LOVELY PROFESSIONAL UNIVERSITY

CSE211

Computer Organization and Design

Lecture : 3 **Tutorial: 1** **Practical: 0** **Credit: 4**

- Introduction
- Logic Gates
- Flip Flops
- Multiplexers
- Demultiplexer
- **Binary counters**
- **Decoder**
- **Encoder**
- **Registers**

BINARY-COUNTERS

Binary counters are used for counting the number of pulses coming at the input line in a specified time period.

The binary counters must possess memory since it has to remember its past states.

As the name suggests, it is a circuit which counts.The main purpose of the counter is to record the number of occurrence of some input.

BINARY-COUNTERS

- A **binary counter** is a hardware circuit that is made out of a series of flip-flops.
- The output of one flip-flop is sent to the input of the next flip-flop in the series.
- A **binary counter** can be either asynchronous or synchronous, depending on how the flip-flops are connected together.

BINARY-DECODER

The name **"Decoder"** means to translate or decode coded information from one format into another, so a digital decoder transforms a set of digital input signals into an equivalent decimal code at its output.

In **digital electronics**, a binary **decoder** is a combinational **logic** circuit that converts binary information from the **n coded inputs to a maximum of 2^n unique outputs.**

They are used in a wide variety of applications,

- data demultiplexing,
- seven segment displays, and
- memory address **decoding**

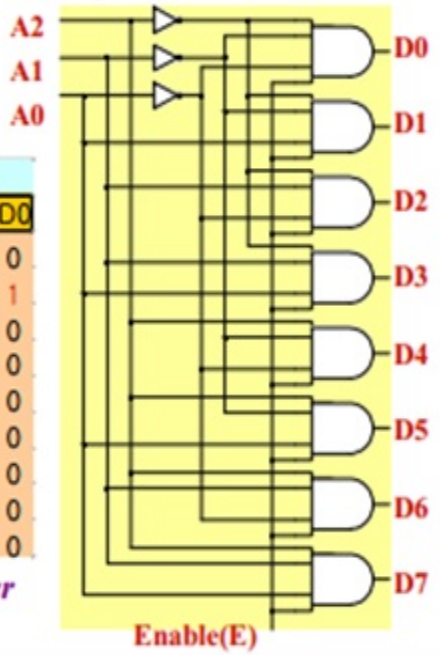
2-2 Decoder/Encoder

- Decoder
  - A combinational circuit that converts binary information from the ***n*** coded inputs to a maximum of 2^n unique outputs
  - ***n***-to-***m*** line decoder = ***n x m*** decoder
    - ***n*** inputs, ***m*** outputs
  - If the n-bit coded information has unused bit combinations, the decoder may have less than 2^n outputs
    - $m \leq 2^n$
- 3-to-8 Decoder
  - A Binary-to-octal conversion
  - Logic Diagram : *Fig. 2-1*
  - Truth Table : *Tab. 2-1*
  - Commercial decoders include one or more Enable Input(E)

Enable	Inputs			Outputs							
E	A2	A1	A0	D7	D6	D5	D4	D3	D2	D1	D0
0	x	x	x	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	1
1	0	0	1	0	0	0	0	0	0	1	0
1	0	1	0	0	0	0	0	0	1	0	0
1	0	1	1	0	0	0	0	1	0	0	0
1	1	0	0	0	0	0	1	0	0	0	0
1	1	0	1	0	0	1	0	0	0	0	0
1	1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	1	0	0	0	0	0	0	0

Tab. 2-1 Truth table for 3-to-8 Decoder

Fig. 2-1 3-to-8 Decoder



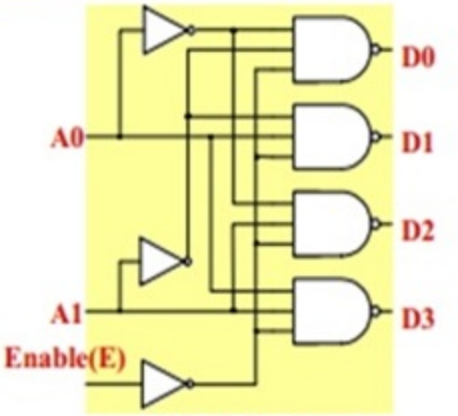
2-2 Decoder/Encoder

- NAND Gate Decoder
  - Constructed with NAND instead of AND gates
  - Logic Diagram/Truth Table : *Fig. 2-2*

