CHAPTER 8

Switching

Solutions to Review Questions and Exercises

Review Questions

1. ***Switching*** provides a practical solution to the problem of connecting multiple devices in a network. It is more practical than using a bus topology; it is more efficient than using a star topology and a central hub. Switches are devices capable of creating temporary connections between two or more devices linked to the switch.
2. The three traditional switching methods are ***circuit switching***, ***packet switching***, and ***message switching***. The most common today are ***circuit switching*** and ***packet switching***.
3. There are two approaches to packet switching: ***datagram approach*** and ***virtual-circuit approach***.
4. In a ***circuit-switched network***, data are not packetized; data flow is somehow a continuation of bits that travel the same channel during the data transfer phase. In a ***packet-switched network*** data are packetized; each packet is somehow an independent entity with its local or global addressing information.
5. The address field defines the ***end-to-end*** (source to destination) addressing.
6. The address field defines the ***virtual circuit number*** (local) addressing.
7. In a ***space-division*** switch, the path from one device to another is spatially separate from other paths. The inputs and the outputs are connected using a grid of electronic microswitches. In a ***time-division*** switch, the inputs are divided in time using TDM. A control unit sends the input to the correct output device.
8. ***TSI*** (time-slot interchange) is the most popular technology in a time-division switch. It used random access memory (RAM) with several memory locations. The RAM fills up with incoming data from time slots in the order received. Slots are then sent out in an order based on the decisions of a control unit.
9. In multistage switching, ***blocking*** refers to times when one input cannot be connected to an output because there is no path available between them—all the possible intermediate switches are occupied. One solution to blocking is to increase the number of intermediate switches based on the Clos criteria.

10. A packet switch has four components: ***input ports***, ***output ports***, the ***routing processor***, and the ***switching fabric***. An input port performs the physical and data link functions of the packet switch. The output port performs the same functions as the input port, but in the reverse order. The routing processor performs the function of table lookup in the network layer. The switching fabric is responsible for moving the packet from the input queue to the output queue.

Exercises

11. We assume that the setup phase is a two-way communication and the teardown phase is a one-way communication. These two phases are common for all three cases. The delay for these two phases can be calculated as three propagation delays and three transmission delays or

$$3\ [(5000\ \text{km})/\ (2 \times 10^8\ \text{m/s})] + 3\ [(1000\ \text{bits}/1\ \text{Mbps})] = 75\ \text{ms} + 3\ \text{ms} = \mathbf{78\ ms}$$

We assume that the data transfer is in one direction; the total delay is then

delay for setup and teardown + propagation delay + transmission delay

a. 78 + 25 + 1 = ***104 ms***
b. 78 + 25 + 100 = ***203 ms***
c. 78 + 25 + 1000 = ***1103 ms***
d. In case a, we have 104 ms. In case b we have 203/100 = 2.03 ms. In case c, we have 1103/1000 = 1.101 ms. The ratio for case c is the smallest because we use one setup and teardown phase to send more data.

12. We assume that the transmission time is negligible in this case. This means that we suppose all datagrams start at time 0. The arrival timed are calculated as:

First:	(3200 Km) / (2×10^8 m/s)	+ (3 + 20 + 20)	= **59.0 ms**
Second:	(11700 Km) / (2×10^8 m/s)	+ (3 + 10 + 20)	= **91.5 ms**
Third:	(12200 Km) / (2×10^8 m/s)	+ (3 + 10+ 20 + 20)	= **114.0 ms**
Fourth:	(10200 Km) / (2×10^8 m/s)	+ (3 + 7 + 20)	= **81.0 ms**
Fifth:	(10700 Km) / (2×10^8 m/s)	+ (3 + 7 + 20 + 20)	= **103.5 ms**

The order of arrival is: **3** → **5** → **2** → **4** → **1**

13.

a. In a ***circuit-switched*** network, end-to-end addressing is needed during the setup and teardown phase to create a connection for the whole data transfer phase. After the connection is made, the data flow travels through the already-reserved resources. The switches remain connected for the entire duration of the data transfer; there is no need for further addressing.
b. In a ***datagram network***, each packet is independent. The routing of a packet is done for each individual packet. Each packet, therefore, needs to carry an end-to-end address. There is no setup and teardown phases in a datagram network (connectionless transmission). The entries in the routing table are somehow permanent and made by other processes such as routing protocols.

c. In a ***virtual-circuit*** network, there is a need for end-to-end addressing during the setup and teardown phases to make the corresponding entry in the switching table. The entry is made for each request for connection. During the data transfer phase, each packet needs to carry a virtual-circuit identifier to show which virtual-circuit that particular packet follows.

