4785 字

利用 SmartCompile 和赛灵思的设计工具进行设计保存

采用像分区、自动命名和拓朴匹配这样的设计保存技术，有可能把解决问题所花费的时间缩短几个月。

作者：Eric Shiflet 和 Kate Kelley
赛灵思公司

在 FPGA 环境下，设计保存是一项复杂的实现挑战，它需要保存的事项包括：一项设计的 HDL 描述、一个模块的综合网表、约束文件内的布局信息，以及在局部比特文件中的配置数据。赛灵思的 Integrated Software Environment (ISE) 9.1i 软件以新的 SmartCompile 技术为特色，其中包含两种新的方法– SmartGuide 和 Partitions-它可保存像布局或布线这样的设计实现数据，并且可以减少解决问题所花费的时间。

SmartGuide 采用命名和拓朴匹配技术来识别一个 FPGA 设计中相对于以前的实现还没有发生改变的各个部分。在新的和已修改过的设计被重新实现时，设计中匹配元器件得到了保存。

Partitions 采用一种技术，其中，FPGA 设计的模块实例被自动地分析，然后，跟以前的模块实现比较，确定该模块实例是最新的还者是过时的。如果Partition 是过时的，它也完全可以重新实现且不发生保存。如果Partition是最新的，它可从以前的实现中被严格地复制且(通过布线从综合网表)完整地保存布局和布线运行时间。

SmartGuide: 它如何运作？

SmartGuide 工作在像查找表(LUT)和触发器这样的 FPGA 中的最低级的物理单元上。这些单元及其连通性依次得到匹配和保存。

要成功引导的第一步是对已经作出较小变更的设计综合一个一致的网表。例如，减法器等式 **regAB=(A"B)**，可以被综合为 **Msub_sub0001** 的逻辑名称。下一个减数就称为 **Msub_sub0002**，并且依此类推。在引入新减数的设计中的任何变更都可能造成设计中所有减数的重新命名。为了解决这个问题，像 Synplicity 公司的 Synplify 和 Synplify Pro 以及赛灵思公司的 XST 这样的综合工具，已经改变了逻辑命名的惯

例。利用这些工具，无论用户有没有对 RTL 进行变更或进行了小的变更，逻辑命名从一个综合运行到下一个综合都变得更加可以预测。按照以上的实例， 减法逻辑就为命名为 **Msub_regAB_sub0001**。通过采用在逻辑中的寄存器名，就能防止改变到其它的减数。

综合命名惯例的另一项增强措施是以本地而不是全局上下文为基础。在 RTL 出现小的变化以及设计沿着非关键路径重新综合时，这可能有所帮助。在综合之后，与非关键时序路径相关的逻辑被修改，而机器给未改变逻辑产生的实例名称保持不变。综合网表变化被本地化到网表内的已修改的或新的逻辑。最优化一向是可复制的，这是因为它们以本地逻辑为基础。例如，作为最优化过程的一个部分，Synplicity 的 Synplify/Pro 目前创建*路径组(path groups)*，它是被分别最优化的各个逻辑组。当逻辑在非关键时序路径上优化时，只有受影响的路径组被改变，以最小化对综合网表的影响。

在生成综合网表后，ISE 9.1i 中的实现工具会处理网表，并把它转换成针对特定 FPGA 架构的经布局和布线的设计。在匹配过程中会发生引导，所有的元器件以匹配元器件名称为基础被引导。如果一个元器件在目前的实现和引导设计中都有相同的名称，那么，该元器件就能被成功地引导。该元器件可能有不同的 LUT 方程式或引脚，并仍可以成功地被引导。

如果在当前实现和引导设计中的源和负载引脚是相同的，就能对网络进行引导。这消除了对两次实现之间的网络名称要保持一致的依赖。它也极大地增加了成功引导网络的机率。

如图 1 所示，例如，LUT4 的逻辑等式已经被修改。LUT4 将被引导，因为即使它的逻辑等式是不同的，它的相关名称也没有被改变。LUT4 和 LUT2 之间的布线将被引导，这是因为 LUT4 和 LUT2 之间的连通性没有被修改。

