# 【 FPGA 】时钟树问题简介

时钟域的相关知识，形象的说就是时钟信号的“势力”范围，它通过时钟树的形式实现。

时钟树不仅可以做到高扇出，还可以做到让时钟信号到达各个触发器的时刻尽可能一致，也即保证时钟信号到达时钟域内不同触发器的时间差最小。

# （1）什么情况下，时钟应该“上树”？

如果一个时钟信号是为FPGA内部的一些逻辑资源提供“脉搏”的，那么强烈建议该时钟“上树”；

如果时钟信号的时钟域实在太小，例如仅控制若干个触发器，那么也许不利用时钟树，FPGA设计也可能通过时序分析，但是仍然建议使用时钟树；

如果时钟信号的时钟域只包括一个触发器，那么也就不存在所谓的时间差了，此时就完全不需要时钟树；

如果一个时钟信号仅仅是为FPGA外部的硬件电路提供时钟激励的，那么外部无论有多少个存储单元需要使用该时钟，都没必要使用时钟树，因为FPGA内部的时钟树无法延伸到FPGA芯片外部。

# （2）如何选择时钟树？

上篇博文提到了时钟树的类型，分为全局时钟树、局部时钟树和IO时钟树。那么具体来说，如果需要使用时钟树，该为时钟选择哪一类时钟树呢？

也许自觉会这么告诉自己，时钟域大的，选择全局时钟树；时钟域小的，选择区域时钟树；时钟域特别小，选择IO时钟树。

事实告诉你，自觉是完全错误的。

IO时钟树分布在FPGA的接口资源中，由于它们离IO管脚最近，所以可以协助FPGA完成一些较高速率的串行数据接收，再经过简单地串并转换之后，以比较低的速率将并行数据丢进FPGA芯片的内部，供其他资源使用。一般来说，每个IO BANK内部会有若干个IO时钟树的资源，因此IO时钟树虽然覆盖范围小，但并不是为小规模的时钟域量身定做的，因此FPGA内部的资源也无法使用该时钟树。

再看全局时钟树，由于全局时钟树可以覆盖到整个FPGA芯片，因此全局时钟树的个数也十分有限，因此使用一定要谨慎，不可滥用。但是如果你硬着头皮省下来一堆全局时钟树，结果却闲置在一边，不派上用场，那简直就是浪费时间，白花心思。因此，全局时钟树这样的资源，不可滥用，也不可不用，要充分利用。

因此，在全局时钟树不紧缺的情况下，无论时钟域的大小，统一建议使用全局时钟树，因此这样也能给编译器提供最大的布局布线自由度，从而让时序约束更容易实现。

最后来看区域时钟树。老实说，区域时钟树覆盖范围也是相当的大，最大可能能到FPGA芯片的几分之一，因此如果时钟域不是特别大，到底使用全局时钟树还是区域时钟树，其实没有一个确定的结论。不过如果不是全局时钟树资源不够用，一般不建议使用区域时钟树。当然了，使用区域时钟树可以让时钟域中资源的分布在物理上更紧凑一些，并且有些功能是必须使用区域时钟树和IO时钟树配合来完成的，因此请注意相关功能的说明。

最后总结下，IO时钟树用于IO接口的串并转换，不可用于FPGA内部时钟域。

全局时钟树，可以覆盖到整个FPGA芯片，在全局时钟树不紧缺的情况下，尽量使用全局时钟树，可以给编译器提供最大的布局布线自由度，让时序约束更容易实现。

局部时钟树，特定情况下可能又用途，但全局时钟树不紧缺的情况下，建议使用全局时钟树。

# （3）时钟信号如何“上树”？

## 1）使用全局时钟树资源

方法一，通过正确的物理连接。

如果时钟信号是由FPGA芯片外部产生的，那么我们可以不通过编程就可以实现时钟树资源的分配。

因为在FPGA芯片的外围管脚中，有一些专门为全局时钟设计的管脚，这点我们可以通过相应的FPGA芯片的数据手册来确认，如果在制作电路板时，直接将外部时钟信号通过这些管脚接入FPGA内部，那么它将自动占据全局时钟树资源。当然了，这些管脚也可以接入普通的数据信号，编译器会对该管脚引入的信号在FPGA设计内部扮演的角色进行分析，如果发现其并没有作为时钟信号来使用，那么将不会为其分配时钟树资源。