14. A ***datagram*** or ***virtual-circuit*** network handles packetized data. For each packet, the switch needs to consult its table to find the output port in the case of a datagram network, and to find the combination of the output port and the virtual circuit identifier in the case of a virtual-circuit network. In a ***circuit-switched*** network, data are not packetized; no routing information is carried with the data. The whole path is established during the setup phase.

15. In ***circuit-switched*** and ***virtual-circuit*** networks, we are dealing with connections. A connection needs to be made before the data transfer can take place. In the case of a circuit-switched network, a physical connection is established during the setup phase and the is broken during the teardown phase. In the case of a virtual-circuit network, a virtual connection is made during setup and is broken during the teardown phase; the connection is virtual, because it is an entry in the table. These two types of networks are considered ***connection-oriented***. In the case of a ***datagram*** network no connection is made. Any time a switch in this type of network receives a packet, it consults its table for routing information. This type of network is considered a ***connectionless*** network.

16. The switching or routing in a ***datagram network*** is based on the final destination address, which is global. The minimum number of entries is two; one for the final destination and one for the output port. Here the input port, from which the packet has arrived is irrelevant. The switching or routing in a ***virtual-circuit*** network is based on the virtual circuit identifier, which has a local jurisdiction. This means that two different input or output ports may use the same virtual circuit number. Therefore, four pieces of information are required: input port, input virtual circuit number, output port, and output virtual circuit number.

17.

Packet 1: **2**
Packet 2: **3**
Packet 3: **3**
Packet 4: **2**

18.

Packet 1: **2, 70**
Packet 2: **1, 45**
Packet 3: **3, 11**
Packet 4: **4, 41**

19.

a. In a ***datagram*** network, the destination addresses are unique. They cannot be duplicated in the routing table.

b. In a ***virtual-circuit*** network, the VCIs are local. A VCI is unique only in relationship to a port. In other words, the (port, VCI) combination is unique. This means that we can have two entries with the same input or output ports. We can

have two entries with the same VCIs. However, we cannot have two entries with the same (port, VCI) pair.

20. When a packet arrives at a router in a ***datagram*** network, the only information in the packet that can help the router in its routing is the ***destination address*** of the packet. The table then is sorted to make the searching faster. Today's routers use some sophisticated searching techniques. When a packet arrives at a switch in a ***virtual-circuit*** network, the pair ***(input port, input VCI)*** can uniquely determined how the packet is to be routed; the pair is the only two pieces of information in the packet that is used for routing. The table in the virtual-circuit switch is sorted based on the this pair. However, since the number of port numbers is normally much smaller than the number of virtual circuits assigned to each port, sorting is done in two steps: first according to the input port number and second according to the input VCI.

21.

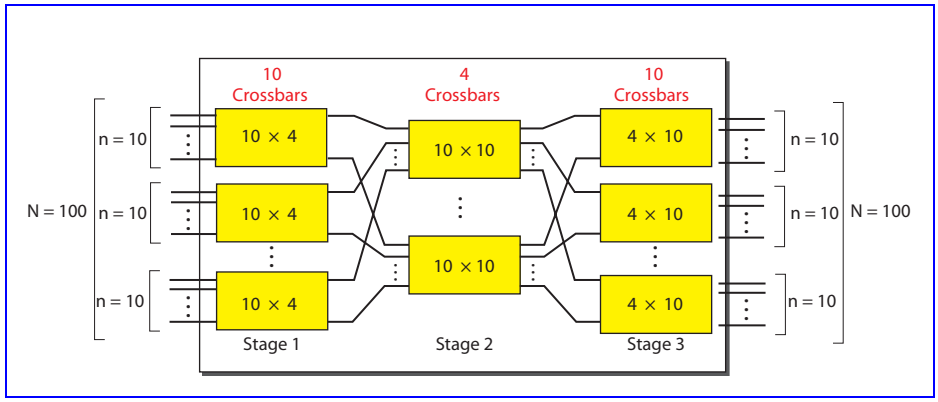
a. If $n > k$, an $n \times k$ crossbar is like a ***multiplexer*** that combines n inputs into k outputs. However, we need to know that a regular multiplexer discussed in Chapter 6 is $n \times 1$.

b. If $n < k$, an $n \times k$ crossbar is like a ***demultiplexer*** that divides n inputs into k outputs. However, we need to know that a regular demultiplexer discussed in Chapter 6 is $1 \times n$.

