Counter as clock divider

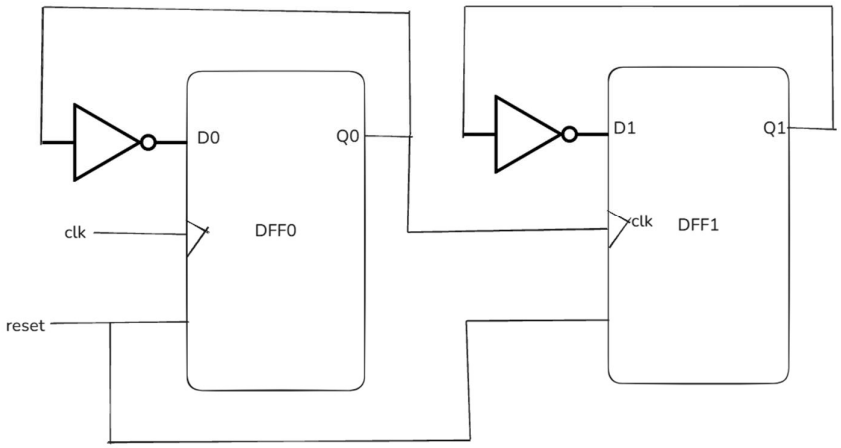

Figure 1: Block diagram

Truth table:

Clock cycle	Q1	Q0
0	0	0
1	0	1
2	1	0
3	1	1
4	0	0
5	0	1
6	1	0
7	1	1

Verilog code:

```
module Counter_as_clk_divider(input clk, input reset, output [1:0] q);
   wire q0, q1;
   dff d1(clk, reset, ~q0, q0);
   dff d2(clk, reset, ~q1, q1);
   assign q = {q1, q0};
endmodule
module dff(input clk, input reset, input d, output reg q);
   always @(posedge clk) begin
      if (reset)
```

```
        q <= 1'b0;
    else
        q <= d;
  end
endmodule
```

Schematics:

