Udemy Sequential Lab

sequential - Google 雲端硬碟

serial2parallel、parallel2serial、combinational_lock、vending_machine

第4組 2023.8.16

C code » Verilog

1. Add clock and reset signal

```
void c_code(
    bool a,
    ap_int<8> b,

    bool &c,
    int &d)
{

}
```

```
module verilog(
    input clk,
    input rst_n,

    input a,
    input [7:0] b,

    output c,
    output [31:0] d
    );

endmodule
```

C code in HLS

- Add signal automatically

Verilog in Vivado

- Add signal by self

C code » Verilog

2. Notice pragma in HLS

- Interface

```
#pragma HLS INTERFACE ap_none port=a
#pragma HLS INTERFACE ap_vld port=a
```

- Unroll

```
#pragma HLS UNROLL
```

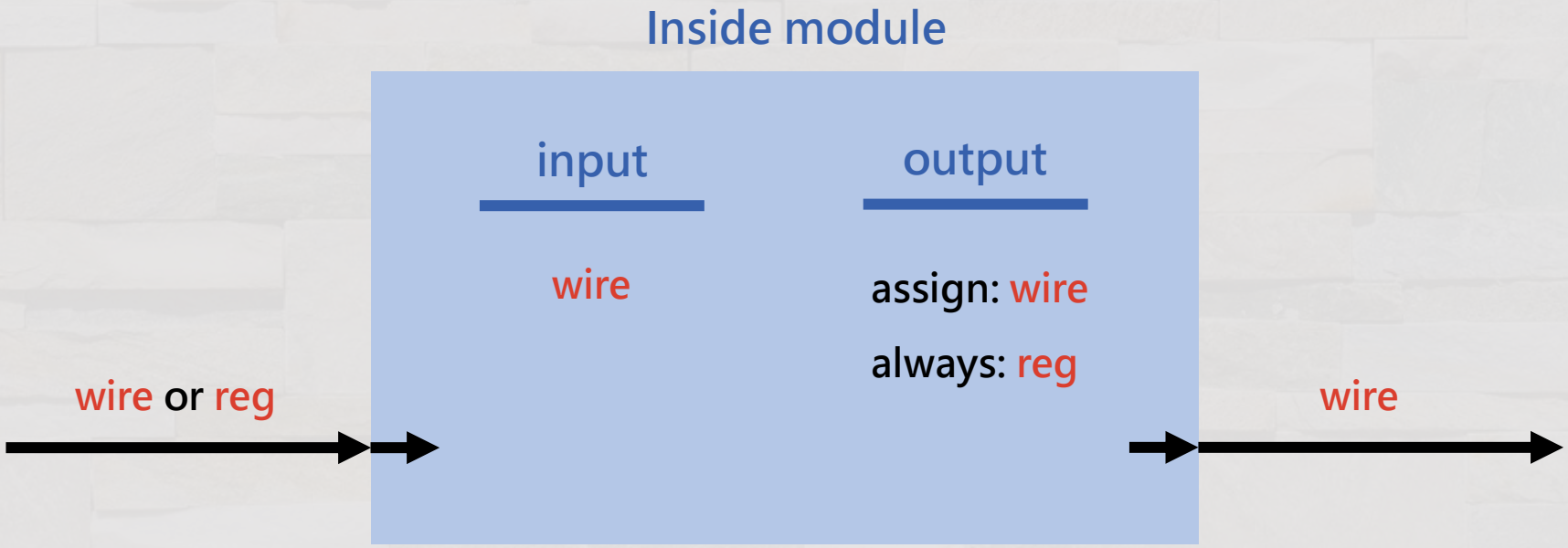
- Pipeline

```
#pragma HLS PIPELINE
```

- Others

C code » Verilog

3. Choose wire or reg in Verilog



C code » Verilog

4. Sequential circuit typing style

```
static bool a = 0;
bool next_a;

// Combinational Part
if(a == 0)
    next_a = 1;
else
    next_a = 0;

// Sequential Part
a = next_a;
```