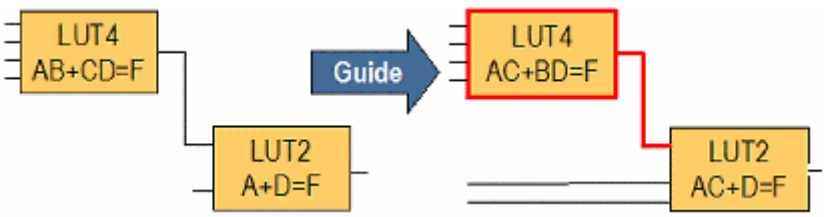


图 1．元器件和网络被成功地引导。

在元器件和网络被引导之后，新的和经修改的元器件及网络被增加到设计中。网表中没有变更的那部分在某些情况下必须被重新布局和布线，但是，这只在设计变化引起电路中另一个部分出现一个重大路径时序冲突的情况下。正是因为该原因，这是一个清除阶段。如果存在关键路径的时序冲突，就会有一些引导逻辑的移动以确定时序。在来自新的和已修改逻辑的关键路径上的清除阶段可能迫使引导逻辑的移动。这个清除阶段极大地增加成功实现一项设计的机率，该设计以稍微降低一些逻辑的保存为代价，以满足时序约束的要求。

Partition: 它如何运作？

为了做到准确无误的保存，设计工程师必须对模块实例设置一个被称为 *Partition* 的属性。这将把 Partition 与该设计的其它部分相隔离。通过隔离模块实例，其接口(跨越 Partition 边界的连通性)被保证不会在各种实现之间被修改。这使得在 Partition 内的元器件和网络可从以前的实现中被复制并被粘贴到当前的实现中。复制和粘贴这些设计信息的过程比重新实现要快得多，并且保证始终是前一次实现的准确复制。

Partition 必须在设计被综合之前就在模块实例上进行设置。然后，综合引擎将把 Partition 接口作为待优化的障碍进行处理。例如，图 2 所示的异步逻辑结构可能被合并成一个更优化的形式。如果 Partition 在这个逻辑的一个部分上已被设置，综合引擎不能对它进一步优化，这是因为它会修改 Partition 接口。类似的情况会出现在像映射这样的下游实现工具中。逻辑内部和外部的 Partition 将被完全最优化。如果关键时序路径需要这种优化出现，它将会被 Partition 边界闭塞。为了避免出现这个问题，要把寄存器添加到跨越 Partition 边界的各种信号上，或者确信这些信号的时序不是关键的。

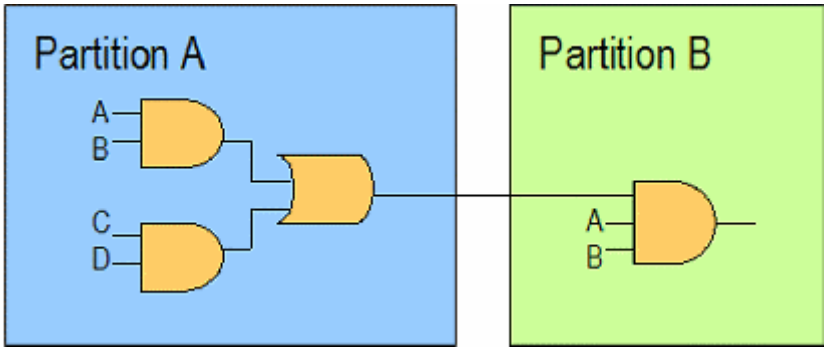


图 2．逻辑优化被 Partition 边界闭塞。

ISE 9.1i 将自动确定要对哪一个 Partitions 进行重新实现。某些类型的设计变更只会影响直接相关的 Partition。这些类型的设计变更包括源代码修改和物理约束。其它类型的设计变更将迫使所有的 Partitions 被重新实现。修改时序约束或优化设置—如映射努力级别—将影响整个设计。

