Lab08

實驗主題:Combinational Logic

實驗日期:2022/10/24

學號姓名:B103040009 尹信淳

實驗內容: Design and verify the following circuits using Verilog HDL and Schematic

Exercise1:

BCD to Excess-3 Code Converter

Circuit1: z=D’ y=CD+C’D’ x=B’C+B’D+BC’D’ w=A+BC+BD

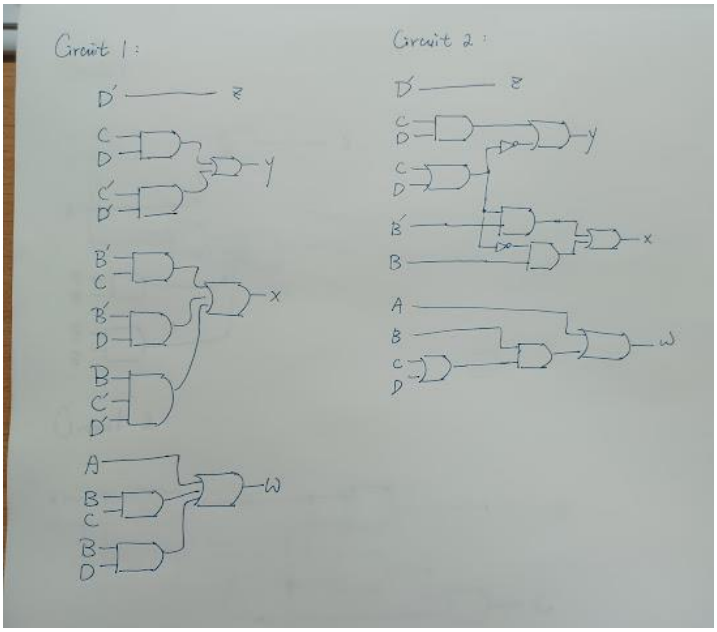
Circuit2: z=D' y=CD+C'D'=CD+(C+D)'

x=B'C+B'D+BC'D'=B'(C+D)+B(C+D)'

w=A+BC+BD=A+B(C+D)

Draw the logic diagrams of Circuit 1 and Circuit 2, and then verify Circuit 1 = Circuit 2 (using Verilog Dataflow modeling)

- Logic Diagram:



- Verification:

Verify Circuit 1 = Circuit 2 (using Verilog Dataflow modeling)

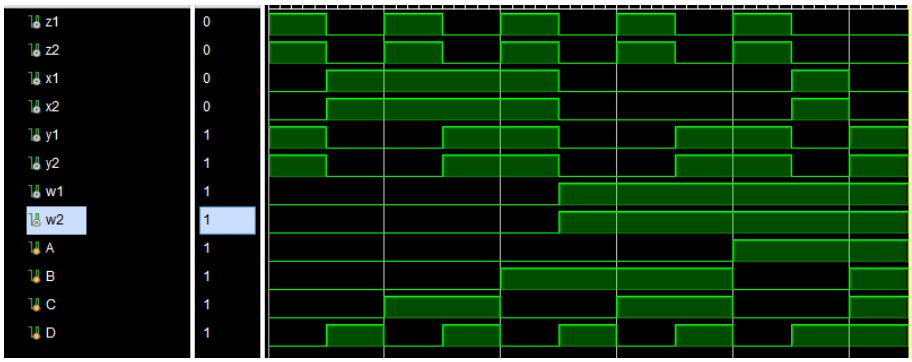
```
module e1(z1,y1,x1,w1,z2,y2,x2,w2,A,B,C,D);
    output z1,z2,y1,y2,x1,x2,w1,w2;
    input A,B,C,D;
    assign z1=~D;
    assign y1=(C&D)|(~C&~D);
    assign x1=(~B&C)|(~B&D)|(B&~C&~D);
    assign w1=A|(B&C)|(B&D);
    ////////////////////////////////////////////
    assign z2=~D;
    assign y2=(C&D)|~(C|D);
    assign x2=(~B&(C|D))|(B&~(C|D));
    assign w2=A|(B&(C|D));
endmodule
```

Testbench:

```
module tb1();
wire z1,z2,x1,x2,y1,y2,w1,w2;
reg A,B,C,D;
e1 UUT(.A(A),.B(B),.C(C),.D(D),.x1(x1),.x2(x2),.y1(y1),.y2(y2),.z1(z1),.z2(z2),.w1(w1),.w2(w2));
initial begin
A=1'b0;B=1'b0;C=1'b0;D=1'b0;#10
A=1'b0;B=1'b0;C=1'b0;D=1'b1;#10
A=1'b0;B=1'b0;C=1'b1;D=1'b0;#10
A=1'b0;B=1'b0;C=1'b1;D=1'b1;#10
A=1'b0;B=1'b1;C=1'b0;D=1'b0;#10
A=1'b0;B=1'b1;C=1'b0;D=1'b1;#10
A=1'b0;B=1'b1;C=1'b1;D=1'b0;#10
A=1'b0;B=1'b1;C=1'b1;D=1'b1;#10
A=1'b1;B=1'b0;C=1'b0;D=1'b0;#10
A=1'b1;B=1'b0;C=1'b0;D=1'b1;#10

A=1'b1;B=1'b1;C=1'b1;D=1'b1;#10 $finish;
end
endmodule
```

波型:



Exercise1 實驗結果與分析:

Circuit1=Circuit2.