C code in HLS

- Sequential part is later

```
reg a = 0;
reg next_a;

// Combinational Part
always @(*)
begin
    if(a == 0)
        next_a = 1;
    else
        next_a = 0;
end

// Sequential Part
always @(posedge clk or negedge rst_n)
begin
    if(!rst_n)
        a <= 0;
    else
        a <= next_a;
end
```

Verilog in Vivado

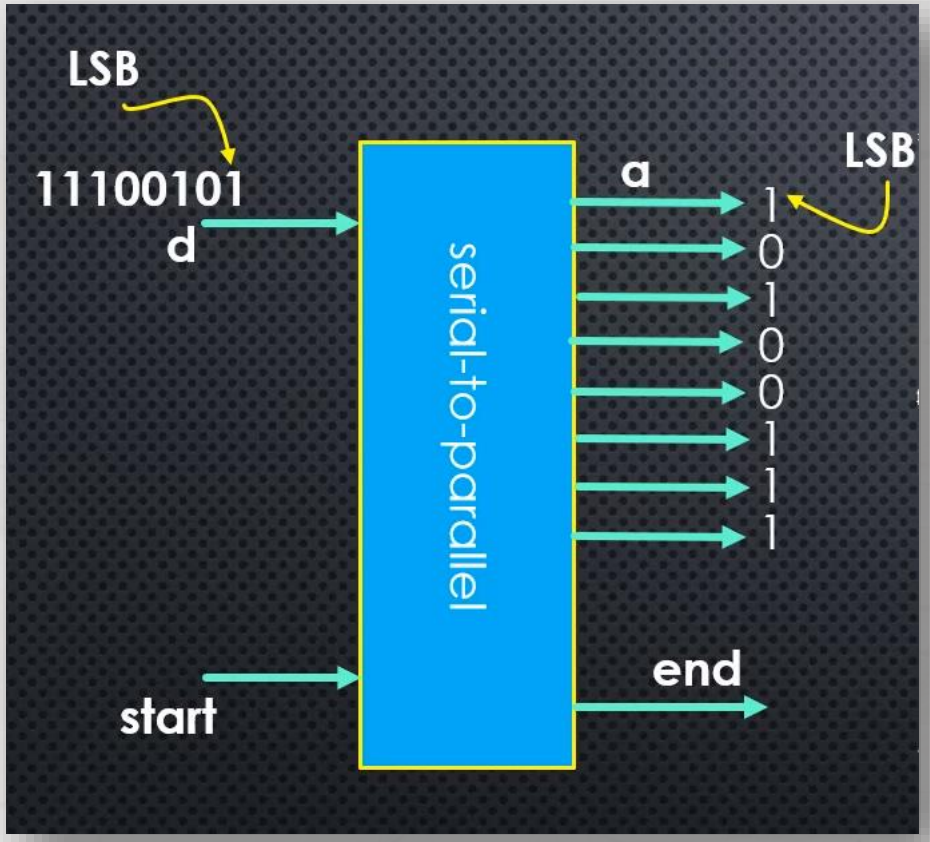
- Sequential part use non-blocking

serial2parallel

FSM design

Lab 1

- **Design a counter count from 0-8**



serial2parallel

Coding

Lab 1

- **Testbench**

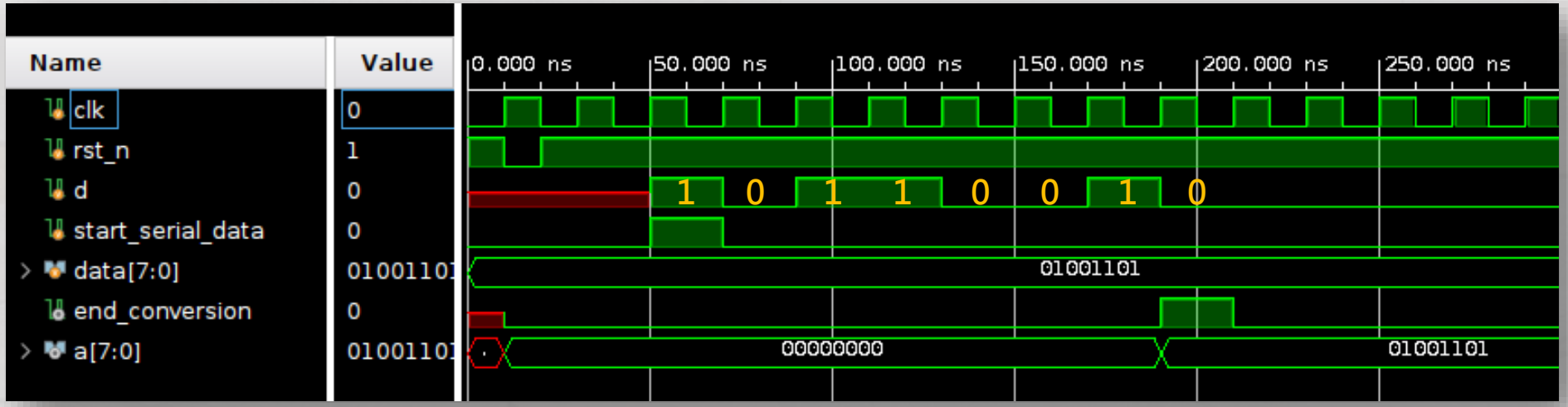
```
module serial2parallel_tb();

reg clk, rst_n, d, start_serial_data;
reg [7:0] data = 8'b01001101;
wire end_conversion;
wire [7:0] a;

serial2parallel S0 (.clk(clk), .rst_n(rst_n),
                    .serial_start(start_serial_data), .d(d),
                    .end_conversion(end_conversion), .a(a));

always #10 clk = ~clk;

initial begin

    clk = 0;
    start_serial_data = 0;
    $display("=== Initialize Success ===");
    rst_n = 1;
    #10 rst_n = 0;
    #10 rst_n = 1;
    $display("=== Reseting ===");

    #30 start_serial_data = 1;
    for(integer i = 0; i < 8; i = i + 1)
    begin
        d = data[i];
        #20 start_serial_data = 0;
    end
    $display("=== Test Result ===");
    #20 $display("a = %b", a);

    if(a == data)
        $display("=== Test Passed ===");
    else
        $display("=== Test Failed ===");

end

endmodule
```