* Active Low Output
* Fig. 2-1 3-to-8 Decoder ≒ Active High Output

Enable	Input		Output			
E	A1	A0	D0	D1	D2	D3
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	0
1	x	x	1	1	1	1

(a) Truth Table



(b) Logic Diagram

Fig. 2-2 2-to-4 Decoder with NAND gates

- Decoder Expansion
  - Constructed decoder : *Fig. 2-3*
  - 3 X 8 Decoder constructed with two 2 X 4 Decoder
- Encoder
  - Inverse Operation of a decoder
  - 2^n input, n output
  - Truth Table : *Tab. 2-2*
    - 3 OR Gates Implementation
      - » A0 = D1 + D3 + D5 + D7
      - » A1 = D2 + D3 + D6 + D7
      - » A2 = D4 + D5 + D6 + D7

Tab. 2-2 Truth Table for Encoder

Inputs								Outputs		
D7	D6	D5	D4	D3	D2	D1	D0	A2	A1	A0
0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	1	0	0	0	1	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	0
0	0	1	0	0	0	0	0	1	0	1
0	1	0	0	0	0	0	0	1	1	0
1	0	0	0	0	0	0	0	1	1	1

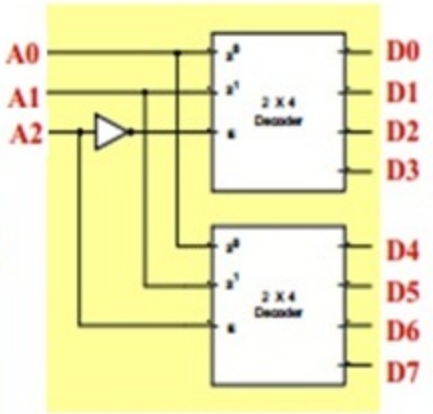


Fig. 2-3 A 3-to-8 Decoder constructed with two with 2-to-4 Decoder

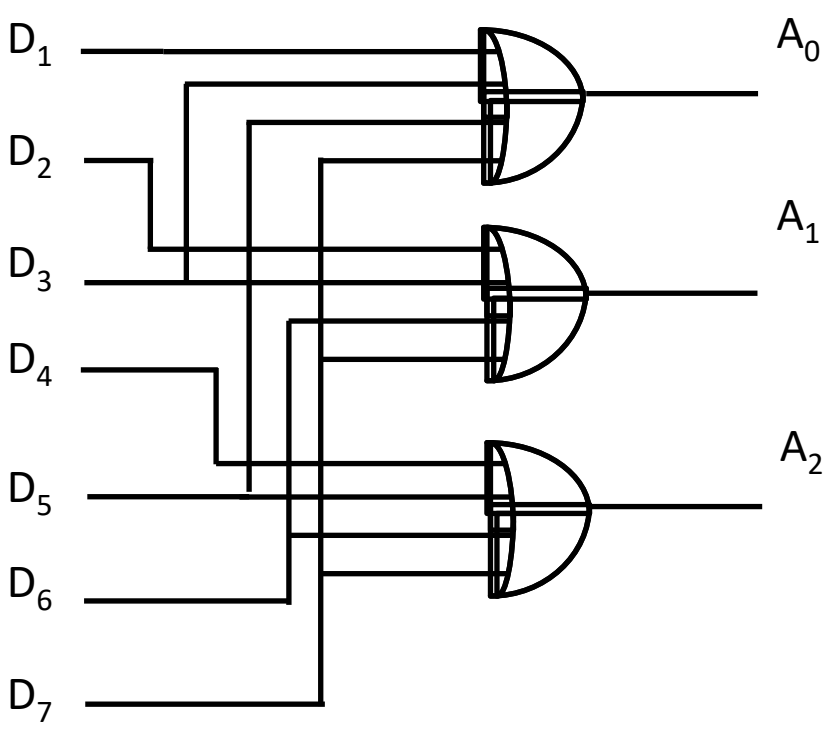
Octal to Binary Encoder

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	A_2	A_1	A_0
0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	1	0	0	0	1	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	0
0	0	1	0	0	0	0	0	1	0	1
0	1	0	0	0	0	0	0	1	1	0
1	0	0	0	0	0	0	0	1	1	1

