
```
`timescale 1 ps /
```

FPGA Note

```
module PLL_ctrl (
```

FPGA 笔记

```
    areset,  
    inclk0,  
    c0,  
    c1,  
    e0,  
    locked);
```

```
    input  areset;  
    input  inclk0;  
    output c0;  
    output c1;  
    output e0;  
    output locked;
```

**FP
GA**

```
    wire [5:0] sub_wire0;  
    wire sub_wire3;  
    wire [3:0] sub_wire4;  
    wire [0:0] sub_wire5 = 1'b0;  
    wire [1:1] sub_wire2 = sub_wire0[1:1];  
    wire [0:0] sub_wire1 = sub_wire0[0:0];  
    wire c0 = sub_wire1;  
    wire c1 = sub_wire2;  
    wire locked = sub_wire3;  
    wire [0:0] sub_wire5 = sub_wire4[0:0];  
    wire e0 = sub_wire5;  
    wire sub_wire6 = inclk0;  
    wire [1:0] sub_wire7 = {sub_wire6,  
sub_wire6};
```

```
endmodule
```

吾生也有涯，而知也无涯。以有涯随无涯，殆已！已
而为知者，殆而已矣！

整理：Yu Wang

整理时间：May 14, 2018

Email: wyu0725@mail.ustc.edu.cn

目 录



I	FPGA 介绍	2
II	Xilinx White Paper	3
1	WP 275: Get your Priorities Right - Make your Design Up to 50% Smaller	5
1.1	基本的逻辑实现	5
1.1.1	Single Level Logic	5
1.1.2	Two Level Logic	5
1.2	D 触发器中的同步和异步	6

说明



最近看到两篇有意思的 Xilinx White Paper，意识到也该把平时遇到的有意思的设计方面的内容记录下来，内容会慢慢的扩充.

本文档使用的模板来自于 ElegantBook，网址已经停止维护了，但是其中一个作者的博客还在更新 [Ethlisan](#)，感谢作者提供模板

部分 I

FPGA 介绍

部分 II

Xilinx White Paper

Xilinx 公司有一系列的 White Paper，口气不小，不过 Xilinx 工程师确实是有这个实力，写 White Paper 的工程师定是扫地僧式的人物。



第 1 章 WP 275: Get your Priorities Right - Make your Design Up to 50% Smaller



[文档下载地址](#)，一开始学 Verilog 的时候把 Verilog 当成 C 一样的语言来学习，逐步了解到代码的背后是有硬件结构支持，也不是所有的代码都是可综合的，这一点暂时不讨论。这篇文档刷新了认识，代码背后不仅由硬件结构，还有基本的 FPGA 单元。这篇文档从 LUT 和 D 触发器入手介绍编译器是如何处理代码的。

1.1 基本的逻辑实现

1.1.1 Single Level Logic

首先考虑一个 4 输入与门的实现，代码如下

Verilog

```
1 always @(posedge clk) begin
2     data_out <= a & b & c & d;
3 end
```

VHDL

```
1 process (clk)
2 begin
3     if clk' event and clk=' 1' then
4         data_out <= a and b and c and d;
5     end if;
6 end process
```

其实现如1.1所示，使用最简单的一个 D 触发器和一个 4 输入 LUT 即可实现

1.1.2 Two Level Logic

当逻辑门的输入个数增加之后，比如说增加到 6 个之后，LUT 输入的个数显然不够了，这个时候就要使用两个 LUT，如下1.2所示，这种情况无论如何 2 个 LUT 是免不了的。

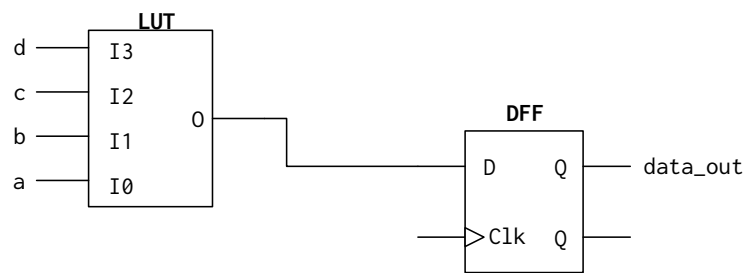


图 1.1: Single level logic use 4 input LUT and D Flip-Flop

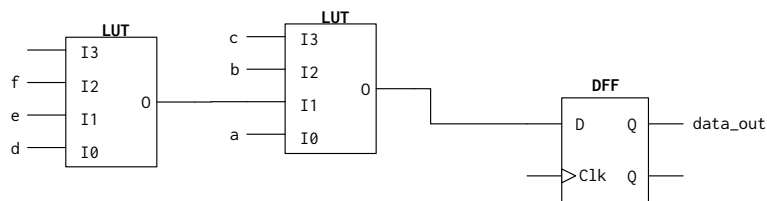


图 1.2: Two Level Logic use two 4-input LUT and a D Flip-Flop

1.2 D 触发器中的同步和异步

首先先在如下的 VHDL 代码中添加异步复位的功能异步复位代码

再看下一个例子，添加 Reset 和 Enable 的功能，如下代码所示 Enable 和 Reset 功能的代码。图下面对这种情况进行解释，首先请回忆

