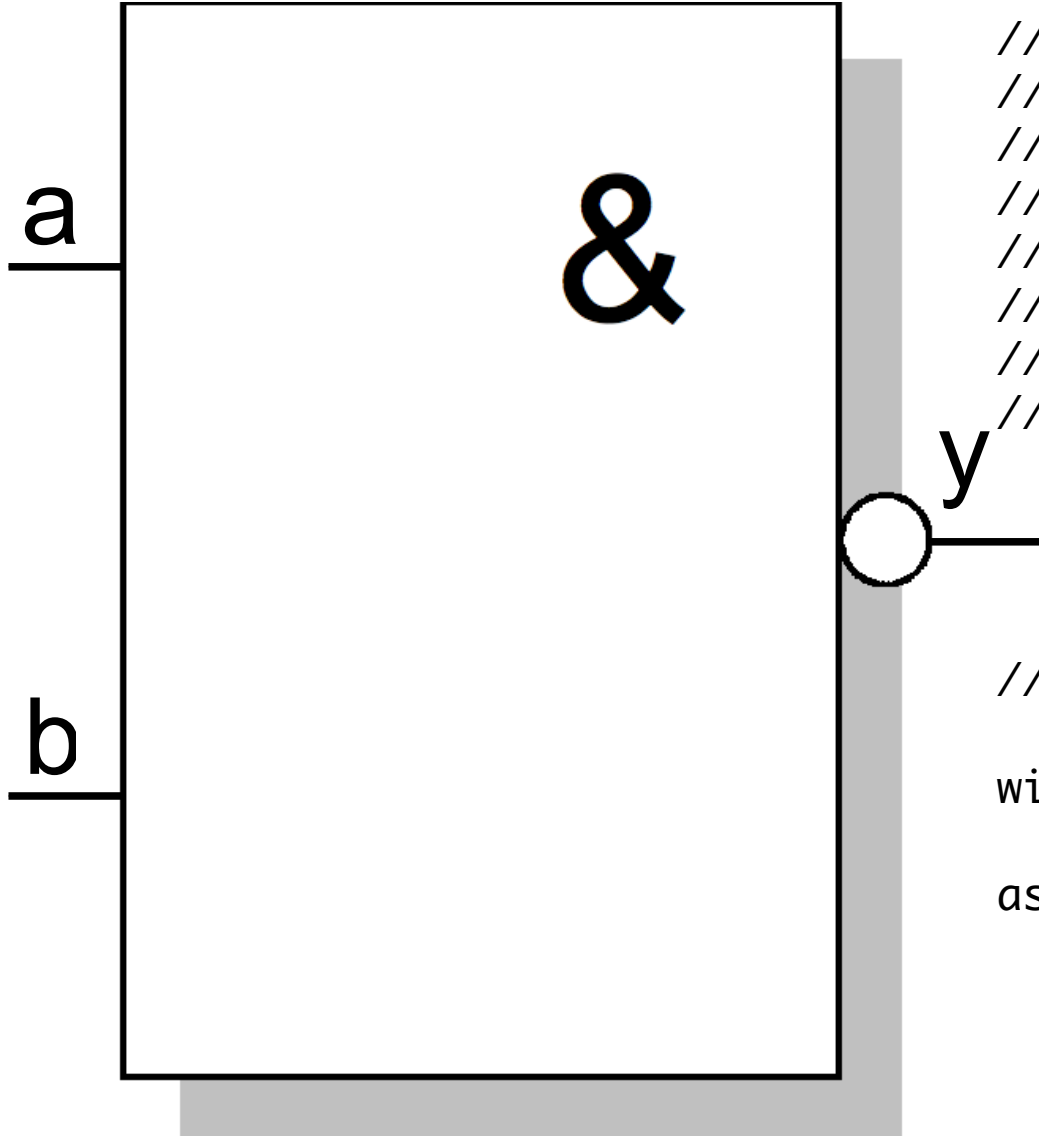


Hacking the Silicon

- A Crash Course in Chip Development -

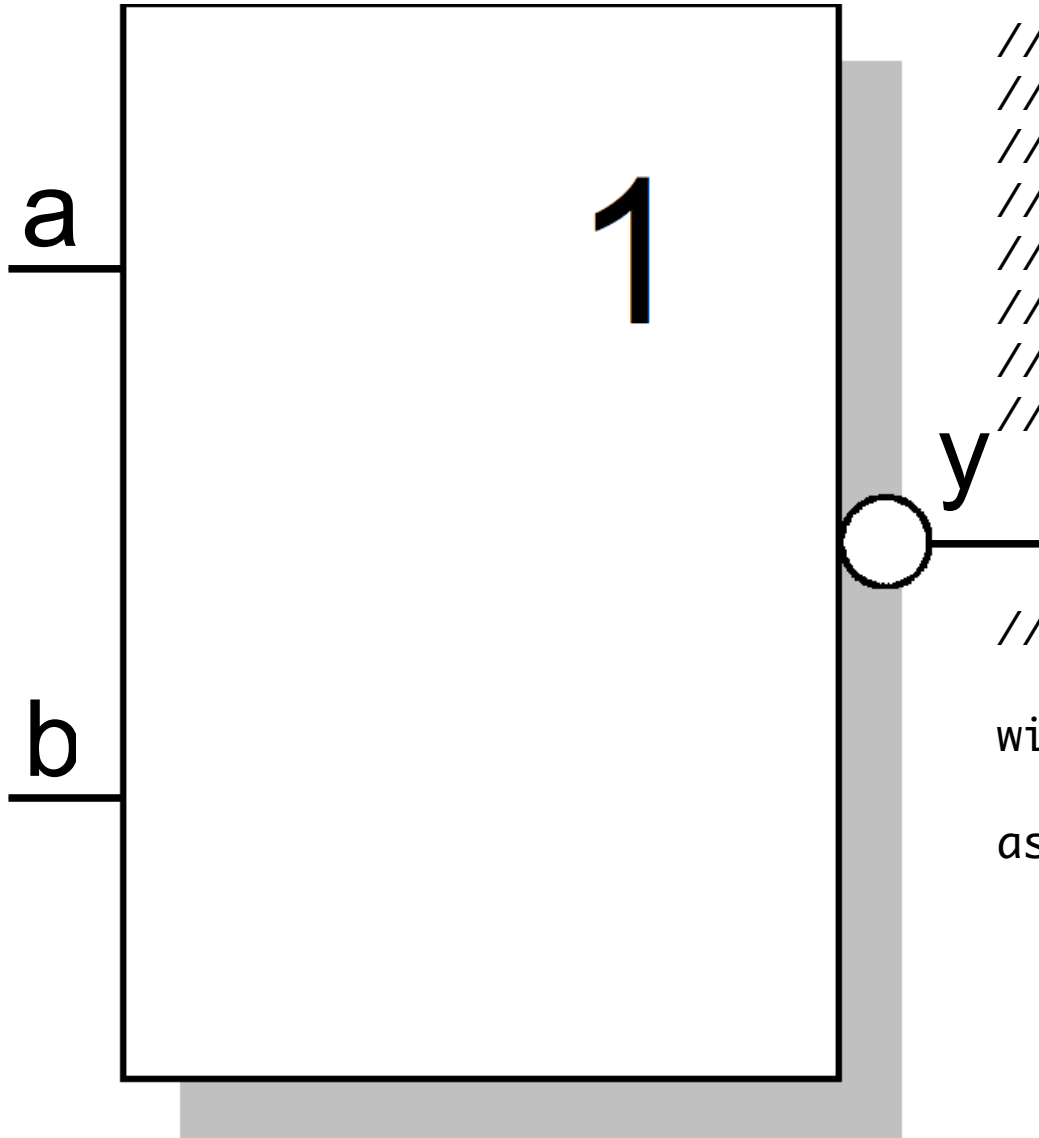
Hagen SANKOWSKI
hsank@nospam.chipforge.org

2nd - 5th September 2007 Cambridge



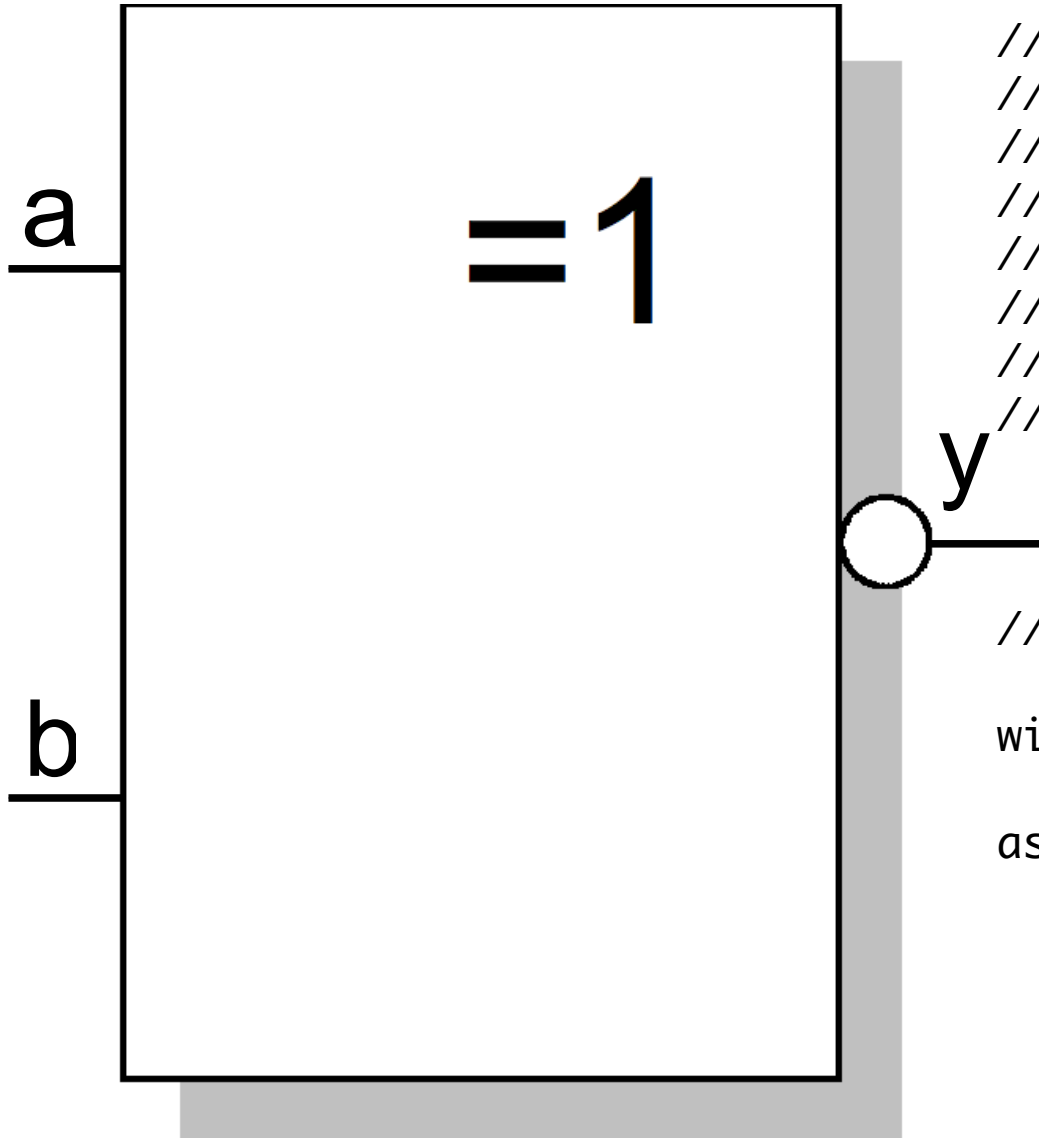
```
// Truth Table:  
//  
// a      b      |      y  
// -----+-----  
// 0      0      |      1  
// 0      1      |      1  
// 1      0      |      1  
// 1      1      |      0
