EC2.101 – Digital Systems and Microcontrollers

Lecture 22 – Registers and Counters 2

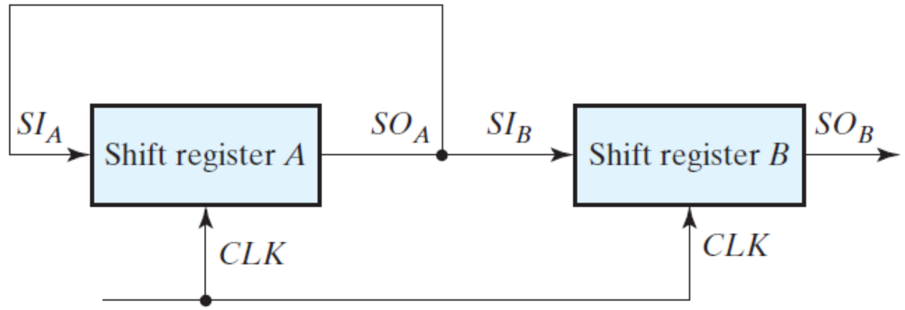
Dr. Aftab M. Hussain,

Assistant Professor, PATRIoT Lab, CVEST

Chapter 6

Serial transfer

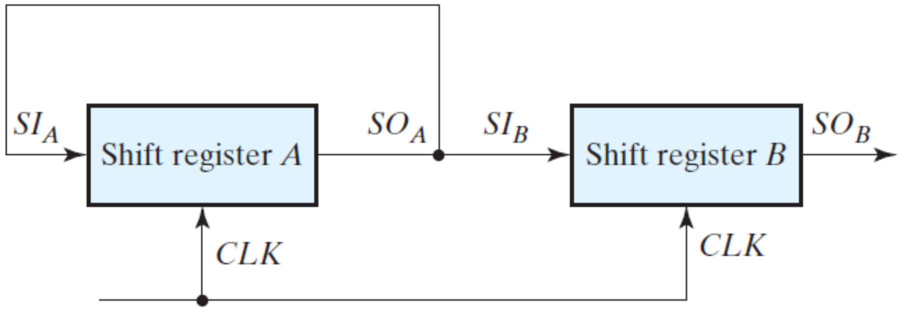
- The datapath of a digital system is said to operate in serial mode when information is transferred and manipulated one bit at a time
- Information is transferred one bit at a time by shifting the bits out of the source register and into the destination register
- This type of transfer is in contrast 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
- The serial output (*SO*) of register *A* is connected to the serial input (*SI*) of register *B*



(a) Block diagram

Serial transfer

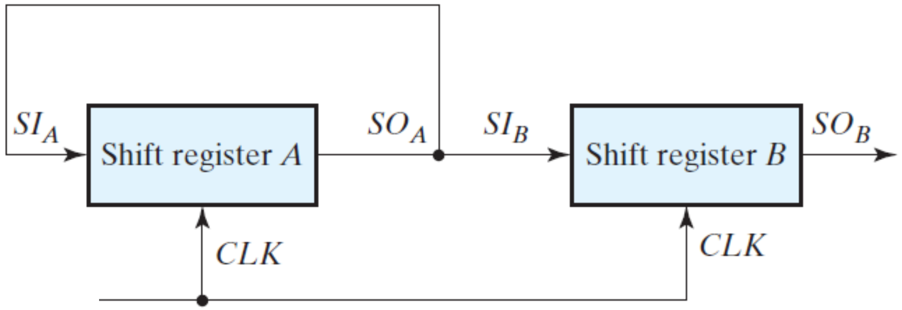
- To prevent the loss of information stored in the source register, the information in register *A* is made to circulate by connecting the serial output to its serial input
- The initial content of register *B* is shifted out through its serial output and is lost unless it is transferred to a third shift register
- The shift control input determines when and how many times the registers are shifted
- For simplicity 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



