**電路實驗報告**

**資工二甲**

**11127137 黃乙家**

1. Verilog Code

count\_hold\_add.v

`timescale 1ns/1ns

module count\_hold\_add(a, b, clk, sel, rst, dout);

input [2:0] a, b;

input clk, rst;

input [1:0] sel;

output wire [3:0] dout;

wire [3:0] cnt\_out;

wire [2:0] adder\_sum;

wire adder\_cout;

wire [3:0] adder\_out;

Counter U\_cnt(.dout(dout), .clk(clk), .rst(rst), .cnt\_out(cnt\_out) );

RCA3 U\_adder(.a(a), .b(b), .s(adder\_sum), .cout(adder\_cout) );

assign adder\_out = {adder\_cout, adder\_sum};

assign dout = (sel == 2'b00) ? adder\_out

: (sel == 2'b10) ? cnt\_out

: (sel == 2'b01) ? dout

: 4'bxxxx;

endmodule

RCA3.v

module RCA3(a, b, cout, s);

input [2:0] a, b;

output [2:0] s;

output cout;

wire [3:0] at, bt, st;

assign at = {1'b0, a};

assign bt = {1'b0, b};

assign st = at + bt;

assign s = st[2:0];

assign cout = st[3];

endmodule

Counter.v

`timescale 1ns/1ns

module Counter( dout, clk, rst, cnt\_out);

input [3:0] dout;

input wire clk, rst;

output reg [3:0] cnt\_out;

always @(posedge clk)

begin

if(rst == 1'b0)

cnt\_out <= 4'b0000;

else

cnt\_out <= dout + 1;

end

endmodule

TM.v

`timescale 1ns/1ns

module TM;

reg [2:0] a, b;

reg clk, rst;

wire [3:0] dout;

reg [1:0] sel;

parameter t = 200;

parameter th = 100;

count\_hold\_add U\_m( .a(a), .b(b), .dout(dout), .clk(clk), .sel(sel), .rst(rst) );

always #th clk = ~clk;

initial begin

rst = 1;

clk = 0;

sel = 2'b00;

a = 3'b000;

b = 3'b000;

#t rst = 0;

#t rst = 1;

sel = 2'b10;

// Counter

#(8\*t) sel = 2'b01;

// Hold

#(2\*t) sel = 2'b00;

// Adder

a = 3'b011;

b = 3'b100;

#t

a = 3'b001;

b = 3'b101;

#t $stop;

end

endmodule

1. 結果

**完整模擬截圖：一張含有 文字, 螢幕擷取畫面, 軟體, 多媒體軟體 的圖片

自動產生的描述**

**一張含有 文字, 螢幕擷取畫面, 陳列, 圖表 的圖片

自動產生的描述第一階段(計數器)**

初始化所有變數後，自第三次正緣觸發後 dout 開始 +1，每次正緣觸發就加一次，直到第 8 次加到 1000 為止進入下一個模式

**一張含有 螢幕擷取畫面, 行, 文字 的圖片

自動產生的描述第二階段(Hold)**

sel(最下行訊號) 切成 0b01 後，兩次正緣觸發不改變 dout 的值，之後進入下一階段

**一張含有 螢幕擷取畫面, 文字, 行 的圖片

自動產生的描述第三階段(Adder)**

sel(最下行訊號) 切成 0b00 後，dout 輸出 a+b 的結果，a 與 b 為 3-bit，而 dout 是 4-bit，當 a = 0b011,b = 0b100，a + b = 0b111, carry = 0，進行位元擴充至 4-bit 變成 0b0111。當 a = 0b001, b = 0b101，a + b = 0b110, carry = 0，進行位元擴充至 4-bit 變成 0b0110，模擬結果合理。