```

```
// Verilog HDL:  
  
wire    a,b,y;           // simple wire  
  
assign  y = ~(a & b);   // NAND
```



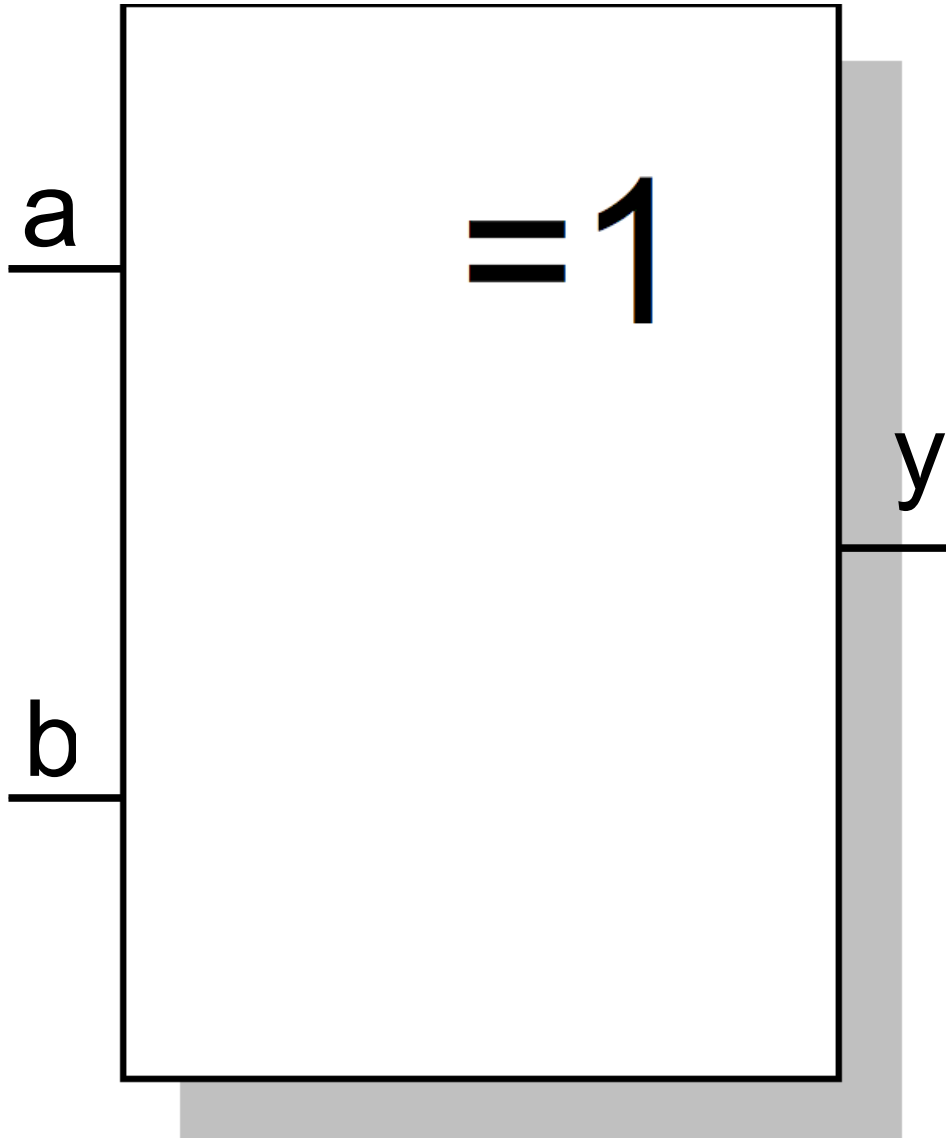
```
// Truth Table:
//
// a      b  |  y
// -----+-----
// 0      0  |  1
// 0      1  |  0
// 1      0  |  0
// 1      1  |  0
```

```
// Verilog HDL:
wire  a,b,y;           // simple wire
assign y = ~(a | b);  // NOR
```



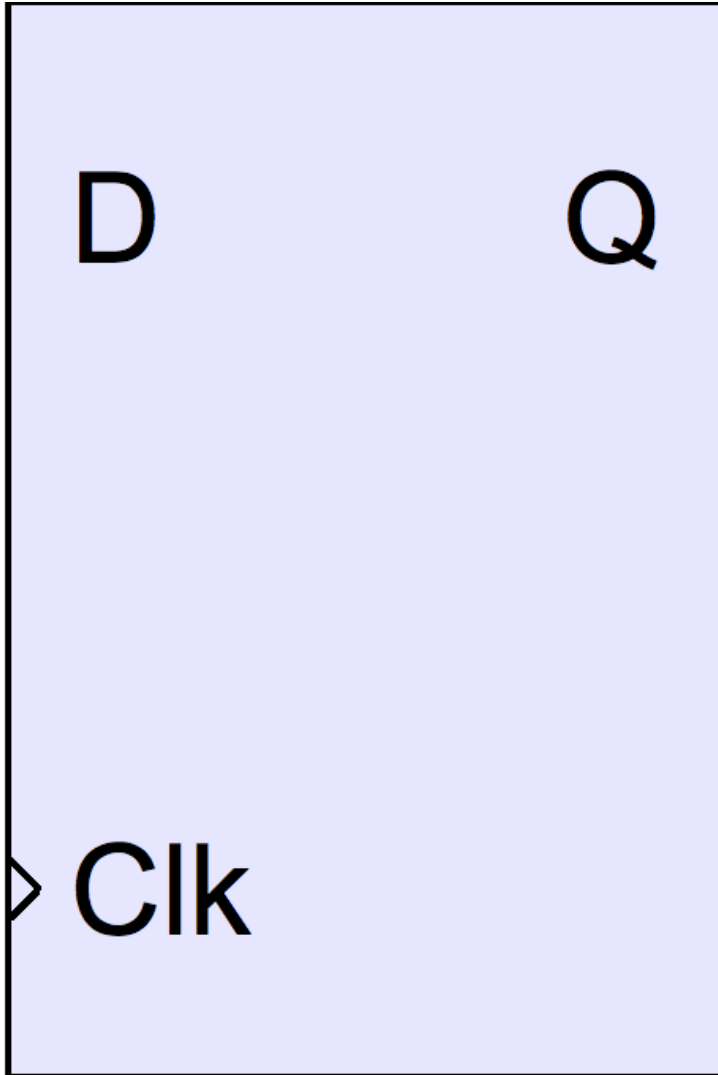
```
