**静态时序分析圣经翻译计划——第一章：绪论**

**[赵俊军](https://www.zhihu.com/people/zhao-jun-jun-19%22%20%5Ct%20%22_blank)**

[**​**](https://www.zhihu.com/question/48510028)

上海交通大学 电子与通信工程硕士在读

327 人赞同了该文章

众所周知，静态时序分析是IC工程师必备知识点，也是秋招中笔试面试的高频考点。
网上不乏优秀视频课，如V3学院尤老师、小梅哥、IC创新学院邸志雄老师的课。
《Static Timing Analysis for Nanometer Designs:A Practical Approach》，可堪称静态时序分析的“圣经”吧！
由于网上找不到此书的翻译，因此笔者决定开坑进行对“圣经”的翻译~！希望可以在加深对静态时序分析的学习同时，完成这一本书的翻译方便大家以后更好地从中学习吧。
有两点需要申明：
​ ● 笔者目前研二，只能利用科研之余推进翻译进度，本书共计588页，预计半年到一年内翻译完成吧~~
​ ● 笔者仅对书中原文语言进行翻译，不会添加个人见解，但求阅读通顺吧！



本章节概述了纳米级（nanometer）设计下的静态时序仿真过程
本章节解决了如下问题：
​ ● 什么是静态时序分析（static timing analysis）？
​ ● 噪声（noise）与串扰（crosstalk）会带来什么影响 ？
​ ● 如何使用静态时序分析？
​ ● 在整个设计流程中的哪个阶段会应用静态时序分析？

**1.1 纳米级设计**

在半导体器件中，常使用金属互连线来连接电路中的不同部分，从而实现设计。随着工艺技术的发展，这些互连线逐渐开始影响设计的性能。对于深亚微米或者纳米级别的工艺技术，互连线间的耦合效应会带来噪声与串扰，而这两者都会限制设计的运行速度。虽然噪声与串扰带来的影响在老一代的工艺技术下是可以忽略不计的，但在如今纳米级别下已经不容忽视了。因此，不论是物理设计还是设计验证都应考虑到噪声与串扰的影响。

**1.2 什么是静态时序分析？**

静态时序分析（简称STA）是用来验证数字设计时序的技术之一，另外一种验证时序的方法是时序仿真，时序仿真可以同时验证功能和时序。“时序分析”这个术语就是用来指代“静态时序分析“或”时序仿真“这两种方法之一，简单来说，时序分析的目的就是为了解决设计中的各种时序问题。

STA被称为静态的原因是其对于设计的分析是静态地执行的，并不依赖于施加在输入端口上的激励。相比之下，时序仿真则可以被视作动态地执行对设计的分析，具体过程描述如下：施加一组激励，观察在这组激励下电路行为是否符合要求，然后换一组激励再重复以上过程，以此类推。

给定了一个设计、输入时钟以及外部环境，STA的目的就是验证这个设计是否能够运行在预期的速度，即这个设计可以安全地运行在给定的时钟频率下且没有时序违例。

图1-1展示了STA的基本功能：

图1-1

上图中DUA即为待分析的设计。时序检查是指建立时间与保持时间检查：建立时间检查是用来保证数据可以在给定时钟周期内到达触发器；保持时间检查是用来保证数据在被触发器采样后还能保持一定时间，即保证触发器不要漏采数据。这些时序检查的目的都是为了保证触发器可以发送并且采样到正确的数据。

STA更重要的意义在于：整个设计只需要被分析一次，就可以对所有情况下设计中的全部路径进行所需的时序检查。因此，STA是能够被用来验证设计时序的一种完全且详尽的方法。

DUA通常使用硬件描述语言，例如VHDL或者Verilog HDL；外部环境，包括时钟定义，通常使用SDC或等价格式进行描述；SDC是一种时序约束的规范语言；而时序报告通常以ASCII格式呈现，一般报告中会有许多列，每一列都会显示路径延时的一个属性。

**1.3 为何使用静态时序分析？**

STA是一种可以验证设计中所有时序要求的详尽方法，而其他时序分析方法例如时序仿真则只能验证到被当前激励执行到的那部分时序路径。基于时序仿真的验证完备性取决于施加激励的完备性。如果使用时序仿真来验证一个千万门级别的设计，速度将会非常慢，并且实际上也无法充分验证。因此，想要基于时序仿真的方法来进行详尽的时序验证是非常困难的。

相比之下，STA则提供了一种更快更简单的方法去分析并检查设计中的全部时序路径。鉴于如今的ASIC设计规模已达千万门级别，STA已经成为了详尽地验证设计时序的必要方法。

**串扰与噪声**

设计的功能和性能会受到噪声的影响，引起噪声的主要原因有：与其他信号的串扰、主要输入端口的噪声、电源等。由于噪声会限制设计所能运行的最高频率，并且也可能导致功能错误，因此一个设计必须保证有足够的鲁棒性，即这个设计可以在原有额定性能的基础上抵御一定的噪声。

