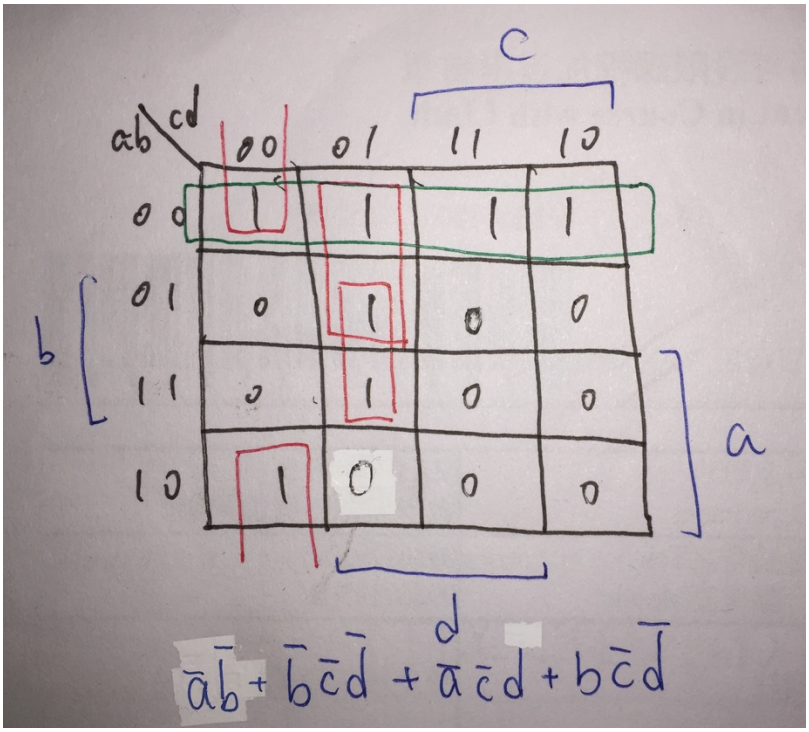
Logic Design

EECS101001

Lab 1: Fibonacci Detector – Report

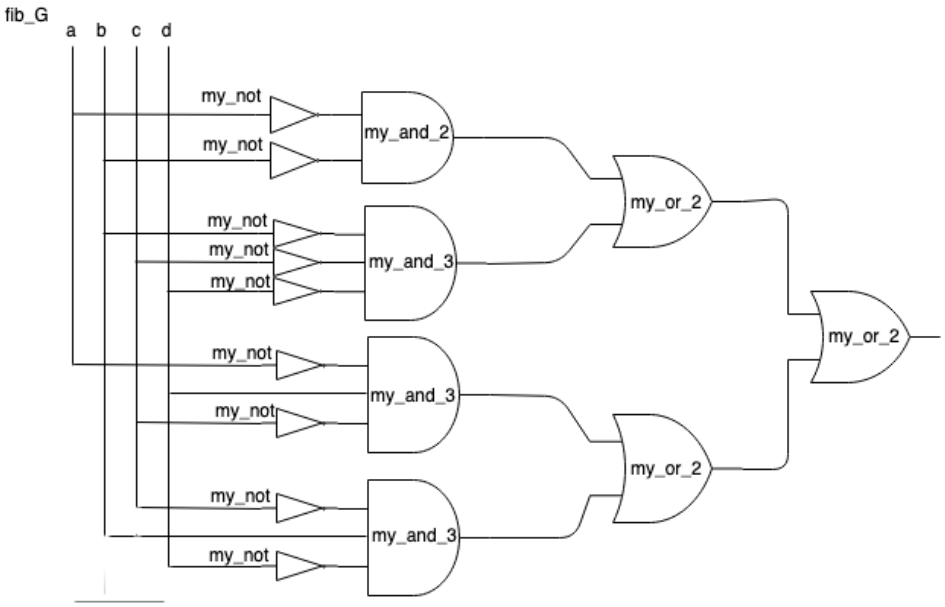
104021215 熊 磊

1. K-Map



In the K-Map the input abcd was storing in array {in[3], in[2], in[1], in[0]}. Since we consider the Hazard-Free, we need to add OR gate to $\bar{a}\bar{c}d$.

2. Graph
   - Schematic Graph



my_not GATE, my_and_2 GATE, my_and_3 GATE, my_or_2 GATE will describe later on.

