实验报告

实验题目：Lab 6　　日期：2018 年 11 月 9 日

姓名:刘紫檀 学号:PB17000232　　成绩:_____

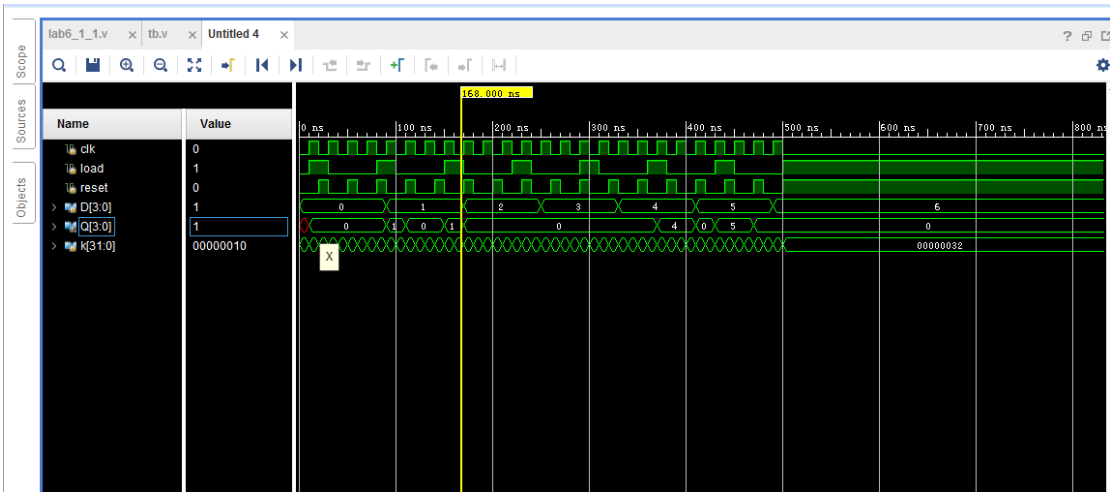
实验目的：

完成 1-1， 2-2， 2-3。

截图：

```
//1-1 代码
`timescale 1ns / 1ps
module reg_sync_reset(
    input clk,
    input load,
    input reset,
    input [3:0] D,
    output reg [3:0] Q
    );
always @ (posedge clk) begin
if (reset)
    Q <= 4'b0000;
else if (load)
    Q <= D;
end
endmodule

//1-1 仿真代码
`timescale 1ns / 1ps
module tb();
    reg clk;
    reg load;
    reg reset;
    reg [3:0] D;
    wire [3:0] Q;

integer k;
reg_sync_reset DUT (clk, load, reset, D, Q);

initial
```

```
begin
clk = 0;
load = 0;
reset = 0;
D = 4'b0;

for (k=0; k < 50; k=k+1)
begin
    #10 clk = ~clk;
    if (k % 7 == 0 || k % 7 == 1)
        load = 1;
    else
        load = 0;
    D = k / 8;
    reset = k % 3;
end


$display("Simulation Done");
end

endmodule
```



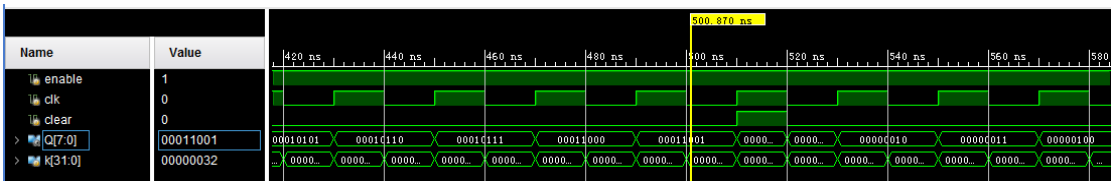
6-1-1 仿真截图

6-1-1 下载照片

```
//6-2-2 代码
`timescale 1ns / 1ps
//clear is async
module sync_counter(
    input enable,
    input clk,
    input clear,
    output [7:0] Q
    );
    wire [7:0] intermediate;
    sync_counter_block b1(enable, clk, clear, Q[0], intermediate[0]);
    sync_counter_block b2(intermediate[0], clk, clear, Q[1], intermediate[1]);
    sync_counter_block b3(intermediate[1], clk, clear, Q[2], intermediate[2]);
    sync_counter_block b4(intermediate[2], clk, clear, Q[3], intermediate[3]);
    sync_counter_block b5(intermediate[3], clk, clear, Q[4], intermediate[4]);
    sync_counter_block b6(intermediate[4], clk, clear, Q[5], intermediate[5]);
    sync_counter_block b7(intermediate[5], clk, clear, Q[6], intermediate[6]);
    sync_counter_block b8(intermediate[6], clk, clear, Q[7], intermediate[7]);
endmodule

module sync_counter_block(
    input Qpprev, // Previous Q product
    input clk,
    input clear,
    output Q,
    output Qpnext
    );
```

```
    assign Qpnext = Qpprev & Q;
    T_ff tf(Q, Qpprev, clk, clear);
endmodule

module T_ff(
    output Q,
    input T,
    input clk,
    input clear
    );

    wire D;
    xor (D, Q, T);
    D_ff df(D, clk, clear, Q);
endmodule



module D_ff(
    input D,
    input clk,
    input clear,
    output reg Q
    );

always @ (posedge clk or negedge clear)
    if (clear)
        Q <= 0;
    else
        Q <= D;
endmodule

//6-2-2 Testbench
module tb();
    reg enable;
    reg clk;
    reg clear;
    wire [7:0] Q;
    integer k;
    sync_counter DUT(enable, clk, clear, Q);

    initial begin
        clk = 0;
        enable = 1;
        clear = 0;
```

```
        for (k = 0; k < 600; k = k + 1) begin
            #10 clk = ~clk;
            if (k % 50 == 0)
                clear = 1;
            else clear = 0;

        end
    end

endmodule
```



6-2-2 仿真截图

6-2-2 下载照片

```
//6-2-3 代码
`timescale 1ns / 1ps
module counter(
    input load,
    input clk,
    input clear,
    input enable,
    output [3:0] Q
    );
```

```
    reg [3:0] count;
    wire cnt_done;
    assign cnt_done = ~| count;
    assign Q = count;
    always @(posedge clk)
    if (clear)
        count <= 0;
    else if (enable)
        if (load | cnt_done)
            count <= 4'b1010; // decimal 10
        else
            count <= count - 1;
endmodule

//6-2-3 仿真代码
`timescale 1ns / 1ps
module tb();
    reg enable;
    reg clk;
    reg clear;
    reg load;
    wire [3:0] Q;
    integer k;
    counter DUT(load, clk, clear,  enable, Q);

    initial begin
        clk = 0;
        enable = 1;
        clear = 0;
        load = 0;
        for (k = 0; k < 60; k = k + 1) begin
            #10 clk = ~clk;
            if (k == 2) load = 1;
            if (k == 4) load = 0;
           // if (k % 7 == 0)
               // clear = ~clear;

        end
    end

endmodule
```

实验总结：这次实验我掌握了触发器的 Verilog HDL 描述。

其中有一点很值得注意：always @(negedge reset)，如果写成 always @ (reset)，就会出问题。