//Daniel Li 50995133 Project 1

//6 Speed Blender

module P1Blender(

input logic clk,

input logic reset,

input logic [3:0] Mode,

output logic [1:0] y); // Same setup as module doesn't change

typedef enum logic [3:0] {

S0 = 4'b0000,

S1 = 4'b0001,

S2 = 4'b0010,

S3 = 4'b0011,

S4 = 4'b0100,

S5 = 4'b0101,

S6 = 4'b0110,

S7 = 4'b0111,

S8 = 4'b1000,

S9 = 4'b1001,

S10 = 4'b1010,

S11 = 4'b1011,

S12 = 4'b1100,

PULSE33 = 4'b1101,

PULSE66 = 4'b1110,

PULSE100 = 4'b1111} statetype;

statetype state, nextstate;

//State Register identical to lecture

always\_ff @(posedge clk)

if (reset) state <= S0;

else state <= nextstate;

//Next state logic identical to lecture as well

always\_comb

case (state)

S0: if (Mode == 4'b1111) nextstate = PULSE100; //This state is the base state, being the off state, and will output 00

else if (Mode == 4'b1110) nextstate = PULSE66;

else if (Mode == 4'b1101) nextstate = PULSE33;

else if (Mode == 4'b0110) nextstate = S6;

else nextstate = S1;

S1: if (Mode == 4'b1111) nextstate = PULSE100;// This state is the next off state, which will output 0 and has more conditions to jump to a 'power' state

else if (Mode == 4'b1110) nextstate = PULSE66;

else if (Mode == 4'b1101) nextstate = PULSE33;

else if (Mode == 4'b0110) nextstate = S7;

else if (Mode == 4'b0101) nextstate = S7;

else nextstate = S2;

S2: if (Mode == 4'b1111) nextstate = PULSE100;// This is the next off state, which will output 0 and has an additional condition to jump to an active 'power' state

else if (Mode == 4'b1110) nextstate = PULSE66;

else if (Mode == 4'b1101) nextstate = PULSE33;

else if (Mode == 4'b0110) nextstate = S8;

else if (Mode == 4'b0101) nextstate = S8;

else if (Mode == 4'b0100) nextstate = S8;

else nextstate = S3;

S3: if (Mode == 4'b1111) nextstate = PULSE100;// Same as S2 but with one more condition

else if (Mode == 4'b1110) nextstate = PULSE66;

else if (Mode == 4'b1101) nextstate = PULSE33;

else if (Mode == 4'b0110) nextstate = S9;

else if (Mode == 4'b0101) nextstate = S9;

else if (Mode == 4'b0100) nextstate = S9;

else if (Mode == 4'b0011) nextstate = S9;

else nextstate = S4;

S4: if (Mode == 4'b1111) nextstate = PULSE100;// Same as S3 but with one more condition

else if (Mode == 4'b1110) nextstate = PULSE66;

else if (Mode == 4'b1101) nextstate = PULSE33;

else if (Mode == 4'b0110) nextstate = S10;

else if (Mode == 4'b0101) nextstate = S10;

else if (Mode == 4'b0100) nextstate = S10;

else if (Mode == 4'b0011) nextstate = S10;

else if (Mode == 4'b0010) nextstate = S10;

else nextstate = S5;

S5: if (Mode == 4'b1111) nextstate = PULSE100;//Last off state, will always cycle back to S0 unless instructed to go to the PULSE states

else if (Mode == 4'b1110) nextstate = PULSE66;

else if (Mode == 4'b1101) nextstate = PULSE33;

else if (Mode == 4'b0110) nextstate = S11;

else if (Mode == 4'b0101) nextstate = S11;

else if (Mode == 4'b0100) nextstate = S11;

else if (Mode == 4'b0011) nextstate = S11;

else if (Mode == 4'b0010) nextstate = S11;

else if (Mode == 4'b0001) nextstate = S11;

else nextstate = S0;

S6: if (Mode == 4'b1111) nextstate = PULSE100;// First power state, will always advance to the next power state, making it so that if S7 is reached at any time

else if (Mode == 4'b1110) nextstate = PULSE66;//Then the duty cycle is capped, unless interrupted by the PULSE state conditions

else if (Mode == 4'b1101) nextstate = PULSE33;

else nextstate = S7;

S7: if (Mode == 4'b1111) nextstate = PULSE100;// Next power state, same function as the first power state

else if (Mode == 4'b1110) nextstate = PULSE66;

else if (Mode == 4'b1101) nextstate = PULSE33;

else nextstate = S8;

S8: if (Mode == 4'b1111) nextstate = PULSE100;// Next power state, same function as the first power state

else if (Mode == 4'b1110) nextstate = PULSE66;

else if (Mode == 4'b1101) nextstate = PULSE33;

else nextstate = S9;

S9: if (Mode == 4'b1111) nextstate = PULSE100;// Same as S8

else if (Mode == 4'b1110) nextstate = PULSE66;

else if (Mode == 4'b1101) nextstate = PULSE33;

else nextstate = S10;

S10: if (Mode == 4'b1111) nextstate = PULSE100;// Same as S9

else if (Mode == 4'b1110) nextstate = PULSE66;

else if (Mode == 4'b1101) nextstate = PULSE33;

else nextstate = S11;

S11: if (Mode == 4'b1111) nextstate = PULSE100; // // Last power state, depending on the input mode, will cycle to the first off or first power state.

else if (Mode == 4'b1110) nextstate = PULSE66;

else if (Mode == 4'b1101) nextstate = PULSE33;

else if (Mode == 4'b0110) nextstate = S6;

else nextstate = S0;

PULSE33: if (Mode == 4'b1101) nextstate = PULSE33; // Lowest full duty cycle output state, with an output of 01

else nextstate = S0;

PULSE66: if (Mode == 4'b1110) nextstate = PULSE66;// Middle full duty cycle output state, with an output of 10

else nextstate = S0;

PULSE100: if (Mode == 4'b1111) nextstate = PULSE100;// Full power, full duty cycle, essentially the same as keeping the input mode on full blend

else nextstate = S0; // Meant to emulate a PULSE function on a blender, such that if the momentary button is released, the output drops to 00 on the next clock cycle

default: nextstate = S0; //

endcase

//Output

assign y = (state == 4'b1110)? 2'b10 :

((state == 4'b1101)? 2'b01 :

((state >= 4'b0110)? 2'b11 : 2'b00)); // Calculating the outputs with conditions. The system won't error as the next clock cycle will rectify the state. The only way to fall into the errors is to have the input modes change at faster than clock cycle speeds at very specific times.

endmodule