- Truth Table

Dec	*A*	*B*	*C*	*D*	*Y*
0	0	0	0	0	1
1	0	0	0	1	1
2	0	0	1	0	1
3	0	0	1	1	1
4	0	1	0	0	0
5	0	1	0	1	1
6	0	1	1	0	0
7	0	1	1	1	0
8	1	0	0	0	1
9	1	0	0	1	0
10	1	0	1	0	0
11	1	0	1	1	0
12	1	1	0	0	0
13	1	1	0	1	1
14	1	1	1	0	0
15	1	1	1	1	0

- GATE Description

```
48  module fib_G(in, out);
49              parameter n=4;
50
51              //IO port declaration
52              input [n-1:0]in;
53              output out;//Output can be wire or reg, default type is wire.
54
55
56              wire not_a, not_b, not_c, not_d;//the default width of wire is 1-bit
57              wire and0, and1, and2, and3;
58              wire or0, or1;
59
60              //<gate><gate name>(output,input1,input2,input3...);
61              my_not not_0(not_a,in[3]);
62              my_not not_1(not_b,in[2]);
63              my_not not_2(not_c,in[1]);
64              my_not not_3(not_d,in[0]);
65
66              my_and_2 and_0(and0, {not_a, not_b});
67              my_and_3 and_1(and1, not_b, not_c, not_d);
68              my_and_3 and_2(and2, not_a, not_c, in[0]);
69              my_and_3 and_3(and3, in[2], not_c, in[0]);
70
71              my_or_2 or_0(or0, and0, and1);
72              my_or_2 or_1(or1, and2, and3);
73              my_or_2 or_2(out, or0, or1);
74
75  endmodule
```

-Explanation-

Module fib_G is start with parameter n = 4, since we just consider

0~15 in decimal. Then we describe the circuit using my own gate which is implemented by **NAND** gate.

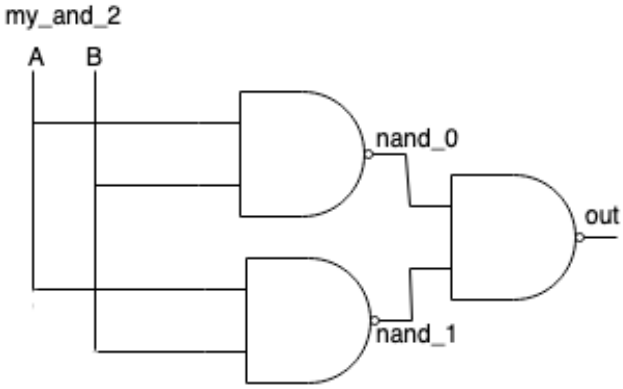
- Data Flow Description
  (!in[3] & !in[2]) |
  (!in[2] & !in[1] & !in[0]) |
  (!in[3] & !in[1] & in[0]) |
  (in[2] & !in[1] & in[0])

  -Explanation-
  4-bit input and 1-bit output, if input is 0, 1, 2, 3, 5, 8, or 13 (Fibonacci Number) output should be 1, otherwise, output should be 0.
  So we got the Behavior Description below…

- Behavior Description

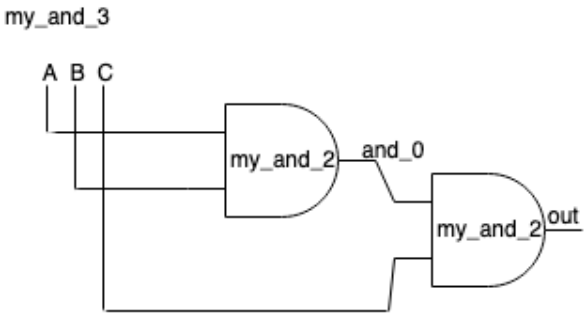
```
always@(*)begin
    case(in)
        0,1,2,3,5,8,13:begin
            out=1'b1;
        end
        default:begin
            out=1'b0;
        end
    endcase
end
```

3. Implementation Details
   Since Lab1 can only use **NAND** Gate to implement **AND, OR, NOT** GATE, I implement some GATE for me.
   - AND GATE for 2-inputs
     ∵ A · B = [(A · B)']'
     = [(A · B)' · (A · B)']'
     Hence, we can use **NAND** GATE to get **my_and_2** GATE.




```
module my_and_2(out, in);
    parameter n=2;

    input [n-1:0]in;
    output out;

    wire nand_0,nand_1;

    nand nand0(nand_0, in[0], in[1]);
    nand nand1(nand_1, in[0], in[1]);

    nand nand3(out, nand_0, nand_1);

endmodule
```

For 3-inputs **AND GATE** is easy deriving from **my_and_2** GATE.



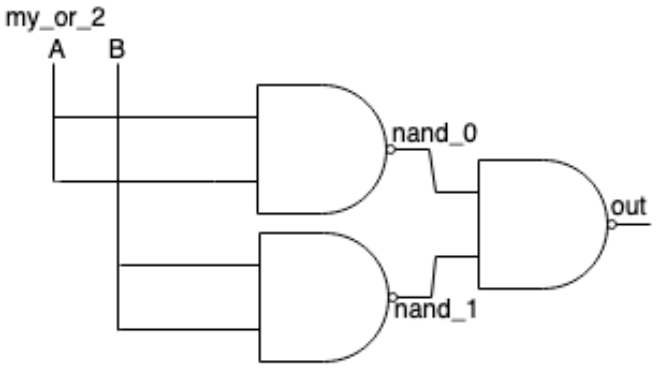

```
module my_and_3(out, in_0, in_1, in_2);
    input in_0, in_1, in_2;
    output out;

    wire and_0;

    my_and_2 and0(and_0, {in_0, in_1});
    my_and_2 and1(out, {and_0, in_2});

endmodule
```

