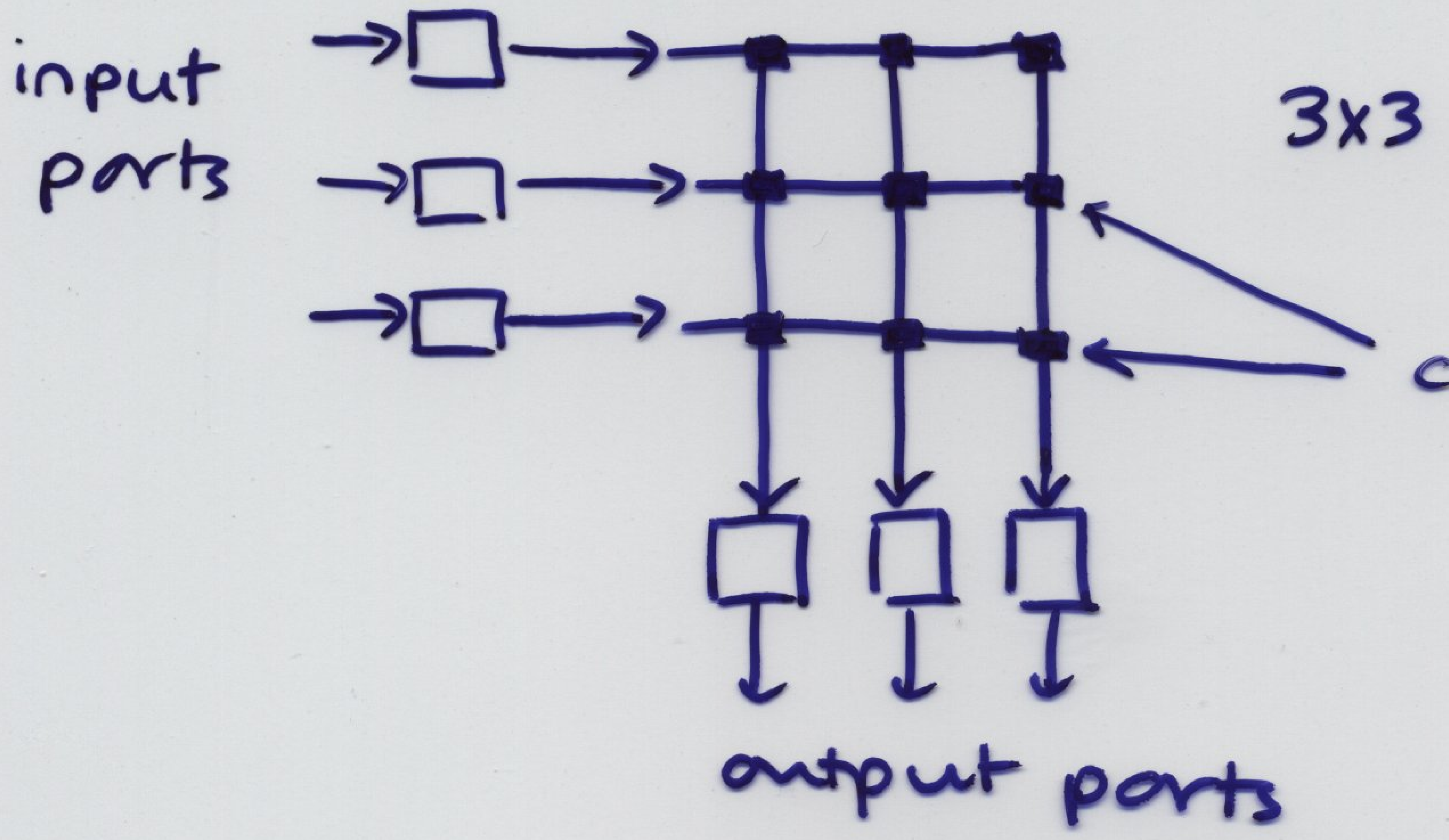
- the memory controller also controls the locations in the shared memory from which the output ports read their packets — the shared memory can be dynamically allocated among the output queues or statically partitioned, or a combination of both
- the switching capacity is limited to half the memory bandwidth (since each connection requires reading & writing at the connection rate)

