Lecture 12

CHAPTER 6

REGISTERS

- A *register* is a group of flip-flops, each one of which shares a common clock and is capable of storing one bit of information.
- An *n* -bit register consists of a group of *n* flip-flops capable of storing *n* bits of binary information.
- In addition to the flip-flops, a register may have combinational gates that perform certain data-processing tasks.
- In its broadest definition, a register consists of a group of flip-flops together with gates that affect their operation.
- The flip-flops hold the binary information, and the gates determine how the information is transferred into the register.

REGISTERS

- The simplest register is one that consists of only flip-flops, without any gates.
- Figure 6.1 shows such a register constructed with four *D* -type flip-flops to form a four-bit data storage register.
- The common clock input triggers all flip-flops on the positive edge of each pulse, and the binary data available at the four inputs are transferred into the register.
- The value of (*I* 3 , *I* 2 , *I* 1 , *I* 0) immediately before the clock edge determines the value of (*A* 3 , *A* 2 , *A* 1 , *A* 0) after the clock edge.
- The input *Clear_b* goes to the active-low *R* (reset) input of all four flip-flops. When this input goes to 0, all flip-flops are reset.

REGISTERS

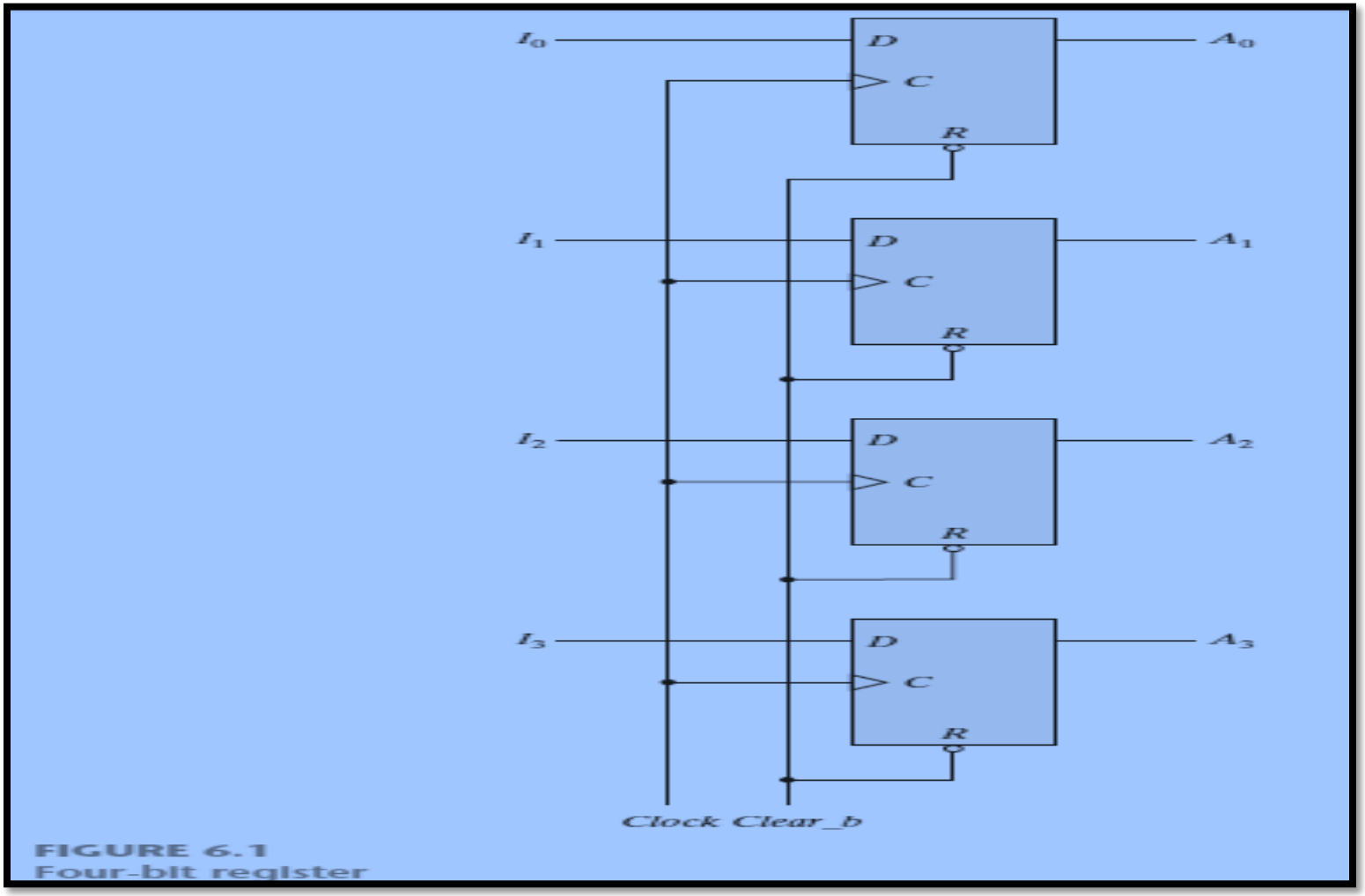

FIGURE 6.1
Four-bit register

Register with Parallel Load

- The transfer of new information into a register is referred to as *loading* or *updating* the register.
- If all the bits of the register are loaded simultaneously with a common clock pulse, we say that the loading is done *in parallel* .
- A clock edge applied to the *C* inputs of the register of Fig. 6.1 will load all four inputs in parallel.
- In this configuration, if the contents of the register must be left unchanged, the inputs must be held constant or the clock must be inhibited from the circuit.