22.

a. See Figure 8.1.

Figure 8.1 *Solution to Exercise 22 Part a*



b. The total number of crosspoints are

$$\text{Number of crosspoints} = 10\ (10 \times 4) + 4\ (10 \times 10) + 10\ (4 \times 10) = \mathbf{1200}$$

c. Only four simultaneous connections are possible for each crossbar at the first stage. This means that the total number of simultaneous connections is **40.**

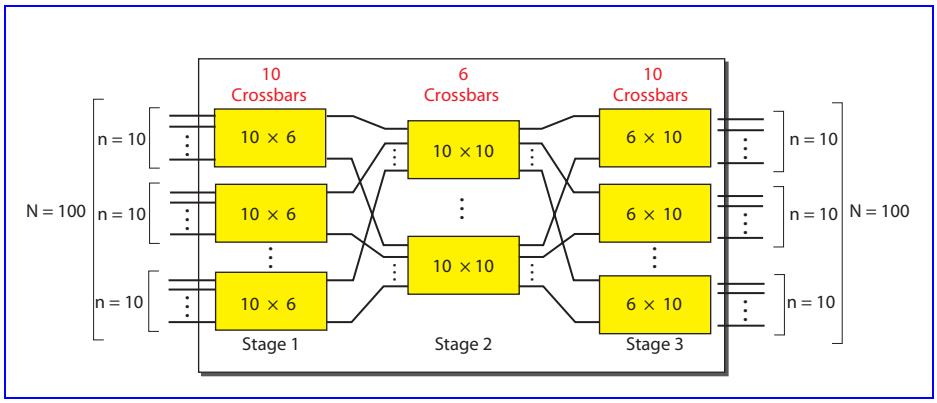
d. If we use one crossbar (100×100), all input lines can have a connection at the same time, which means **100** simultaneous connections.

e. The blocking factor is 40/100 or ***40 percent***.

23.

a. See Figure 8.2.

Figure 8.2 *Solution to Exercise 23 Part a*



b. The total number of crosspoints are

$$\text{Number of crosspoints} = 10\ (10 \times 6) + 6\ (10 \times 10) + 10\ (6 \times 10) = \mathbf{1800}$$

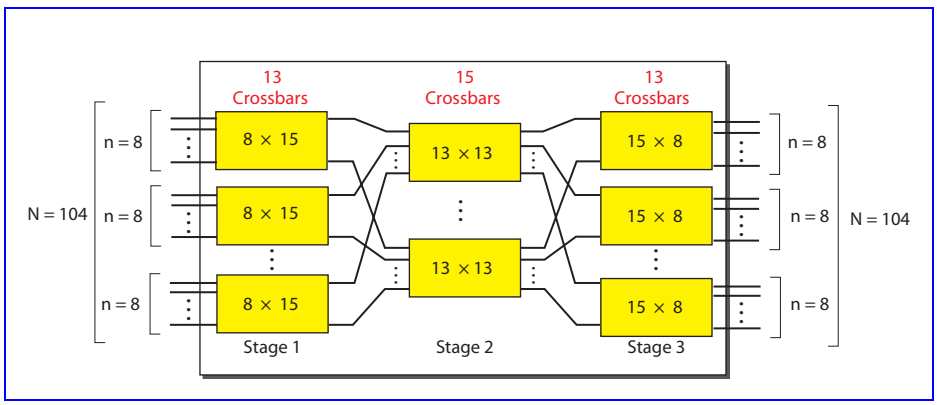
c. Only six simultaneous connections are possible for each crossbar at the first stage. This means that the total number of simultaneous connections is **60**.

d. If we use one crossbar (100 × 100), all input lines can have a connection at the same time, which means **100** simultaneous connections.

e. The blocking factor is 60/100 or ***60 percent.***

24. According to Clos, $n = (N/2)^{1/2} = 7.07$. We can choose n = 8. The number of crossbars in the first stage can be 13 (to have similar crossbars). Some of the input lines can be left unused. We then have $k = 2n - 1 = 15$. Figure 8.3 shows the configuration.

