Lab 2 6-Bit by 2-Bit Binary Multiplier

Verilog Tutorials

Always block

Please refer to Always block tutorial

1. Do not assign too many signals in a single always block, it will be really hard to debug.
2. Use one main case statement or if/else statement in a always block.
3. Construct the module with several always blocks.
4. It is forbidden to assign a signal at several places of your code simultaneously.

Wrong example：

```
always@(*)begin
        case(in)
                1'b1:     out = 1'b1;
                default : out = 1'b0;
        endcase
end

always@(*)begin
        if (a == b && c == d)    out = 1'b0;
        else                     out = 1'b1;
end
```

In the above example, “out” is assigned in two different always blocks. When two always blocks assign value to the same signal simultaneously, it is a fatal bug.

Use { } in verilog

You can use “{ }” to concatenate bits.
Example:

```
module Bracket_example1 (a, b, c, out);
        input a, b, c;
        output [5:0] out;

        wire a, b, c;
        wire [5:0] out;

        assign out = {3'b000, a,b,c};

endmodule
```

Lab 2 Description

The main goal of this lab is to implement a 6-Bit by 2-Bit Binary Multiplier.

Modules

There are four main modules we need to design in this lab.

6-Bit by 2-Bit Binary Multiplier - (1)

- Half-Adder - (2)
- 2-Bit by 2-Bit Binary Multiplier - (3)
- 4-bit carry lookahead adder - (4)

Half-Adder

First, construct a Half-Adder.

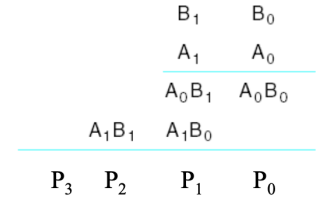
Half Adder

x	y	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Truth Table of Half-Adder (hint: You can refer to CH4 p.13~p.15)

2-Bit by 2-Bit Binary Multiplier

Make use of two Half-Adders to construct a 2-bit by 2-bit binary multiplier.

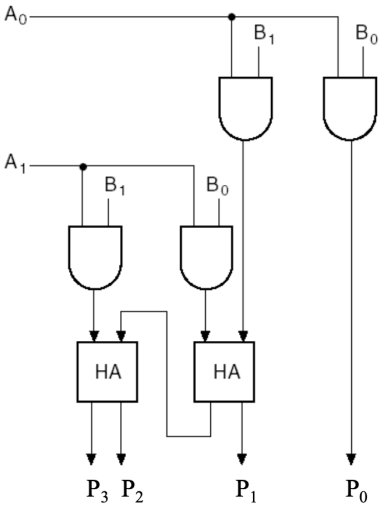


$P_0 = A_0B_0$

$P_1 = A_0B_1 + A_1B_0$

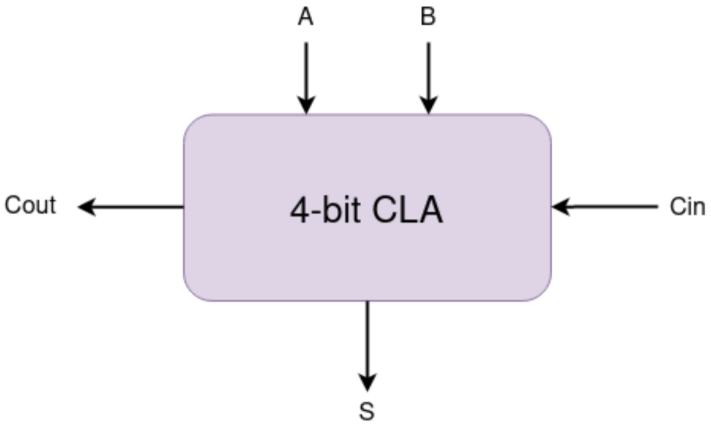
$P_2 = A_1B_1 + c_2$

$P_3 = c_3$



4-bit Carry Lookahead Adder

Construct a 4-bit CLA.



6-Bit by 2-Bit Binary Multiplier

Lastly, construct a 6-bit by 2-bit multiplier by using 2-bit by 2-bit multipliers and 4-bit CLAs as the building blocks.

The name of the “6-Bit by 2-Bit Binary Multiplier” module should be “Multiplier”. Its 6-bit input’s name should be “in1”, its 2-bit input’s name should be “in2”, and its 8-bit output’s name should be “out”.

Files to upload

1.Multiplier.v

2.{studentID}_report.pdf (For example, 111062000_report.pdf)

(1) Explain the construction of your 6-bit by 2-bit multiplier, and provide a block diagram of your design.
(2) Execution result (screen shot).

Please upload these two files on the eeclass system.

Be careful of the file name, module name, and whatever listed above.

Wrong format will result in the failure(0 points) of your assignment, please double check before uploading it.

Deadline

2023/05/11 23:58

Upload the files before the deadline!

Late assignment will get a 10% penalty per day, and no assignment will be accepted after 3 days from the given due date.

Do not plagiarize!