Register with Parallel Load

- A four-bit data-storage register with a load control input that is directed through gates and into the *D* inputs of the flip-flops is shown in Fig. 6.2 .
- The load input to the register determines the action to be taken with each clock pulse.
- When the load input is 1, the data at the four external inputs are transferred into the register with the next positive edge of the clock.
- When the load input is 0, the outputs of the flip-flops are connected to their respective inputs.
- The transfer of information from the data inputs or the outputs of the register is done simultaneously with all four bits in response to a clock edge

Register with Parallel Load

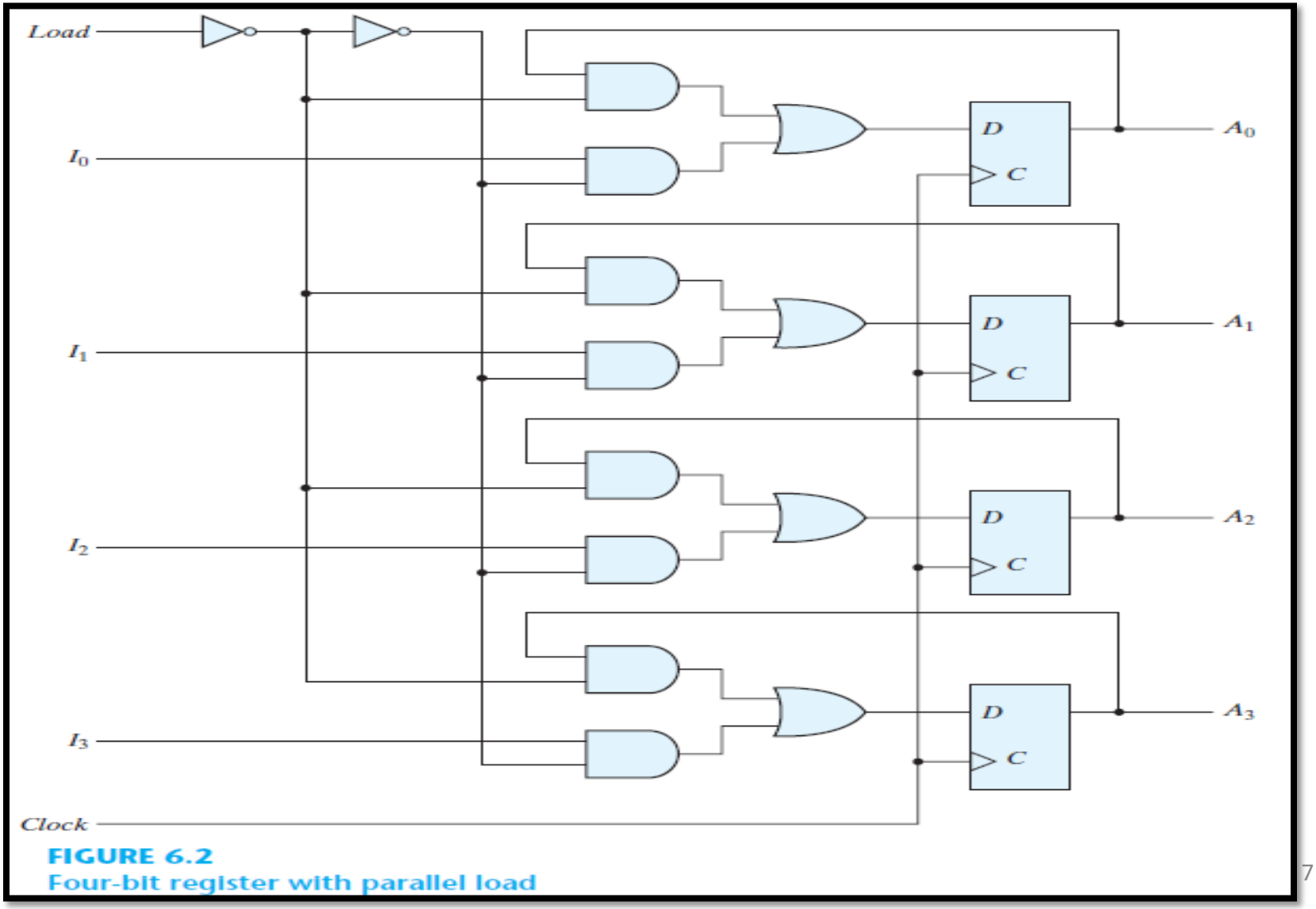

FIGURE 6.2
Four-bit register with parallel load

SHIFT REGISTERS

- A register capable of shifting the binary information held in each cell to its neighboring cell, in a selected direction, is called a *shift register.*
- The logical configuration of a shift register consists of a chain of flip-flops in cascade, with the output of one flip-flop connected to the input of the next flip-flop.
- All flip-flops receive common clock pulses, which activate the shift of data from one stage to the next.
- The simplest possible shift register is one that uses only flip-flops, as shown in Fig. 6.3 .
- The output of a given flip-flop is connected to the *D* input of the flip-flop at its right.
- This shift register is unidirectional (left-to-right).

