

邏輯系統實驗

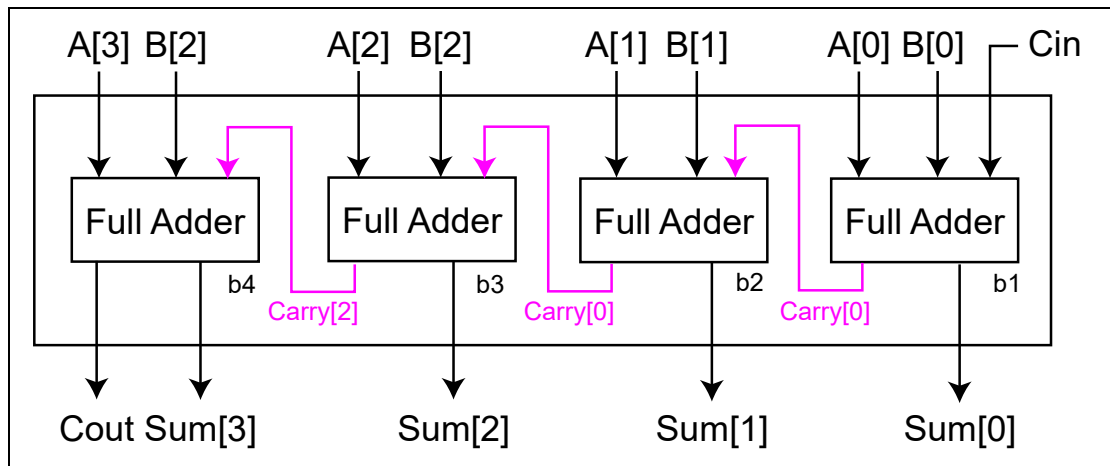
Lab 5

2021/04/08(四)

第 1 組	
組員姓名	學號
陳旭祺	E24099059
張振杰	E24085034
何啟造	E34085337

● 實作題(一): 4 bit Adder

1. Schematic View



2. Verilog Code

fulladder
<pre> module fulladder(A, B, C, Carry, Sum); input A, B, C; output Carry, Sum; assign Carry = (((A ^ B) & C) (A & B)); assign Sum = A ^ B ^ C; endmodule </pre>
adder4bits
<pre> module adder4bits(A, B, Cin, Cout, Sum); input [3:0] A, B; input Cin; output [3:0] Sum; output Cout; wire [3:0] Carry; fulladder b1(A[0], B[0], Cin , Carry[0], Sum[0]); fulladder b2(A[1], B[1], Carry[0], Carry[1], Sum[1]); fulladder b3(A[2], B[2], Carry[1], Carry[2], Sum[2]); fulladder b4(A[3], B[3], Carry[2], Cout , Sum[3]); endmodule </pre>

