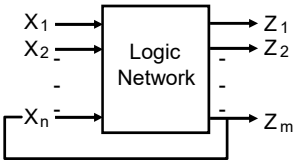
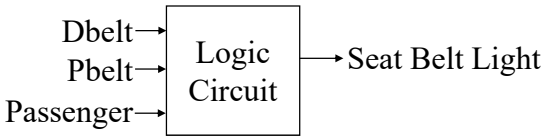
Combinational vs. Sequential Logic

❖ Readings: 5-5.4.4



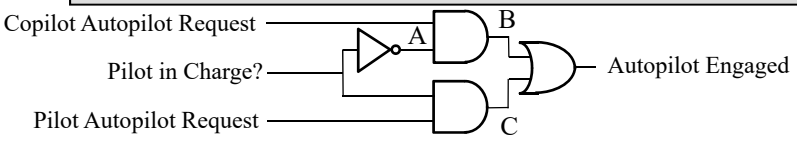
Network implemented from logic gates. The presence of feedback distinguishes between ***sequential*** and ***combinational*** networks.

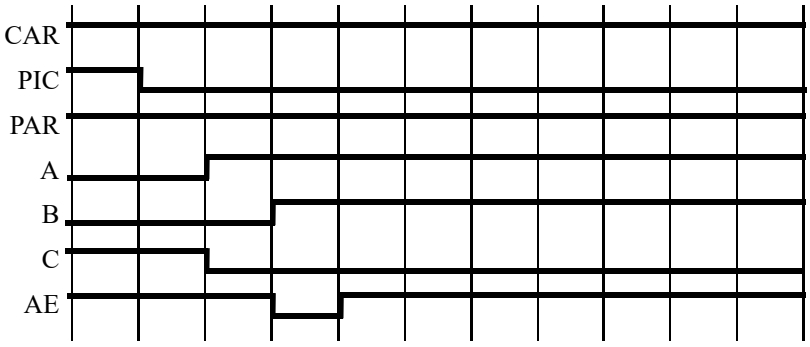
Combinational logic
no feedback among inputs and outputs
outputs are a pure function of the inputs
e.g., seat belt light:
(Dbelt, Pbelt, Passenger) mapped into (Light)



Hazards/Glitches

❖ Circuit can temporarily go to incorrect states

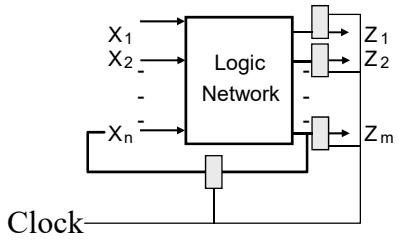


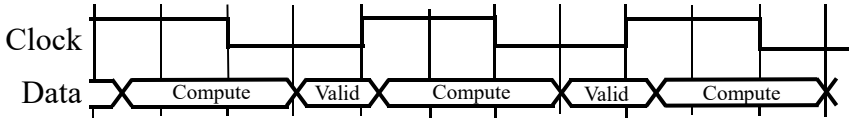


❖Must filter out temporary states

Safe Sequential Circuits

- Clocked elements on feedback, perhaps outputs
  - Clock signal synchronizes operation
  - Clocked elements hide glitches/hazards



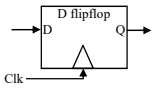


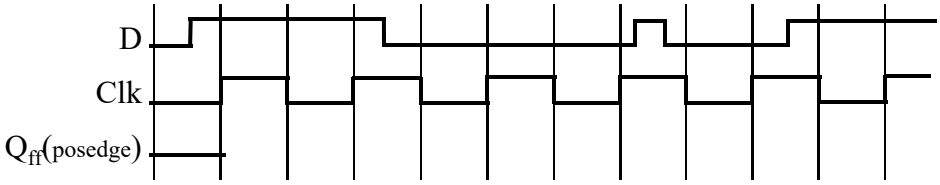
Basic D Flip Flop

```
// Basic D flip-flop

module basic_D_FF (q, d, clk);
  output logic  q;
  input  logic  d, clk;

  always_ff @(posedge clk) begin
    q <= d; // ALWAYS use <= to assign to clocked elements
  end
endmodule
```





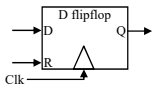
D Flip Flop w/Synchronous Reset

```
// D flip-flop w/synchronous reset

module D_FF (q, d, reset, clk);
  output logic  q;
  input  logic  d, reset, clk;

  always_ff @(posedge clk) begin
    if (reset)
      q <= 0;  // On reset, set to 0
    else
      q <= d; // Otherwise out = d
  end

endmodule
```



Verilog Testbench

```
module stimulus;
  logic  clk, reset, d, q;

  parameter ClockDelay = 100;

  D_FF dut (.q, .d, .reset, .clk); // Instantiate the D FF

  initial begin // Set up the clock
    clk <= 0;
    forever #(ClockDelay/2) clk <= ~clk;
  end

  initial // Set up the reset signal
  begin
    d <= 0; reset <= 1; @(posedge clk);
            reset <= 0; @(posedge clk);
    d <= 1;             @(posedge clk);
    d <= 0;             @(posedge clk);
                        @(posedge clk);
    $stop(); // end the simulation
  end

endmodule
```

Testbench Waveforms