serial2parallel

Coding

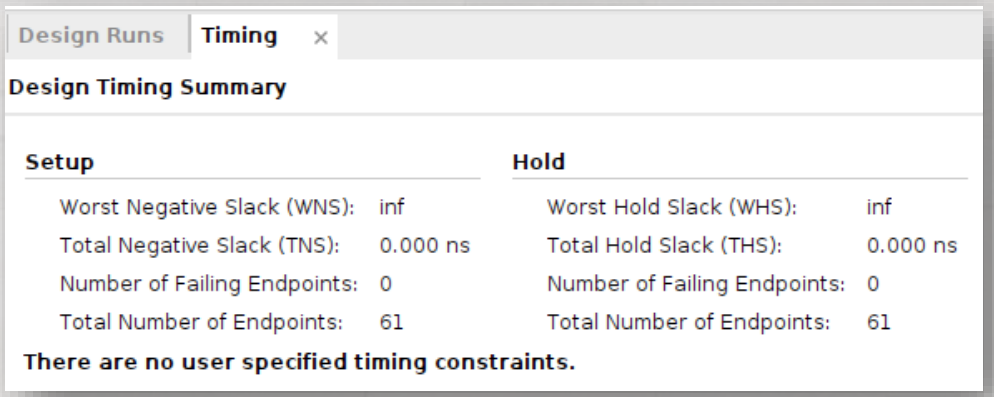
Lab 1

■ Simulation result



■ Compare to HLS

	FF	LUT	Timing
HLS	49	163	5.140 ns
Vivado	28	11	0 ns

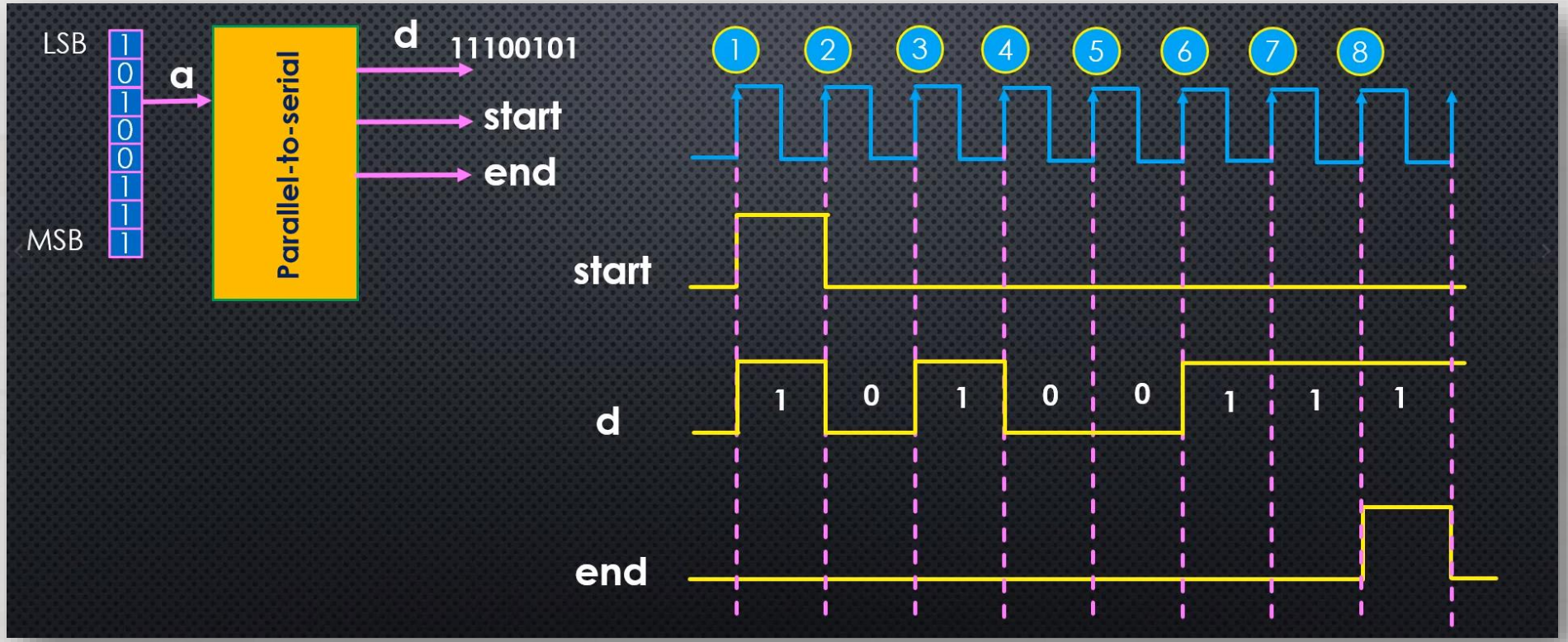


parallel2serial

FSM design

Lab 2

- **Design a counter count from 0-8**



parallel2serial

Coding

Lab 2

- **Testbench**

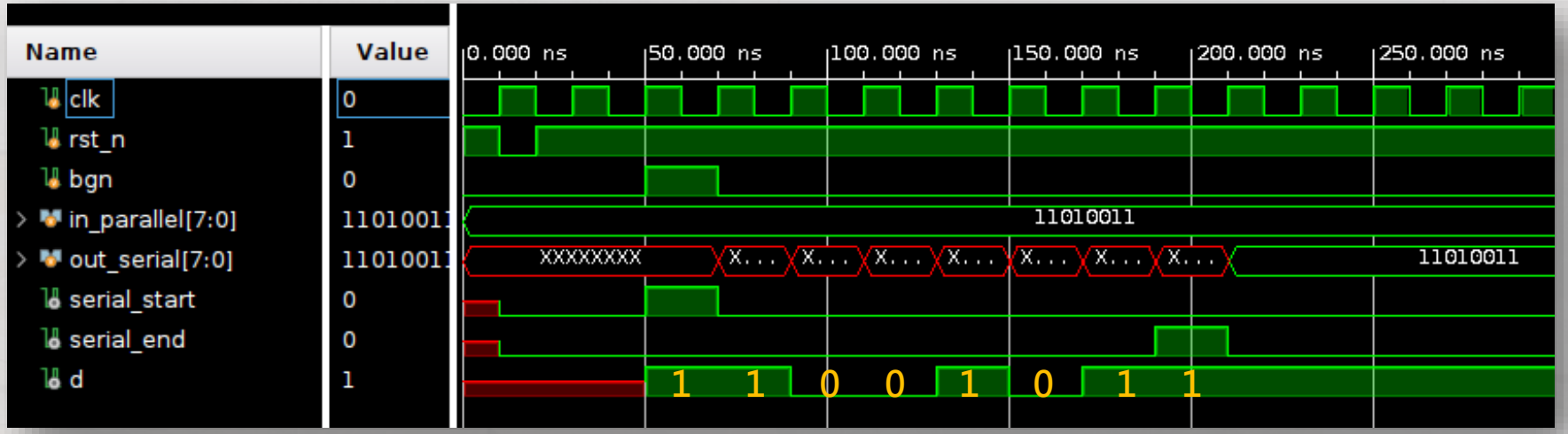
```
module parallel2serial_tb();

reg clk, rst_n, bgn;
reg [7:0] in_parallel = 8'b11010011, out_serial;
wire serial_start, serial_end, d;

parallel2serial P0 (.clk(clk), .rst_n(rst_n), .a(in_parallel), .bgn(bgn),
                    .d(d), .serial_start(serial_start), .serial_end(serial_end));

always #10 clk = ~clk;

initial begin

    clk = 0;
    bgn = 0;
    $display("=== Initialize Success ===");
    rst_n = 1;
    #10 rst_n = 0;
    #10 rst_n = 1;
    $display("=== Reseting ===");

    $display("=== Test Result ===");
    #30 bgn = 1;
    for(integer i = 0; i < 8; i = i + 1)
    begin
        #20 bgn = 0;
        out_serial[i] = d;
        $display("When i = %1d:", i);
        $display("d = %b, serial_start = %b, serial_end = %b", d, serial_start, serial_end);
    end
    #20$display("out_serial = %b", out_serial);

    if(in_parallel == out_serial)
        $display("=== Test Passed ===");
    else
        $display("=== Test Failed ===");

end

endmodule
```

