计算机系统结构实验报告 实验 2

姓名：卞思沅

学号：518021910656

日期：2020/05/27

目录：

1. 实验概述

1.1 实验名称

FPGA 基础实验：4-bitAdder

1.2 实验目的

1）掌握 Xilinx 逻辑设计工具 Vivado 的基本操作

2）掌握 VerilogHDL 进行简单的逻辑设计

3）使用功能仿真；

4）约束文件的使用和直接写法

5）添加时序约束

6）生成 Bitstream 文件

2. 实验描述

2.1 实验简述

1. 按照要求创建工程并添加文件

2. 在源代码编辑区双击 adder_1bit，添加如下方框中的代码：

```
module adder_1bit(
    input a,
    input b,
    input ci,
    output s,
    output co
    );
    wire s1, c1, c2, c3;
    and  (c1, a, b),
         (c2, b, ci),
         (c3, a, ci);

    xor  (s1, a, b),
         (s, s1, ci);

    or   (co, c1, c2, c3);

endmodule
```

3. 实现 4 位加法器。创建源文件，命名为 adder_4bits，添加如下代码

```
module adder_4bits(
        input [3:0] a,
        input [3:0] b,
        input ci,
        output [3: 0] s,
        output co
        );

        wire [2:0] ct;

        adder_1bit a1(.a(a[0]), .b(b[0]), .ci(ci), .s(s[0]),.co(ct[0])),
                   a2(.a(a[1]), .b(b[1]), .ci(ct[0]), .s(s[1]),.co(ct[1])),
                   a3(.a(a[2]), .b(b[2]), .ci(ct[1]), .s(s[2]),.co(ct[2])),
                   a4(.a(a[3]), .b(b[3]), .ci(ct[2]), .s(s[3]),.co(co));

endmodule
```

4. 功能仿真，创建激励文件，输入以下代码

```
module adder_4bits_tb(   );

   reg [3:0] a;
   reg [3:0] b;
   reg ci;

   wire [3:0] s;
   wire co;

   adder_4bits u0 (
       .a(a),
       .b(b),
       .ci(ci),
       .s(s),
       .co(co)
       );

   initial begin
       a = 0;
       b = 0;
       ci = 0;

       #100;
       a = 4'b0001;
       b = 4'b0010;
       #100;
       a = 4'b0010;
       b = 4'b0100;

       #100;
       a = 4'b1111;
       b = 4'b0001;
       #100;
       ci = 1'b1;

   end

endmodule
```

5. 点击左侧区的 Run Simulation 并选择 Run Behavioral Simulation。得到正确的仿真波形。

2.2 代码及结果

Adder_1bit.v:

```
23 module adder_1bit(
24     input a,
25     input b,
26     input ci,
27     output s,
28     output co
29     );
30     wire s1, c1, c2, c3;
31     and (c1, a, b),
32         (c2, b, ci),
33         (c3, a, ci);
34 
35     xor (s1, a, b),
36         (s, s1, ci);
37 
38     or (co, c1, c2, c3);
39 endmodule
```

adder_4bit:

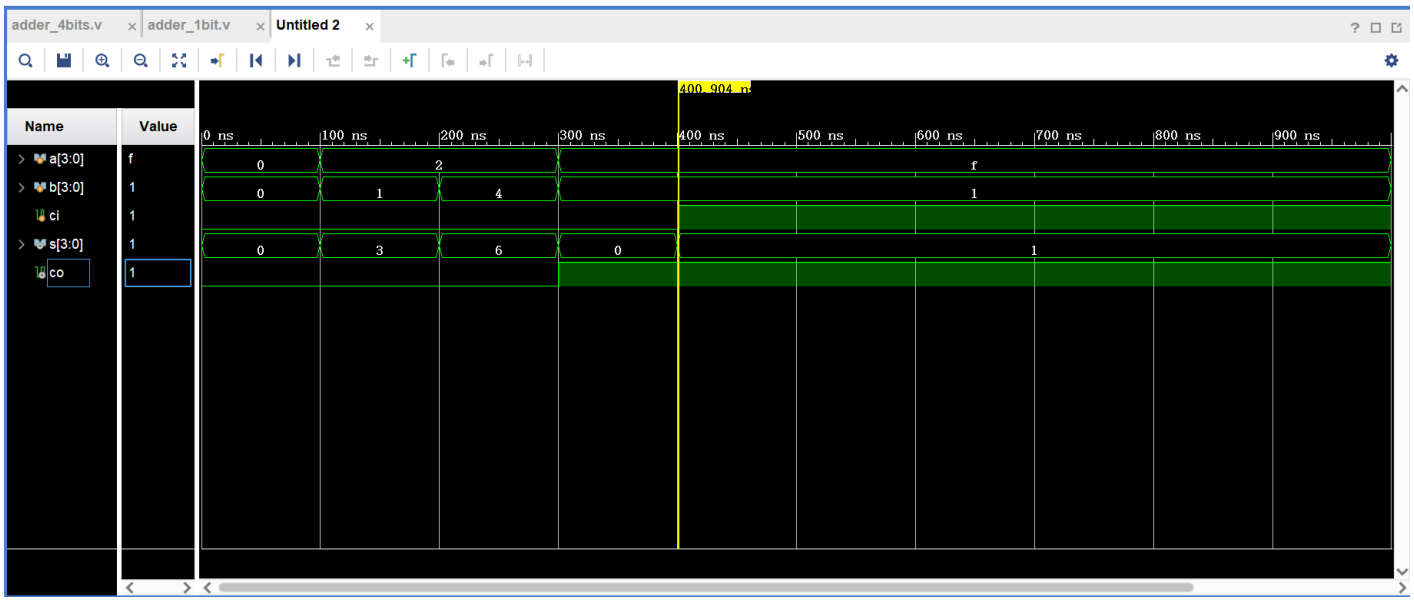
```
module adder_4bits(
    input [3:0] a,
    input [3:0] b,
    input ci,
    output [3:0] s,
    output co
    );

    wire [2:0] ct;

    adder_1bit  a1(.a(a[0]), .b(b[0]), .ci(ci), .s(s[0]), .co(ct[0])),
                a2(.a(a[1]), .b(b[1]), .ci(ct[0]), .s(s[1]), .co(ct[1])),
                a3(.a(a[2]), .b(b[2]), .ci(ct[1]), .s(s[2]), .co(ct[2])),
                a4(.a(a[3]), .b(b[3]), .ci(ct[2]), .s(s[3]), .co(co));

endmodule
```

激励文件：

```
module adder_4bit_tb(
    );

    reg [3:0] a;
    reg [3:0] b;
    reg ci;

    wire [3:0] s;
    wire co;

    adder_4bits u0 (
        .a(a),
        .b(b),
        .ci(ci),
        .s(s),
        .co(co)
        );
```

```
    initial begin
        a = 0;
        b = 0;
        ci = 0;

        #100;
        a = 4'b0010;
        b = 4'b0001;

        #100;
        a = 4'b0010;
        b = 4'b0100;

        #100;
        a = 4'b1111;
        b = 4'b0001;
        #100
        ci = 1'b1;

    end

endmodule
```

仿真结果：



3. 实验心得与总结

本次实验依旧无需自己编写代码，按照教程便可以得到结果，较为容易。

在实验中，我注意观察了示例代码的结构与各文件之间的关联，大致理解了模块间的关系。模块文件定义某一模块（可以理解为函数），而激励文件调用此模块，并用具体数据将模块实例化，并通过仿真模拟出结果。此外，通过代码阅读以及上网搜索资料，我弄清楚了示例代码各部分的含义，理解了代码

编写过程中的各项注意点，感觉实验很有收获。

4、参考资料

2020 计算机系统结构实验指导书-LAB02_M