计算机系统结构实验报告

Lab02

FPGA 基础实验：4-bit Adder

姓　　名：徐薪

学　　号：519021910726

日　　期：2021 年 6 月 09 日

摘 要

本实验实现了 FPGA 基础实验中的 4-bit adder。该器件支持带进位的 4 位二进制数的加法，输入两个 4 位二进制数，输出一个 4 位二进制数，同时也是输入的和。我们通过与、或、异或等简单的逻辑运算实现 1-bit adder，并通过对 1-bit adder 进行组合得到 4-bit adder。本实验通过软件仿真的形式进行实验结果的验证。

目录

1. 实验概述

1.1 实验内容

本实验实现了 FPGA 基础实验中的 4-bit adder。该器件支持带进位的 4 位二进制数的加法，输入两个 4 位二进制数，输出一个 4 位二进制数，同时也是输入的和。我们通过与、或、异或等简单的逻辑运算实现 1-bit adder，并通过对-bit adder 进行组合得到 4-bit adder。本实验通过软件仿真的形式进行实验结果的验证。

1.2 实验目的

（1）熟悉 Xilinx 逻辑设计工具 Vivado 的基本操作；

（2）掌握使用 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）上板验证。

2. 原理分析

2.1 1-bit adder 的原理

一位全加器包括 A_i、B_i 和 C_i 三个一位输入端，其中 A_i 和 B_i 表示输入的两位数，C_i 表示上一次计算的进位；同时还包括 S_i 和 C_0 两个输出端，S_i 表示加法的运算结果，C_0 表示向下一位传递的进位。其真值表如下：

Ai	Bi	Cin	Si	Co
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

表 1. 1-bit adder 的真值表

从中我们可以得到各变量的逻辑表达式为：

$S=A\oplus B\oplus C_{in}$

$C_0=C_{in}(A\oplus B)+AB$

2.2 4-bit adder 的原理

四位全加器可以看成四个一位全加器串联而成，其示意图如下所示：

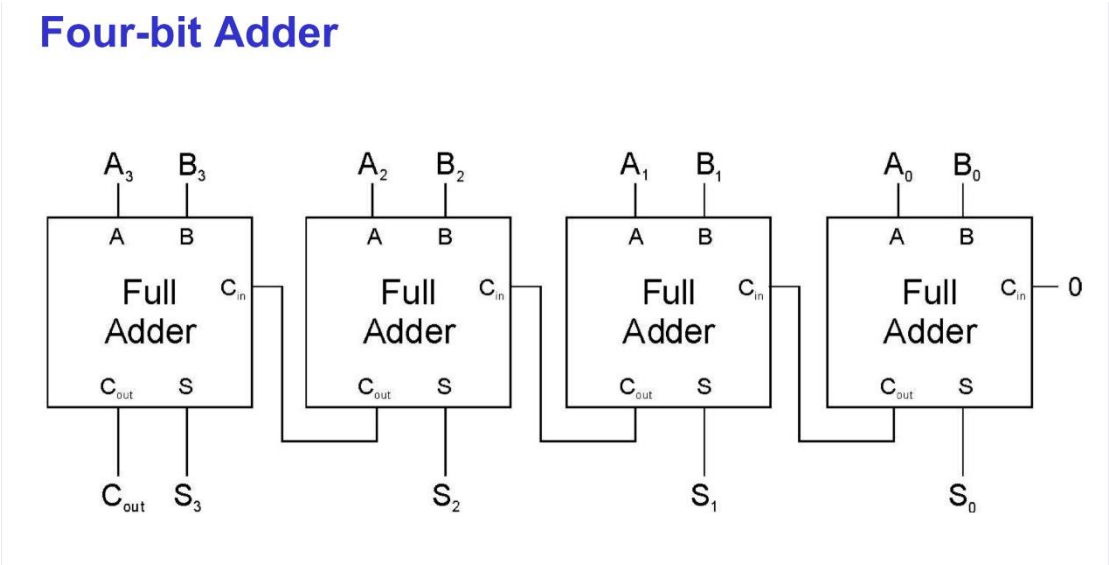

图 1. 4-bit adder 的示意图

其中 A_i、B_i 表示输入 A、B 的第 i 位，S_i 表示输出 S 的第 i 位。

C_{out} 表示最终结果的溢出。

2.3 上板验证的原理

因为 Flow Navigator 可以自动完成综合、实现、生成 FPGA 配置文件，所以上板验证的实验原理与 4-bit adder 的实验原理相同，这里不再赘述。