在缺省条件下，Partition将促成实现工具保存从综合网表到已布线设计的所有实现数据。在某些情况下，它可能需要让像布线器这样的实现工具能修改某一个已保存的Partition，同时保存它的布局。Partition属性,即众所周知的*保存(preserve)*，能实现这一级别的控制。Preserve可以被设置到综合、布局、布线或继承上。继承属性将采用与Partition的父亲相同的特性。综合属性只保存综合网表；所有实现的其它方面可能会被修改。布局属性通过布局保存已综合的网表。此外，输出也可能被修改。布线属性将保证包括布线在内的所有实现数据得到保存。在 Synplify Pro Partition的流程中，由Partition流程保存的信息的级别(综合；布局与综合；或布线、布局与综合)作为一种属性在tcl 文件中有详细的说明，然后，再通过ISE布局和布线工具读出。

通过放宽要保存的信息级别，一些 Partition 的实现可以被修改。实现工具将尽可能多地保存 Partition，同时，仍能满足时序约束的要求，而且还能成功的对设计进行布局和布线。 要促使 Partition 在没有任何保存的情况下完全地被重新实现，就要使用 Partition 的"rerun"命令。

采用 SmartGuide

SmartGuide 对于像修改逻辑等式这样的小逻辑变更最为有用。像增加新的模块或实例这样大的变更，将会影响设计层次，并减少从以前的实现中成功匹配元器件的可能性。与 SmartGuide 能够很好配合的变更类型有：

- 在一或两个模块中的小逻辑变更(不到 10%)；
- 移动引脚位置；
- 改变一个元器件的属性；
- 改变一个时序约束；

SmartGuide 仅仅是在映射实现过程中开启的一个选项。SmartGuide 自动地使用以时序约束为基础直接实现的映射时序算法。为了在 ISE Project Navigator 软件工具中启动 SmartGuide，可从顶层菜单中选择 **Source > Use SmartGuide**，或者右击顶层模块并选择 **Use SmartGuide**，如图 3 所示。

