

Digital Logic Design Experiments Experiment Report

WIREFRAME FORMATION

電機一甲 110310138 劉千榮



**TAIPEI
TECH**
Since 1912

國立臺北科技大學
National Taipei University of Technology

數位邏輯設計實習報告

第 7 週 : 四位元加法器與加減法器

組員

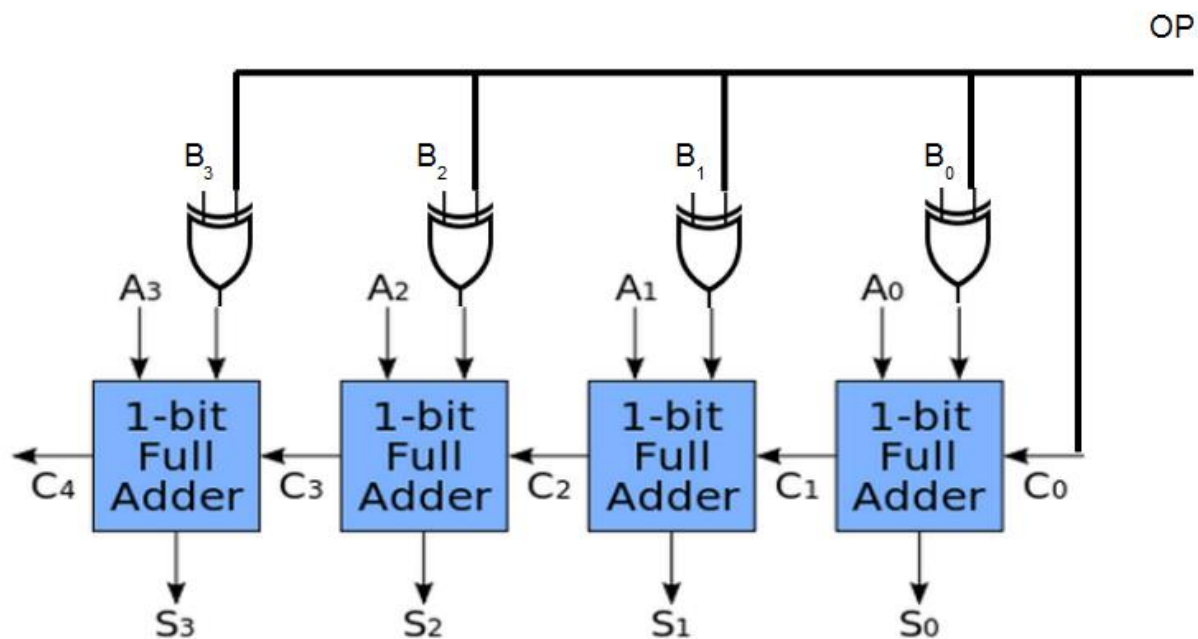
110310138 劉千榮

110310147 王瑞鴻

壹、基本題

模擬驗證四位元加減法器

請以以上之實驗模擬結果及詳細討論說明為例,列舉出四位元加減法器之加減法,具代表性例子至少6組(加減法各至少3組以上之 Group Values 及 Count Values),模擬波形輸出結果,並詳細說明其結果為何符合四位元加減法器之邏輯,愈詳細愈好! (請解釋減法器之正負號,1 or 2的補數)

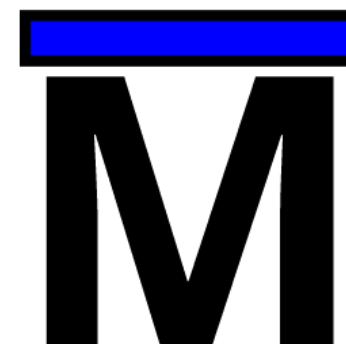


實驗原理

使用Quartus prime lite編輯邏輯閘電路，燒入DE-10 stander板子，則可確認電路功能。只要將上次加分題的作業稍作更改，便能完成電路之布置。在完成實驗後，撰寫 TestBench，利用內建 ModelSim 驗證波型輸出，再將其燒錄製至開發板，執行功能。



Quartus
Prime



設計程序

先確認電路功能，畫出真值表並且轉換成布林代數，則可以使用邏輯閘組合成電路。

模擬驗證

先確認電路功能，畫出真值表並且轉換成布林代數，則可以使用邏輯閘組合成電路。

設計程序 – 1/3 (Verilog HDL)

```
// Top-Level Entity
```

```
module hw6(sub, a, b, s, cout);
```

```
input sub;
```

```
input [3:0] a;
```

```
input [3:0] b;
```

```
output cout;
```

```
output [3:0] s;
```

```
wire [3:0] bT04bits;
```

```
assign bT04bits = b ^ {4{sub}};
```

```
adder4bit adder4_bit(a, bT04bits, sub, s, cout);
```

```
endmodule
```