(a) Block diagram

Serial transfer

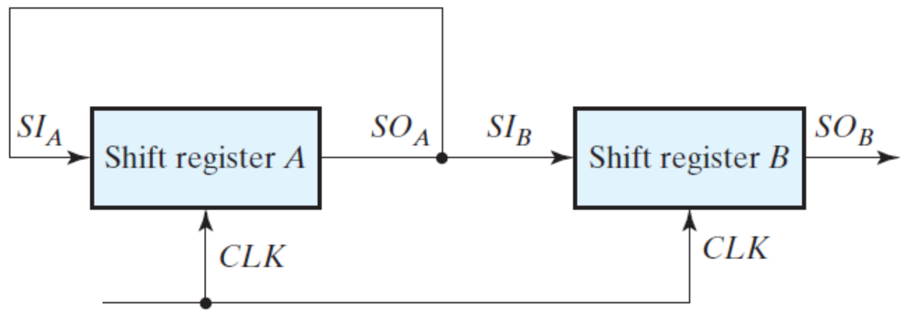
- Assume that the binary content of *A* before the shift is 1011 and that of *B* is 0010
- The serial transfer from *A* to *B* occurs in four steps
- With the first pulse, T_1, the rightmost bit of *A* is shifted into the leftmost bit of *B* and is also circulated into the leftmost position of *A*
- At the same time, all bits of *A* and *B* are shifted one position to the right



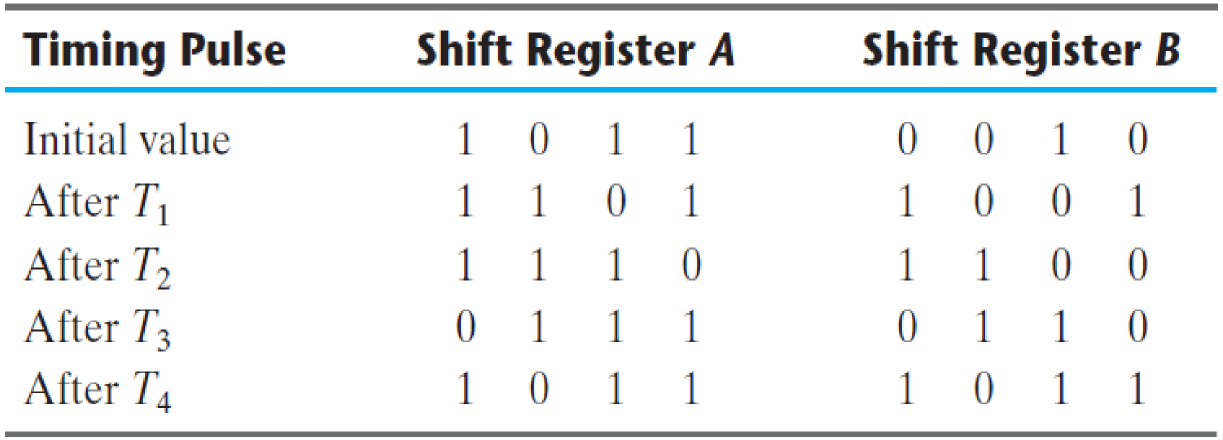
(a) Block diagram

Serial transfer

- The previous serial output from *B* in the rightmost position is lost, and its value changes from 0 to 1
- The next three pulses perform identical operations, shifting the bits of *A* into *B,* one at a time
- After the fourth shift, the shift control goes to 0, and registers *A* and *B* both have the value 1011
- Thus, the contents of *A* are copied into *B*, so that the contents of *A* remain unchanged i.e., the contents of *A* are restored to their original value