parallel2serial

Coding

Lab 2

■ Simulation result



■ Compare to HLS

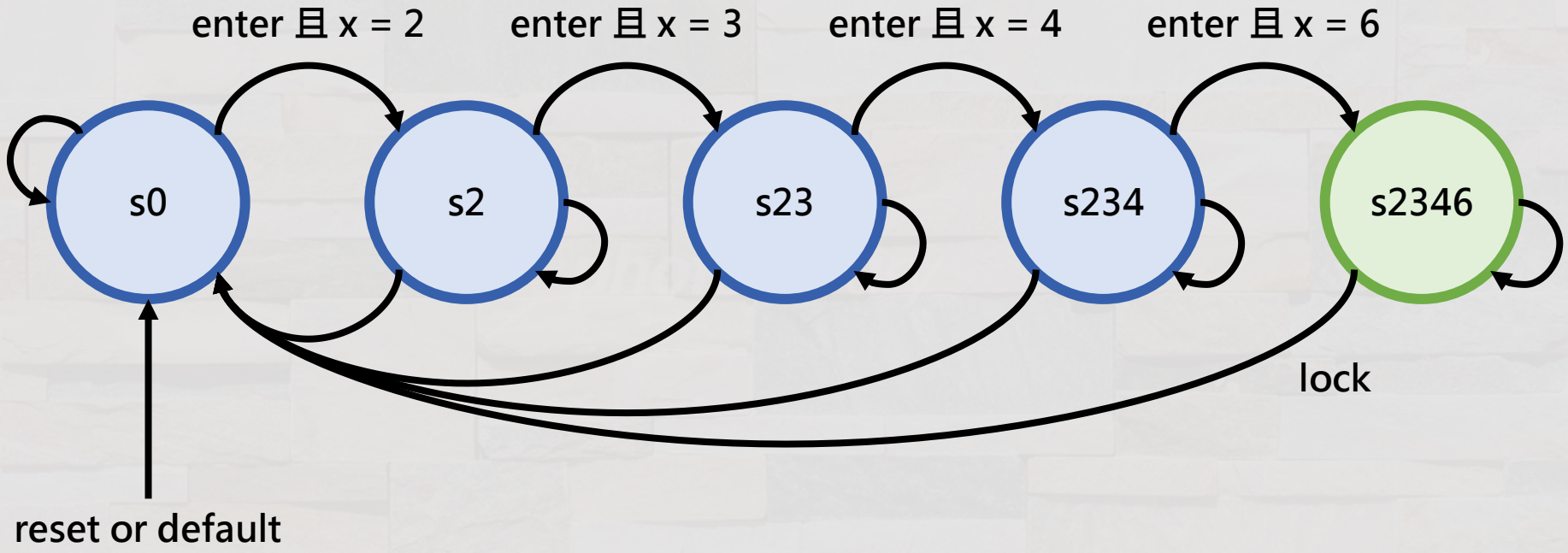
	FF	LUT	Timing
HLS	33	151	5.055 ns
Vivado	7	8	0 ns

