Review Problem: Basic Verilog

- Write the Verilog for a 2-input gate that is TRUE when an odd number of inputs are true.

Review Problem: flip-flops

- The following two flip-flops are subtly different, but both useful. The difference in code is shown in bold. What is the difference in their behavior?

```
module D_FF1 (q, d, reset, clk);
  output q;
  input d, reset, clk;
  reg q;

  always @(posedge clk)
  if (reset)
    q <= 0;
  else
    q <= d;

endmodule
```

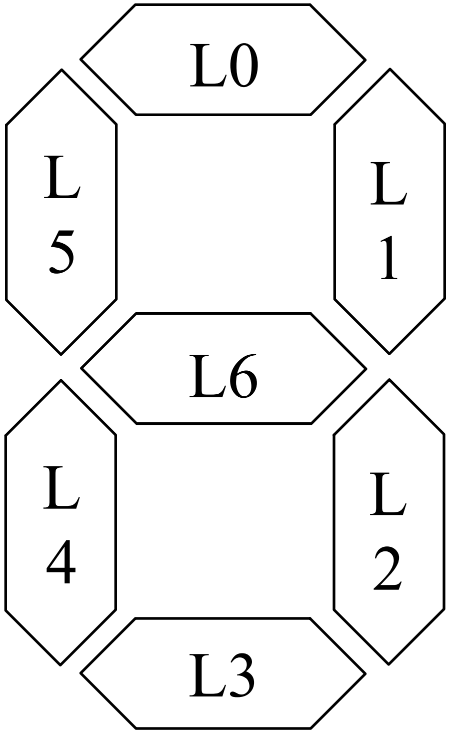
```
module D_FF2 (q, d, reset, clk);
  output q;
  input d, reset, clk;
  reg q;

  always @(posedge clk or posedge reset)
  if (reset)
    q <= 0;
  else
    q <= d;

endmodule
```

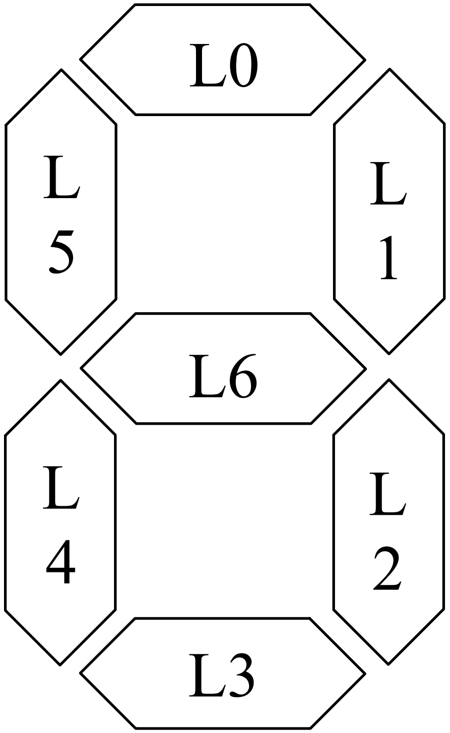
Case Study: Seven Segment Display

- Chip to drive digital display

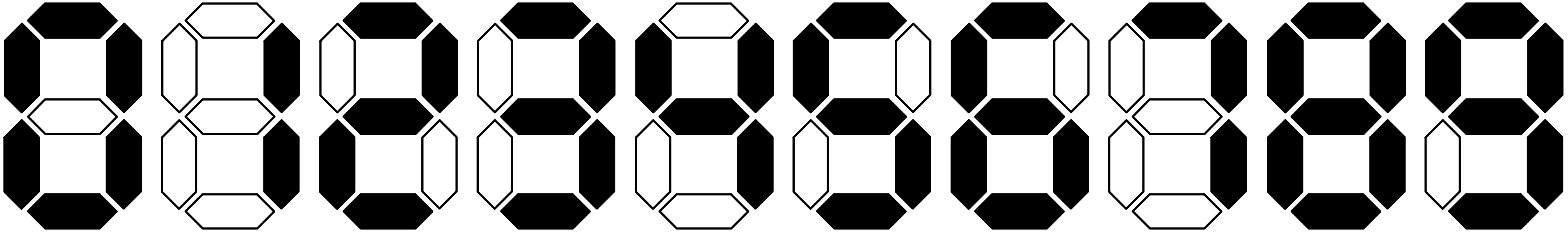


B3	B2	B1	B0
0	0	0	0
0	0	0	1
0	0	1	0
0	0	1	1
0	1	0	0
0	1	0	1
0	1	1	0

Case Study (cont.)