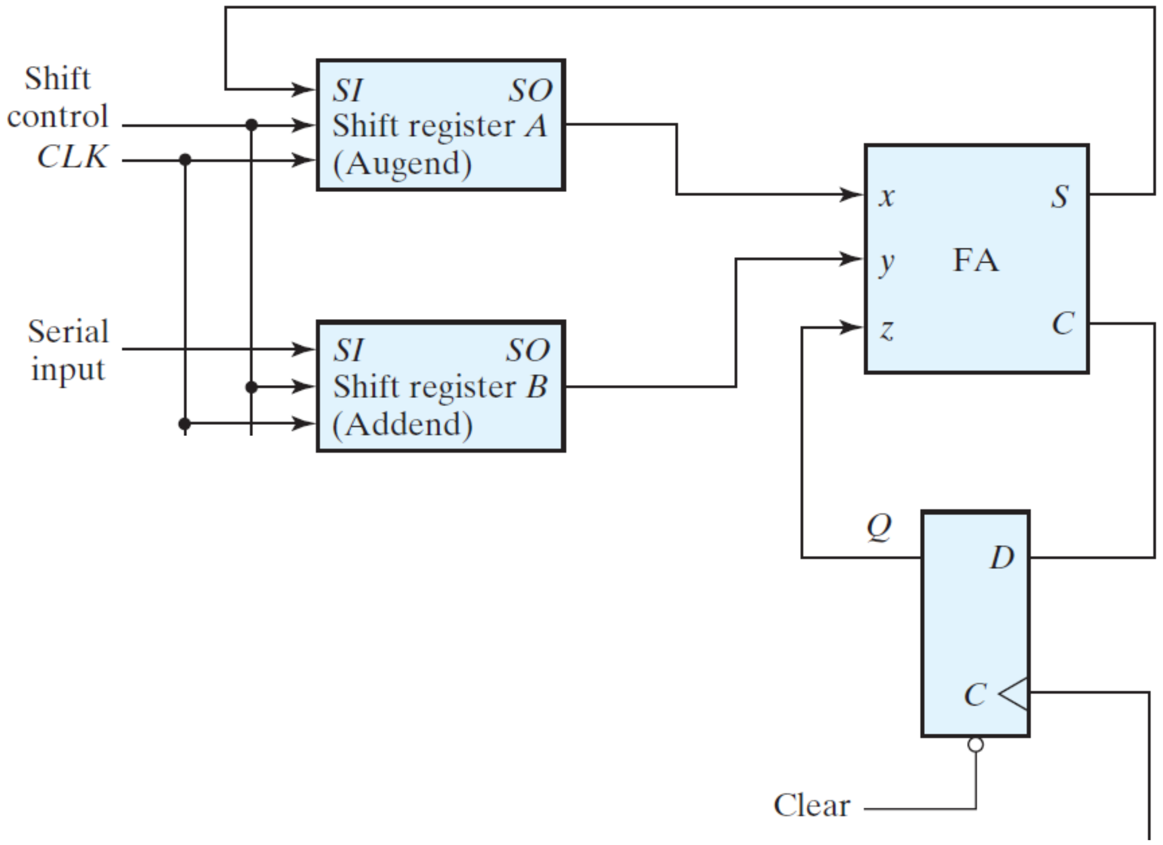
(a) Block diagram



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

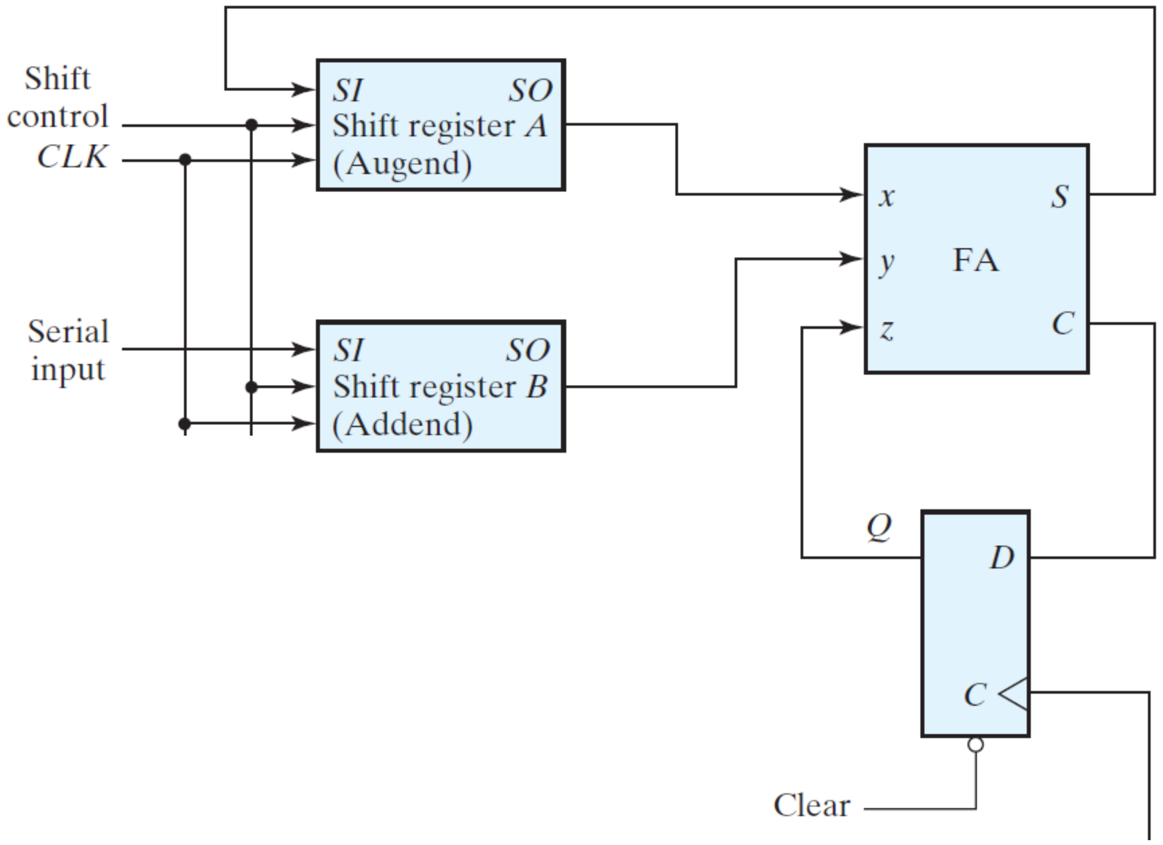
Serial addition

- Can we design a serial addition circuit? The two binary numbers to be added serially are stored in two shift registers
- This is similar to the algorithm we use for adding manually
- Beginning with the least significant pair of bits, the circuit adds one pair at a time through a single full-adder (FA) circuit
- 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



Serial addition

- Comparing the serial adder with the parallel adder (4-bit adder), we note several differences
- The parallel adder uses registers with a parallel load, whereas the serial adder uses shift registers
- The number of full-adder circuits in the parallel adder is equal to the number of bits in the binary numbers, whereas the serial adder requires only one full-adder circuit and a carry flip-flop