SHIFT REGISTERS

- Each clock pulse shifts the contents of the register one bit position to the right.
- The configuration does not support a left shift.
- The *serial input* determines what goes into the leftmost flip-flop during the shift.
- The *serial output* is taken from the output of the rightmost flip-flop.
- It will be shown later that the shift operation can be controlled through the *D* inputs of the flip-flops rather than through the clock input.
- If, however, the shift register of Fig. 6.3 is used, the shift can be controlled with an input by connecting the clock through an AND gate.

SHIFT REGISTERS

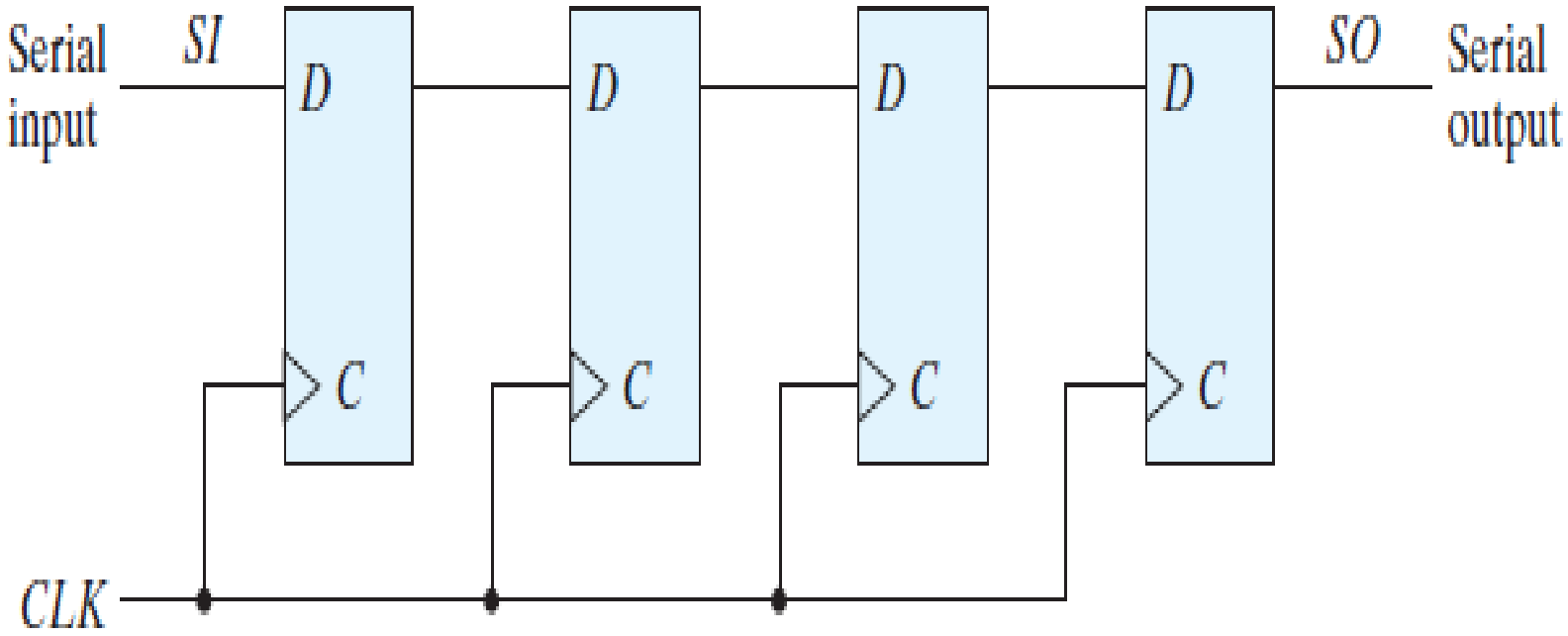

FIGURE 6.3

Four-bit shift register

SHIFT REGISTERS (Serial Transfer)

- The datapath of a digital system is said to operate in serial mode when information is transferred and manipulated one bit at a time.
- Information is transferred one bit at a time by shifting the bits out of the source register and into the destination register.
- This type of transfer is in contrast to parallel transfer, whereby all the bits of the register are transferred at the same time.
- The serial transfer of information from register *A* to register *B* is done with shift registers, as shown in the block diagram of Fig. 6.4 (a).
- The serial output (*SO*) of register *A* is connected to the serial input (*SI*) of register *B.*

SHIFT REGISTERS (Serial Transfer)

- To prevent the loss of information stored in the source register, the information in register *A* is made to circulate by connecting the serial output to its serial input.
- The initial content of register *B* is shifted out through its serial output and is lost unless it is transferred to a third shift register.
- The shift control input determines when and how many times the registers are shifted.
- For illustration here, this is done with an AND gate that allows clock pulses to pass into the *CLK* terminals only when the shift control is active.
- Suppose the shift registers in Fig. 6.4 have four bits each.