B3	B2	B1	B0	Val	L0	L1	L2	L3	L4	L5	L6
0	0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	1	0	1	1	0	0	0	0
0	0	1	0	2	1	1	0	1	1	0	1
0	0	1	1	3	1	1	1	1	0	0	1
0	1	0	0	4	0	1	1	0	0	1	1
0	1	0	1	5	1	0	1	1	0	1	1
0	1	1	0	6	1	0	1	1	1	1	1
0	1	1	1	7	1	1	1	0	0	0	0
1	0	0	0	8	1	1	1	1	1	1	1
1	0	0	1	9	1	1	1	1	0	1	1



Case Study (cont.)

- Implement L5:

B3	B2	B1	B0	L5
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	0
1	0	0	0	1
1	0	0	1	1

7-seg display in Verilog

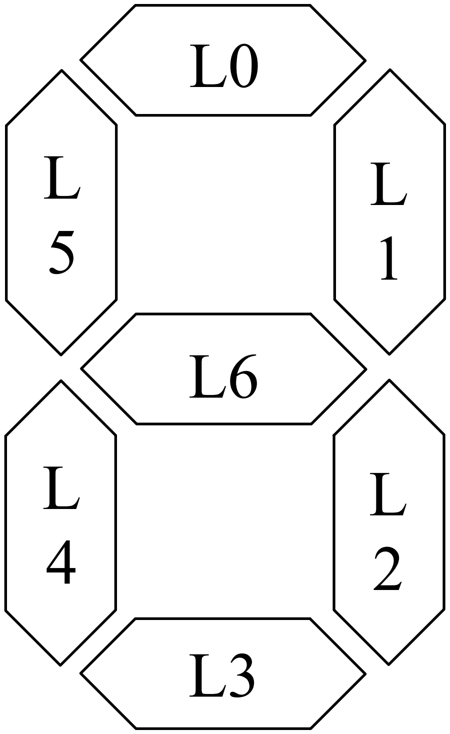
- Verilog RTL: just describe what you want

```
module seg7 (bcd, leds);
  input      [3:0] bcd;
  output reg [6:0] leds;

  always @(*)
    case (bcd)
      // 3210            6543210
      4'b0000: leds = 7'b0111111;
      4'b0001: leds = 7'b0000110;
      4'b0010: leds = 7'b1011011;
      4'b0011: leds = 7'b1001111;
      4'b0100: leds = 7'b1100110;
      4'b0101: leds = 7'b1101101;
      4'b0110: leds = 7'b1111101;
      4'b0111: leds = 7'b0000111;
      4'b1000: leds = 7'b1111111;
      4'b1001: leds = 7'b1101111;
      default: leds = 7'bX;
    endcase
endmodule
```

Review Problem

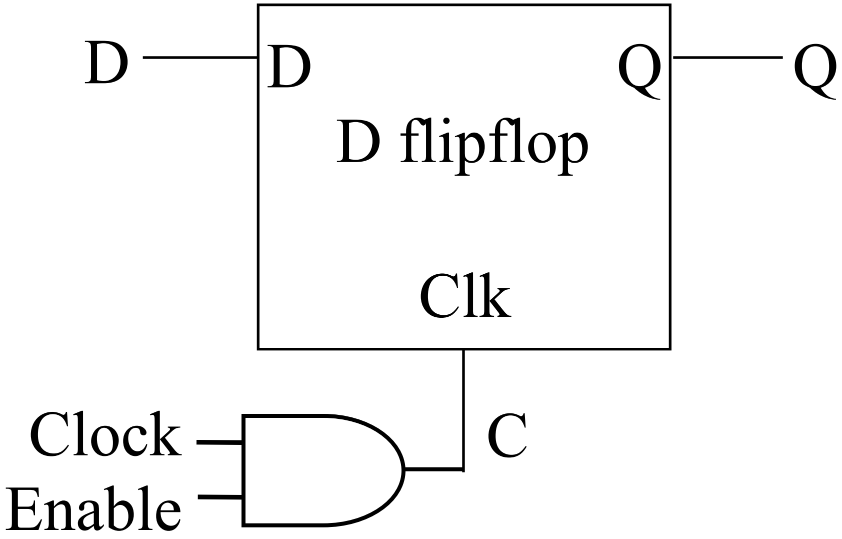
- Extend this Verilog code to also show the letter "A" on input pattern 1010 (ten) and "F" on pattern 1111 (fifteen).