$$A_0 = D_1 + D_3 + D_5 + D_7$$

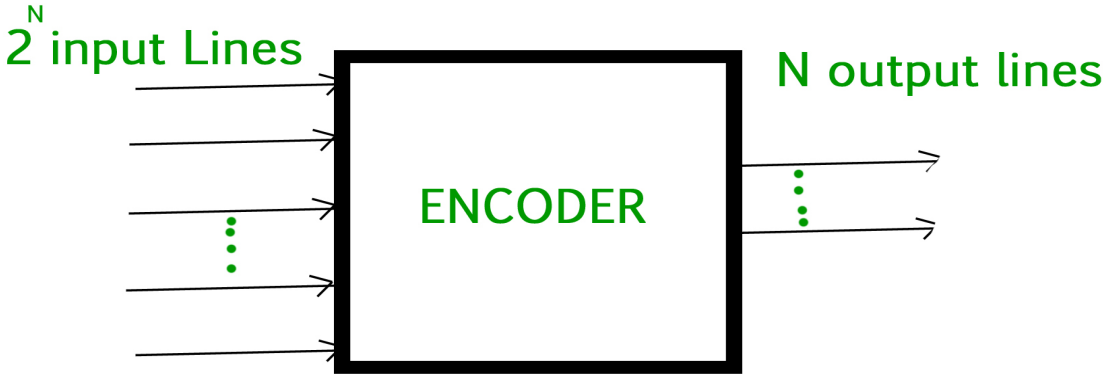
$$A_1 = D_2 + D_3 + D_6 + D_7$$

$$A_2 = D_4 + D_5 + D_6 + D_7$$

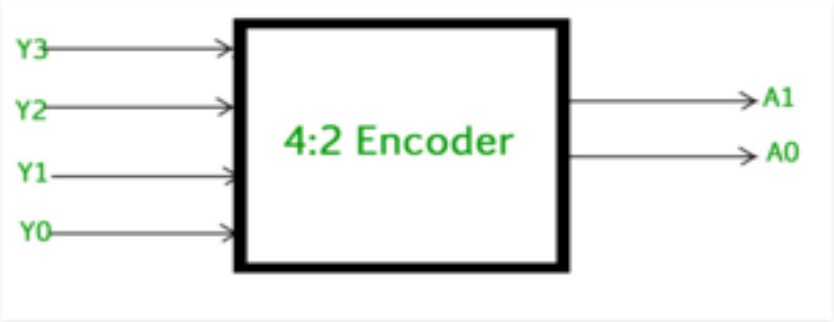


BINARY-ENCODER

- An Encoder is a **combinational circuit** that performs the reverse operation of Decoder.
- It has maximum of **2^n input lines** and **'n' output lines**, hence it encodes the information from 2^n inputs into an n-bit code.
- It will produce a binary code equivalent to the input, which is active High. Therefore, the encoder encodes **2^n input lines with 'n' bits.**



The 4 to 2 Encoder consists of **four inputs Y3, Y2, Y1 & Y0 and two outputs A1 & A0.** At any time, only one of these 4 inputs can be '1' in order to get the respective binary code at the output. The figure below shows the logic symbol of 4 to 2 encoder :



The Truth table of 4 to 2 encoder is as follows :

INPUTS				OUTPUTS	
Y3	Y2	Y1	Y0	A1	A0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

If we record any music in any recorder, such types of process is called

_____

a) Multiplexing
b) Encoding
c) Decoding
d) Demultiplexing

A decoder converts n inputs to _____ outputs.

a) n
b) n^2
c) 2^n
d) n^n

2-4 Registers

- Register
  - A group of flip-flops with each flip-flop capable of storing one bit of information
  - An n-bit register has a group of n flip-flops and is capable of storing any binary information of n bits
  - The simplest register consists only of flip-flops, with no external gate : *Fig. 2-6*
  - A clock input C will load all four inputs in parallel
    - The clock must be *inhibited* if the content of the register must be left unchanged
- Register with Parallel Load
  - A 4-bit register with a load control input : *Fig. 2-7*
  - The clock inputs receive clock pulses at all times
  - The buffer gate in the clock input will increase "fan-out"
  - Load Input
    - 1 : Four input transfer
    - 0 : Input inhibited, Feedback from output to input(**no change**)