SHIFT REGISTERS (Serial Transfer)

- Then the control unit that supervises the transfer of data must be designed in such a way that it enables the shift registers, through the shift control signal, for a fixed time of four clock pulses in order to pass an entire word.
- This design is shown in the timing diagram of Fig. 6.4 (b).
- The shift control signal is synchronized with the clock and changes value just after the negative edge of the clock.
- The next four clock pulses find the shift control signal in the active state, so the output of the AND gate connected to the *CLK* inputs produces four pulses: *T*1, *T*2, *T*3, and *T*4.
- Each rising edge of the pulse causes a shift in both registers.
- The fourth pulse changes the shift control to 0, and the shift registers are disabled.

SHIFT REGISTERS (Serial Transfer)

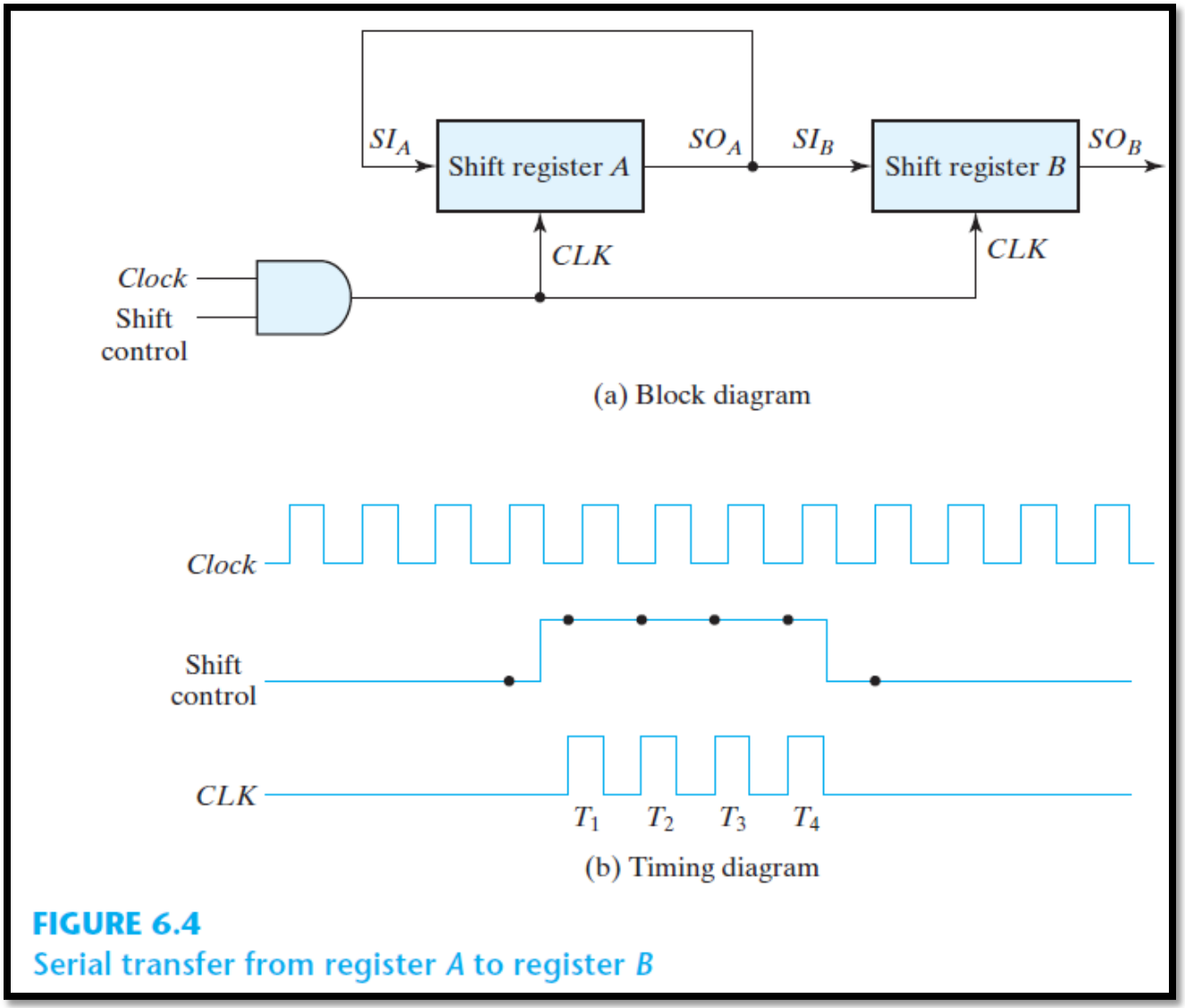

FIGURE 6.4
Serial transfer from register *A* to register *B*

SHIFT REGISTERS (Serial Transfer)

Table 6.1
Serial-Transfer Example

Timing Pulse	Shift Register *A*	Shift Register *B*
Initial value	1 0 1 1	0 0 1 0
After T_1	1 1 0 1	1 0 0 1
After T_2	1 1 1 0	1 1 0 0
After T_3	0 1 1 1	0 1 1 0
After T_4	1 0 1 1	1 0 1 1

SHIFT REGISTERS (Serial Addition)

- Operations in digital computers are usually done in parallel because that is a faster mode of operation.
- Serial operations are slower because a datapath operation takes several clock cycles, but serial operations have the advantage of requiring fewer hardware components.
- To demonstrate the serial mode of operation, we present the design of a serial adder.
- The two binary numbers to be added serially are stored in two shift registers.
- Beginning with the least significant pair of bits, the circuit adds one pair at a time through a single full-adder (FA) circuit, as shown in Fig. 6.5 .