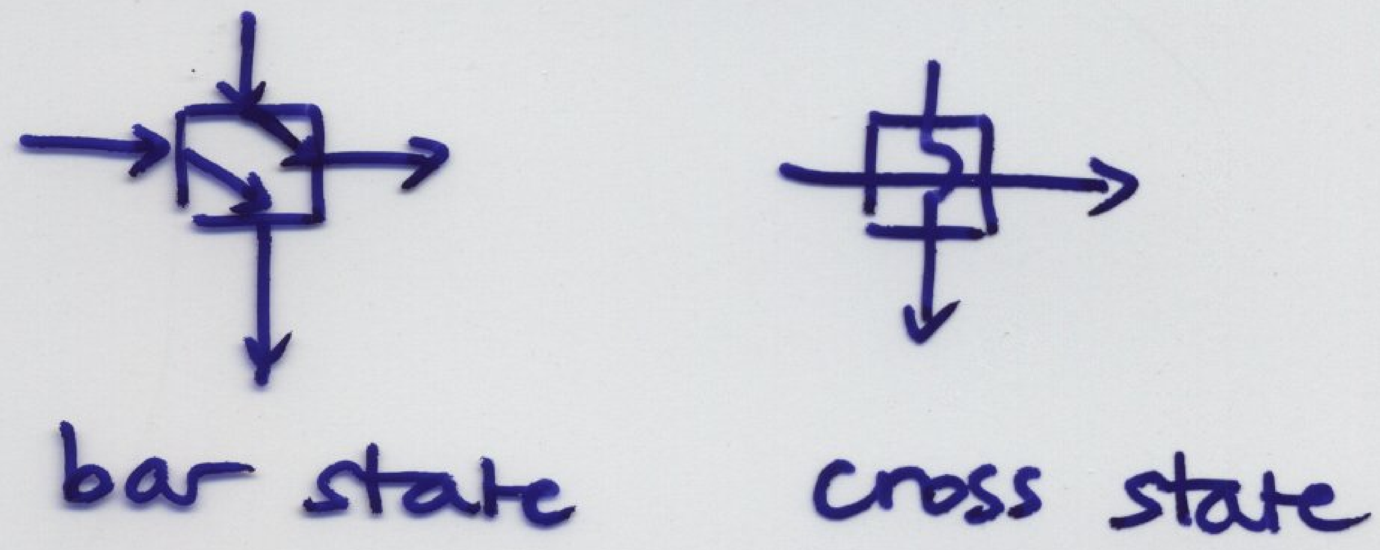
b) <u>Shared-medium switch fabric</u> eg. Cisco 5600 series



- input ports transfer packets to output ports via a shared bus
- only 1 packet can be transferred at a time over the shared bus; access to the bus is controlled by polling, TDM or handshaking between input & output ports
- each output port has a separate queue
- the switching capacity is limited by the bus bandwidth (can be as high as 32 Gbps; sufficient for access & enterprise network routers)