- Excluding the registers, the parallel adder is a combinational circuit, whereas the serial adder is a sequential circuit which consists of a full adder and a flip-flop that stores the output carry



Universal register

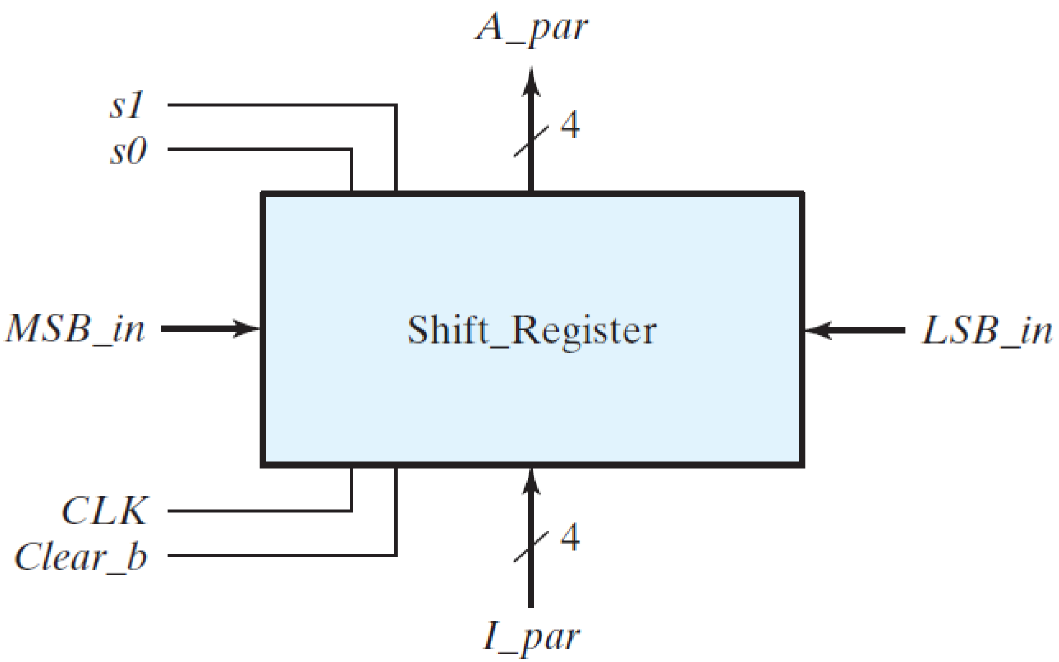
- If the flip-flop outputs of a shift register are accessible, then information entered serially by shifting can be taken out in parallel from the outputs of the flip-flops
- If a parallel load capability is added to a shift register, then data entered in parallel can be taken out in serial fashion by shifting the data stored in the register
- Some shift registers provide the necessary input and output terminals for parallel transfer
- They may also have both shift-right and shift-left capabilities