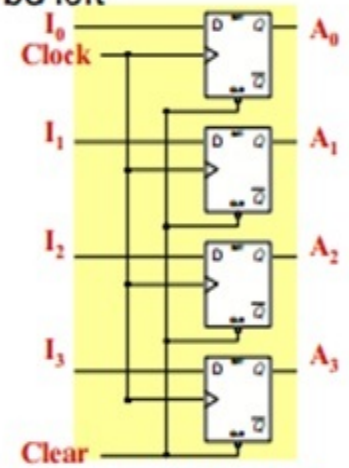


Fig. 2-6 4-bit register

2-4 Registers

- When the load input is 1 , the data in the four inputs are transferred into the register with the next positive transition of a clock pulse

- When the load input is 0, the data inputs are inhibited and the D-output of flip flop are connected to their inputs.

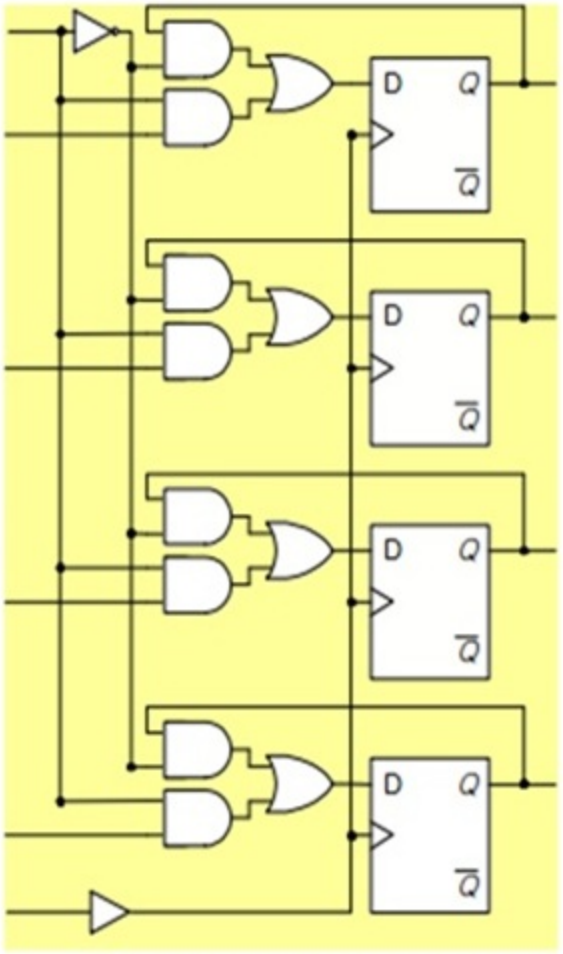


Fig. 2-7 4-bit register with parallel load

2-5 Shift Registers

- Shift Register
  - A register capable of shifting its binary information in one or both directions
  - The logical configuration of a shift register consists of a chain of flip-flops in cascade
  - The simplest possible shift register uses only flip-flops : *Fig. 2-8*
  - The ***serial input*** determines what goes into the leftmost position during the shift
  - The ***serial output*** is taken from the output of the rightmost flip-flop

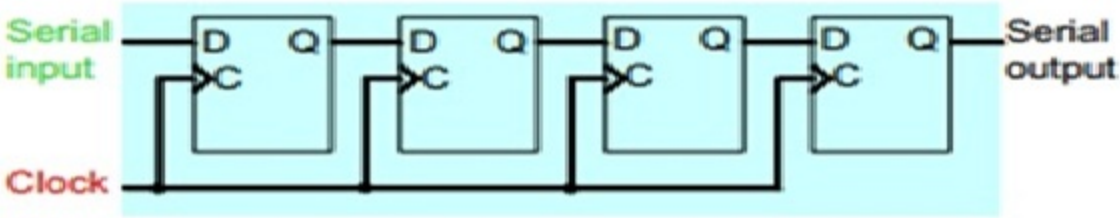


Fig. 2-8 4-bit shift register

2-5 Shift Registers

- Bidirectional Shift Register with Parallel Load
  - A register capable of shifting in *one direction only* is called a ***unidirectional shift register***
  - A register that can shift in *both directions* is called a ***bidirectional shift register***
  - The most general shift register has all the capabilities listed below:
    - An input clock pulse to synchronize all operations
    - A shift-right /left (serial output/input)
    - A parallel load, n parallel output lines
    - The register unchanged even though clock pulses are applied continuously
  - 4-bit bidirectional shift register with parallel load : *Fig. 2-9*
    - 4 X 1 Mux = 4 , D F/F = 4

