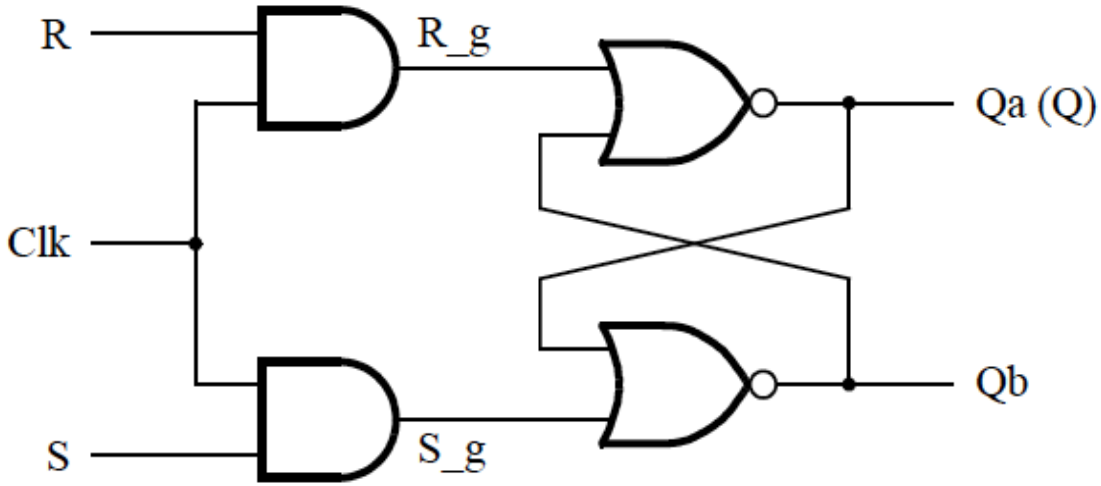
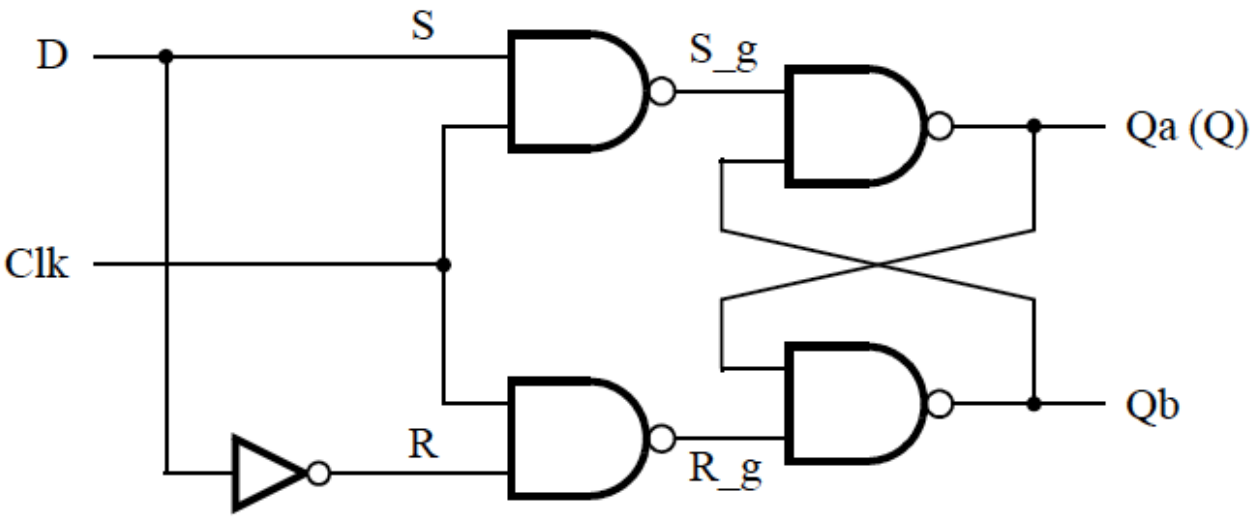
DE0 Lab5

Latches, Flip-flops, and Registers




```
// A gated RS latch
module  part1 (Clk, R, S, Q);
    input Clk, R, S;
    output Q;

    wire R_g, S_g, Qa, Qb /* synthesis keep */ ;

    and (R_g, R, Clk);
    and (S_g, S, Clk);
    nor (Qa, R_g, Qb);
    nor (Qb, S_g, Qa);

    assign Q = Qa;

endmodule
```

```
// A gated RS latch
module  part1 (Clk, R, S, Q);
    input Clk, R, S;
    output Q;

    wire R_g, S_g, Qa, Qb /* synthesis keep */ ;

    assign R_g = R & Clk;
    assign S_g = S & Clk;
    assign Qa = ~(R_g | Qb);
    assign Qb = ~(S_g | Qa);

    assign Q = Qa;

endmodule
```