// Truth Table:
//
// a      b  |  y
// -----+-----
// 0      0  |  1
// 0      1  |  0
// 1      0  |  0
// 1      1  |  1
```

```
// Verilog HDL:
wire  a,b,y;           // simple wire
assign y = ~(a ^ b);  // XNOR
```



```
// Truth Table:  
//  
// a      b  |  y  
// -----+-----  
// 0      0  |  0  
// 0      1  |  1  
// 1      0  |  1  
// 1      1  |  0
```

```
// Verilog HDL:  
  
wire  a,b,y;           // simple wire  
  
assign y = a ^ b;     // XOR
```

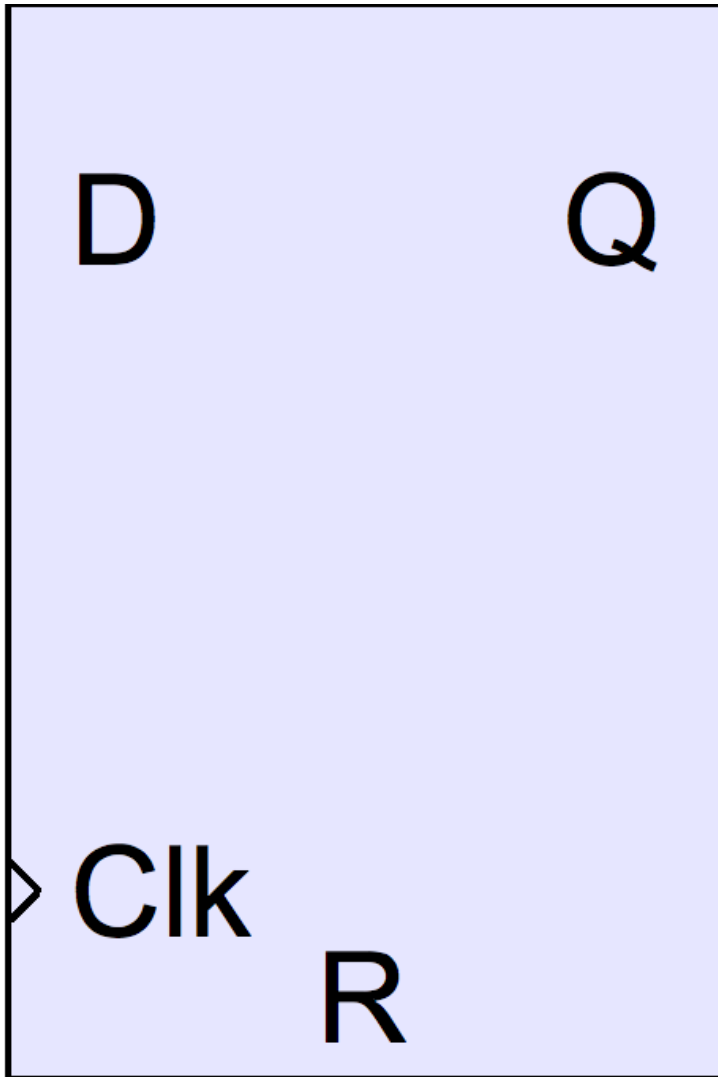


```
// Sequential Device Table:
//
// d      clk |  q
// -----+-----
// 0      /   |  0   rising edge
// 1      /   |  1   rising edge
// ?      ?   |  q   store
```

```
// Verilog HDL:

reg    q;           // register output

always @ (posedge clk)
begin
    q <= d;        // latch data
end
```

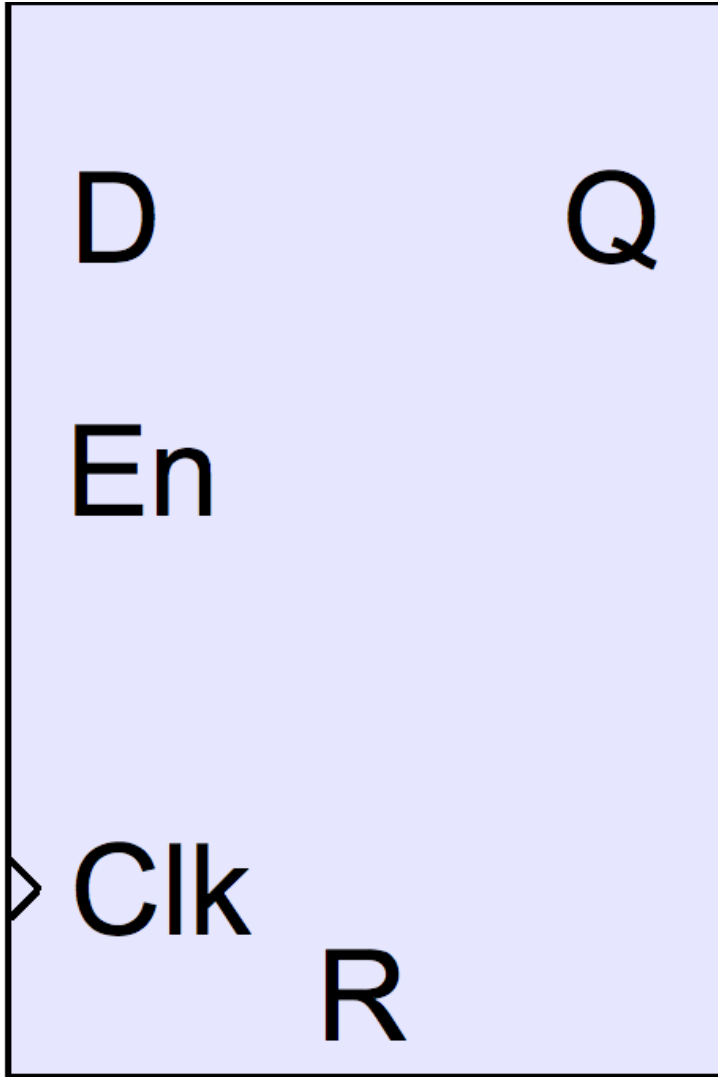


```
// Sequential Device Table:
//
// d   r   clk |   q
// -----+-----
// ?   1   ?   |   0   asynchronous
// 0   0   /   |   0   rising edge
// 1   0   /   |   1   rising edge
// ?   0   ?   |   q   store
```

```
// Verilog HDL:
```

```
reg    q;           // register output

always @ (posedge r or posedge clk)
begin
    if (r)           // reset
        q <= 1'b0;
    else
        q <= d;
end
```



```
// Sequential Device Table:
//
// d   r   en  clk |   q
// -----+-----
// ?   1   ?   ?   |   0
// ?   0   0   /   |   q
// 0   0   1   /   |   0
// 1   0   1   /   |   1
// ?   0   1   ?   |   q
```

```
// Verilog HDL:
```

```
reg    q;           // register output

always @ (posedge r or posedge clk)
begin
    if (r)           // reset
        q <= 1'b1;
    else
        if (en)      // data enable
            q <= d;
        else
            q <= q;
end
```

Design Flow