Tab. 2-4 Function Table for Register of Fig. 2-9

Mode		Operation
S1	S0	
0	0	No chage
0	1	Shift right(down)
1	0	shift left(up)
1	1	Parallel load

A 4-bit bidirectional shift register with parallel load is shown in Fig. 2-9. Each stage consists of a D flip-flop and a 4×1 multiplexer. The two selection inputs S_1 and S_0 select one of the multiplexer data inputs for the D flip-flop. The selection lines control the mode of operation of the register according to the function table shown in Table 2-4. When the mode control $S_1S_0 = 00$, data input 0 of each multiplexer is selected. This condition forms a path from the output of each flip-flop into the input of the same flip-flop. The next clock transition transfers into each flip-flop the binary value it held previously, and no change of state occurs. When $S_1S_0 = 01$, the terminal marked 1 in each multiplexer has a path to the D input of the corresponding flip-flop. This causes a shift-right operation, with the serial input data transferred into flip-flop A_0 and the content of each flip-flop A_{i-1} transferred into flip-flop A_i for $i = 1, 2, 3$. When $S_1S_0 = 10$ a shift-left operation results, with the other serial input data going into flip-flop A_3 and the content of flip-flop A_{i+1} transferred into flip-flop A_i for $i = 0, 1, 2$. When $S_1S_0 = 11$, the binary information from each input I_0 through I_3 is transferred into the corresponding flip-flop, resulting in a parallel load operation. Note that the way the diagram is drawn, the shift-right operation shifts the contents of the register in the down direction while the shift left operation causes the contents of the register to shift in the upward direction.

TABLE 2-4 Function Table for Register of Fig. 2-9

Mode control		
S_1	S_0	Register operation
0	0	No change
0	1	Shift right (down)
1	0	Shift left (up)
1	1	Parallel load

