***LAB 03: Latches, Flip-Flops and Registers***

**Task 1: RS Latch via Gates/Equations**

File: B\_latchRSg.v, B\_latchRSe.v

FPGA: X.

Code:

RS LATCH (GATES)

module B\_latchRSg(

input clock, R, S,

output Q

);

wire Qa, Qb, R\_g, S\_g/\* synthesis keep \*/;

and(R\_g, R, clock);

and(S\_g, S, clock);

or(Qa, R\_g, Qb);

or(Qb, S\_g, Qa);

assign Q = Qa;

endmodule

RS LATCH (EQUATIONS)

module B\_latchRSe(

input clock, R, S,

output Q

);

wire Qa, Qb;

assign Qa = ~((R & clock) | Qb);

assign Qb = ~((S & clock) | Qa);

assign Q = Qa;

endmodule

**Task 2: D Latch Implementation using Equations**

File: B\_latchDe.v

FPGA: X.

Code:

MAIN

module lab03part02(

input [1:0]SW,

output[0:0]LEDR

);

B\_latchDe testDe(SW[1], SW[0], LEDR[0]);

endmodule

D LATCH (EQUATIONS)

module B\_latchDe(

input CLK, D,

output Q

);

wire Qa, Qb;

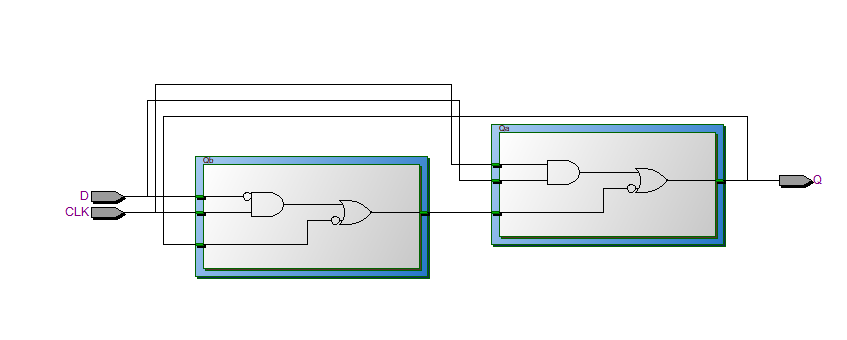
assign Qa = ~(~(D & CLK) & Qb);

assign Qb = ~(~(~D & CLK) & Qa);

assign Q = Qa;

endmodule

Output (Technology Map):



**Task 3: Master Slave D Flip Flop**

File: lab03part03.v, B\_flipflopDmse.v, B\_latchDe.v

FPGA: DONE.

Code:

MAIN

module lab03part03(

input [1:0]SW,

output [0:0]LEDR/\*,

output [2:0]LEDG\*/

);

/\* assign LEDG[0] = LEDR[0];

assign LEDG[1] = SW[0];

assign LEDG[2] = SW[1]; \*/

B\_flipflopDmse ff1(SW[1], SW[0], LEDR[0]);

endmodule

MASTER SLAVE D-FLIPFLOP [POSITIVE EDGE]

module B\_flipflopDmse(

input CLCK, D,

output Q

);

wire Qm, Qs, CL2;

assign CL2 = ~CLCK;

B\_latchDe ffa(CL2, D, Qm);

B\_latchDe ffb(CLCK, Qm, Qs);

assign Q = Qs;

endmodule

**Task 4: D LATCH, POSEDGE/NEGEDGE D FLIPFLOPS**

File: lab03part04.v, B\_memoryFlops.v

FPGA: X.

Code:

MAIN

module lab03part04(

output Q1, Q2, Q3,

input D, Clk

);

mem\_DLatch dl1(Q1, D, Clk);

mem\_Dflippos dff1(Q2, D, Clk);

mem\_Dflipneg dff2(Q3, D, Clk);

endmodule

MEMORY (LATCHES, FLIPFLOPS)

/\* GATED D LATCH \*/

module mem\_DLatch(

output reg Q,

input D, clk

);

always@(D, clk)

begin

if(clk)

Q <= D;

end

endmodule

/\* POSITIVE EDGE TRIGGERED - D FLIP FLOP \*/

module mem\_Dflippos(

output reg Q,

input D, clk

);

always@(posedge clk)

begin

Q <= D;

end

endmodule

/\* NEGATIVE EDGE TRIGGERED - D FLIP FLOP \*/

module mem\_Dflipneg(

output reg Q,

input D, clk

);

always@(negedge clk)

begin

Q <= D;

end

endmodule

**Task 5: D Latch Implementation using Equations**

File:

FPGA: X.

Code:

MAIN

**Task 2: D Latch Implementation using Equations**

File: B\_latchDe.v

FPGA: X.

Code:

D LATCH (EQUATIONS)