設計程序 – 2/3 (Verilog HDL)

```
// 4bits a/'s module
```

```
module adder4bit(a, b, cin, s, cout);
```

```
input cin;
```

```
input [3:0] a;
```

```
input [3:0] b;
```

```
output [3:0] s;
```

```
output cout;
```

```
wire f1T0f2, f2T0f3, f3T0f4;
```

```
fulladder fa1_LSB (a[0], b[0], cin, s[0], f1T0f2);
```

```
fulladder fa2      (a[1], b[1], f1T0f2, s[1], f2T0f3);
```

```
fulladder fa3      (a[2], b[2], f2T0f3, s[2], f3T0f4);
```

```
fulladder fa4_MSB  (a[3], b[3], f3T0f4, s[3], cout);
```

```
endmodule
```

設計程序 - 3/3 (Verilog HDL)

```
//fulladder module
```

```
module fulladder(a, b, cin, sum, cout);
```

```
// Input Port(s)
```

```
input a, b, cin;
```

```
// Output Port(s)
```

```
output sum, cout;
```

```
assign {cout, sum} = a + b + cin;
```

```
endmodule
```


TestBench – 1/3 (Verilog HDL)

```
`timescale 1ns/1ps
```

```
module hw6_tb;
```

```
reg cin;
```

```
reg [3:0] a;
```

```
reg [3:0] b;
```

```
wire cout;
```

```
wire [3:0] s;
```

```
hw6 addsub4Bits(cin, a, b, s, cout);
```

```
integer i;
```

TestBench – 2/3 (Verilog HDL)

```
initial  
begin
```

```
    $display("Testing cin = 0, the adder mode.");
```

```
    cin = 1'b0;
```

```
    for(i = 0; i < 256; i = i + 1)  
    begin
```

```
        {b, a} = i;
```

```
        #10;
```

```
        $display($time, "\t %b + %b = %b", a, b, s);
```

```
    end
```

TestBench – 3/3 (Verilog HDL)

```
$display("Testing cin = 1, the subtractor mode.");
```

```
cin = 1' b1;
```

```
for(i = 0; i < 256; i = i + 1)  
begin
```

```
    {b, a} = i;  
    #10;
```

```
    $display($time, "\t %b - %b = %b", a, b, s);
```















```
end
```

```
end
```

```
endmodule
```

接腳設定簡介

本次實驗我們把輸入與輸出端接角以下列表格定義。

Node Name	Direction	Location	I/O Bank	VREF Group	I/O Standard
 a[3]	Input	PIN_AC30	5B	B5B_N0	2.5 V (default)
 a[2]	Input	PIN_AB28	5B	B5B_N0	2.5 V (default)
 a[1]	Input	PIN_Y27	5B	B5B_N0	2.5 V (default)
 a[0]	Input	PIN_AB30	5B	B5B_N0	2.5 V (default)
 b[3]	Input	PIN_AD30	5B	B5B_N0	2.5 V (default)
 b[2]	Input	PIN_AC28	5B	B5B_N0	2.5 V (default)
 b[1]	Input	PIN_V25	5B	B5B_N0	2.5 V (default)
 b[0]	Input	PIN_W25	5B	B5B_N0	2.5 V (default)
 cout	Output	PIN_AG25	4A	B4A_N0	2.5 V (default)
 s[3]	Output	PIN_AD24	4A	B4A_N0	2.5 V (default)
 s[2]	Output	PIN_AC23	4A	B4A_N0	2.5 V (default)
 s[1]	Output	PIN_AB23	5A	B5A_N0	2.5 V (default)
 s[0]	Output	PIN_AA24	5A	B5A_N0	2.5 V (default)
 sub	Input	PIN_AC29	5B	B5B_N0	2.5 V (default)

