Chapter 4
Sequential Basics
Sequential Basics

- Sequential circuits
  - Outputs depend on current inputs and previous inputs
  - Store *state*: an abstraction of the history of inputs
- Usually governed by a periodic clock signal
D-Flipflops

- 1-bit storage element
  - We will treat it as a basic component

- Other kinds of flipflops
  - SR (set/reset), JK, T (toggle)
**Registers**

- **Store a multi-bit encoded value**
  - One D-flipflop per bit
  - Stores a new value on each clock cycle

```verilog
wire [n:0] d;
reg [n:0] q;
...
always @(posedge clk)
  q <= d;
```

**Nonblocking assignment**

**Event list**
Pipelines Using Registers

Total delay = Delay\(_1\) + Delay\(_2\) + Delay\(_3\)

Clock period = Delay\(_1\) + Delay\(_2\) + Delay\(_3\)

Interval between outputs > Total delay

Clock period = \(\text{max}(\text{Delay}\(_1\), \text{Delay}\(_2\), \text{Delay}\(_3\))\)

Total delay = 3 \times \text{clock period}

Interval between outputs = 1 clock period
Pipeline Example

- Compute the average of corresponding numbers in three input streams
- New values arrive on each clock edge

```verilog
module average_pipeline (output reg signed [0:13] avg,
                         input signed [0:13] a, b, c,
                         wire signed [0:14] a_plus_b;
wire signed [0:15] sum;
  wire signed [0:22] sum_div_3;
  reg signed [0:14] saved_a_plus_b
Reg signed [0:13] saved_c
Reg [0:15] saved_sum;
... input clk );
```
Pipeline Example

...
Verilog

Block diagram of the non pipelined design example
Block diagram of the Pipelined design Example