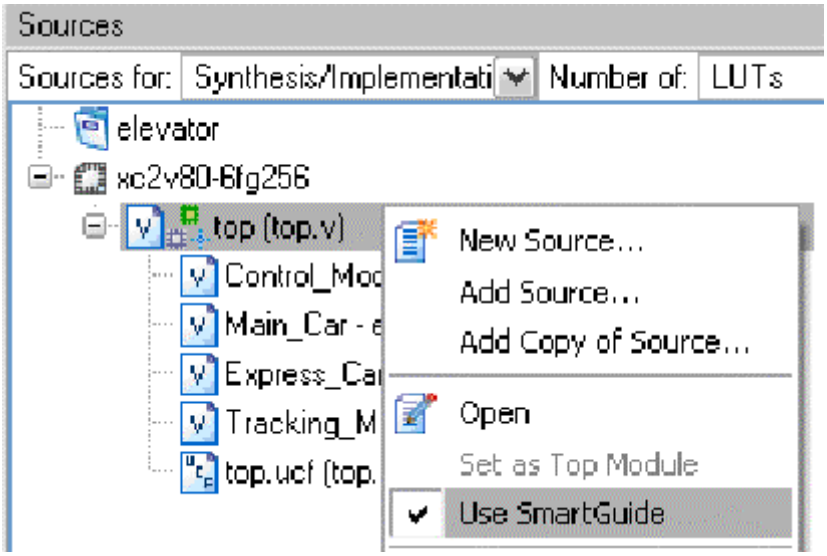


图3．在 Project Navigator 软件工具中开启 SmartGuide。

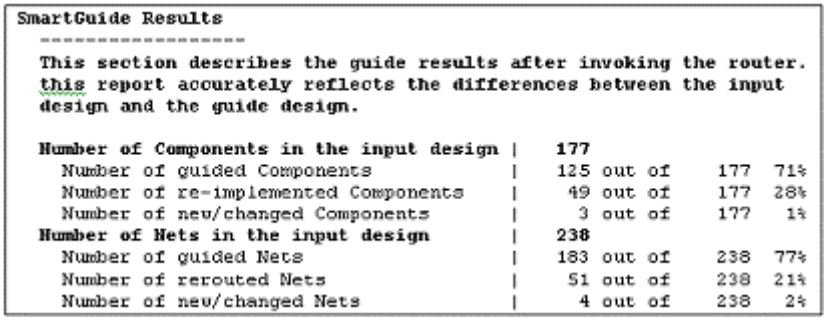
要在 tcl 接口上开启 SmartGuide，需要调用如下命令:

% project set "Use Smartguide" True

用批处理命令使用 SmartGuide，要采用这种语法:

map "smartguide <placed_and_routed.ncd>
par "smartguide <placed_and_routed.ncd>

在设计被实现后，映射和标准报告将表明有多少设计被成功地引导。在以下所示的报告文件中，有 71%的元器件及 77%的网络被引导。



```
SmartGuide Results
  ------------------
  This section describes the guide results after invoking the router.
  this report accurately reflects the differences between the input
  design and the guide design.

  Number of Components in the input design |   177
    Number of guided Components           |   125 out of    177  71%
    Number of re-implemented Components   |    49 out of    177  28%
    Number of new/changed Components      |     3 out of    177   1%
  Number of Nets in the input design      |   238
    Number of guided Nets                 |   183 out of    238  77%
    Number of rerouted Nets               |    51 out of    238  21%
    Number of new/changed Nets            |     4 out of    238   2%
```

有关元器件和网络的详细信息被存储在引导报告文件(GRF)。该文件列出了被重新实现的新文件和网络的名称。

大约在 10 次的引导实现后，建议无需引导就进行重新实现以最优化整个的设计。这将使得以前已被引导的逻辑与新的或已修改过的逻辑之间得到最优化。

采用 Partitions

有若干策略可用于决定设计中什么逻辑模块成为待 Partition 的最佳候选模块。如果缩短运行时间是主要的目标，要把设计分成具有类似数量逻辑的 4 到 10 个之间的 Partitions。如果其中的一个 Partitions 被修改，其它的将被保存。因此，保存数量与设计中 Partitions 的数量是成比例的。另一个策略是当难以满足时序约束时实例创建一个 Partition。一旦针对这个 Partition 的时序约束得到满足，即使在该 Partition 外部的逻辑被修改时，它也会被保存起来。

为了向设计添加更多的 Partitions，存在一个逐渐减小的返回点。Partition 接口是最优化的障碍。如果只能通过对 Partition 接口最优化来解决一个关键路径或包装问题， 那么，应该从设计中把那个 Partition 消除。在 Partition 接口上创建寄存器将排除出现时序或包装问题的可能性。

XST 和 Synplify Pro 两者都可被用详细说明 RTL Partitions。

在 Partition 流程中使用 XST

如图 4 所示，要采用 XST 综合工具在 ISE Project Navigator 中创建 Partitions，右击 Sources 列表中的实例并选择 New Partition。从这个菜单可以修改其它的 Partition 属性，如保存。