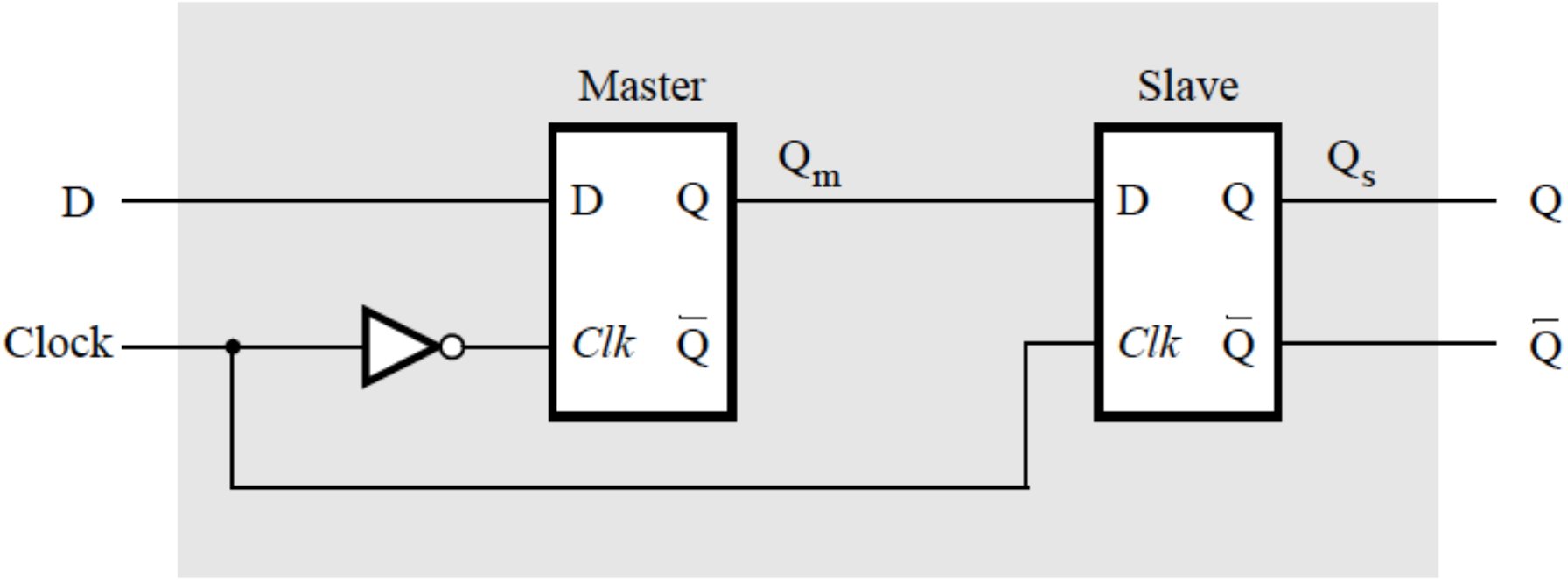



```
module D_latch (input Clk, D, output Q);

    wire S, R, S_g, R_g, Qa, Qb

    assign S = D;
    assign R = ~D;
    assign S_g = ~(S & Clk);
    assign R_g = ~(R & Clk);
    assign Qa = ~(S_g & Qb);
    assign Qb = ~(R_g & Qa);

    assign Q = Qa;

endmodule
```

```
module D_latch (D, Clk, Q);
input D, Clk;
output reg Q;
always @ (D, Clk)
if (Clk)
Q = D;
endmodule
```

Part 1
Implement a master-slave D flip-flop

A master-slave D flip-flop



Steps

1. Create a new Quartus II project. Generate a Verilog file that instantiates two copies of your gated D latch module to implement the master-slave flip-flop.
2. Include in your project the appropriate input and output ports for the Altera DE0 board. Use switch SW0 to drive the D input of the flip-flop, and use SW1 as the Clock input. Connect the Q output to LEDG0 .
3. Compile your project.
4. Use the Technology Viewer to examine the D flip-flop circuit, and use simulation to verify its correct operation.
5. Download the circuit onto the DE0 board and test its functionality by toggling the D and Clock switches and observing the Q output.

Verilog code (1)

```
// SW[0] is the flip-flop's D input,
//SW[1] is the edge-sensitive Clock, LEDG[0] is Q
module DFF(input [1:0] SW, output [0:0] LEDG);
        wire Qm, Qs;
        // module D_latch (input Clk, D, output Q);
        D_latch U1 (     ,      ,      );
        D_latch U2 (     ,      ,      );

        assign LEDG[0] = Qs;
endmodule
```

Part 2

To design a 8-bit register

Steps

1. Create a new Quartus II project which will be used to implement the desired circuit on the Altera DE0 board.
2. Write a Verilog file that provides the necessary functionality. Use KEY0 as an active-low asynchronousreset, and use KEY1 as a clock input.
3. Include the Verilog file in your project and compile the circuit.
4. Assign the pins on the FPGA to connect to the switches and 7-segment displays, as indicated in the User Manual for the DE0 board.
5. Recompile the circuit and download it into the FPGA chip.
6. Test the functionality of your design by toggling the switches and observing the output displays.

Verilog code

```
module Lab5_part2 (SW, KEY, HEX3, HEX2, HEX1, HEX0);
        input [7:0] SW;
        input [1:0] KEY;                    // Used for reset and enable for A_reg
        output [0:6] HEX3, HEX2, HEX1, HEX0;
        wire [7:0] A, B;
        // instantiate module regn (R, Clock, Resetn, Q);
        regn A_reg (SW, KEY[1], KEY[0], A);
        assign B = SW;
        // drive the displays through 7-seg decoders
        hex7seg digit_3 (A[7:4], HEX3);
        hex7seg digit_2 (A[3:0], HEX2);
        hex7seg digit_1 (B[7:4], HEX1);
        hex7seg digit_0 (B[3:0], HEX0);
endmodule
```

Verilog code

```
module regn (R, Clock, Resetn, Q);
        parameter n = 8;
        input [n-1:0] R;
        input Clock, Resetn;
        output [n-1:0] Q;
        reg [n-1:0] Q;
        always @(posedge Clock or negedge Resetn)
                if (Resetn == 0)
                        Q <= {n{1'b0}};
                else
                        Q <= R;
endmodule
```