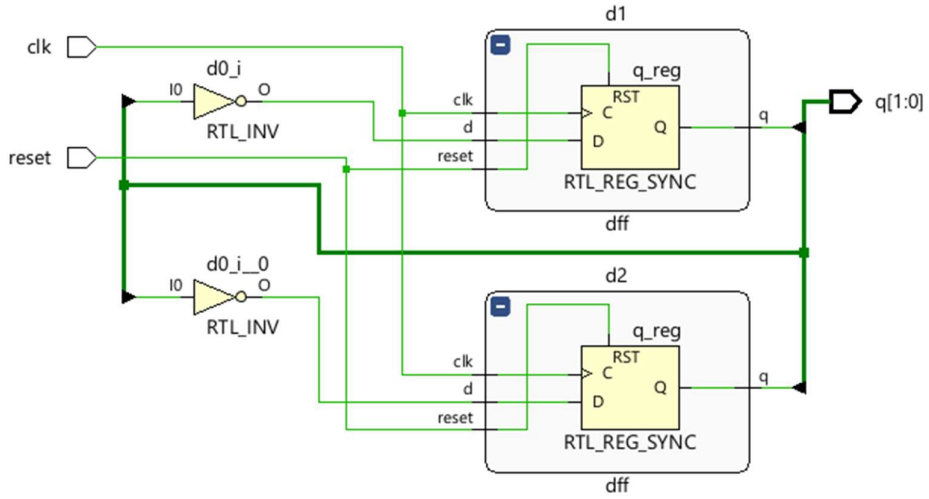

Figure 2: RTL Schematic

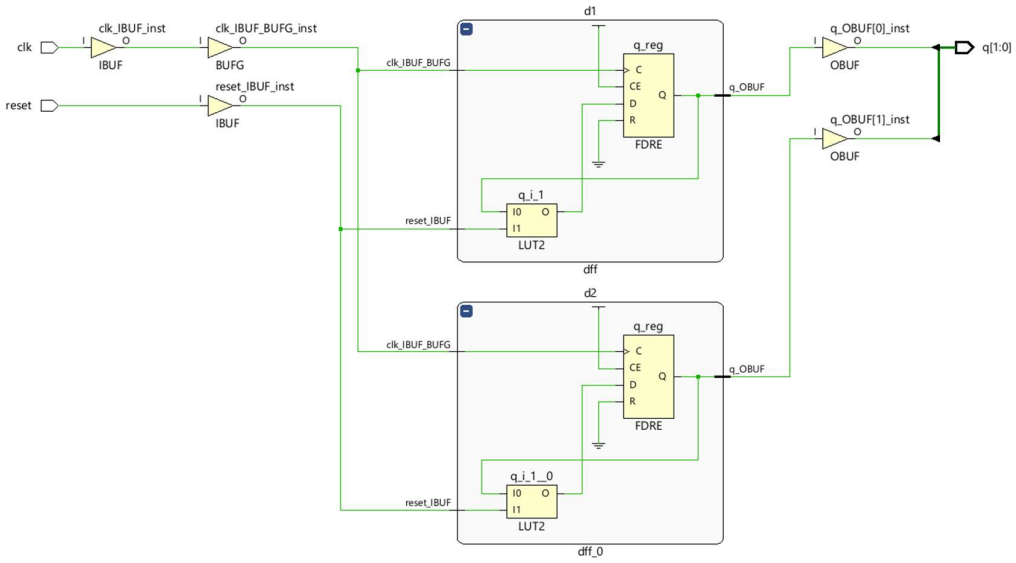

Figure 3: Synthesized Schematic

LUT's:

I1	I0	O=!I0 & !I1
0	0	1
0	1	0
1	0	0
1	1	0

Figure 4: LUT2

q_i_1__0

I1	I0	O=!I0 & !I1
0	0	1
0	1	0
1	0	0
1	1	0

Figure 5: LUT2

Testbench:

```
module counter_2bit_tb;
   reg clk;
   reg reset;
   wire [1:0] q;

   counter_2bit uut (
      .clk(clk),
      .reset(reset),
      .q(q)
   );

   initial begin
      clk = 0;
      forever #5 clk = ~clk;
```

```
    end

    initial begin
        reset = 1;
        #10;
        reset = 0;
        #100;
        $finish;
    end

    initial begin
        $monitor("time = %0t | clk = %b | reset = %b | q = %b", $time, clk, reset, q);
    end
endmodule
```

Timing diagram:

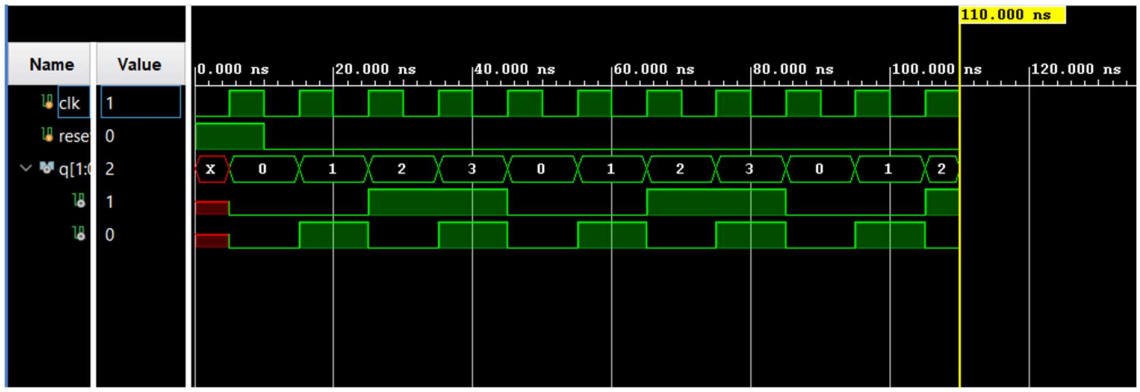

Figure 6: Timing diagram