module p7\_1(Z, D, S);

output Z;

input[3:0] D;

input[1:0] S;

not n0 (inv\_S0, S[0]);

not n1 (inv\_S1, S[1]);

and a0 (a0\_o, D[0], inv\_S1, inv\_S0);

and a1 (a1\_o, D[1], inv\_S1, S[0]);

and a2 (a2\_o, D[2], S[1], inv\_S0);

and a3 (a3\_o, D[3], S[1], S[0]);

or o0 (Z, a0\_o, a1\_o, a2\_o, a3\_o);

endmodule

module p7\_2(Z, D, S);

output Z;

input[1:0] S;

input[3:0] D;

wire Z;

wire[1:0] S;

wire[3:0] D;

assign Z = D[S];

endmodule

module p7\_3(Z, D, S);

output Z;

input[3:0] D;

input[1:0] S;

reg Z;

wire[1:0] S;

wire[3:0] D;

always @(S or D)

begin

Z = D[S];

end

endmodule

module p7\_4(Z, D, S);

output Z;

input[1:0] S;

input[3:0] D;

wire Z;

wire[1:0] S;

wire[3:0] D;

assign Z = ( S == 0 )? D[0] : (( S == 1 )? D[1] : ((S == 2 )? D[2] : D[3]));

endmodule

module p7\_5(Z, D, S);

output Z;

input[1:0] S;

input[3:0] D;

reg Z;

wire[1:0] S;

wire[3:0] D;

always @( S or D )

begin

case( S )

0 : Z = D[0];

1 : Z = D[1];

2 : Z = D[2];

3 : Z = D[3];

endcase

end

endmodule

module p7\_6(Z, D, S);

output Z;

input[1:0] S;

input[3:0] D;

reg Z;

wire[1:0] S;

wire[3:0] D;

always @( S or D)

begin

Z = ( ~S[0] & ~S[1] & D[0] ) | ( S[0] & ~S[1] & D[1] ) | ( ~S[0] & S[1] & D[2] ) | ( S[0] & S[1] & D[3] );

end

endmodule

module tb\_p7();

reg[3:0] D;

reg[1:0] S;

wire Z;

integer i;

p7\_4 UUT(Z, D, S);

initial

begin

#10 $monitor("UUT | D = %b", D, " | S = %b", S, " | Z = ", Z);

for( i = 0; i <= 15; i = i + 1)

begin

D = i;

S = 0; #10;

S = 1; #10;

S = 2; #10;

S = 3; #10;

$display("\n\n");

end

end

endmodule