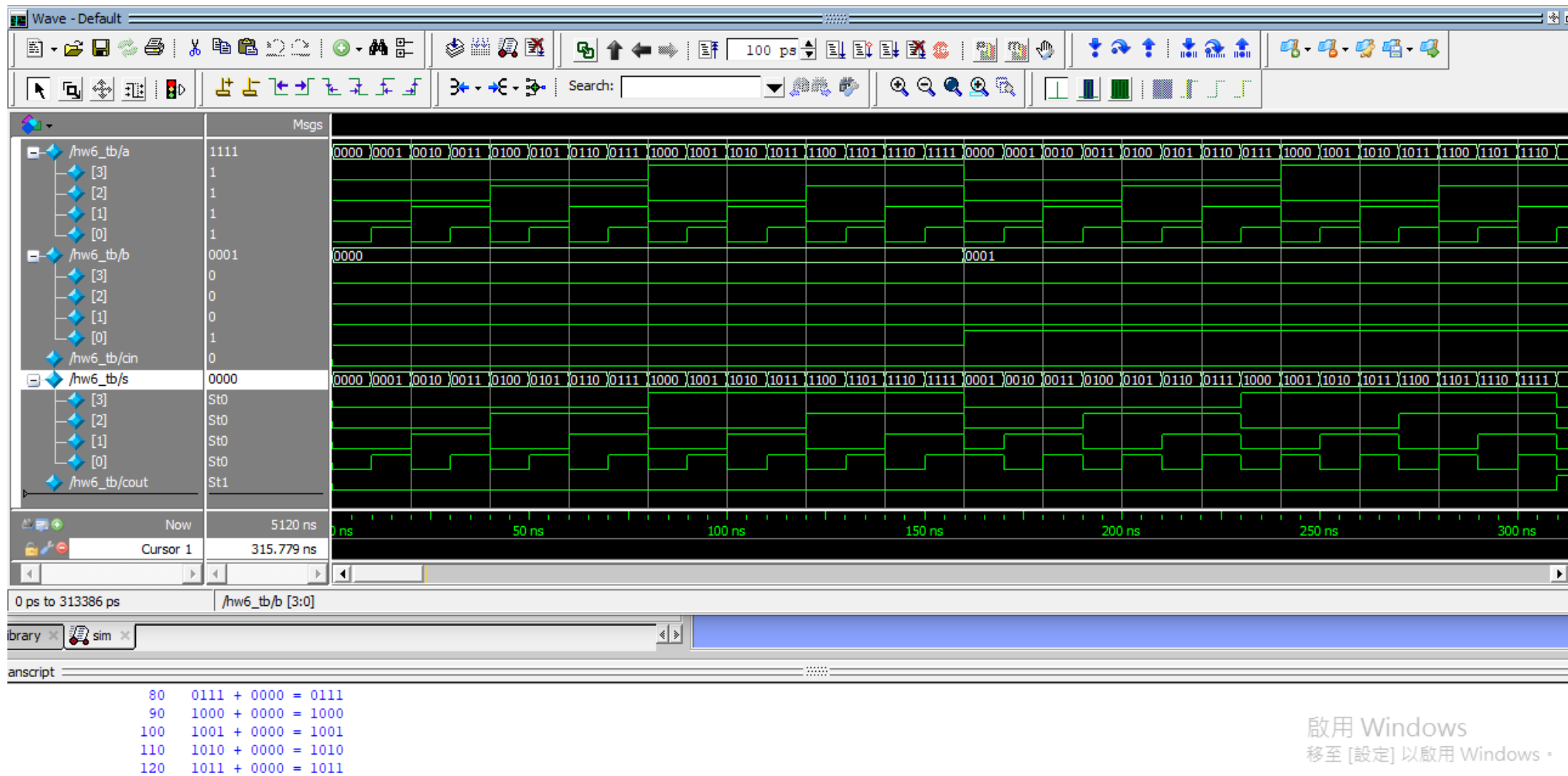
Pin Assignment of Slide Switches

<i>Signal Name</i>	<i>FPGA Pin No.</i>	<i>Description</i>	<i>I/O Standard</i>
SW[0]	PIN_AB30	Slide Switch[0]	Depend on JP3
SW[1]	PIN_Y27	Slide Switch[1]	Depend on JP3
SW[2]	PIN_AB28	Slide Switch[2]	Depend on JP3
SW[3]	PIN_AC30	Slide Switch[3]	Depend on JP3
SW[4]	PIN_W25	Slide Switch[4]	Depend on JP3
SW[5]	PIN_V25	Slide Switch[5]	Depend on JP3
SW[6]	PIN_AC28	Slide Switch[6]	Depend on JP3
SW[7]	PIN_AD30	Slide Switch[7]	Depend on JP3
SW[8]	PIN_AC29	Slide Switch[8]	Depend on JP3
SW[9]	PIN_AA30	Slide Switch[9]	Depend on JP3

