Sequential Circuits

So far we have only been discussing combinational citcuits. This was good as far as our learning of Verilog language and its constructs are concerned. Practical FPGA circuits, however, almost always contains sequential circuits.

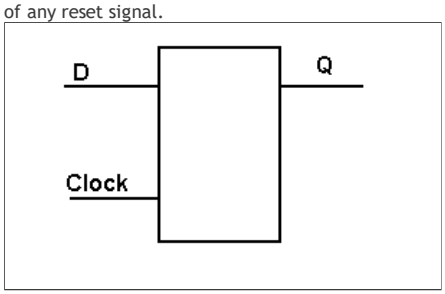
Combinational circuts do not have memory and its present output is a function only of present inputs. A sequential circuit, on the other hand, has memory and its present output depends not only upon present input but also upon past input(s).

A better term for past inputs is "state". A sequential circuit consists of finite states and its output depends upon present input and one of these states.

Implicit in the design of the sequential circuits is a global clock and the circuit operates on the rising or falling edge of the clock.

D Flip Flop

A D Flip Flop is the most basic building block of sequential circuit. From the abstraction at the top level, a D Flip Flop has an Clock and a Data D as input. It has one output designated as Q. For simplicity we do not assume presence of any reset signal.



This D Flip Flop functions as follows

1. When the Clock stays low or the clock stays high, the output does not change, it stays at its previous value.
2. When the Clock edge changes from low to high, the Output Q gets the value of the input D.

Here is the verilog implemmentation of D Flip Flop.

```
1.  // referencedesigner.com
2.  // D Flip Flop without Reset
3.
4.  module d_ff
5.     (
6.      input wire Clock,
7.      input wire D,
8.      output reg Q
9.     );
10.
11.    always @(posedge Clock)
12.       Q = D;
13. endmodule
```

The new thing to note is the line

```
always @(posedge Clock)
```

that triggers the posistive edge of clock event in the sensitivity list.

The posedge keyword specifies the direction of the clock signal changing from 0 to 1. Similarly, negedge keywork specifies the clock direction from 1 to 0.

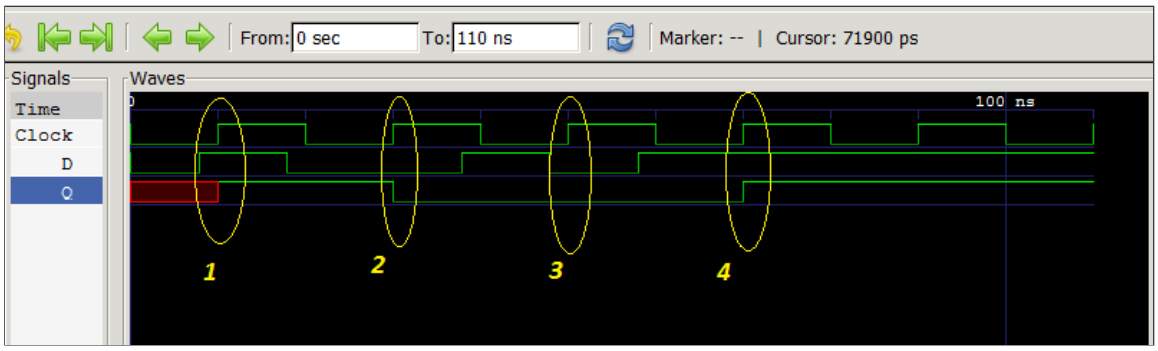
This D flip flop is a positive edge-triggered FF. An important thing to note is that the input signal D is not present in the sensitive list. The D signal is sampled only at the rising edge of the clk signal.

Let us now write a test bench fo the D Flip flop and verify its behavior. We will also add capability to see its waveform in GTK wave.

```
1.  `timescale 1ns / 1ps
```

```
module stimulus;
        reg Clock;
        reg D;
        wire Q;
        // Instantiate the Unit Under Test (UUT)
        d_ff d1 (
                .Clock(Clock),
                .D(D),
                .Q(Q)
        );

        integer i;

        initial begin
        $dumpfile("test.vcd");
    $dumpvars(0,stimulus);
        D = 0;
        #8  D= 1;
        #10 D= 0;
        #10 D= 0;
        #10 D =1;
    #10 D =0;
        #10 D = 1;
        #40;
    end

        initial begin
        Clock = 0;
        for ( i =0; i <=10; i= i+1)
        #10 Clock = ~Clock;
    end


                initial begin
                 $monitor("Clock=%d,D=%d,Q=%d \n",Clock,D,Q);
                 end

endmodule
```

One thing you will note that there are two initial statements in the stimulus. One that contains the clock and other that contains Data. It is a good idea to let the clock run freely using a for loop etc. And then we can focus on change in the Data. We have randomly assigned some values to Data that changes (or does not change) 2 ns before the rising edge of the clock . Look at the waveform generated using this stimulus

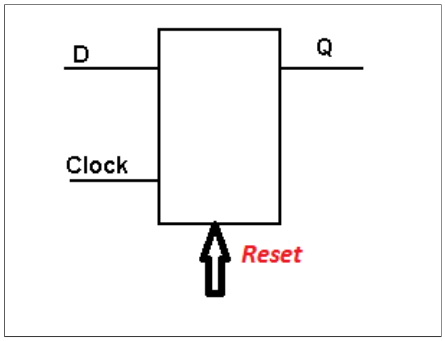


1. At rising edge of clock marked 1 - the Data D Input is 1, hence Q becomes 1 right after that. It is undefined x before that.
2. At rising edge of clock marked 2, the Data is 0, hence Q becomes 0 right after the clock rising edge.
3. At 3, the Data is again 0, hence Q stays 0.
4. At 4, Data is 1 at the rising egdge of clock, hence Q becomes 1

.

D Flip Flop with Asyncronous Reset

A D Flip Flop with Asynchronous Reset adds another Reset Input.



At the falling edge of the reset, the Output Q is reset to zero. This happens irrespective of the clock value. In other words, the reset is aysnc.

The following is the implemmentation of D Flip Flop with asynchronous reset.

```
1. // referencedesigner.com
2. // D Flip Flop with Asyncronous Reset
3.
4. module d_ff
5.    (
6.     input wire Clock,
7.     input wire D,
8.         input Rst,
9.     output reg Q
10.    );
11.
12.    always @(negedge Rst or  posedge Clock  )
13.       begin
14.           if (!Rst)
15.       Q = 0;
16.           else
17.           Q = D;
18.           end
19. endmodule
```

We will leave it as an exercize to write a testbench and verify the functionality of D Flip Flop with asynchronous reset.

D Flip Flop with Syncronous Reset

A D Flip Flop with Synchronous Reset also allows the reset, but the reset takes place only at clock edge. Let us say, we are allowing reset at its negative edge and the effect takes place at positive edge of clock.

The following is the implemmentation of D Flip Flop with synchronous reset in verilog.

```
1. // referencedesigner.com
2. // D Flip Flop with Syncronous Reset
3.
4. module d_ff
5.    (
6.     input wire Clock,
7.     input wire D,
8.         input Rst,
9.     output reg Q
10.    );
11.
12.    always @(posedge Clock  )
13.       begin
14.           if (!Rst)
15.       Q = 0;
16.           else
17.           Q = D;
18.           end
19. endmodule
```

< Previous Next >