Exercise2: Full Adder

Derive the following Boolean functions using Karnaugh maps

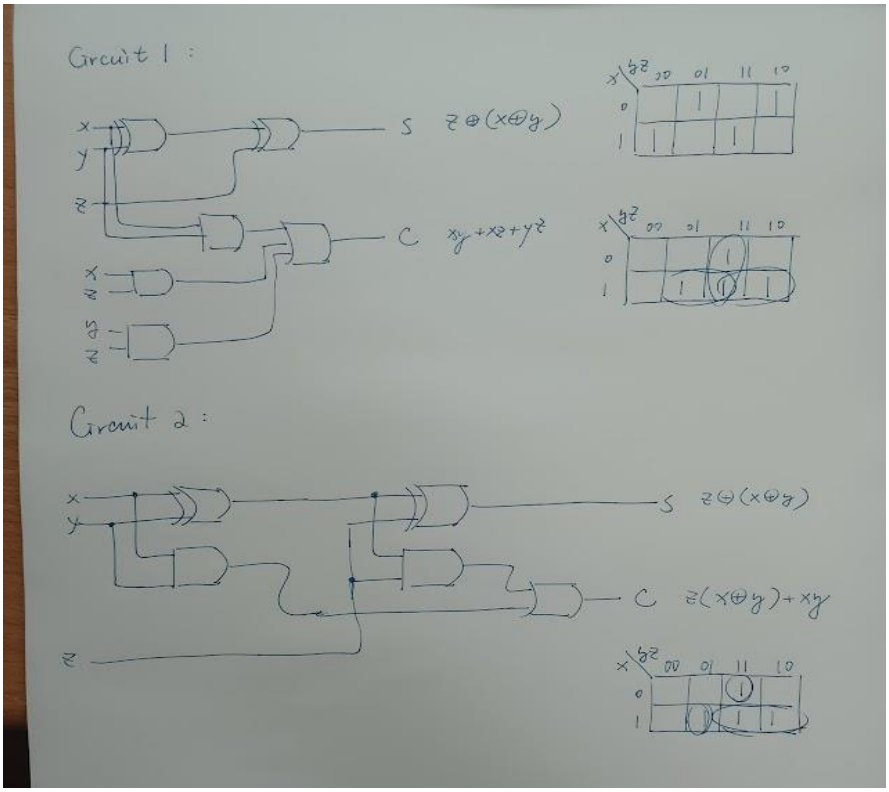
S = z⊕(x⊕y), C = xy + xz + yz (Circuit 1)

S = z⊕(x⊕y), C = z(x⊕y)+xy (Circuit 2)

Draw the logic diagram of Circuit 1 & Circuit 2.

Verify Circuit 1 = Circuit 2.

◆ Derivation&diagram



- Verification:

  Verify Circuit 1 = Circuit 2.

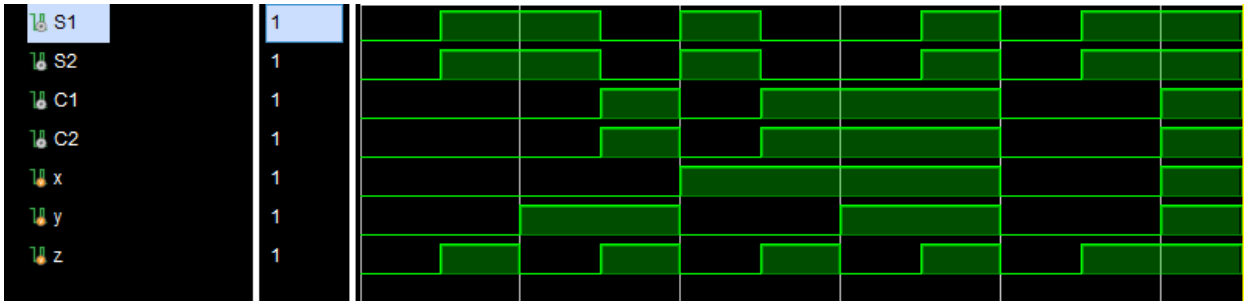
```
module e2(S1,C1,S2,C2,x,y,z);
output S1,C1,S2,C2;
input x,y,z;
wire n1,n2,n3,n4,n5,n6,n7,n8;
xor(n1,x,y);
xor(S1,n1,z);
and(n2,x,y);
and(n3,y,z);
and(n4,x,z);
or(C1,n2,n3,n4);
////////////////////////////////////
xor(n5,x,y);
xor(S2,n5,z);
and(n6,x,y);
xor(n7,x,y);
and(n8,n7,z);
or(C2,n8,n6);

endmodule
```