Universal register

- The most general shift register has the following capabilities:

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

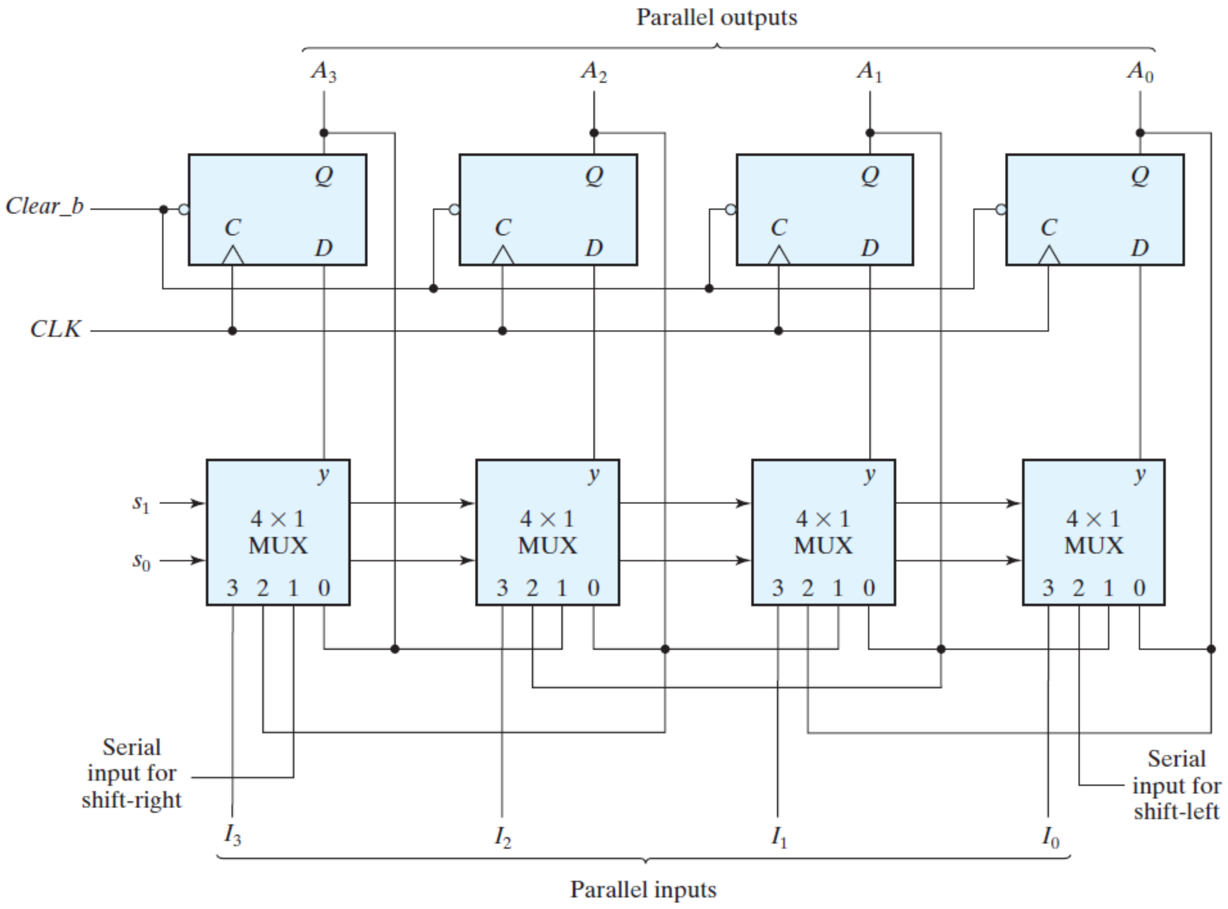
Universal register



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

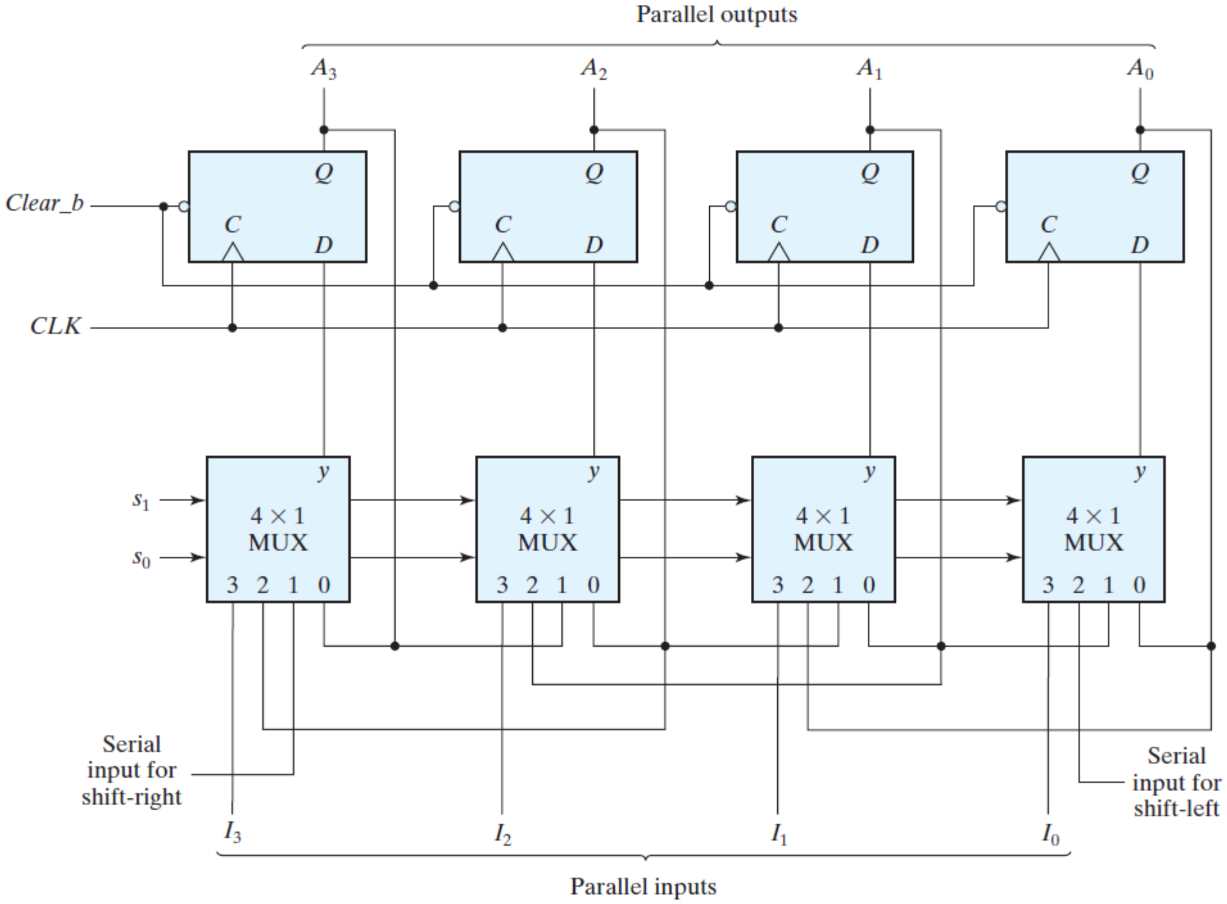
Universal register

- The circuit consists of four *D* flip-flops and four multiplexers
- The four multiplexers have two common selection inputs s_1 and s_0
- The selection inputs control the mode of operation of the register according to the function required
- The output of the MUXes are applied to the inputs of the FFs which controls the next state of the FF