combinational_lock

FSM design

Lab 3

- **Moore machine: Door open only in s2346**



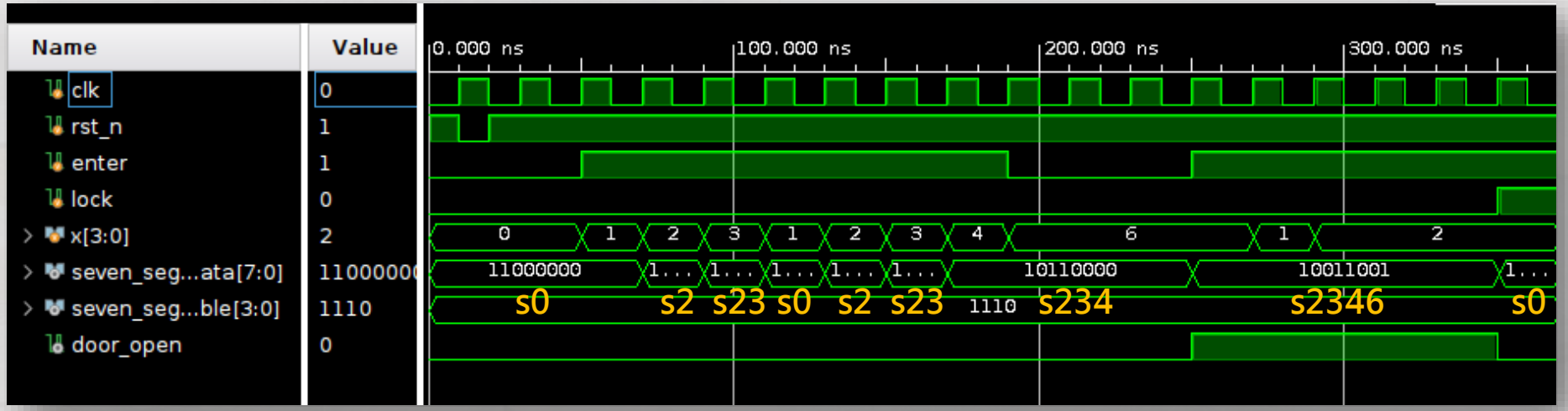
- **Use 7-segment to display states**

combinational_lock

Coding

Lab 3

■ Simulation result



■ Compare to HLS

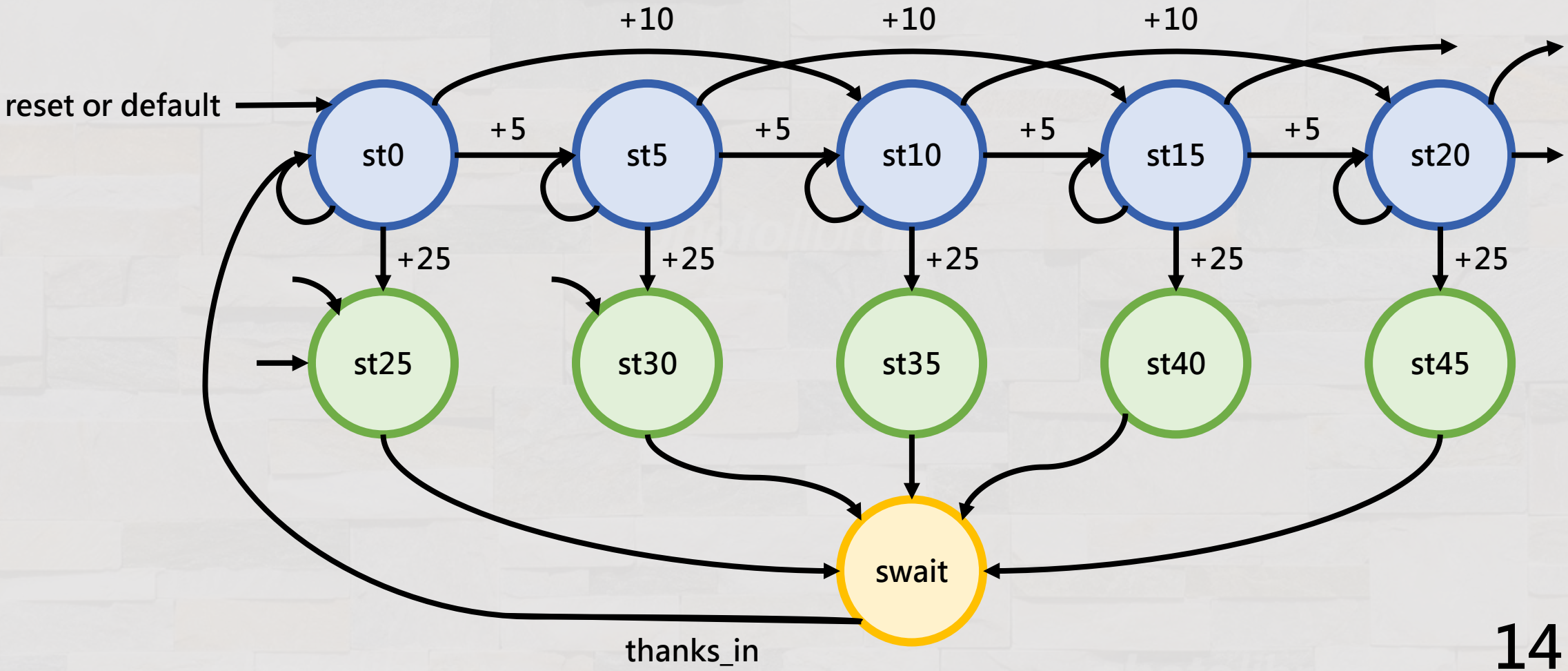
	FF	LUT	Timing
HLS	8	220	6.202 ns
Vivado	4	11	0 ns