SHIFT REGISTERS (Serial Addition)

- The carry out of the full adder is transferred to a *D* flip-flop, the output of which is then used as the carry input for the next pair of significant bits.
- The sum bit from the *S* output of the full adder could be transferred into a third shift register.
- By shifting the sum into *A* while the bits of *A* are shifted out, it is possible to use one register for storing both the augend and the sum bits.
- The serial input of register *B* can be used to transfer a new binary number while the addend bits are shifted out during the addition.

SHIFT REGISTERS (Serial Addition)

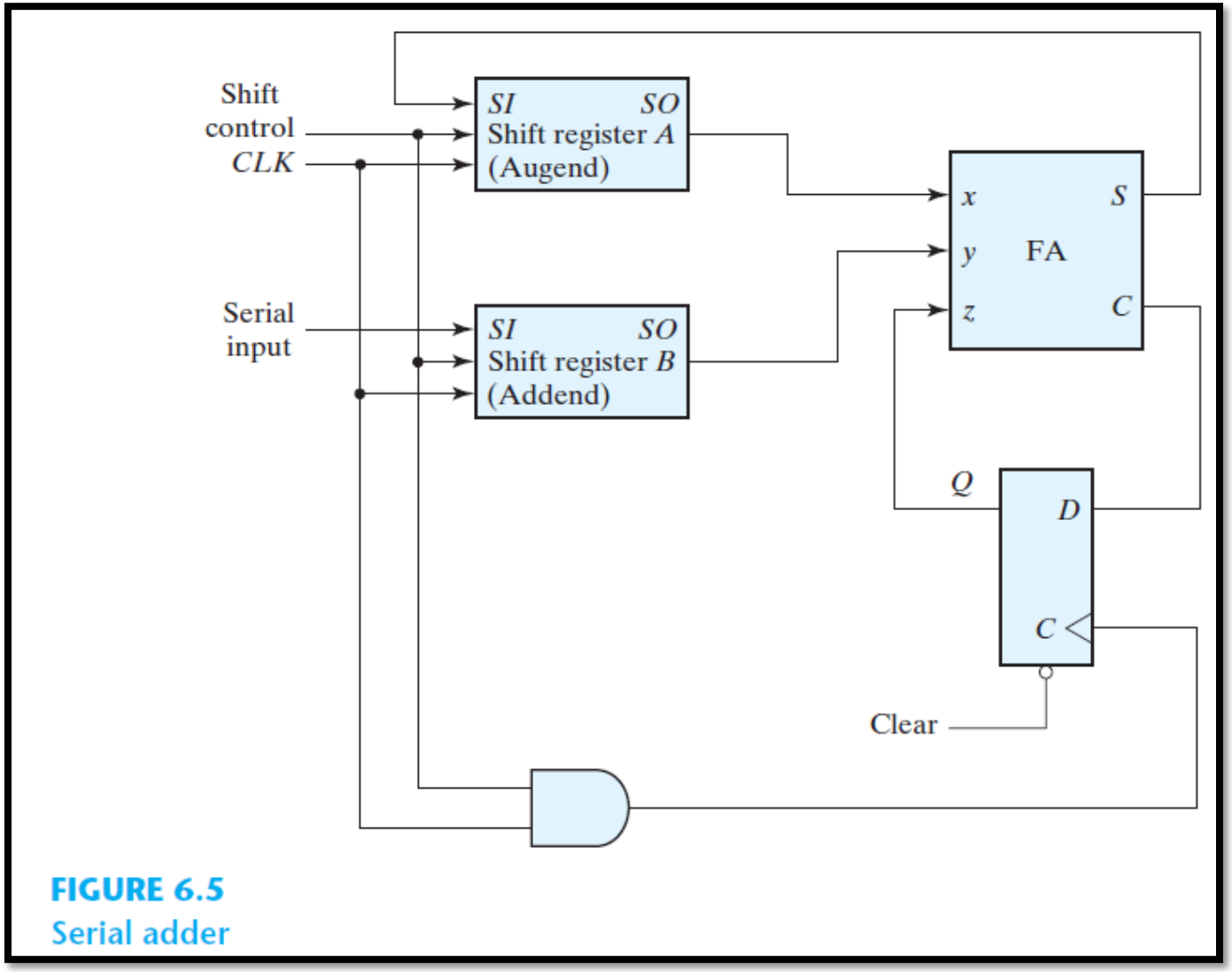

FIGURE 6.5
Serial adder

SHIFT REGISTERS (Serial Addition)