Pin Assignment of LEDs

<i>Signal Name</i>	<i>FPGA Pin No.</i>	<i>Description</i>	<i>I/O Standard</i>
LEDR[0]	PIN_AA24	LED [0]	3.3V
LEDR[1]	PIN_AB23	LED [1]	3.3V
LEDR[2]	PIN_AC23	LED [2]	3.3V
LEDR[3]	PIN_AD24	LED [3]	3.3V
LEDR[4]	PIN_AG25	LED [4]	3.3V
LEDR[5]	PIN_AF25	LED [5]	3.3V
LEDR[6]	PIN_AE24	LED [6]	3.3V
LEDR[7]	PIN_AF24	LED [7]	3.3V
LEDR[8]	PIN_AB22	LED [8]	3.3V
LEDR[9]	PIN_AC22	LED [9]	3.3V

輸出模擬 (ModelSim)



啟用 Windows
移至 [設定] 以啟用 Windows。

問題與心得

110310138 劉千榮

考慮到需依照需求進行加法或減法，且題目重點為「驗證」四位元加減法器電路，故須再撰寫TestBench驗證電路，以程式的方式指定輸入資料，測試一共 $2^9 = 512$ 種所有狀態。

當我們在加減法選擇線輸入「1」時，XOR閘的一端將輸入「1」，輸出端則為另一輸入端的反向，代表通過XOR閘時，就進行了1的補數轉換，最後cin端會再加一，即完成2的補數轉換。

使用2的補數進行運算時，最高位元(MSB)的數字代表正負，所以 n-bits 只能表示出 $\pm 2^{n-1}$ 大小的數。因此運算數值中，若總和超過 $(2^{n-1} - 1)$ 或低於 -2^{n-1} 時，將會產生溢位，最終產生錯誤結果。

貳、進階題

模擬驗證四位元正數大小判別器

在進行大小判別時，都是將兩數進行相減，大於時、結果會大於0，等於時、結果會等於0，小於時、結果會小於0（也就是會借位），所以請依照下圖所示，設計一個「四位元大小判別器」。（ $>$ $=$ $<$ 符號請自行用七段顯示）

實驗原理

使用Quartus prime lite編輯邏輯閘電路，燒入DE-10 stander板子，則可確認電路功能。利用減法器，觀察最高位元之數值。在完成實驗後，驗證波型輸出，再將其燒錄製至開發板，執行功能驗證。



