VERILOG 3:

TIME AND DELAY

3 Realms of Time and Delay

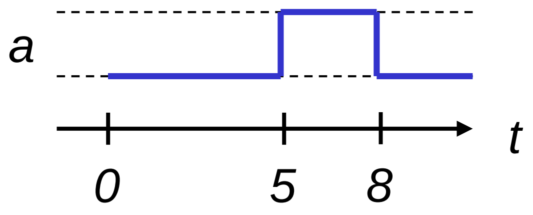
Discussed in this Verilog 3 section

1) Verilog simulation: “wall clock” time
2) ⭐ Verilog simulation: timing within the simulation
   a) These delays are set by “#” delays discussed in the following slides
3) Circuit delays (in circuits created by the synthesizer tool + the fabrication technology library)
   a) Simple models using “#” delays in a cell library
   b) More sophisticated Static Timing Analysis (STA) which takes into account things like circuit capacitive loading and delays due to wires (briefly covered at the end of 180)
   c) Propagation delays (t_p) found by spice simulations
   d) Measured silicon

Delays

- Delays may be inserted into always and initial blocks to cause the simulator to let "simulation time" advance
- Syntax:
  - **`#n        // delay of n time units`**
  - Example:

```
always @(...) begin
   a = 1'b0;
   #5;               // 5-unit delay
   a = 1'b1;
   #3;               // 3-unit delay
   a = (c|d)^(e|f);  // a = 0 here
end
```



Verilog "#" Delays are normally used in three places

1) Testbench verilog where it is essential

- Example: to time input signals
- Example: the clock generator (see Verilog Testing notes)
- Example code:

```
//Example testbench to generate input signals
always @(...) begin
   reset = 1'b1;
   in    = 16'h0000;
   #10;             // 10-unit delay
   reset = 1'b0;
   in    = 16'h0001;
   #10;
   in    = 16'h0002;
   #10;
   in    = 16'h0003;
   #10;
   ...
end
```

Verilog "#" Delays are normally used in three places

2) In flip-flop declarations in "hardware(!) verilog"

- To set a *clock-to-Q* delay for the purpose of increasing waveform readability
- Usage will normally produce a warning from synthesis tools
- Details and syntax are given in a later lecture

3) In gate libraries to provide crude delay estimations

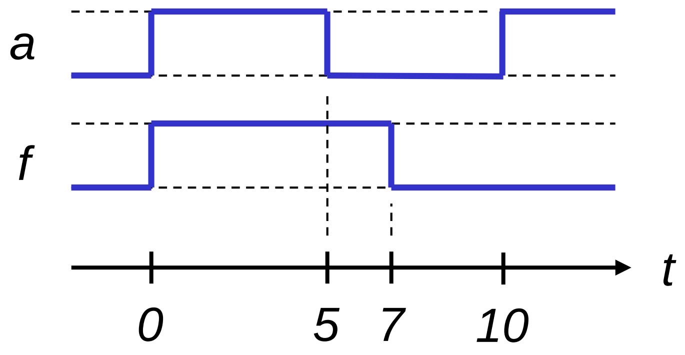
- We will not work on this in this class

Concurrent Operation

- Think of verilog modules as operating on independent circuits (remember it describes *hardware*).

```
always begin    // this block executes repeatedly without pausing
   a = (b&c) | d; // 1
   #5;            // 5-unit delay
   a = ~a;        // 0
   #5;
end

always begin
   f = ~(g ^ h);  // 1
   #7;            // 7-unit delay
   f = ~f;        // 0
   #7;
end
```



Setting the timescale of "#" delays

- **`` `timescale ``** *time_unit base / precision base*
- The first argument specifies "#1" delay
- The second argument specifies the precision with which delays may be specified
- Base is {s, ms, us, ns, ps, fs}
- Ex: **`` `timescale 1ns/10ps ``**
  - **`#5`** would produce a 5 ns delay