3. 功能实现

3.1 1-bit adder

我们通过巧妙设置中间变量 s1、c1、c2、c3 来储存中间结果，从而简化逻辑运算表达：

```
module adder_1bit(
    input a,
    input b,
    input ci,
    input 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
```

3.2 4-bit adder

根据图 1 所示的四位全加器的示意图，我们可以通过串联四个一

位全加器来实现四位全加器。前一个一位全加器的 C_0 设为后一个一位全加器的 C_{in}，这样就将四个一位全加器串联了起来。四位全加器的代码实现如下：

```
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. 仿真测试

我们使用 Verilog 编写激励文件，采用软件仿真的形式对于 4-bit adder 进行测试，测试结果如图 1 所示：

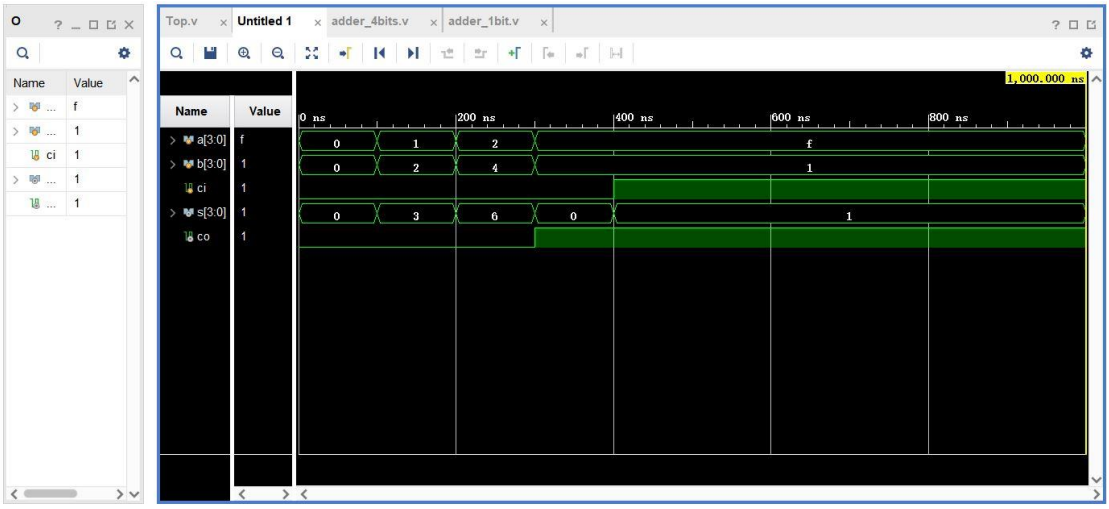

图 2. 4-bit adder 仿真结果

从图 1 可以看出，我们完成了 4-bit adder 的功能实现，并且仿真结果正确。

5. 实验总结

5.1 实验评价

图 1 的验证结果以及上板验证的结果表明 4-bit adder 能够实现正常的功能，因此本次实验是成功的。

5.2 实验心得

这个实验总体来讲是简单的，只要按照实验操作指导书一步一步来，就能得到正确的实验结果。

在这一次实验中，我最大的收获就是深刻理解了藏在 Verilog 语言下的 module 思想。简而言之，就是在设计一个复杂的元件时，可以先把它拆成几个小的功能逻辑单元，然后再依次实现这些小的功能逻辑单元，最后将它们组合起来，就实现了整个元件的功能。正如 4-bit adder 的设计，我们先把它的功能拆成 4 个 1-bit adder，实现了 1-bit adder 的功能之后，我们将它们串联起来，就实现了 4-bit adder 的功能。这一思想在之后的实验中会经常用到，尤其是在 MIPS 处理器的设计中，我是把 MIPS 的功能拆成一个个独立的运算逻辑单元，然后再分别实现这些逻辑单元，最终将它们有机联系起来得到 MIPS 处理器。