In this recitation, the main objective is to get the students familiar with the entire process from truth table to Verilog simulation. Please ask the students to implement the two functions according to the following truth table for D1 and D0.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **A** | **B** | **C** | **D1** | **D0** |
| 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 1 |

D1 = (~A&C) | (~A&B) | (A&~B&~C)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| AB  C | 00 | 01 | 11 | 10 |
| 0 |  | 1 |  | 1 |
| 1 | 1 | 1 |  |  |

D0 = (B&C) |(~B&~C)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| AB  C | 00 | 01 | 11 | 10 |
| 0 | 1 |  |  | 1 |
| 1 |  | 1 | 1 |  |

module RCP1 (A, B, C, D0);

input C, A, B;

output D0;

assign D0 = (~A&C) | (~A&B) | (A&~B&~C);

endmodule

module RCP2 (A, B, C, D1);

input C, A, B;

output D1;

assign D1 = (B&C) |(~B&~C);

endmodule

module counter (clr, clk, OC);

input clr, clk;

output reg [2:0] OC;

initial begin

OC = 0;

end

always @(posedge clk) begin

if (clr == 0)

OC = 0;

else

OC = OC + 1;

end

endmodule

module test\_bench ();

wire P1A, P1B, P1C, P1D0;

wire P2B, P2C, P2A, P2D1;

wire clr, clk;

wire [2:0] counterO;

reg osc;

initial begin

osc = 0;

end

always begin

#10 osc = ~osc;

end

assign clr=1;

assign clk=osc;

counter C1(clr, clk, counterO);

assign P1A = counterO[2];

assign P1B = counterO[1];

assign P1C = counterO[0];

assign P2A = counterO[2];

assign P2B = counterO[1];

assign P2C = counterO[0];

RCP1 P1(P1A, P1B, P1C, P1D0);

RCP2 P2(P2A, P2B, P2C, P2D1);

endmodule