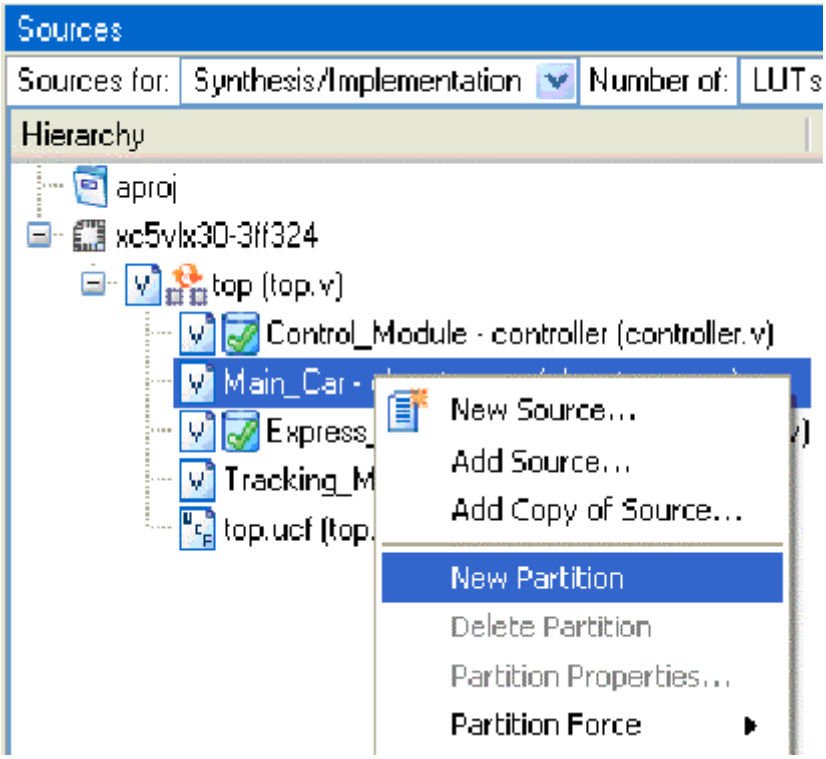


图 4．在 ISE Project Navigator 软件工具 (XST 流程)中创建 Partitions。

在 tcl 接口中创建 Partition，要使用这条命令:

% partition new

Partitions 不能用批处理命令创建，因为他们需要在设计中的特定逻辑模块上设置一种属性。

实现 FPGA 设计(XST, ngdbuild, map, par)的个别应用生成关于哪一个 Partitions 在它们各自的报告文件中被保存和实现的信息。例如，在这个 XST 综合报告中，一些 Partitions 被保存，而其它的被重新实现。每一个已实现的 Partition 都有一个原因解释它为什么没有被保存。

```
Partition Implementation Status
-------------------------------
  Preserved Partitions:
     Partition "/top"
     Partition "/top/Control_Module"
  Implemented Partitions:
     Partition "/top/Express_Car":
        HDL source file(s) were modified.
     Partition "/top/Main_Car":
        There was no implementation for this Partition.
```

在 9.1i 版本 ISE 中，对时序约束或命令行变更做出的修改—如努力级别—将迫使所有的 Partitions 被重新实现。

在 Partition 流程中采用 Synplify Pro

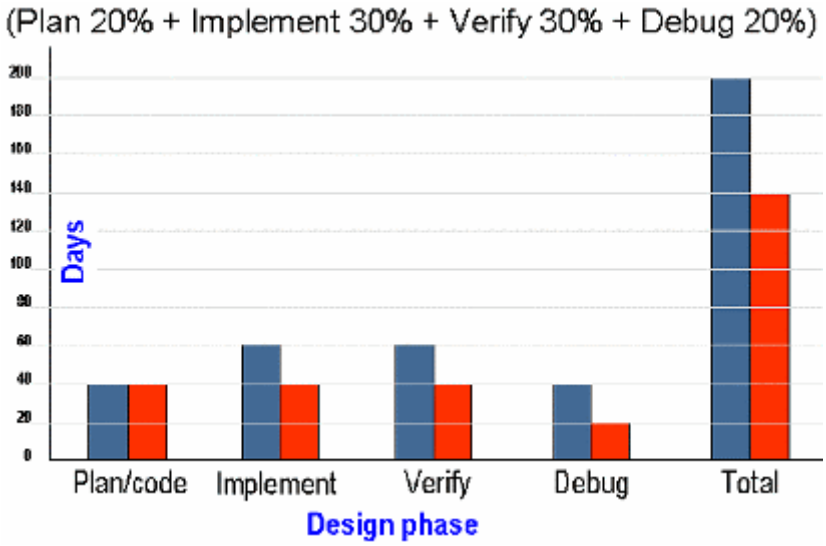
在 Synplify Pro Partition 流程的情况下，用户在运行综合之前，指定 RTL 模块/子模块(Partitions)作为 Synplify Pro 中的编译点。在整个设计中运行最初的布局和布线之后，该工具检测哪些模块/子模块已经发生了改变，并且能利用这一变化对所选择的任意模块执行后续的增量布局和布线，与此同时，使其它已布局/布线的模块保持不动。在已传递到布局和布线的 tcl 文件中，有可能详细说明布局或布局*和*布线是否被保存(缺省是布局和布线)。

