Design a 2-bit counter counting in the order of 0,3,1,2,0,3,1,2…. You may start with the next-state table:

|  |  |  |  |
| --- | --- | --- | --- |
| Q1 | Q0 | D1 | D0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 |

Then, get D1=~Q1 and D0=~(Q1^Q0).

You may then ask them to write a Verilog module of the 2-bit counter with using this basic D flip-flop module, and run the simulator to get the waveform:

module Dff1 (D, clk, Q, Qbar);

input D, clk;

output reg Q, Qbar;

initial begin

Q = 0;

Qbar = 1;

end

always @(posedge clk) begin

#1

Q = D;

#1

Qbar = ~Q;

end

endmodule

module counter\_2\_bit (clk, Q);

input clk;

output [1:0] Q;

wire Q1, Q1bar, Q0, Q0bar, D1, D0;

assign D0 =~(Q1^Q0);

Dff1 C0 (D0, clk, Q0, Q0bar);

assign D1 = ~Q1;

Dff1 C1 (D1, clk, Q1, Q1bar);

assign Q[1] = Q1;

assign Q[0] = Q0;

endmodule

module test\_bench ();

wire clr, clk;

reg osc;

initial begin

osc = 0;

end

always begin

#10 osc = ~osc;

end

assign clr=1;

assign clk=osc;

wire [1:0] c2bitoutput;

counter\_2\_bit counter2bit(clk, c2bitoutput);

endmodule