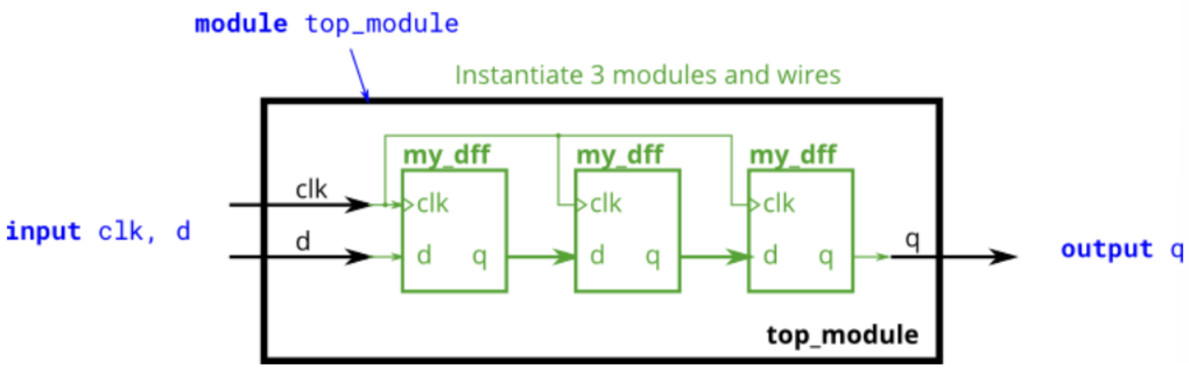
Module shift

← module_name   Previous   Next   module_shift8 →

You are given a module `my_dff` with two inputs and one output (that implements a D flip-flop). Instantiate three of them, then chain them together to make a shift register of length 3. The `clk` port needs to be connected to all instances.

The module provided to you is: `module my_dff ( input clk, input d, output q );`

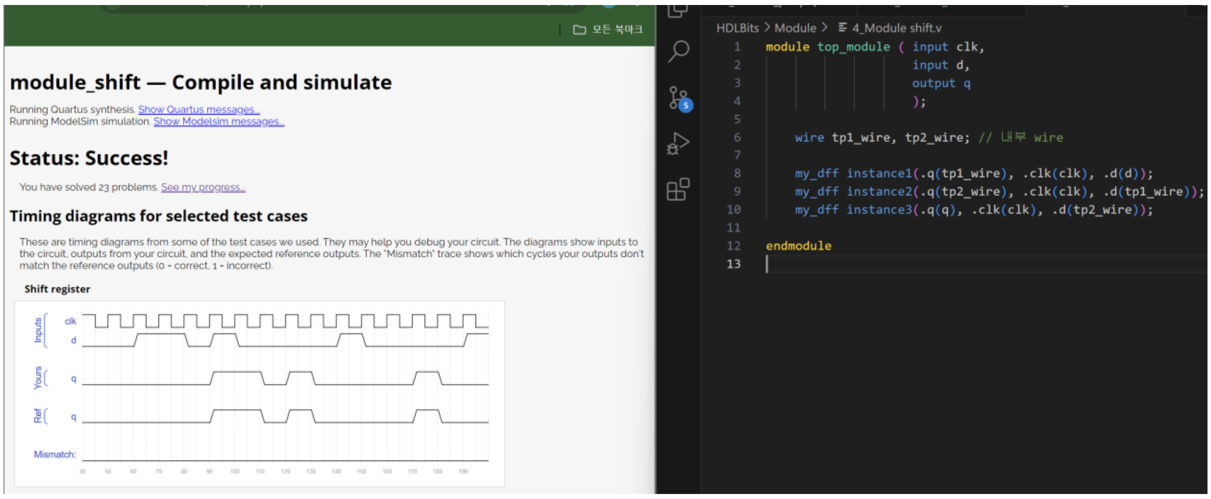
Note that to make the internal connections, you will need to declare some wires. Be careful about naming your wires and module instances: the names must be unique.



설명

- my_dff 3개를 엮어 shift register를 만들어라. Clock port는 모든 인스턴스에 연결되어 있어야 한다. 내부 연결을 만드려면, 몇 개의 wire를 선언할 필요가 있다.

코드 및 결과



module_shift — Compile and simulate

Running Quartus synthesis. Show Quartus messages...
Running ModelSim simulation. Show Modelsim messages...

Status: Success!

You have solved 23 problems. See my progress...

Timing diagrams for selected test cases

These are timing diagrams from some of the test cases we used. They may help you debug your circuit. The diagrams show inputs to the circuit, outputs from your circuit, and the expected reference outputs. The "Mismatch" trace shows which cycles your outputs don't match the reference outputs (0 - correct, 1 - incorrect).

Shift register

```verilog
module top_module ( input clk,
                    input d,
                    output q
                    );

    wire tp1_wire, tp2_wire; // 내부 wire

    my_dff instance1(.q(tp1_wire), .clk(clk), .d(d));
    my_dff instance2(.q(tp2_wire), .clk(clk), .d(tp1_wire));
    my_dff instance3(.q(q), .clk(clk), .d(tp2_wire));

endmodule

```

top_module 내부에서 따로 wire를 선언해 인스턴스끼리 신호를 주고 받을 수 있게 하자.