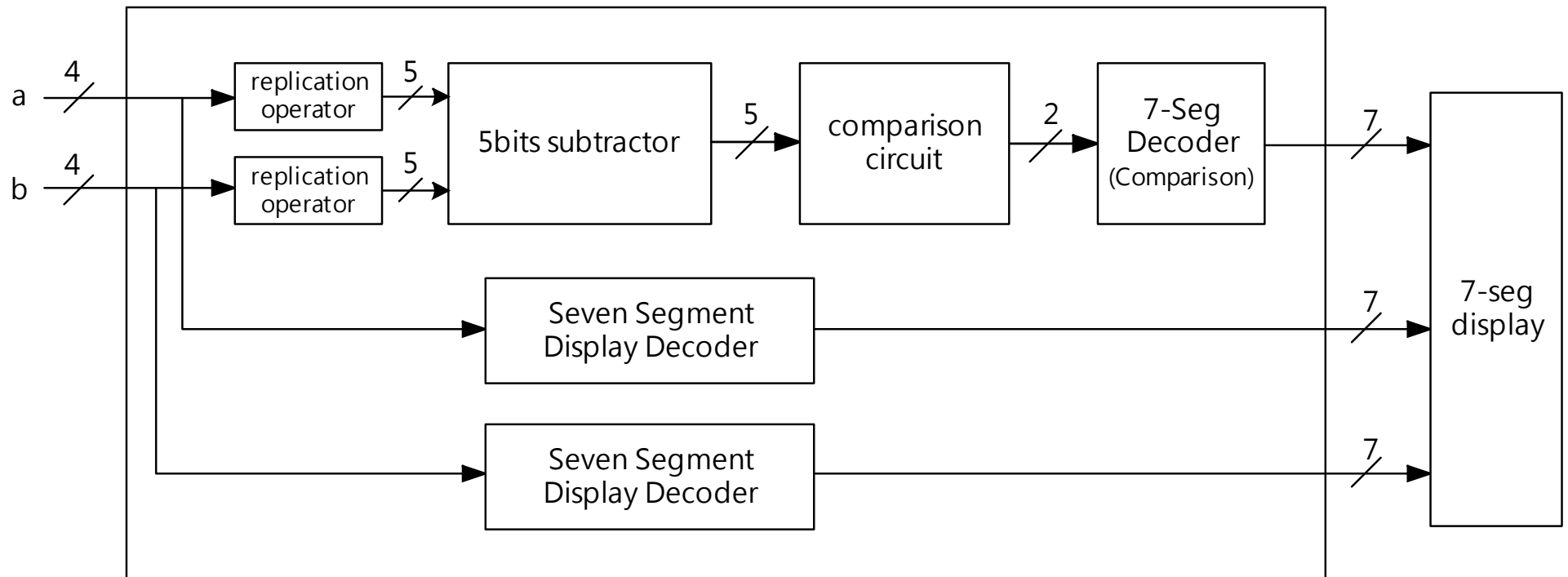
設計程序

先確認電路功能，畫出真值表並且轉換成布林代數，則可以使用邏輯閘組合成電路。

模擬驗證

先確認電路功能，畫出真值表並且轉換成布林代數，則可以使用邏輯閘組合成電路。

Circuit Architecture



設計程序 – 1/5 (Verilog HDL)

```
module hw6_extra(a, b, seg7_a, seg7_b, seg7_compare);

input [3:0] a;
input [3:0] b;

output [6:0] seg7_a;
output [6:0] seg7_b;
output [6:0] seg7_compare;

wire [4:0] s;
wire [4:0] ex_a;
wire [4:0] ex_b;
wire [1:0] compare_sel;

assign ex_a = {1'b0, a};
assign ex_b = {{1'b0, b} ^ {5{1'b1}}};

seg7 seg7_1(a, seg7_a);
seg7 seg7_2(b, seg7_b);
seg7_com compare(compare_sel, seg7_compare);

adder5bit adder5_bit(ex_a, ex_b, 1, s);
compare_mux (s, compare_sel);

endmodule
```

設計程序 – 2/5 (Verilog HDL)

```
// 5bits subtractor module
module adder5bit(a, b, cin, s, cout);

input cin;
input [4:0] a;
input [4:0] b;

output [4:0] s;
output cout;

wire f1T0f2, f2T0f3, f3T0f4, f4T0f5;

fulladder fa1_LSB (a[0], b[0], cin, s[0], f1T0f2);
fulladder fa2 (a[1], b[1], f1T0f2, s[1], f2T0f3);
fulladder fa3 (a[2], b[2], f2T0f3, s[2], f3T0f4);
fulladder fa4 (a[3], b[3], f3T0f4, s[3], f4T0f5);
fulladder fa5_MSB (a[4], b[4], f4T0f5, s[4], cout);

endmodule
```

設計程序 – 3/5 (Verilog HDL)

```
// Seven Segment Display Decoder
module seg7 (input [3:0] in, output reg [6:0] Out);
always@(in) begin
    case(in)
        4'b0000: Out <= 7'b000_0001; // 0
        4'b0001: Out <= 7'b100_1111; // 1
        4'b0010: Out <= 7'b001_0010; // 2
        4'b0011: Out <= 7'b000_0110; // 3
        4'b0100: Out <= 7'b100_1100; // 4
        4'b0101: Out <= 7'b010_0100; // 5
        4'b0110: Out <= 7'b110_0000; // 6
        4'b0111: Out <= 7'b000_1111; // 7
        4'b1000: Out <= 7'b000_0000; // 8
        4'b1001: Out <= 7'b000_1100; // 9
        4'b1010: Out <= 7'b000_1000; // A
        4'b1011: Out <= 7'b110_0000; // b
        4'b1100: Out <= 7'b011_0001; // C
        4'b1101: Out <= 7'b100_0010; // d
        4'b1110: Out <= 7'b011_0000; // E
        4'b1111: Out <= 7'b011_1000; // F
    endcase
end
endmodule
```

設計程序 – 4/5 (Verilog HDL)

```
// comparison circuit module
module compare_mux (input [4:0] s,
                    output reg [1:0] compare_sel);

always@(s)

begin
    if(s == 5'b00000) // 等於
        compare_sel <= 2'b10;
    else if(s[4] == 1) // 小於
        compare_sel <= 2'b01;
    else // 大於
        compare_sel <= 2'b00;

end

endmodule
```