Testbench

```
module tb2();
wire S1,S2,C1,C2;
reg x,y,z;
e2 UUT(.S1(S1),.S2(S2),.C1(C1),.C2(C2),.x(x),.y(y),.z(z));
initial begin
x=1'b0;y=1'b0;z=1'b0;#10
x=1'b0;y=1'b0;z=1'b1;#10
x=1'b0;y=1'b1;z=1'b0;#10
x=1'b0;y=1'b1;z=1'b1;#10
x=1'b1;y=1'b0;z=1'b0;#10
x=1'b1;y=1'b0;z=1'b1;#10
x=1'b1;y=1'b1;z=1'b0;#10
x=1'b1;y=1'b1;z=1'b1;#10
x=1'b0;y=1'b0;z=1'b0;#10
x=1'b0;y=1'b0;z=1'b1;#10
x=1'b1;y=1'b1;z=1'b1;#10 $finish;
end
endmodule
```

波型



Exercise2 實驗結果與分析:

Circuit1=Circuit2.

Exercise 3: 8-bit Ripple-carry Adder

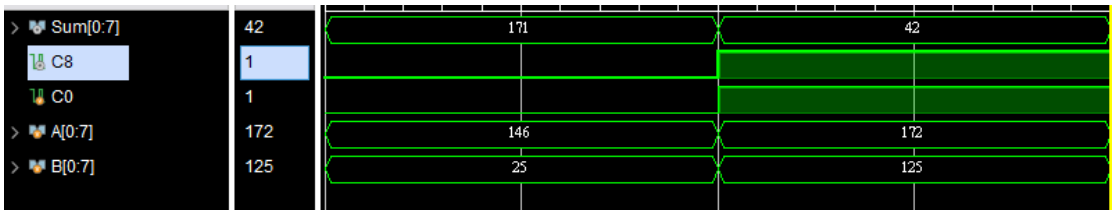
Design and verify the 8-bit ripple-carry adder composed of eight full-adders (using Verilog Structural level modeling)

```
module half_adder (output S, C, input x, y);
    xor (S, x, y);
    and (C, x, y);
endmodule
module full_adder (output S, C, input x, y, z);
    wire S1, C1, C2;
    half_adder HA1 (S1, C1, x, y);
    half_adder HA2 (S, C2, S1, z);
    or G1 (C, C2, C1);
endmodule
module e3(output[7:0]Sum,output C8,input[7:0] A,B,input C0);
    wire C1,C2,C3,C4,C5,C6,C7;
    full_adder  FA0 (Sum[0], C1, A[0], B[0], C0),
                FA1 (Sum[1], C2, A[1], B[1], C1),
                FA2 (Sum[2], C3, A[2], B[2], C2),
                FA3 (Sum[3], C4, A[3], B[3], C3),
                FA4 (Sum[4], C5, A[4], B[4], C4),
                FA5 (Sum[5], C6, A[5], B[5], C5),
                FA6 (Sum[6], C7, A[6], B[6], C6),
                FA7 (Sum[7], C8, A[7], B[7], C7);
endmodule
```

Testbench

```
module tb3();
wire[0:7]Sum;
wire C8;
reg C0;
reg[0:7] A,B;
e3 UUT(.Sum(Sum),.C8(C8),.C0(C0),.A(A),.B(B));
initial begin
    C0=1'b0;A[0]=1'b1;A[1]=1'b0;A[2]=1'b0;A[3]=1'b1;A[4]=1'b0;A[5]=1'b0;A[6]=1'b1;A[7]=1'b0;
    B[0]=1'b0;B[1]=1'b0;B[2]=1'b0;B[3]=1'b1;B[4]=1'b1;B[5]=1'b0;B[6]=1'b0;B[7]=1'b1;#10
    C0=1'b1;A[0]=1'b1;A[1]=1'b0;A[2]=1'b1;A[3]=1'b0;A[4]=1'b1;A[5]=1'b1;A[6]=1'b0;A[7]=1'b0;
    B[0]=1'b0;B[1]=1'b1;B[2]=1'b1;B[3]=1'b1;B[4]=1'b1;B[5]=1'b1;B[6]=1'b0;B[7]=1'b1;#10
 $finish;
end
endmodule
```

波型



Exercise3 實驗結果與分析:

運算結果符合預期。

實驗心得

這次的實驗設計了三種組合電路:第一題是 BCD 轉超三碼轉換器；第二題是全加器；第三題則是八位元的波紋進位加法器。前兩題的內容較著重於組合電路的精簡化，增加邏輯閘的共用。組合電路的簡化沒有一定的規則，因此這兩題會是非常好的經驗，應該好好學起來並運用。第三題雖然要自己設計，但其實四位元版本老師已經提供在 ppt 裡，因此只要看懂，仔細對好輸入輸出，把兩個四位元加法器合成八位元版本並不是難事。

總結這次的實驗，除了過程中再次運用了過去學習的技能與知識，也開始練習了真正組合電路的設計，雖然還不算是非常複雜的電路，但是一個好的基礎與開端。