- The operation of the serial adder is as follows: Initially, register *A* holds the augend, register *B* holds the addend, and the carry flip-flop is cleared to 0.
- The outputs (*SO*) of *A* and *B* provide a pair of significant bits for the full adder at *x* and *y.*
- Output *Q* of the flip-flop provides the input carry at *z.*
- The shift control enables both registers and the carry flip-flop, so at the next clock pulse, both registers are shifted once to the right, the sum bit from *S* enters the leftmost flip-flop of *A*, and the output carry is transferred into flip-flop *Q.*

SHIFT REGISTERS (Serial Addition)

- The shift control enables the registers for a number of clock pulses equal to the number of bits in the registers.
- For each succeeding clock pulse, a new sum bit is transferred to *A*, a new carry is transferred to *Q*, and both registers are shifted once to the right.
- This process continues until the shift control is disabled.
- Thus, the addition is accomplished by passing each pair of bits together with the previous carry through a single full-adder circuit and transferring the sum, one bit at a time, into register *A*.

SHIFT REGISTERS (Serial Addition)

- To show that serial operations can be designed by means of sequential circuit procedure, we will redesign the serial adder with the use of a state table.
- The state table that specifies the sequential circuit is listed in Table 6.2 .
- The present state of *Q* is the present value of the carry.
- The present carry in *Q* is added together with inputs *x* and *y* to produce the sum bit in output *S.*
- The next state of *Q* is equal to the output carry.
- Note that the state table entries are identical to the entries in a full-adder truth table, except that the input carry is now the present state of *Q* and the output carry is now the next state of *Q.*

SHIFT REGISTERS (Serial Addition)

Table 6.2
State Table for Serial Adder

Present State	Inputs		Next State	Output	Flip-Flop Inputs	
Q	x	y	Q	S	J_Q	K_Q
0	0	0	0	0	0	X
0	0	1	0	1	0	X
0	1	0	0	1	0	X
0	1	1	1	0	1	X
1	0	0	0	1	X	1
1	0	1	1	0	X	0
1	1	0	1	0	X	0
1	1	1	1	1	X	0

SHIFT REGISTERS (Serial Addition)

- If a *D* flip-flop is used for *Q*, the circuit reduces to the one shown in Fig. 6.5 .
- If a *JK* flipflop is used for *Q*, it is necessary to determine the values of inputs *J* and *K* by referring to the excitation table (Table 5.12).
- This is done in the last two columns of Table 6.2 .
- The two flip-flop input equations and the output equation can be simplified by means of maps to
- $J_Q = xy$
- $K_Q = x'y' = (x + y)'$
- $S = x \otimes y \otimes Q$

SHIFT REGISTERS (Serial Addition)