c) Crossbar switch fabric

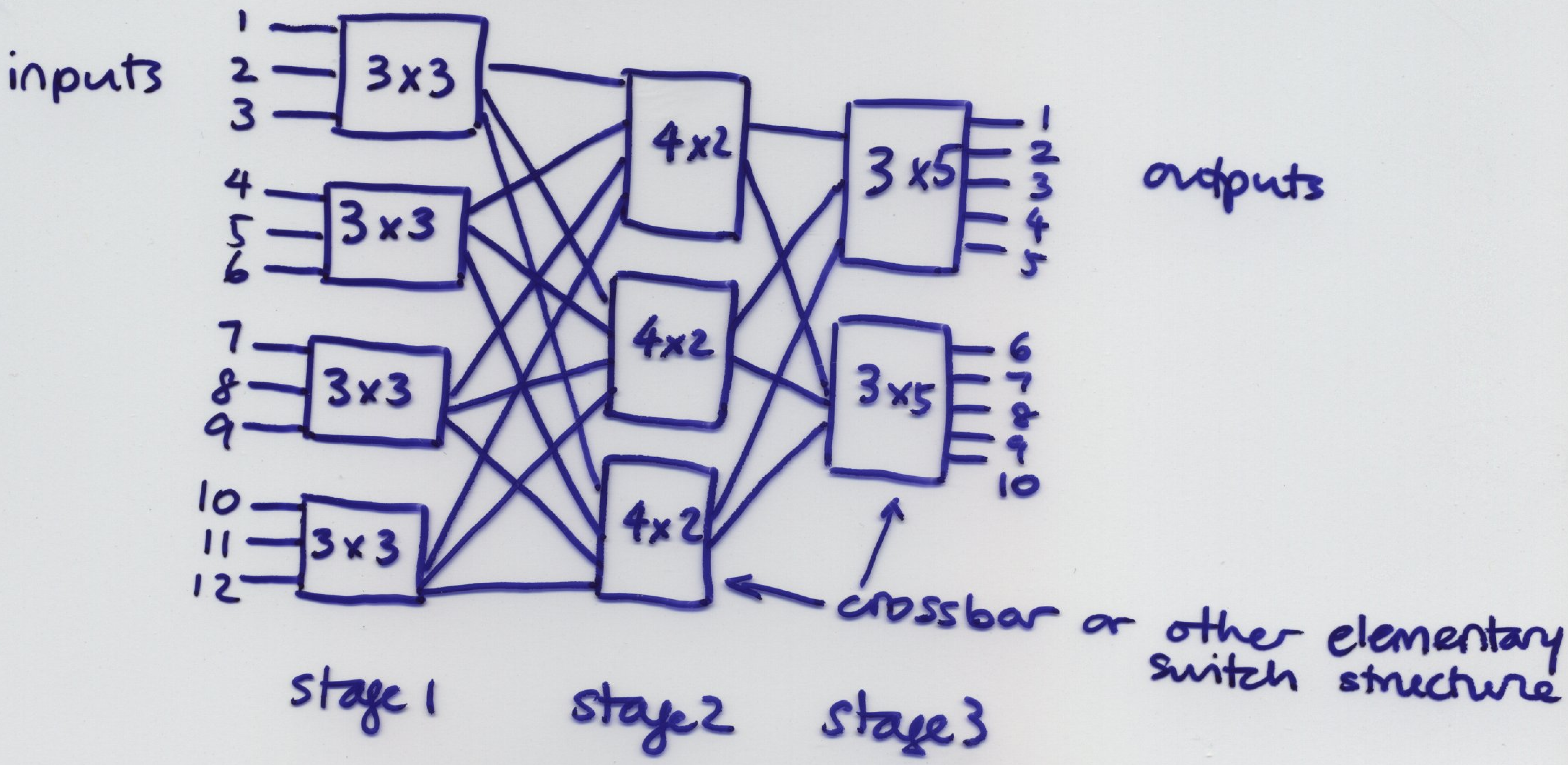




- an N×N crossbar has N input lines & N output lines, & N^2 crosspoints
- a packet received at an input port travels along the corresponding (horizontal) input line until it intersects the (vertical) output line corresponding to the desired output port
- internally nonblocking
- the input & output ports of received packets are used to set the crosspoint switches to set up the desired input-output paths
- only 1 packet at a time is transferred on each line
- the complexity of a crossbar in terms of crosspoints or paths grows as N^2
- each crosspoint has 2 states, so an N×N crossbar has 2^{N^2} states (substantially more than the lower bound of N! states needed for an N×N nonblocking switch)

- these elementary switching structures do not scale well
- most large switches are formed by interconnecting smaller switches into a <u>switching network</u>

eg. 12× 10 switching network



- <u>blocking in switching networks</u>
  - let $c=(i,j)$ denote a connection request from input i to output j
  - a set $\mathcal{C}$ of connection requests is called feasible if all the connection requests in $\mathcal{C}$ have distinct inputs & distinct outputs
  - a switching network may contain >1 way to set up paths satisfying a given $\mathcal{C}$; the chosen set of paths is called a <u>routing</u> of $\mathcal{C}$, denoted $R(\mathcal{C})$
  - $\{\mathcal{C}, R(\mathcal{C})\}$ is called a <u>state</u> of the switching network

- state $\{\mathcal{C}, R(\mathcal{C})\}$ is called a <u>nonblocking state</u> if for any feasible $\{c\} \cup \mathcal{C}$, $c \notin \mathcal{C}$, the additional connection c can be routed through the switch without changing the existing routing $R(\mathcal{C})$
- a switching network is
  - <u>rearrangeably nonblocking</u> (RNB) if any feasible set of connections can be routed, possibly with rearrangements if connections are routed sequentially
  - <u>strictly nonblocking</u> (SNB) if there are no blocking states
  - <u>wide sense nonblocking</u> (WSNB) under a routing algorithm $\mathcal{R}$ if the additional connection (feasible) connection c can be routed according to $\mathcal{R}$ & the resulting state $\{\mathcal{C} \cup \{c\}, \mathcal{R}(\mathcal{C} \cup \{c\})\}$ is also a nonblocking state (i.e. a route is always available for a new feasible connection if a predefined routing algorithm is used for all connections)
- SNB, WSNB networks are useful in continuous-time switches; RNB networks are better suited to cell switches (a new set of cells in each slot)
- tradeoff between hardware complexity & routing complexity

- in high-speed packet switches, the time available for the routing decision is on the order of the packet transmission time, so low complexity routing algorithms are needed for scalability
- a switching network is <u>self-routing</u> if the input & output port addresses of each cell can be used to set up the appropriate input-output paths in a distributed manner
  - scalable
- <u>delta networks</u> are a large class of networks that can be made self-routing
  - an $a^n \times b^n$ delta network is built using n stages of $a \times b$ crossbars, with $a^{n-i} b^{i-1}$ crossbars in stage $i = 1, 2, \ldots, n$
  - the interconnection pattern is such that there is a unique path from each input port to each output port
  - Eg. a $3^2 \times 3^2$ delta network