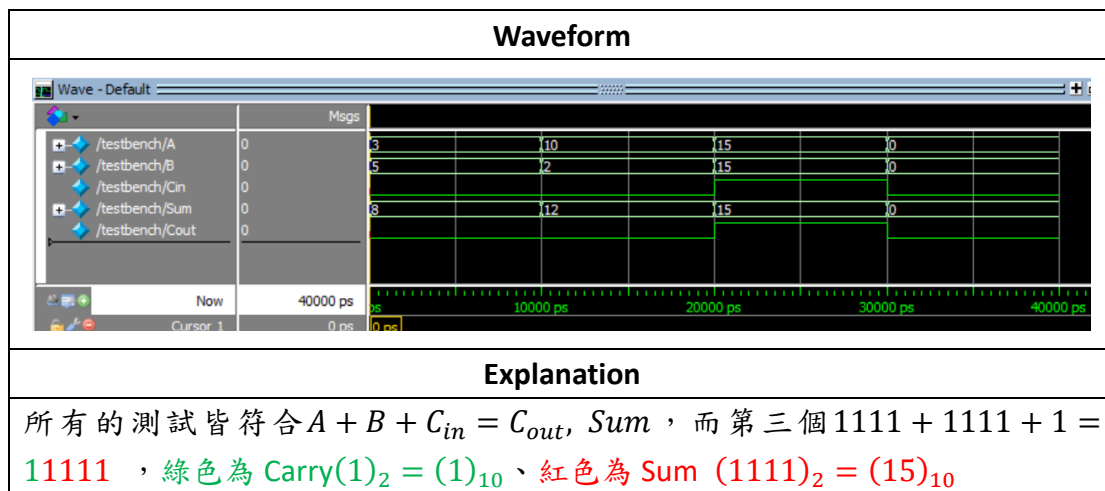
3. Testbench for Verilog Code

```
`timescale 1ns/10ps
module testbench();
    reg [3:0] A, B;
    reg      Cin;
    wire [3:0] Sum;
    wire      Cout;

    adder4bits
n0(.A(A), .B(B), .Cin(Cin), .Sum(Sum), .Cout(Cout));

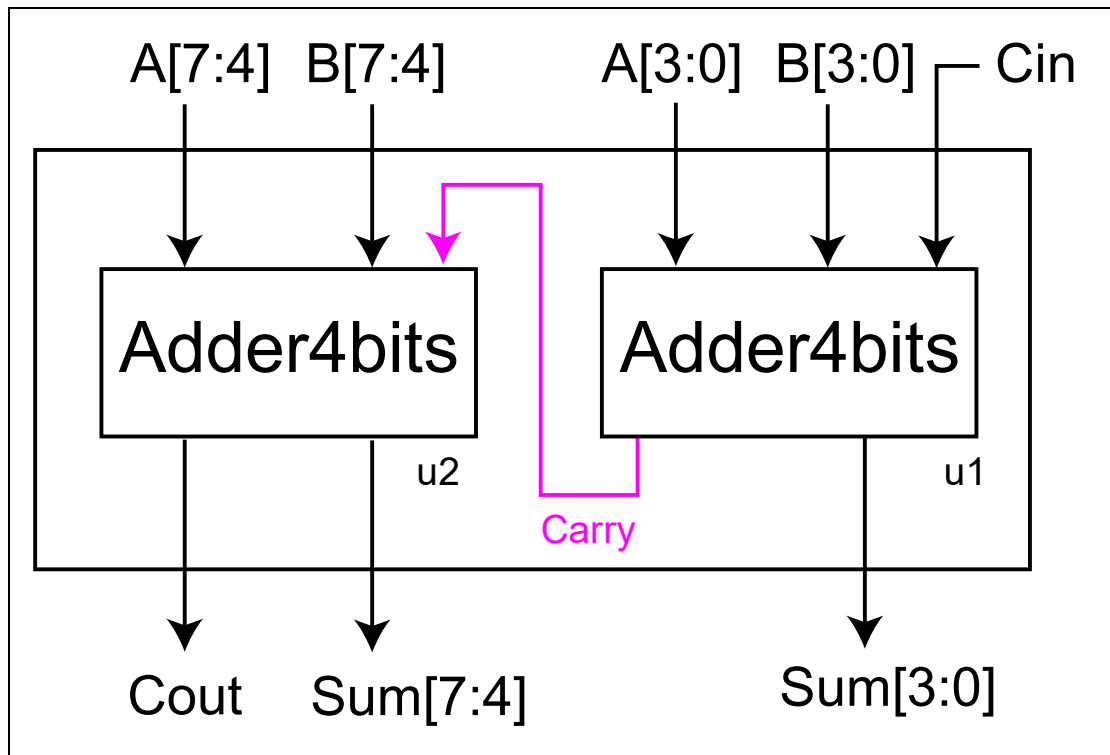
    initial begin
        A = 03; B = 05; Cin = 0;
        #10 A = 10; B = 02; Cin = 0;
        #10 A = 15; B = 15; Cin = 1;
        #10 A = 00; B = 00; Cin = 0;
        #10 $finish;
    end
endmodule
```

4. Simulation Result



● 實作題(二): 4-bit Ripple Carry Adder

1. Schematic View



2. Verilog Code

```
`include "fulladder.v" "adder4bits.v"

adder8bits

module adder8bits(A,B,Cin,Cout,Sum);
    input  [7:0] A,B;
    input      Cin;
    output [7:0] Sum;
    output      Cout;
    wire        Carry;

    adder4bits
    u1(.A(A[3:0]), .B(B[3:0]), .Cin(Cin) , .Cout(Carry), .Sum(Sum[3:0]));

    adder4bits
    u2(.A(A[7:4]), .B(B[7:4]), .Cin(Carry), .Cout(Cout) , .Sum(Sum[7:4]));
endmodule
```

