Digital System Design Lab

Lab 11

Counters/Shift Registers

Student ID: D1166506

Name: 周嘉禾

Date: 2023/12/13

1. **Objectives**
   * To become familiar with shift register in verilog
   * To learn how to use counter in verilog
2. **Theorem**
   1. **Counters:**

Counters are essential digital circuits used in electronics to count clock pulses or specific events. They're versatile components found in various applications, including digital clocks, frequency dividers, and sequential circuit designs. These circuits can count up, down, or in more complex patterns, facilitating tasks like generating specific sequences or controlling the flow of operations within a system. Counters are integral to the functioning of many digital systems, providing the capability to manage and monitor events or timing within electronic devices.

* 1. **Shift Registers:**

Shift registers are another crucial component in digital electronics used for storing and moving data in a serial manner. They allow the shifting of data bits, enabling serial-to-parallel or parallel-to-serial conversion. Shift registers come in different configurations, such as serial-in-serial-out (SISO), serial-in-parallel-out (SIPO), parallel-in-serial-out (PISO), and parallel-in-parallel-out (PIPO), offering flexibility in data manipulation and transfer. These registers find applications in data storage, data transfer between different parts of a system, and serial data communication.

1. **Experimental Results**
   1. **Step 1**
      1. *Think*
2. Initialize the led
3. If reset, reset led to initial status
4. If clk, let led shift right
   * 1. *Code*

module step\_1(clk, reset, led);

input clk;

input reset;

output reg[7:0] led=8'b10000000;

initial begin

led = 8'b10000000;

end

always @(posedge clk, negedge reset) begin

if (!reset) begin

led <= 8'b10000000; // Reset the shift register

end

else begin

led <= {led[0], led[7:1]}; // Shift right

end

end

endmodule

* + 1. *Simulation*

一張含有 文字, 螢幕擷取畫面, 行, 數字 的圖片

自動產生的描述

* 1. **Step 2**
     1. *Think*
        1. Initialize the led
        2. If reset, reset led to initial status 0
        3. If clk, let count +1/-1 according to dir
        4. Set led value as count
     2. *Code*

module step\_2(dir, clk, reset, led);

input dir;

input clk;

input reset;

output reg[3:0] led=4'b0000;

reg [3:0] count=0;

initial begin

led = 4'b0000;

end

always @(posedge clk, negedge reset) begin

if (!reset) begin

count = 0;

end

else begin

case (dir)

1'b0: count = count - 1;

1'b1: count = count + 1;

endcase

end

led <= count;

end

endmodule

* + 1. *Simulation*

**一張含有 行, 文字, 數字, 字型 的圖片

自動產生的描述**

* 1. **Step 3**
     1. *Think*
        1. If reset, reset led to initial status 0/1 according to EO
        2. If clk, let count +2
        3. If EO isn’t same as last state, reset the count
        4. Set led value as converted led according to count
     2. *Code*

module step\_3(EO, clk, reset, led);

input EO, clk;

input reset;

output reg[0:6] led;

reg [3:0] count;

initial begin

case (EO)

1'b0: count = 0; // initialize to 0

1'b1: count = 1; // initialize to 1

endcase

end

always @(posedge clk or negedge reset) begin

if (!reset) begin

case (EO)

1'b0: count = 0; // reset to 0

1'b1: count = 1; // reset to 1

endcase

end

else if (EO && !(count&1)) count <= 1;

else if (!EO && count&1) count <= 0;

else count <= (count + 2) % 10;

end

always @\* begin

led = converted\_led(count);

end

function [0:6] converted\_led;

input [3:0] number;

begin

case (number)

0: converted\_led = 7'b0000001; // 0

1: converted\_led = 7'b1001111; // 1

2: converted\_led = 7'b0010010; // 2

3: converted\_led = 7'b0000110; // 3

4: converted\_led = 7'b1001100; // 4

5: converted\_led = 7'b0100100; // 5

6: converted\_led = 7'b0100000; // 6

7: converted\_led = 7'b0001101; // 7

8: converted\_led = 7'b0000000; // 8

9: converted\_led = 7'b0000100; // 9

default: converted\_led = 7'b1111111; // default

endcase

end

endfunction

endmodule

* + 1. *Simulation*

*一張含有 文字, 行, 螢幕擷取畫面, 數字 的圖片

自動產生的描述*

1. **Comments**

None

1. **Problems & Solutions**

None

1. **Feedback**

None