設計程序 – 5/5 (Verilog HDL)

```
// Seven Segment Display Decoder for >, =, <
module seg7_com (input [1:0] in, output reg [6:0] Out);

always@(in)
begin // synopsys full_case

    case(in)
        2'b00: Out <= 7'b111_1000; // 大於
        2'b01: Out <= 7'b100_1110; // 小於
        2'b10: Out <= 7'b100_1000; // 等於
        default: Out <= 7'b000_0000;
    endcase

end
endmodule
```


改良方法

若以行為模型直接敘述電路功能，就能直接將減法器與解碼器的功能合併。如下：

```
module compare_dec (input [3:0] a, input [3:0] b, output reg [1:0]  
compare_sel);
```

```
always @(a or b)
```

```
  begin
```

```
    if(a == b)
```

```
      compare_sel <= 2'b10;
```

```
    else if(a < b)
```

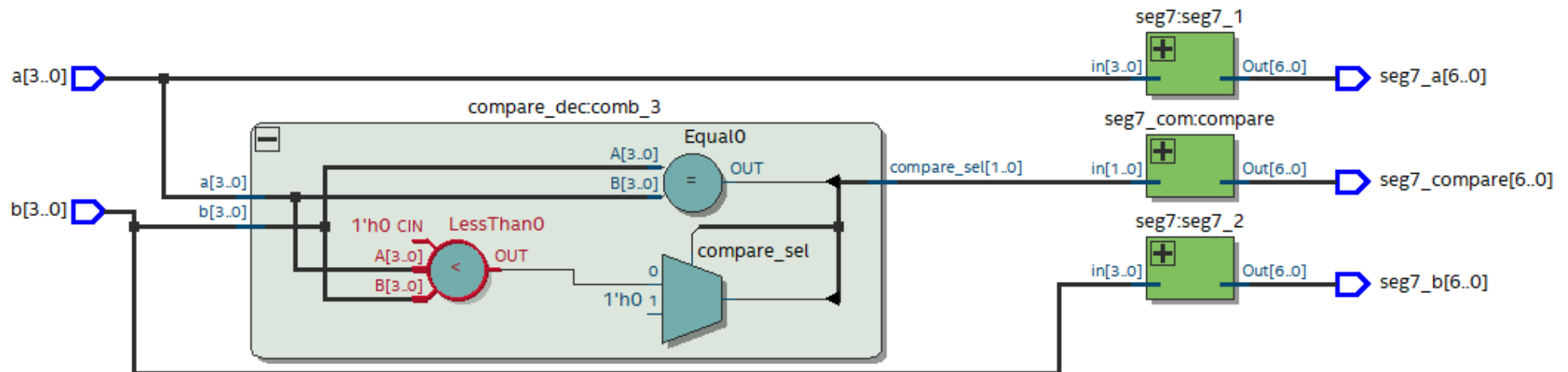
```
      compare_sel <= 2'b01;
```

```
    else
```





```
      compare_sel <= 2'b00;
```

```
end endmodule
```

RTL Module



模擬 Simulation

	Name	Value at 0 ps	0 ps	10.0 ns	20.0 ns	30.0 ns	40.0 ns	50.0 ns	60.0 ns	70.0 ns	80.0 ns	90.0 ns
			0 ps									
	a	B 0110	6 0110	5 0101	10 1010	8 1000	12 1100	14 1110	3 0011	10 1010	13 1101	
	b	B 0001	1 0001	4 0100	13 1101	5 0101	13 1101	7 0111	8 1000	13 1101	14 1110	
	seg7_a	B 1100000	1100000	0100100	0001000	0000000	0110001	0110000	0000110	0001000	1000010	
	seg7_b	B 1001111	1001111	1001100	1000010	0100100	1000010	0001111	0000000	1000010	0110000	
	seg7_co...	B 1111000	> 1111000	>	< 1001110	> 1111000	< 1001110	> 1111000	<	< 1001110	<	
			<p>大於 [1 111000 小於 [1001110 等於 [1001000</p>									

問題與心得

110310138 劉千榮

這題比較困難的地方，大概只有用減法器的特性製作比較器。實作上我把輸入的4位元先拼接成最高位元為0另四位元為原輸入，再將兩數相減，最後判斷最高位元之數值，即可完成該電路的實作，如電路架構所示。沒去學校導致沒有實體電路展示，蠻可惜的。

WIREFRAME FORMATION

工業推手一世紀 · 企業搖籃一百年

100 Years of Excellence · Cultivating Entrepreneurs of Tomorrow



國立臺北科技大學
National Taipei University of Technology