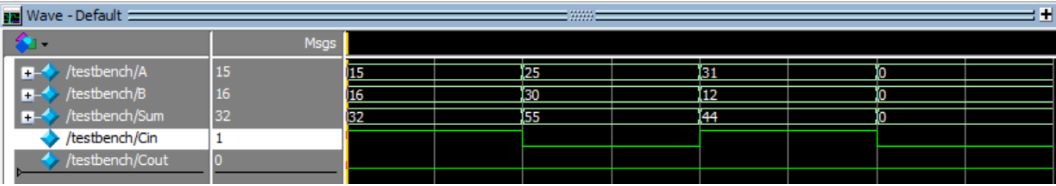
3. Testbench for Verilog Code

```
`timescale 1ns/10ps
module testbench();
    reg [7:0]A,B;
    reg      Cin;
    wire [7:0]Sum;
    wire      Cout;

    adder8bits
n1(.A(A), .B(B), .Cin(Cin), .Sum(Sum), .Cout(Cout));

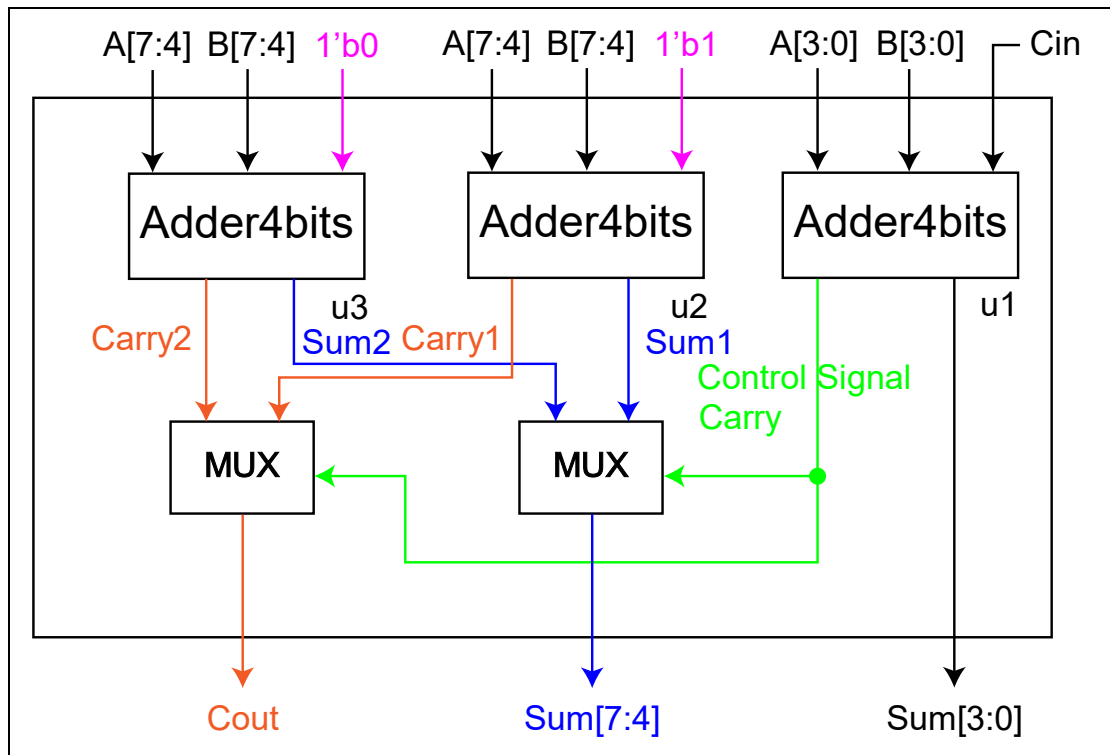
    initial begin
        A=15; B=16; Cin=1;
        #10 A=25; B=30; Cin=0;
        #10 A=31; B=12; Cin=1;
        #10 A= 0; B= 0; Cin=0;
        #10 $finish;
    end
endmodule
```

4. Simulation Result

Waveform							
							
Explanation							
<p>利用 4-bit adder 組合出 8-bit ripple carry adder 加法器，與前者一樣為所有的測試皆符合 $A + B + C_{in} = C_{out}$。Sum 有 8bit，故最大值為 $2^8 - 1 = 255$，所以 C_{out} 皆為 0</p>							