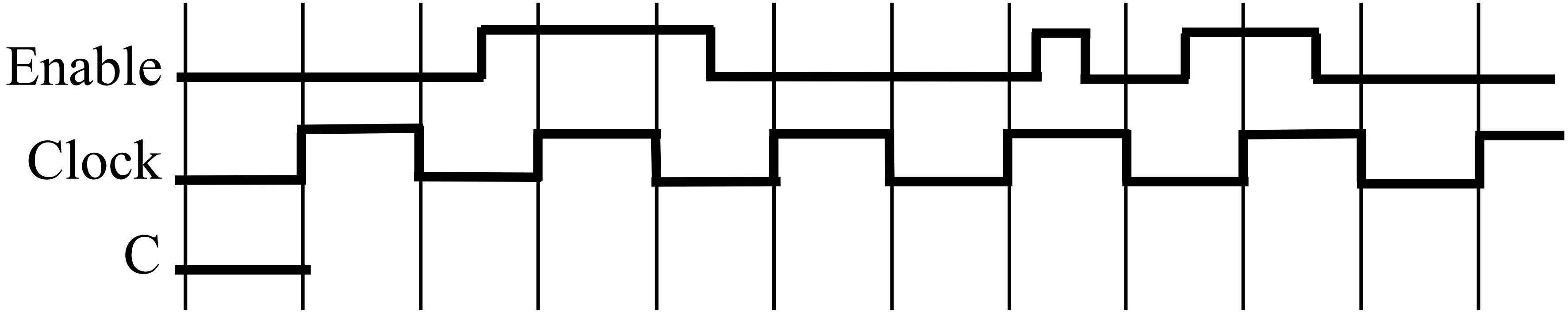


```
module seg7 (bcd, leds);
  input      [3:0] bcd;
  output reg [6:0] leds;

  always @(*)
    case (bcd)
      // BCD[]           LEDS[]
      // 3210            6543210
      4'b0000: leds = 7'b0111111;
      4'b0001: leds = 7'b0000110;
      4'b0010: leds = 7'b1011011;
      4'b0011: leds = 7'b1001111;
      4'b0100: leds = 7'b1100110;
      4'b0101: leds = 7'b1101101;
      4'b0110: leds = 7'b1111101;
      4'b0111: leds = 7'b0000111;
      4'b1000: leds = 7'b1111111;
      4'b1001: leds = 7'b1101111;
      default: leds = 7'bX;
    endcase
endmodule
```