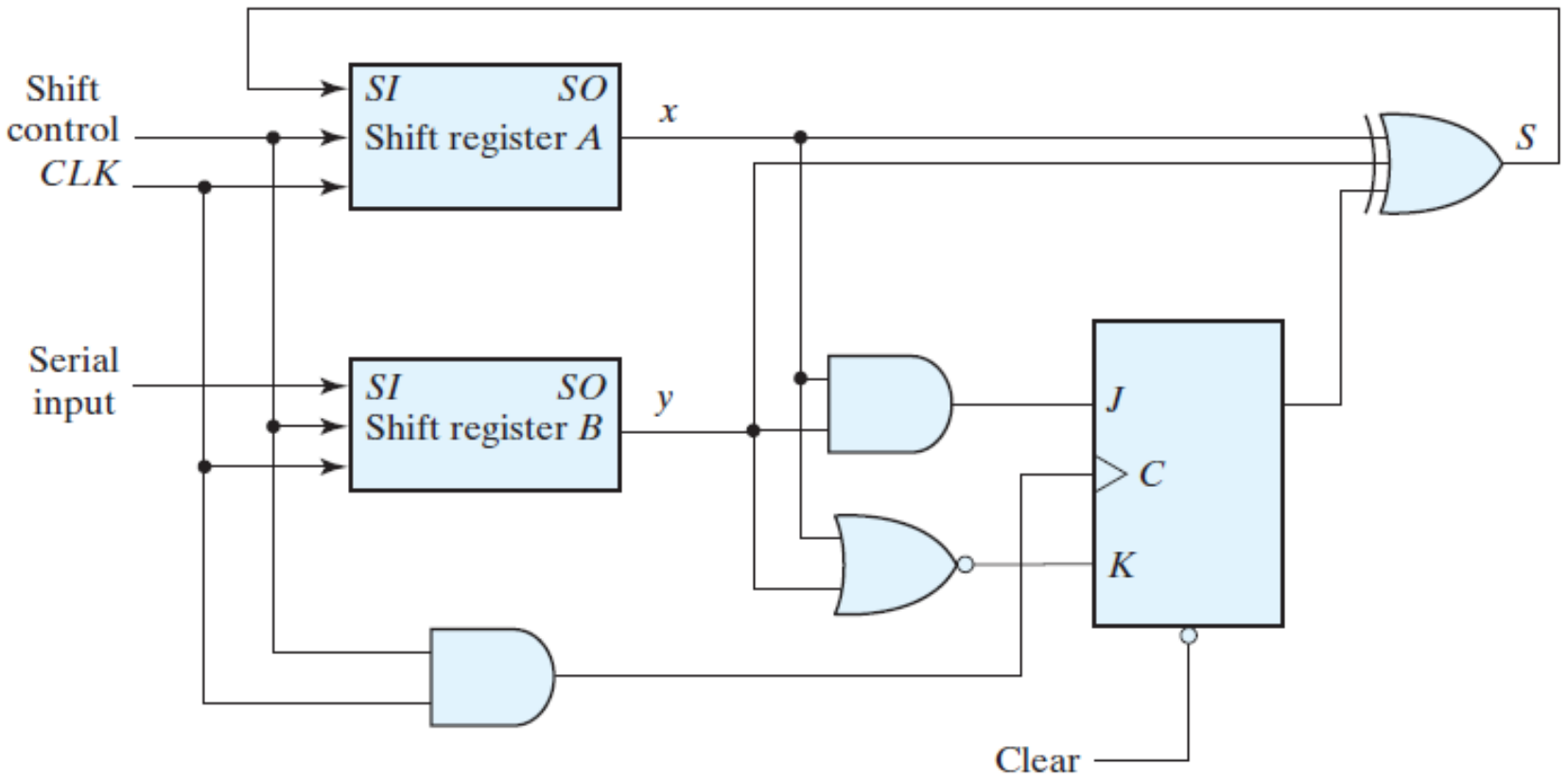

FIGURE 6.6
Second form of serial adder

SHIFT REGISTERS
(Universal Shift Register)

- **1.** A *clear* control to clear the register to 0.
- **2.** A *clock* input to synchronize the operations.
- **3.** A *shift-right* control to enable the shift-right operation and the *serial input* and *output* lines associated with the shift right.
- **4.** A *shift-left* control to enable the shift-left operation and the *serial input* and *output* lines associated with the shift left.
- **5.** A *parallel-load* control to enable a parallel transfer and the *n* input lines associated with the parallel transfer.
- **6.** *n* parallel output lines.
- **7.** A control state that leaves the information in the register unchanged in response to the clock. Other shift registers may have only some of the preceding functions, with at least one shift operation.

SHIFT REGISTERS
(Universal Shift Register)

- A register capable of shifting in one direction only is a *unidirectional* shift register.
- One that can shift in both directions is a *bidirectional* shift register.
- If the register has both shifts and parallel-load capabilities, it is referred to as a *universal shift register.*
- The block diagram is shown in Fig. 6.7 .
- The circuit consists of four *D* flip-flops and four multiplexers.
- The four multiplexers have two common selection inputs *s*1 and *s*0.
- Input 0 in each multiplexer is selected when *s*1*s*0 = 00, input 1 is selected when *s*1*s*0 = 01, and similarly for the other two inputs.

SHIFT REGISTERS
(Universal Shift Register)

- The selection inputs control the mode of operation of the register according to the function entries in Table 6.3 .
- When *s*1*s*0 = 00, the present value of the register is applied to the *D* inputs of the flip-flops.
- This condition forms a path from the output of each flip-flop into the input of the same flip-flop, so that the output recirculates to the input in this mode of operation.
- The next clock edge transfers into each flip-flop the binary value it held previously, and no change of state occurs.

SHIFT REGISTERS
(Universal Shift Register)

- When *s*1*s*0 = 01, terminal 1 of the multiplexer inputs has a path to the *D* inputs of the flip-flops.
- This causes a shift-right operation, with the serial input transferred into flip-flop *A*3.
- When *s*1*s*0 = 10, a shift-left operation results, with the other serial input going into flip-flop *A*0.
- Finally, when *s*1*s*0 = 11, the binary information on the parallel input lines is transferred into the register simultaneously during the next clock edge.
- Note that data enters *MSB_in* for a shift-right operation and enters *LSB_in* for a shift-left operation.
- *Clear_b* is an active-low signal that clears all of the flip-flops.

SHIFT REGISTERS
(Universal Shift Register)

