Module shift8

← module_shift (/wiki/module_shift)

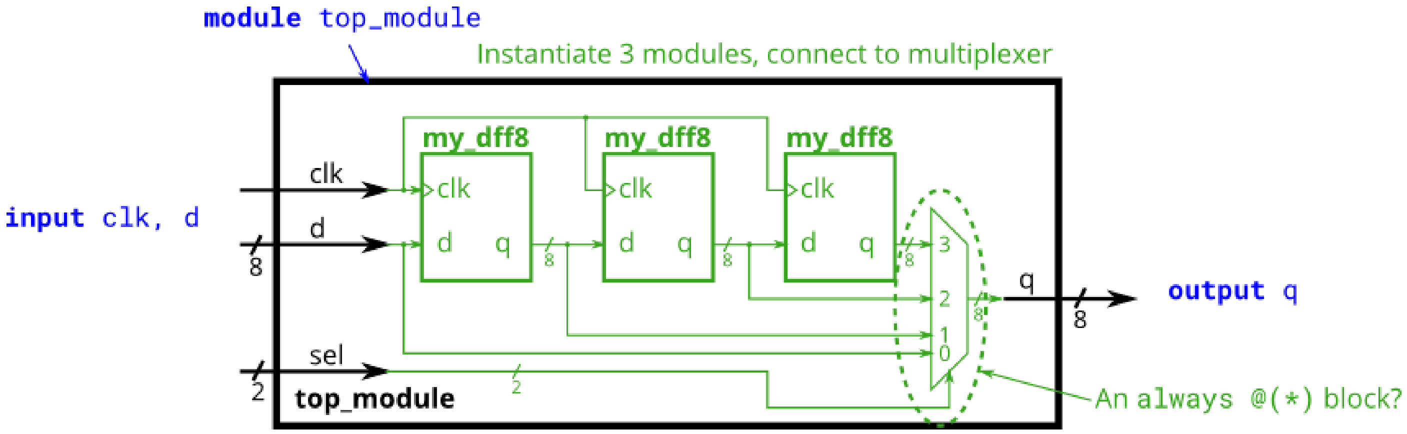
module_add (/wiki/module_add) →

This exercise is an extension of module_shift (/wiki/module_shift). Instead of module ports being only single pins, we now have modules with vectors as ports, to which you will attach wire vectors instead of plain wires. Like everywhere else in Verilog, the vector length of the port does not have to match the wire connecting to it, but this will cause zero-padding or trucation of the vector. This exercise does not use connections with mismatched vector lengths.

You are given a module `my_dff8` with two inputs and one output (that implements a set of 8 D flip-flops). Instantiate three of them, then chain them together to make a 8-bit wide shift register of length 3. In addition, create a 4-to-1 multiplexer (not provided) that chooses what to output depending on `sel[1:0]`: The value at the input d, after the first, after the second, or after the third D flip-flop. (Essentially, `sel` selects how many cycles to delay the input, from zero to three clock cycles.)

The module provided to you is: `module my_dff8 ( input clk, input [7:0] d, output [7:0] q );`

The multiplexer is not provided. One possible way to write one is inside an `always` block with a `case` statement inside. (See also: mux9to1v (/wiki/mux9to1v))



(/wiki/File:Module_shift8.png)

Module Declaration

```
module top_module (
    input clk,
    input [7:0] d,
    input [1:0] sel,
    output [7:0] q
);
```

Write your solution here

```
module top_module (
    input clk,
    input [7:0] d,
    input [1:0] sel,
    output [7:0] q
);

endmodule

```

Submit

Submit (new window)

Upload a source file... ⌄

Solution

Show solution

```
module top_module (
    input clk,
    input [7:0] d,
    input [1:0] sel,
    output reg [7:0] q
);

    wire [7:0] o1, o2, o3;      // output of each my_dff8

    // Instantiate three my_dff8s
    my_dff8 d1 ( clk, d, o1 );
    my_dff8 d2 ( clk, o1, o2 );
    my_dff8 d3 ( clk, o2, o3 );

    // This is one way to make a 4-to-1 multiplexer
    always @(*)     // Combinational always block
        case(sel)
            2'h0: q = d;
            2'h1: q = o1;
            2'h2: q = o2;
            2'h3: q = o3;
        endcase

endmodule

```

← module_shift ✓ (/wiki/module_shift)

module_add ✓ (/wiki/module_add) →

Retrieved from "http://hdlbits.01xz.net/mw/index.php?title=Module_shift8&oldid=1448 (http://hdlbits.01xz.net/mw/index.php?title=Module_shift8&oldid=1448)"

Category (/wiki/Special:Categories): **Modules (/wiki/Category:Modules)**

Problem Set Contents