Figure 8.3 *Solution to Exercise 24 Part a*



We can calculate the total number of crosspoints as

$$13\ (8 \times 15) + 15\ (13 \times 13) + 13\ (15 \times 8) = \textbf{\textit{5655}}$$

The number of crosspoints is still much less than the case with one crossbar (10,000). We can see that there is no blocking involved because each 8 input line has 15 intermediate crossbars. The total number of crosspoints here is a little greater than the minimum number of crosspoints according to Clos using the formula $4N[(2N)^{1/2} - 1]$, which is 5257.

25.

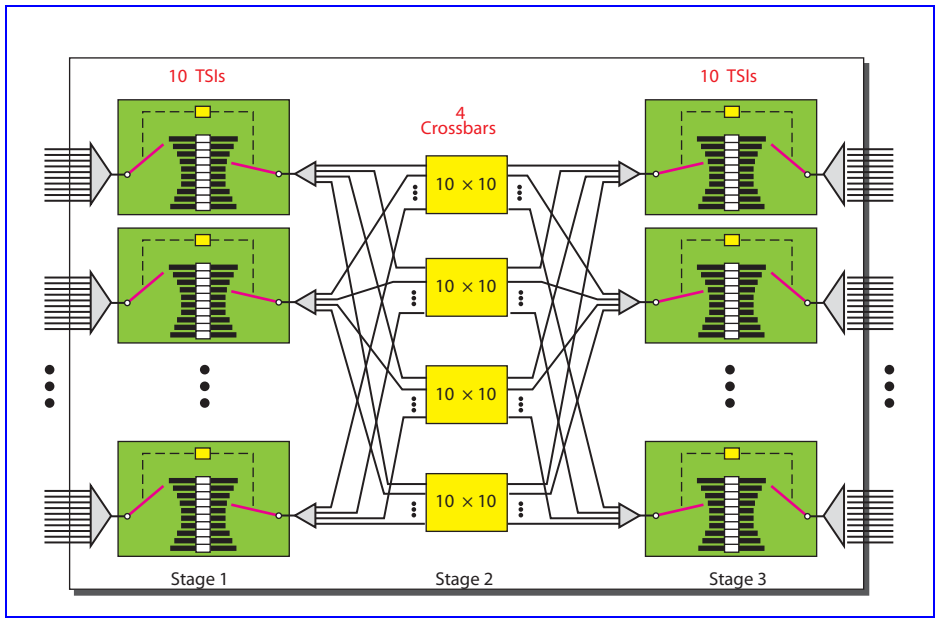
a. Total crosspoints = $N^2 = 1000^2 =$ ***1,000,000***

b. Total crosspoints $\geq 4N[(2N)^{1/2} - 1] \geq$ ***174,886.*** With less than 200,000 crosspoints we can design a three-stage switch. We can use $n = (N/2)^{1/2} = 23$ and choose k = 45. The total number of crosspoints is ***178,200***.

26. We give two solutions.

a. We first solve the problem using only crossbars and then we replace the crossbars at the first and the last stage with TSIs. Figure 8.1 shows the solution using only crossbars. We can replace the crossbar at the first and third stages with TSIs as shown in Figure 8.4. The total number of crosspoints is ***400*** and the total number of memory locations is ***200***. Each TSI at the first stage needs one TDM multiplexer and one TDM demultiplexer. The multiplexer is 10×1, but the demultiplexer is 1×4. In other words, the input frame has 10 slots and the output frame has only 4 slots. The data in the first slot of all input TSIs are directed to the first switch, the output in the second slot are directed to the second switch, and so on.

Figure 8.4 *First solution to Exercise 26*



b. We can see the inefficiency in the first solution. Since the slots are separated in time, only one of the switches at the middle stage is active at each moment. This means that, instead of 4 crossbars, we could have used only one with the same result. Figure 8.5 shows the new design. In this case we still need ***200*** memory locations but only ***100*** crosspoints.

Figure 8.5 *Second solution to Exercise 26*