方法二，通过恰当的代码描述。

如果很不巧，外部的时钟信号（外部时钟）没有通过专用的全局时钟管脚连接到FPGA内部，又或者某一个时钟信号是FPGA内部产生（再生时钟）的，例如FPGA内部PLL的输出，那么此时就需要通过编写程序来完成时钟的“上树”工作了。有些时候，即使不使用代码显示指定，编译器也会根据代码的分析结果，来为时钟信号分配全局时钟资源。不过这种靠“天”吃饭的思想不可取，FPGA工程师一定要让FPGA芯片尽可能的处于自己，而不是编译器的掌控之下，因此强烈建议通过自己的代码来指明时钟树的使用。

那么具体要怎么通过HDL代码来实现时钟树资源的分配呢？答案就是使用原语。

由于原语是跟FPGA芯片的生产厂商息息相关的，因此同一个功能的原语在不同的编译器中的名称很可能大相径庭，例如用于全局时钟树分配的最主要的原语，Xilinx公司叫它BUFG，而Altera公司却称其为global。

这里，以Xilinx公司的FPGA产品为例，来介绍代码的描述方法，其他公司的FPGA产品方法类似，只不过需要替换原语的名称罢了。

如果FPGA内部有一个名为innerClk的时钟信号，我们想为它分配一个全局时钟树，Verilog HDL描述为：

wire globalClk;

BUFG onTree(.O(globalClk), .I(innerClk));

按照上述HDL代码描述以后，我们就可以在后续的逻辑功能中放心使用上树后的innerClk——globalClk了。

实际上，直接从外部全局时钟管脚引入的时钟信号，相当于在HDL代码中使用了IBUFG + BUFG原语。

除此之外，如果希望多个时钟信号分享一个时钟树，也可以使用BUGMUX这个原语，相当于MUX +BUFG，例如，希望当前FPGA设计中的某一部分逻辑其时钟是可以在40Hz和60Hz之间切换的。

## 2）使用区域时钟树、IO时钟树资源

与全局时钟管脚类似，FGPA芯片的外围管脚中也有专门为区域时钟和IO时钟设计的专有管脚，但是，光将时钟信号连接到这些管脚上，还并不一定能完成相应时钟树的使用，还必须要在代码中显式地进行描述才行。

以Xilinx公司为例，使用原语BUFIO，将会为这些专用管脚上的信号分配IO时钟树资源，使用BUFR，将会为这些专用管脚上的信号分配区域时钟树资源。由于区域时钟常配合IO时钟完成串并转换，因此，BUFR还具有神奇的分频功能。最后，由于这两个时钟树的覆盖范围并不是整个FPGA芯片，所以在进行HDL代码编写时，也请注意资源的使用。

# （4）被“拉下树”的时钟信号

已经上树的时钟信号，若不小心，也可能被拉下树，因此，在HDL代码编写的时候，一定要避免这种情况。

是什么导致时钟信号脱离了时钟树了呢？

通过前面的介绍，我们知道时钟树是由若干级缓冲器再加上一些近似等长的连线组成的，这也就是说，时钟树仅能对时钟信号起到一个基本的传递作用，除此以外，别无它用。

因此，凡是相对时钟树上的时钟信号进行任何逻辑操作，来生成一个新的信号，那么新的信号已经不再位于时钟树上了（注意，原来的时钟信号仍在时钟树上）。如果希望新的信号仍然作为时钟来驱动一些逻辑，那么必须重新调用相应原语来让新的时钟信号获得空闲的时钟树资源，所以，之前介绍的FPGA内部生成的再生时钟，门控时钟，行波时钟，如果需要使用，一定要先使用原语为它们分配好时钟树资源。

下面举例说明，原始时钟信号被拉下树以及在此上树的过程：

// gClkOnTreeA is on the clock tree

assign midClk0 = ~gClkOnTreeA; // midClk0 is not on the clock tree

assign midClk1 = en & gClkOnTreeA; // midClk1 is not on the clock tree

BUFG reOnTree0(.O(gClkOnTreeB),.I(midClk0)); //gClkOnTreeB is on the clock tree

BUFG reOnTree1(.O(gClkOnTreeC),.I(midClk1)); //gClkOnTreeC is on the clock tree