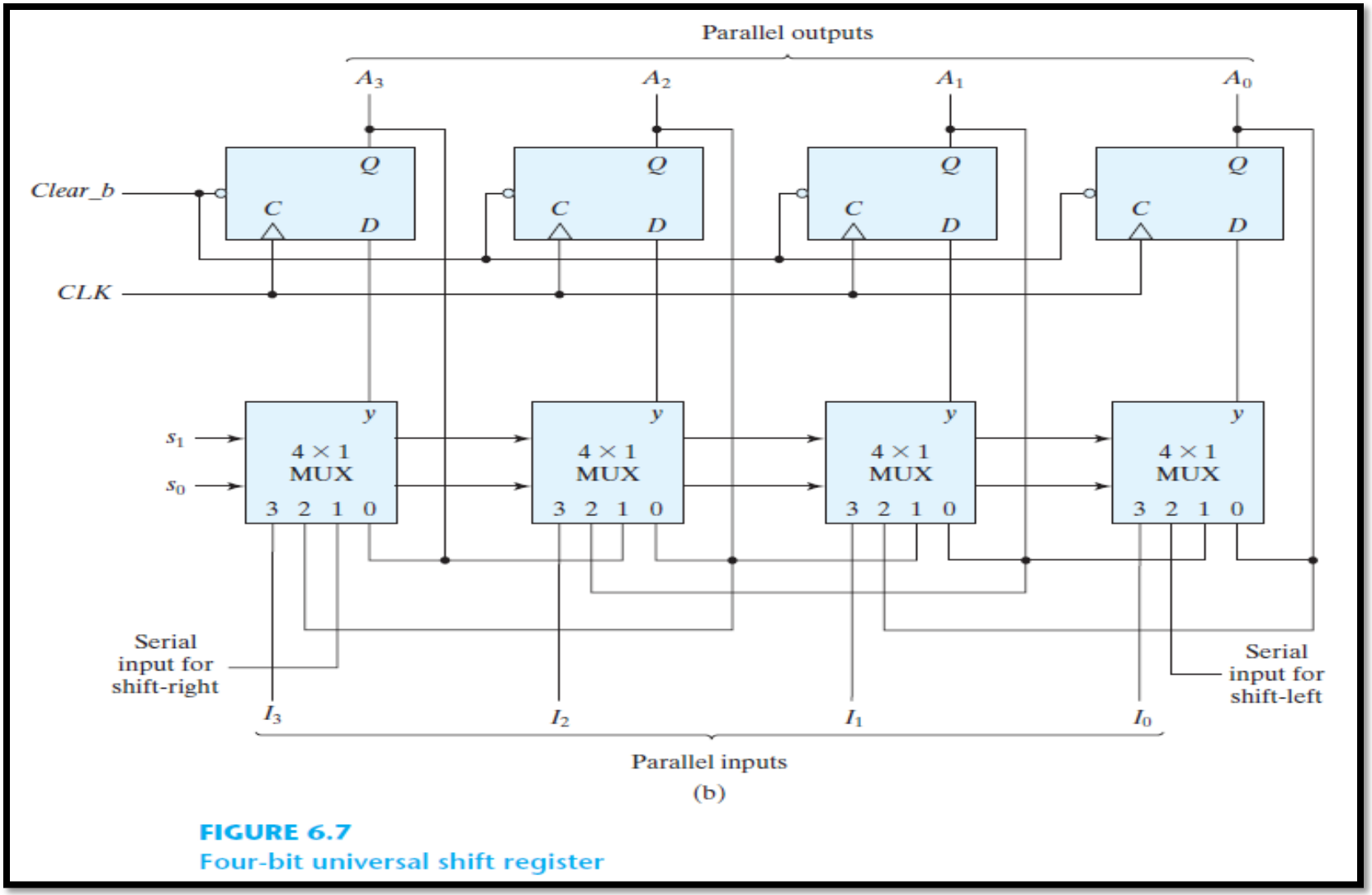

FIGURE 6.7
Four-bit universal shift register

SHIFT REGISTERS
(Universal Shift Register)

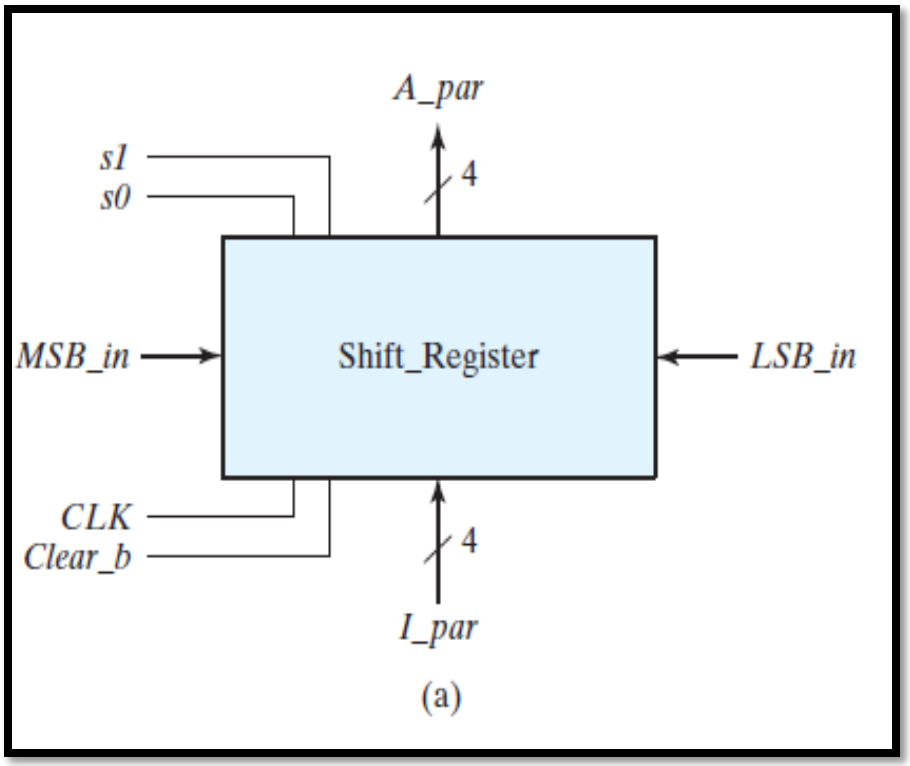

Table 6.3
Function Table for the Register of Fig. 6.7

Mode Control		
s_1	s_0	Register Operation
0	0	No change
0	1	Shift right
1	0	Shift left
1	1	Parallel load