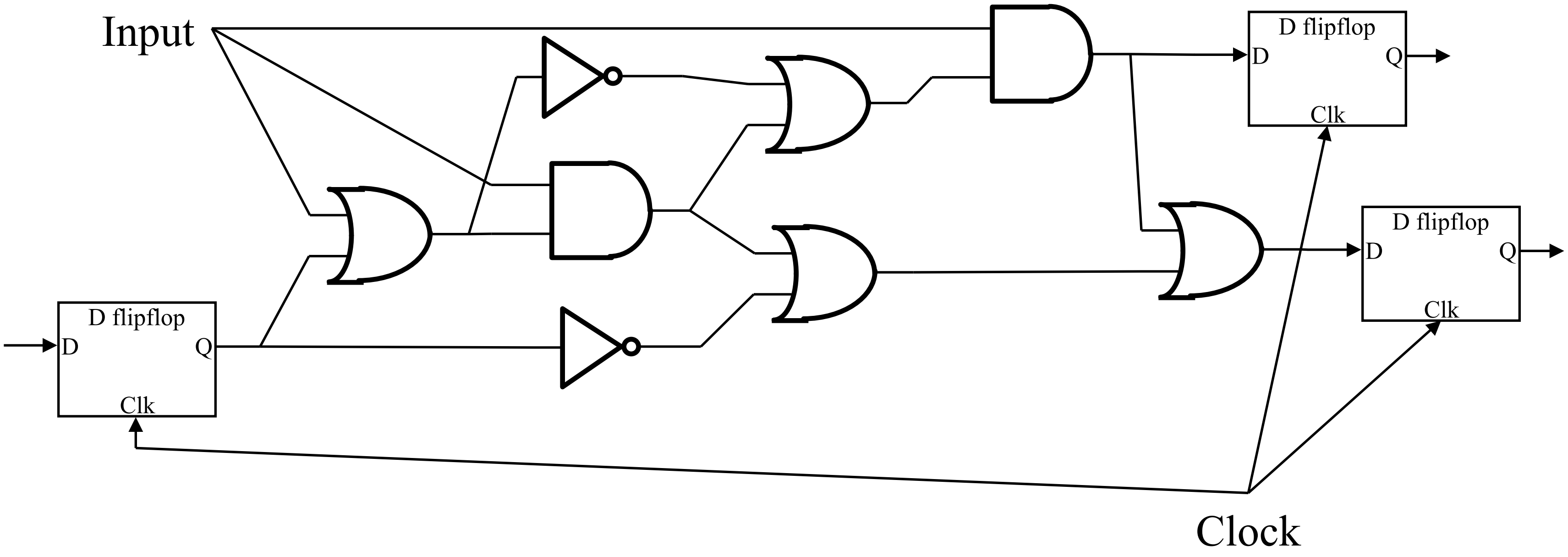
Flipflop Realities 1: Gating the Clock



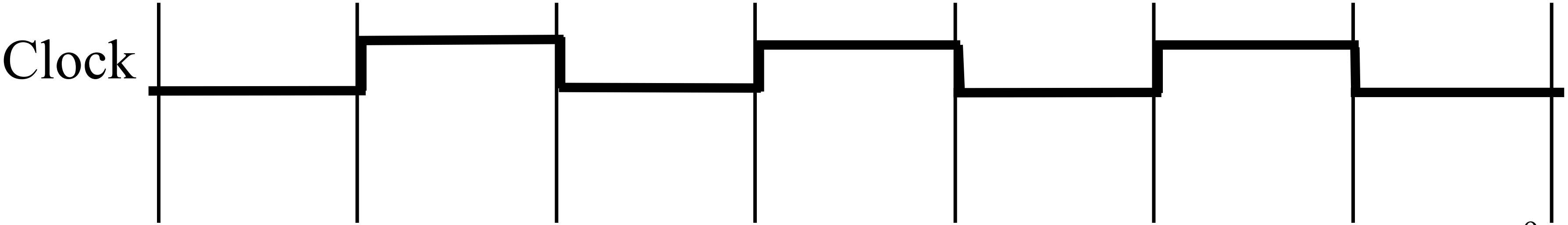


- **NEVER put a logic gate between the clock and DFF’s CLK input.**

Flipflop Realities 2: Clock Period, Applying Stimulus

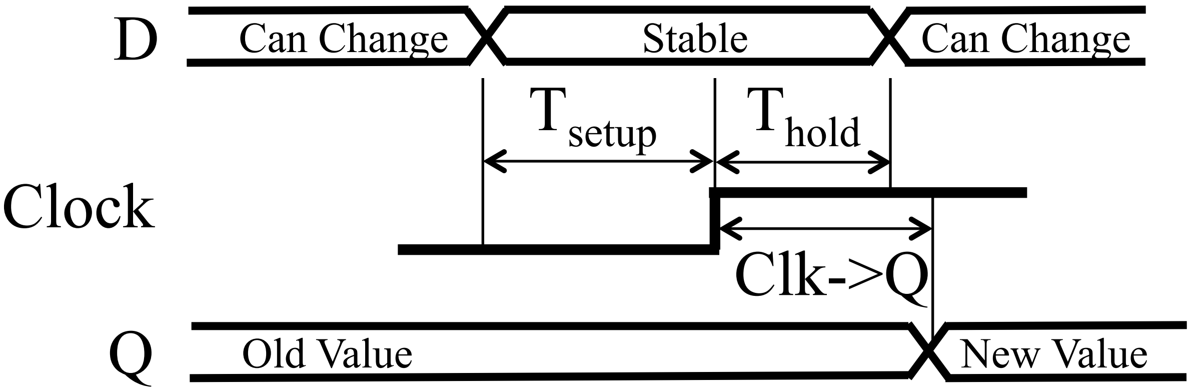


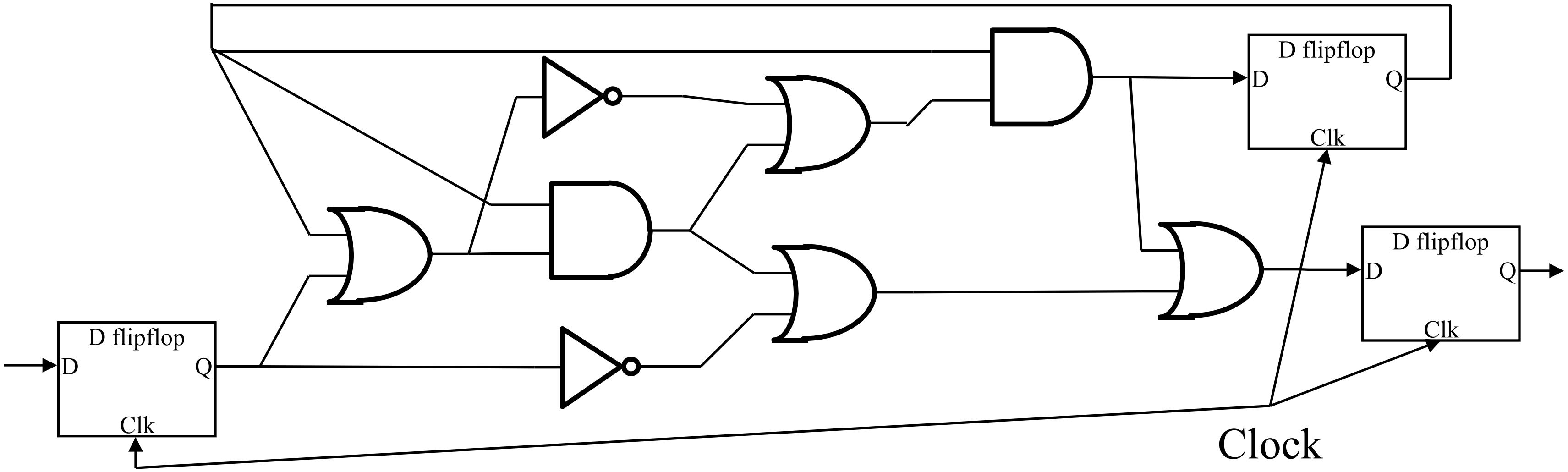
- Clock Period?
- Apply Inputs when?



T_{setup}, T_{hold}, Clk->Q

- Flipflops require their inputs be stable for time period around clock edge





Timing Definitions

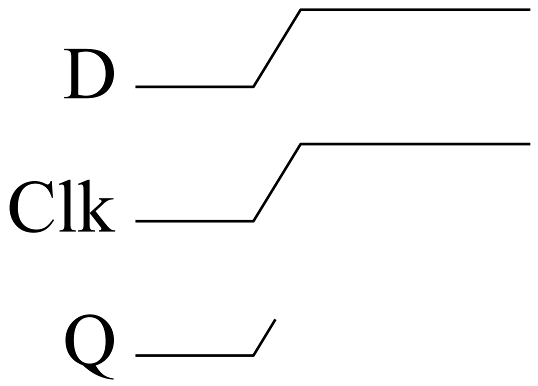
- T_{setup}: Time D must be stable BEFORE clock edge
