Digital System Design Lab

Lab 13

Finite State Machine Applications

Student ID: D1166506

Name: 周嘉禾

Date: 2023/12/20

1. **Objectives**
   * To learn how to build a frequency divider
2. **Theorem**

When designing a frequency division circuit, our goal is to reduce or "divide" the frequency of an input signal. Here's a common method using D-type or T-type flip-flops (also known as inverters or toggle flip-flops) to achieve division:

* 1. Choose Flip-Flop Type: The output of a D-type flip-flop is directly equal to the D input, while a T-type flip-flop toggles its output at each clock cycle.
  2. Connect Flip-Flops: Chain multiple flip-flops together so that the output of one flip-flop is connected to the clock input of the next. In this way, the output frequency of each flip-flop is half that of the previous one.
  3. Set Division Ratio: Adding more flip-flops can further reduce the output frequency. For example, if we connect three flip-flops, the output frequency will be one-eighth of the input frequency (because 23 = 8).
  4. Verify Design: Use circuit simulation software to verify whether your design can produce the expected output frequency.

This design method is suitable for digital signals and can be easily expanded to achieve higher division ratios. However, for analog signals, different techniques may be needed, such as mixers or injection-locked oscillators.

1. **Experimental Results**
   1. **Step 1**
      1. *Think*
2. Build a 1 Hz frequency divider
3. When positive edge of frequency divider, led shift left or right
   * 1. *Code*

module step\_1(clock, SW, led);

input SW, clock;

output reg [0:5] led=6'b011111;

reg Q=1'b0;

integer count=0 ;

always @(posedge Q) begin

case (SW)

0: led <= {led[5], led[0:4]};

1: led <= {led[1:5], led[0]};

endcase

end

always @(posedge clock) begin

if (count==24999999) begin

Q <= !Q;

count <= 0;

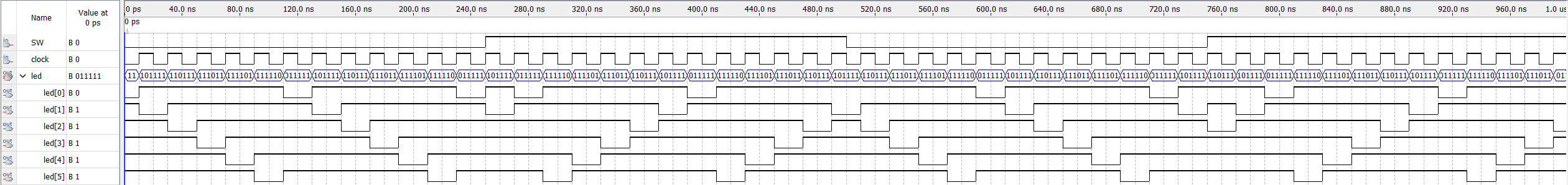
end

else count <= count + 1;

end

endmodule

* + 1. *Simulation*



* 1. **Step 2**
     1. *Think*

1. Build a 1 Hz frequency divider
2. Initialize led0~3 according to SW
3. When positive edge of frequency divider, led0~3 shift left or right
   * 1. *Code*

module step\_2(clock, SW, led3, led2, led1, led0);

input SW, clock;

output reg [0:6] led3, led2, led1, led0;

reg Q=1'b0;

reg last\_SW=0;

integer count=0;

parameter led\_d=7'b1000010, led\_E=7'b0110000, led\_0=7'b0000001, led\_NULL=7'b1111111;

always @(posedge Q) begin

if (last\_SW!=SW) begin

case (SW)

0: begin led3 <= led\_NULL; led2 <= led\_d; led1 <= led\_E; led0 <= led\_0; end

1: begin led3 <= led\_d; led2 <= led\_E; led1 <= led\_0; led0 <= led\_NULL; end

endcase

last\_SW <= SW;

end

else

case (SW)

0: begin led3 <= led2; led2 <= led1; led1 <= led0; led0 <= led3; end

1: begin led3 <= led0; led2 <= led3; led1 <= led2; led0 <= led1; end

endcase

end

always @(posedge clock) begin

if (count==24999999) begin

Q <= !Q;

count <= 0;

end

else count <= count + 1;

end

endmodule

* + 1. *Simulation*

一張含有 文字, 行, 收據, 樣式 的圖片

自動產生的描述

1. **Comments**

* In Lab 2 step 4, we used a 12 MHz oscillator and connected it to a D Flip-Flop, which divided the frequency and could generate a 6 MHz output.

1. **Problems & Solutions**

None

1. **Feedback**

None