● 實作題(三): 8-bit Carry Select Adder

1. Schematic View



2. Verilog Code

```
`include "fulladder.v" "adder4bits.v"

adder8bits_v2

module adder8bits_v2(A,B,Cin,Cout,Sum);
    input  [7:0] A,B;
    input      Cin;
    output [7:0] Sum;
    output      Cout;
    wire      Carry;
    wire  [3:0] Sum1;
    wire  [3:0] Sum2;
    wire      Cout1;
    wire      Cout2;

    adder4bits
    u1(.A(A[3:0]), .B(B[3:0]), .Cin(Cin) , .Cout(Carry), .Sum(Sum
    [3:0]));
    adder4bits
```

```

u2(.A(A[7:4]), .B(B[7:4]), .Cin(1'b1), .Cout(Cout1) , .Sum(Su
m1));
    adder4bits
u3(.A(A[7:4]), .B(B[7:4]), .Cin(1'b0), .Cout(Cout2) , .Sum(Su
m2));

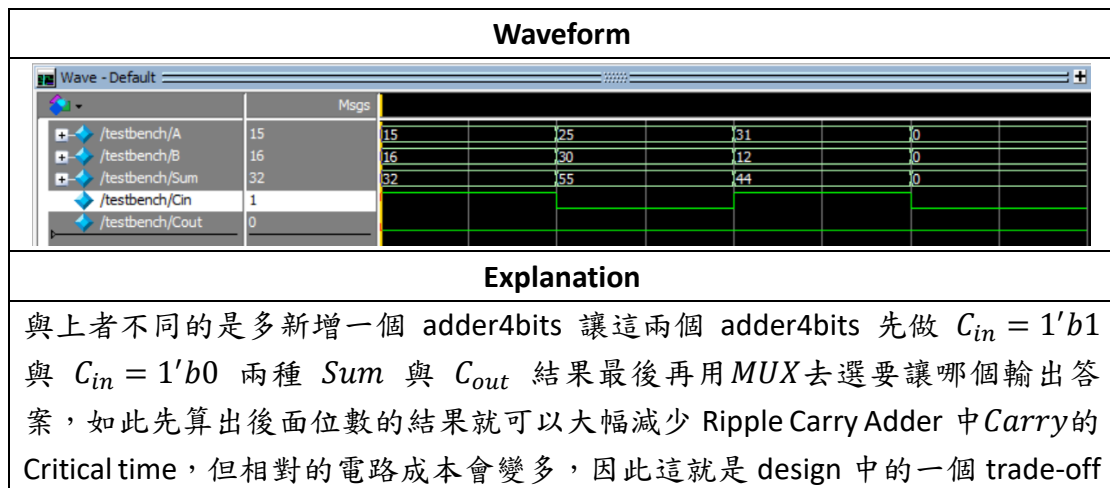
    assign Cout      = (Carry == 1'b1)?  Cout1:Cout2;
    assign Sum[7:4]  = (Carry == 1'b1)?  Sum1 :Sum2;
endmodule

```

3. Testbench for Verilog Code

As some as Problem2

4. Simulation Result



● 心得

1. 組員一 陳旭祺

延續上次實驗這次一樣打 Verilog，由於第一與第二題我與組員分開打，但兩者其實是 hierarchical coding 的從屬關係，而我又未注意到腳位的順序而導致看 Waveform 驗證時 Sum 值為高阻抗(z)即未有輸出值，最後把 Module Ports 的接法從 Position sensitive 換成比較保險的 Explicitly declared，希望下次我不要再犯這種基本的錯。

2. 組員二 張振杰

這次實驗主要還是在 verilog 的 coding 上，其實這次實驗我才發現原來一個檔案在視覺上可以當作一個元件就像講義上畫格子寫 4-bit adder 的樣子。然後如果需要對接的話則要注重兩個檔案之間相連的函數例如 4-bit adder 的 carry 對接另一個 4-bit adder 的 Cin。整體下來如果說問題的話比較麻煩還是在 debug 上面，因為要一直確保檔案之間的函數連接，又要檢查各個函數裡的值是否正確。

3. 組員三 何啟造

這次的實驗也是延續上一週的 verilog 練習，所以整體來說不難。而這一次是用 verilog 實作出 4-bits adder 和 8-bits adder 只要思考出它的邏輯觀念，就不難做到，只是因為偶爾會出現一些小問題，所以 debug 要很久。