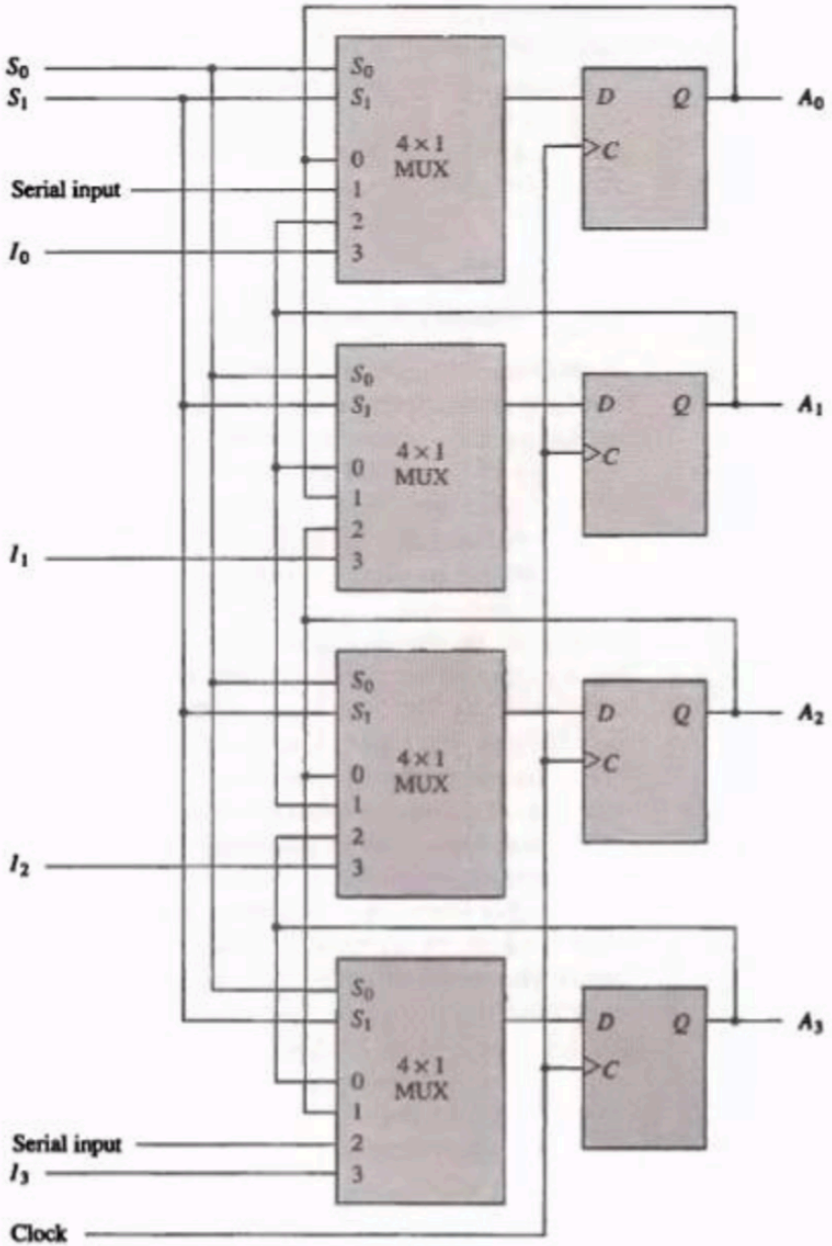


Figure 2-9 Bidirectional shift register with parallel load.

2-5 Shift Registers

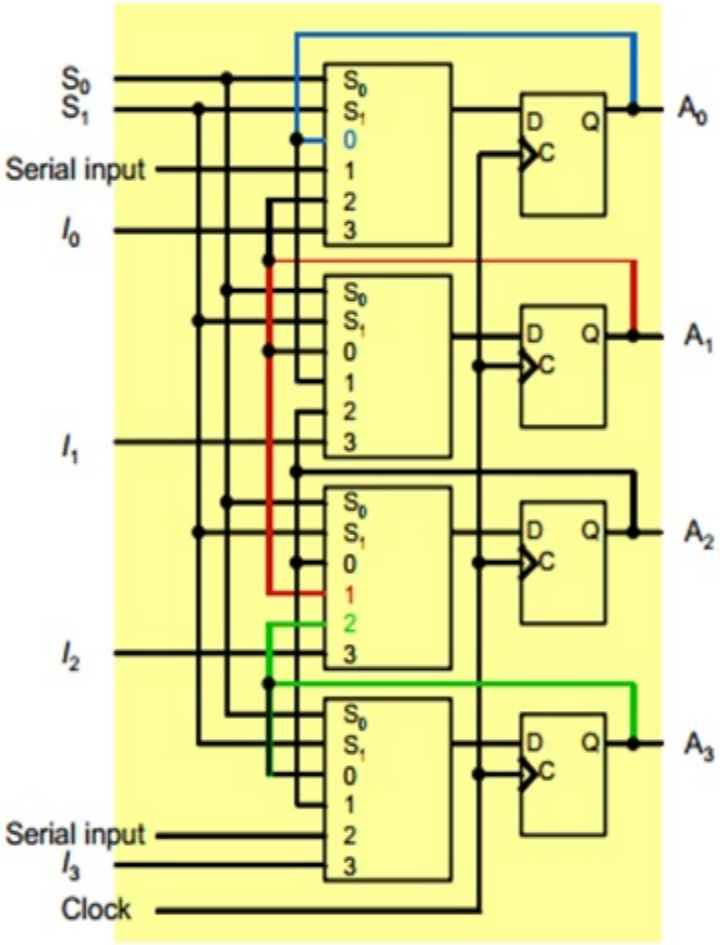


Fig. 2-9 Bidirectional shift register

$S_1S_0 = 00$: $A_i \rightarrow A_i$ (No change)
$S_1S_0 = 01$: $A_{i-1} \rightarrow A_i$ (Shift)
$S_1S_0 = 10$: $A_{i+1} \rightarrow A_i$ (Shift)
$S_1S_0 = 11$: Parallel load

- Shift Register

Interface digital systems situated remotely from each other

Shift registers are often used to interface digital systems situated remotely from each other. For example, suppose that it is necessary to transmit an n-bit quantity between two points. If the distance between the source and the destination is too far, it will be expensive to use n lines to transmit the n bits in parallel. It may be more economical to use a single line and transmit the information serially one bit at a time. The transmitter loads the n-bit data in parallel into a shift register and then transmits the data from the serial output line. The receiver accepts the data serially into a shift register through its serial input line. When the entire n bits are accumulated they can be taken from the outputs of the register in parallel. Thus the transmitter performs a parallel-to-serial conversion of data and the receiver converts the incoming serial data back to parallel data transfer.