- OR GATE for 2-inputs

  ∵ A + B = [(A+B)']'

$= [(A \cdot A)' \cdot (B \cdot B)']'$

Hence

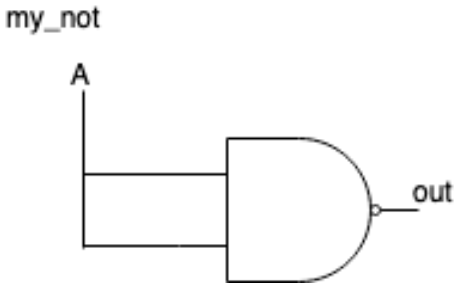



```
27  module my_or_2(out, in_0, in_1);
28      input in_0, in_1;
29      output out;
30
31      wire nand_0,nand_1;
32
33      nand nand0(nand_0, in_0, in_0);
34      nand nand1(nand_1, in_1, in_1);
35
36      nand nand3(out, nand_0, nand_1);
37
38  endmodule
```

- NOT GATE

  $\because A' = (A \cdot A)'$

  Hence




```
40  module my_not(out, in_0);
41      input in_0;
42      output out;
43
44      nand nand0(out, in_0, in_0);
45
46  endmodule
```

4. Simulation Result

```
time=   5,in=0000,out_G=1,out_D=1,out_B=1
time=  10,in=0001,out_G=1,out_D=1,out_B=1
time=  15,in=0010,out_G=1,out_D=1,out_B=1
time=  20,in=0011,out_G=1,out_D=1,out_B=1
time=  25,in=0100,out_G=0,out_D=0,out_B=0
time=  30,in=0101,out_G=1,out_D=1,out_B=1
time=  35,in=0110,out_G=0,out_D=0,out_B=0
time=  40,in=0111,out_G=0,out_D=0,out_B=0
time=  45,in=1000,out_G=1,out_D=1,out_B=1
time=  50,in=1001,out_G=0,out_D=0,out_B=0
time=  55,in=1010,out_G=0,out_D=0,out_B=0
time=  60,in=1011,out_G=0,out_D=0,out_B=0
time=  65,in=1100,out_G=0,out_D=0,out_B=0
time=  70,in=1101,out_G=1,out_D=1,out_B=1
time=  75,in=1110,out_G=0,out_D=0,out_B=0
time=  80,in=1111,out_G=0,out_D=0,out_B=0
All pass!! subarashii
Simulation complete via $finish(1) at time 80 NS + 0
```

5. Encountered Problems and Solving

- It's my very first time writing VERILOG, there are a lot of unfamiliar things. The first problem I got is how to compile a Verilog program. But I solved it later on.
- Only can using NAND GATE is a little bit challenging, but only need some Law from Boolean algebra can easily implement the GATE needed.
- The "parameter" in module. The first time I saw it I had no idea what it is, and then I saw something like *input [n-1:0]in* , *in[3]* , *in[2]* , … I thought maybe its mean the input may be an array. Therefore, I try to use in my GATE *my_and_2*.
- Finally, when I almost finish and get this error

```
ncelab: *W,CUVMPW (./fib.v,22|19): port sizes differ in port connection (1/2).
        my_and_2 and0(and_0, {in_0, in_1});
                             |
ncelab: *W,CUVMPW (./fib.v,22|22): port sizes differ in port connection (2/1).
        my_and_2 and1(out, {and_0, in_2});
                           |
ncelab: *W,CUVMPW (./fib.v,23|17): port sizes differ in port connection (1/2).
        my_and_2 and1(out, {and_0, in_2});
                           |
ncelab: *W,CUVMPW (./fib.v,23|20): port sizes differ in port connection (2/1).
        my_and_2 and1(out, {and_0, in_3});
                             |
ncelab: *W,CUVMPW (./fib.v,34|17): port sizes differ in port connection (1/2).
        my_and_2 and1(out, {and_0, in_3});
                           |
ncelab: *W,CUVMPW (./fib.v,34|20): port sizes differ in port connection (2/1).
        my_and_2 and0(and_0, {in_0, in_1});
                             |
ncelab: *W,CUVMPW (./fib.v,22|19): port sizes differ in port connection (1/2).
        my_and_2 and0(and_0, {in_0, in_1});
                             |
ncelab: *W,CUVMPW (./fib.v,22|22): port sizes differ in port connection (2/1).
        my_and_2 and1(out, {and_0, in_2});
```

I just put my function’s parameter in wrong order. So when input isn’t array will get this error.

- It’s really good opening in Verilog, hoping that I will keep enjoy in this class. Thanks to all TAs and Professor!!