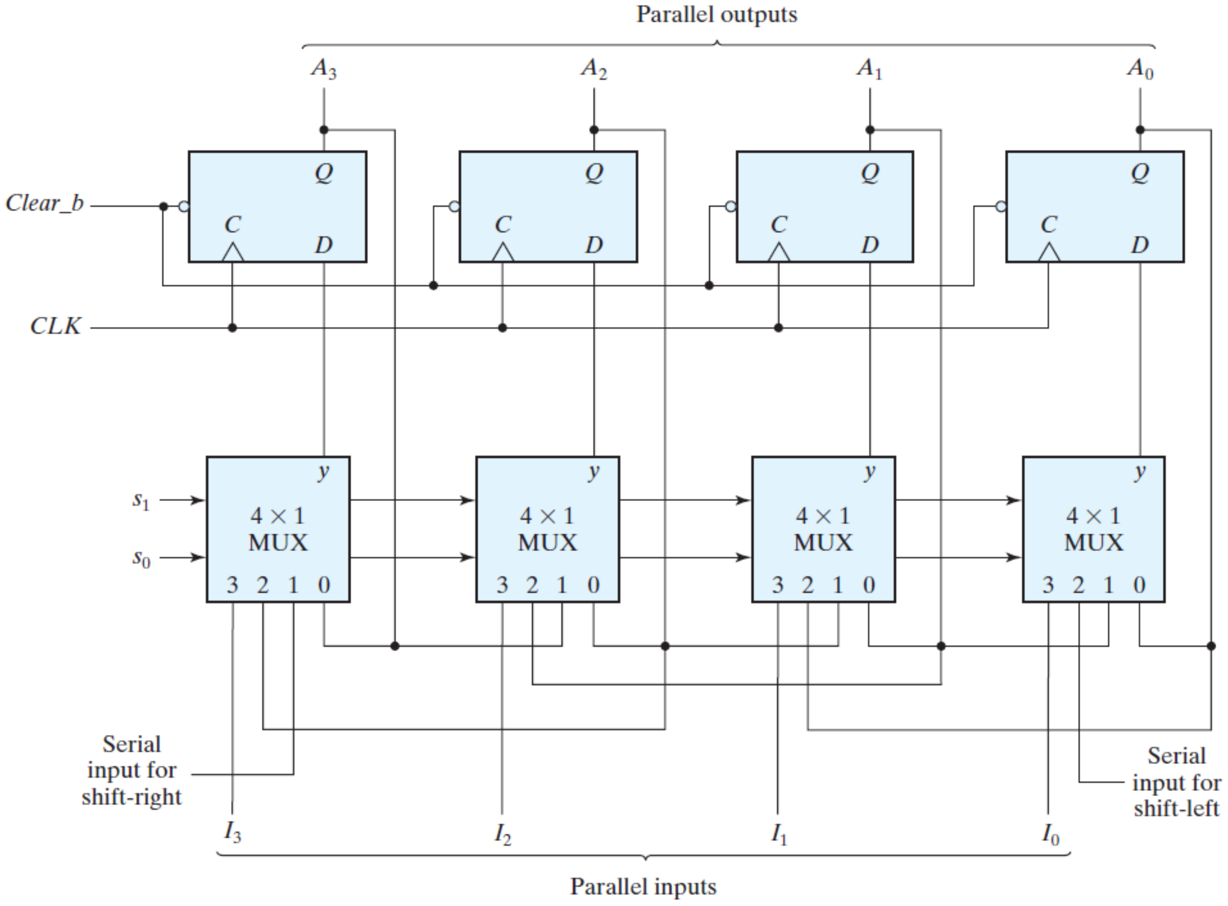
Universal register

- 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



Universal register

- When s_1s_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_1s_0 = 10, a shift-left operation results, with the other serial input going into flip-flop A_0
- Finally, when s_1s_0 = 11, the binary information on the parallel input lines is transferred into the register simultaneously during the next clock edge


Parallel outputs
A_3
A_2
A_1
A_0
Clear_b
CLK
Q
C
D
y
s_1
s_0
4 × 1 MUX
3 2 1 0
Serial input for shift-right
Serial input for shift-left
I_3
I_2
I_1
I_0
Parallel inputs