vending_machine

FSM design

Lab 4

- **Moore machine: Make change only in st25 - st45**



vending_machine

Coding

Lab 4

- **Testbench**

```
    #30 $display("=== Test Result ===");
    put_coin(0, 0, 0); // no coin
    put_coin(0, 1, 0); // dime
    put_coin(0, 1, 0); // dime
    put_coin(0, 0, 1); // quarter

    $display("=== Test Finished ===");
    $finish;

end

endmodule
```

```
task put_coin;
    input nickel_in_num, dime_in_num, quarter_in_num;

    begin
        #20
        nickel_in = nickel_in_num;
        dime_in = dime_in_num;
        quarter_in = quarter_in_num;

        if(nickel_in)
            $display("Put 1 nickel inside: +5 $");
        else if(dime_in)
            $display("Put 1 dime inside: +10 $");
        else if(quarter_in)
            $display("Put 1 quarter inside: +25 $");
        else
            $display("Don't put any coins");

        $display("candy_out = %b, nickel_out = %b, dime_out = %b", candy_out, nickel_out, dime_out);
        $display("====================");
```

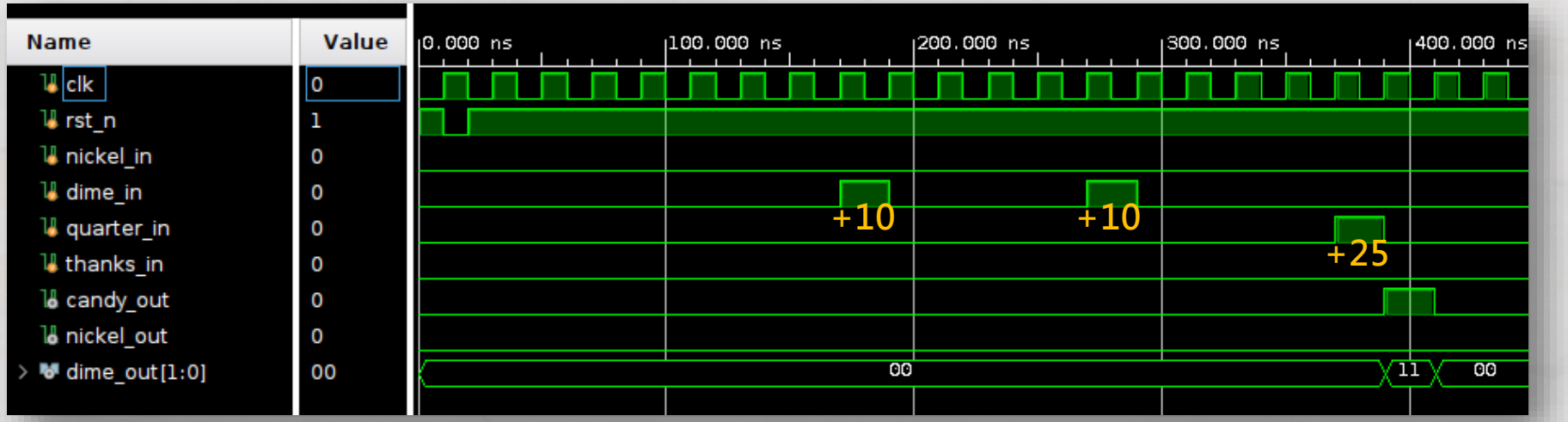
```
        #20
        nickel_in = 0;
        dime_in = 0;
        quarter_in = 0;

        for(integer i = 0; i < 3; i = i + 1)
        begin
            #20
            $display("Waiting...");
            $display("candy_out = %b, nickel_out = %b, dime_out = %b", candy_out, nickel_out, dime_out);
            $display("====================");
        end

    end
endtask
```

vending_machine

Coding

Lab 4

■ Simulation result



■ Compare to HLS

	FF	LUT	Timing
HLS	33	142	3.935 ns
Vivado	15	12	0 ns

Question

① Whether use pipeline design in Lab 3 ?

	FF	LUT	Timing
Pipeline	9	208	6.202 ns
No pipeline	8	220	6.202 ns

② Glitches will appear in HLS design sometimes