  - Adds to critical path delay

- Clk->Q: Time from clock edge to Q changing
  - Adds to critical path delay

- T_{hold}: Time D must be stable AFTER clock edge
  - Sets minimum path from Q of one DFF to D of another

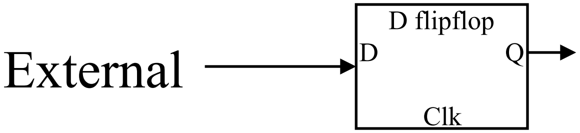
Flipflop Realities 3: External Inputs

- External inputs aren’t synchronized to the clock

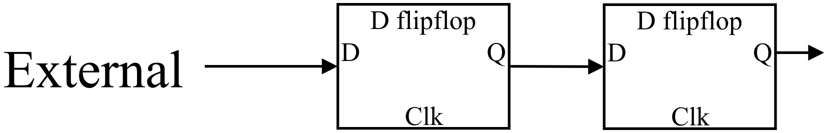


Dealing with Metastability

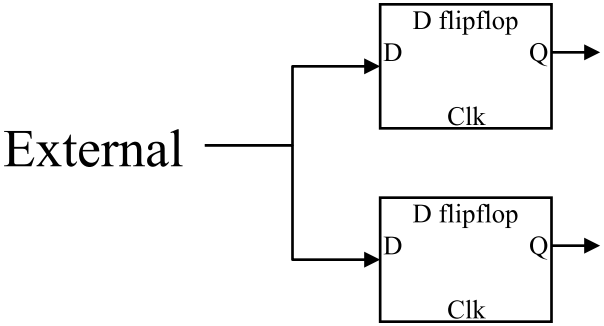
- Single DFF



- 2 DFFs in series



- 2 DFFs in parallel


D flipflop
D
Q
Clk
External
D flipflop
D
Q
Clk