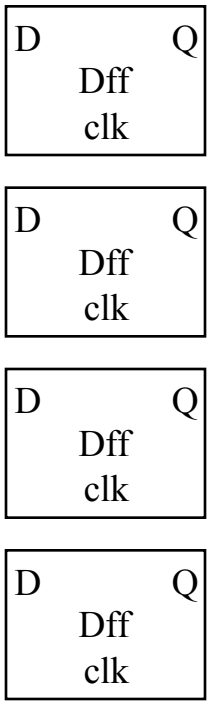
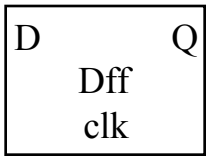
Registers

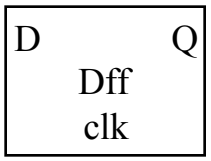
- Readings: 5.8-5.9.3
- Storage unit. Can hold an n-bit value
- Composed of a group of n flip-flops
  - Each flip-flop stores 1 bit of information

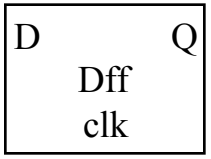


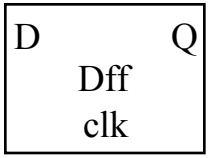
Controlled Register

Reset	Load	Action
0	0	Q = old Q
1	0	Q = 0
0	1	Q = D



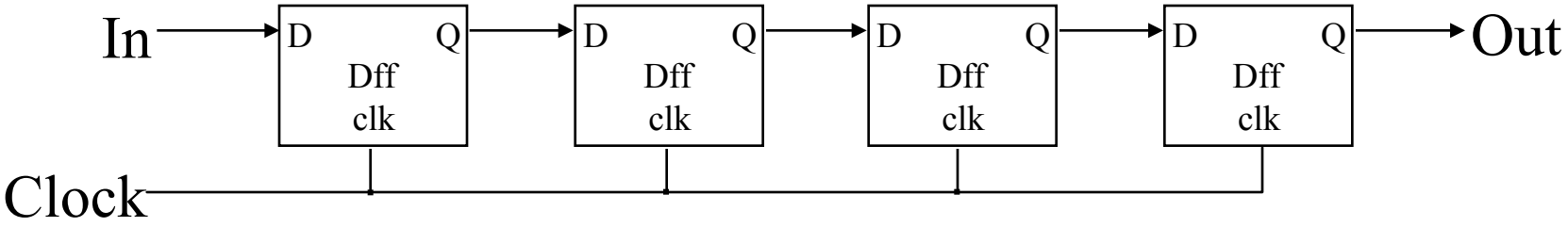






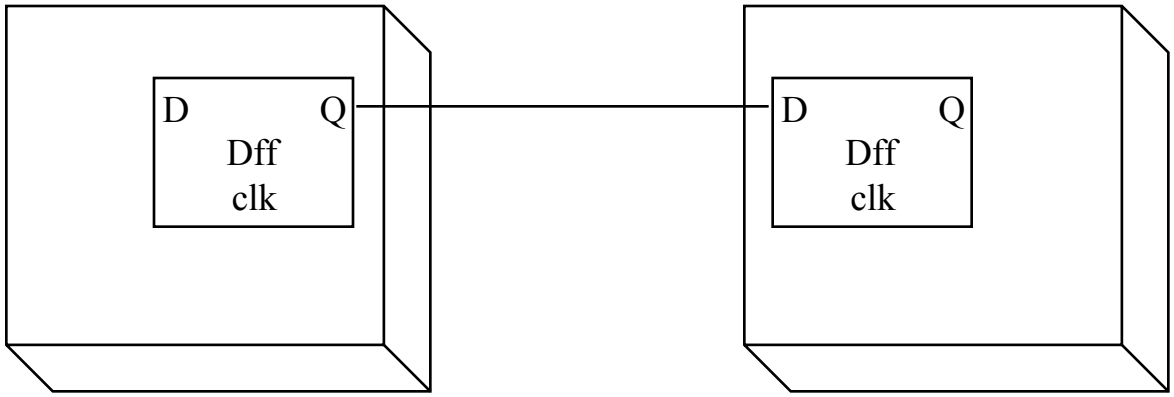
Shift Register

- Register that shifts the binary values in one or both directions



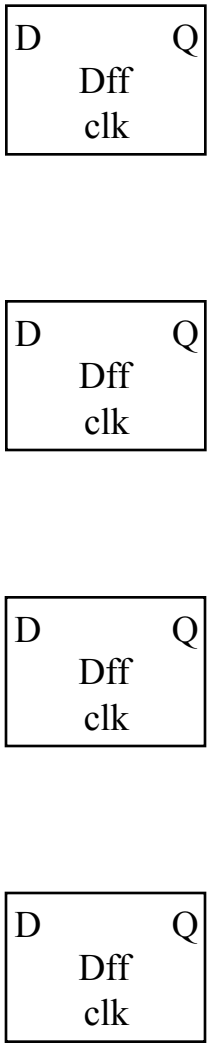
Transfer of Data

- 2 modes of communication: Parallel vs. Serial
  - Parallel: all bits transferred at the same time
  - Serial: one bit transferred at a time
- Shift register can be used for serial transfer

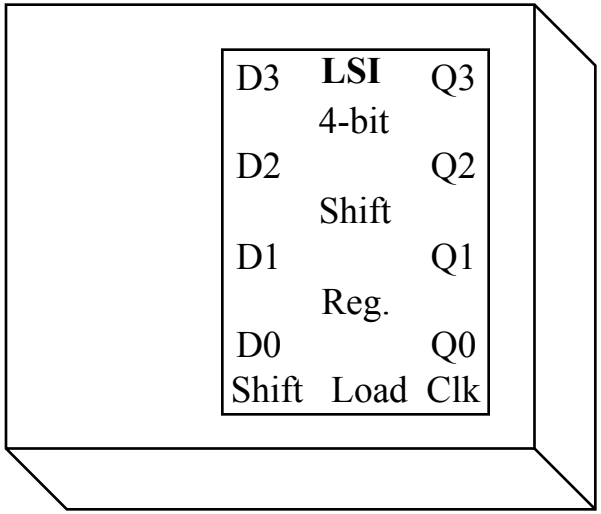


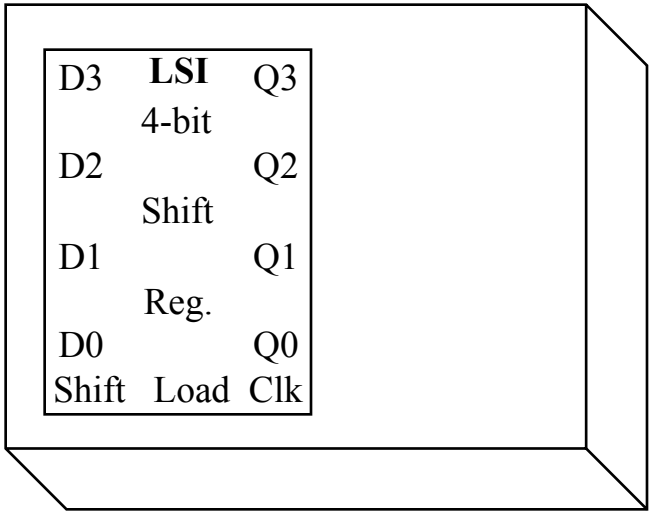
Shift Register w/Parallel Load

Shift	Load	Action
0	0	Q = old Q
1	0	Shift
X	1	Parallel Load



Conversion between Parallel & Serial





Counters

- A reg. that goes through a specific state sequence
- *n-bit Binary Counter:* counts from 0 to 2^N-1 in binary
- *Up Counter:* Binary value increases by 1
- *Down Counter:* Binary value decreases by 1
- 3-bit binary up counter state diagram:

Binary Up-Counter Imp.

Complex Binary Counter

Load	Count	Action
0	0	Q = old Q
0	1	Up Count
1	0	Reset
1	1	Load Parallel

Arbitrary Sequence Counters

- Design a 3-bit count that goes through the sequence 000->010->100->101->111->110->001->011->000->...

Counters in Verilog

```
module upcounter #(parameter WIDTH=8)
  (out, incr, reset, clk);

  output reg [WIDTH-1:0] out;
  input                  incr, reset, clk;

endmodule
```

Verilog Memories

```
module memory16x6 (data_out, data_in, addr, we, clk);
  output reg [5:0] data_out;
  input      [5:0] data_in;
  input      [3:0] addr;
  input            we, clk;

  reg        [5:0] mem       [15:0];

  always @(*)
    data_out = mem[addr];

  always @(posedge clk)
    if (we)
      mem[addr] <= data_in;

endmodule
```