Synplify 把时间戳写入到 EDIF，它可用于确定每一个编辑点自上一次的布局和布线运行后是否发生改变。一份 tcl 文件也由 Synplify 写入，从而为赛灵思布局和布线定义分区。当用户指定”syn_hier=locked,physical”时， Synplify 把额外属性写入 EDIF 之中，要观察的内容包括该模块最近被综合时的时间戳。如果从上一次运行开始模块一直没有发生变化，旧的时间戳被存入 EDIF 之中。每个标有“locked, physical”的编辑点将含有这样一个时间戳。赛灵思 ISE 9.1i 拾取该时间戳，与以前的布局和布线时间戳进行比较，并且只在发生改变的模块上进行增量布局和布线。

总结和优点

一般说来，保存一项设计要比重新实现一项设计更快。因此，如果大部分设计能通过采用 Partitions 或 SmartGuide 进行保存，实现这项设计所需的时间将会更少。在用于测试这些技术的广泛的工具套件中平均运行时间的改善，比最初的实现要快 2.5 倍。例如，如果一项设计要花 3 个小时实现，那么，利用设计保存的实现所需时间就要少 1 个小时。在一些最佳的情形下，运行时间比最初实现要快到 6 倍之多。

设计保存的另一个优点是减少了验证。如果一个设计模块被严格地保存，那么，那部分设计在设计修改后就不需要再重新验证。因为已保存的实现与前一次的实现是完全相同的，布局、布线和时序是相同的，这就使得重新验证没有必要了。



5. Design cycle reduction for 200 day FPGA project using preservation techniques.

图5．采用设计保存技术把一个200 天的FPGA 项目设计周期缩短的情况。

要花大约 200 天来实现的 FPGA 设计通常涉及设计定义、实现、验证和调试。分配到每个阶段的预计时间如图 5 所示。采用设计保存技术，就有可能使解决问题所花费的时间减少几个月。这是基于这样的假定：FPGA 的实现时间是几个小时，因此，运行时间可能被减少。它还假定该设计采用基于模块的验证测试基准，因此，在该设计内已保存的模块不需要被重新验证。如果一项设计采用了这种技术，采用 SmartCompile 技术获得解决方案所花费的时间就被大幅缩短。

Eric Shiflet*是赛灵思公司从事部分重配置和增量设计软件产品的技术市场工程师。他在这些软件产品的职责包括：产品定义、市场营销、新产品展示和支持。Eric 于1998 年加入赛灵思公司。在入职赛灵思前，**Eric** 曾受雇于 Altera。他有超过10 年在 FPGA行业的经验，除了各种营销职务外，还从事过产品开发和现场应用的工作。Eric拥有乔治亚理工学院电子工程学士学位。您可通过邮址eric.shiflet@xilinx.com与他取得联系。*

Kate Kelley*是赛灵思设计软件部的高级技术市场工程师。她主要负责ISE软件中的 SmartGuide。Kate 于1996 加入赛灵*

*思，并且拥有超过 18 年的硬件和软件设计经验。这些年大部分时间从事赛灵思FPGA的引导工作。在入职赛灵思前，**Kate** 曾在Wandel、Goltermann和IBM任职。Kate拥有罗彻斯特理工学院计算机工程学士学位。你可通过邮址kate.kelley@xilinx.com与她取得联系。*