基于逻辑仿真的验证是无法处理由串扰、噪声以及片上变化（on-chip variations）所带来的的影响的。

本书中所描述的分析方法不仅包括了传统时序分析技术，还包括了能够验证设计中噪声问题的噪声分析。

**1.4 设计流程**

本节主要介绍了本书其余部分使用的CMOS数字设计流程，同时也简要说明了其在FPGA和异步设计中的适用性。

**1.4.1 CMOS数字设计**

在CMOS数字设计流程中，STA会在实现的各个阶段里被使用到。图1-2展示了一个基本的流程：

图1-2

STA很少在RTL级完成，因为在这一抽象层级上，验证设计的功能更为重要，而非时序。 同样，由于块（block）的描述处于行为级，因此时序信息也并非都是可用的。 一旦将RTL级的设计综合到门级，就可以使用STA来验证设计的时序。 STA也可以在执行逻辑优化之前运行，其目标是确定最差或关键的时序路径。 可以在逻辑优化后重新运行STA，以查看是否还有剩余的时序违例路径需要优化，或者确定关键路径。

在物理设计的一开始，时钟树被认为是理想的，即它们具有零延迟。 一旦物理设计开始并且建立了时钟树之后，就可以执行STA来再次检查时序。 实际上，在物理设计过程中，可以在每一步都执行STA以确定最差的路径。

在物理实现中，逻辑单元通过金属互连走线连接。 金属走线的寄生RC（电阻和电容）会影响通过这些走线的信号路径延迟。 在典型的纳米级设计中，大部分延迟和功耗都是由互连线带来的寄生因素所导致的。 因此，对设计的任何分析都应评估互连线对性能（速度，功耗等）的影响。 综上所述，信号走线之间的耦合会导致噪声，并且设计验证必须考虑到噪声对性能的影响。

在逻辑设计阶段，由于没有与布局有关的物理信息，因此可以假设互连线是理想状态的，此阶段会更关注查看导致最差路径的逻辑。 在这个阶段使用的另一种技术是采用线负载模型（wireload model）来估算互连线的长度， 线负载模型会基于逻辑单元的扇出提供一个估计的RC值。

在完成走线的布线之前，设计实现工具会使用布线距离的估算值来获得该路线的寄生RC值。 由于布线尚未完成，因此该阶段称为全局布线（global route）阶段，以将其与最终布线（final route）阶段区分开来。 在物理设计的全局布线阶段，简化的布线用于估计布线长度，而对布线的估计用于确定计算走线延迟所需的电阻和电容值。 在此阶段中，无法考虑耦合效应带来的影响。在实际精细的布线完成后，就可以提取实际的RC值，并且可以分析耦合效应带来的影响。 但是，物理设计工具仍可以使用近似值来帮助缩短计算RC值的运行时间。

提取工具用于从布线设计中提取详细的寄生参数（RC值），这样的提取工具一般具有以下选项：在迭代优化期间以较少的运行时间和较低精确度的RC值来获取寄生参数，以及在最终验证期间以较长的运行时间来提取非常精确的RC值。

总结一下，可以根据以下条件在门级网表上执行静态时序分析：

​ 1.互连线的建模方式：理想互连线，线负载模型，具有近似RC值的全局布线以及具有精确RC值的实际布线。

​ 2.时钟的建模方式：理想时钟（零延迟）或是传播时钟（实际延迟）。

​ 3.是否考虑信号之间的耦合效应以及是否分析串扰噪声。

图1-2似乎暗示着STA是在实现步骤之外完成的，即STA是在综合、逻辑优化和物理设计步骤中的每个步骤之后完成的。 实际上，这些步骤中的每一个都在其功能范围内集成有STA。 例如，逻辑优化步骤中的时序分析引擎可用于识别优化器需要处理的关键路径。 同样，布局工具中集成的时序分析引擎可用于在布局逐步进行过程中保持设计的时序。

**1.4.2 FPGA设计**

STA的基本流程在FPGA中仍然有效，即使FPGA中的布线受限于通道，提取寄生参数和执行STA的机制也与CMOS数字设计流程相同。 例如，可以在假设互连为理想状态的情况下执行STA，或使用线负载模型，在时钟树为理想状态或真实状态的情况下执行STA，或者对寄生参数情况采用全局布线或真实布线来执行STA。

**1.4.3 异步设计**

STA的原理也适用于异步设计，但在异步设计中会更加关注从一个信号到另一个信号的时序，而不是进行有可能不存在的建立时间与保持时间检查。 因此，异步设计中的时序检查一般是点到点时序检查或偏斜检查。 用于分析由耦合效应引起的毛刺的噪声分析适用于任何设计，包括同步设计与异步设计。同样，考虑到耦合响应对时序带来影响的噪声分析，对于异步设计也同样有效。

**1.5 不同阶段的静态时序分析**

在逻辑级（未进行物理设计的门级），STA可采用以下模型：

​ 1.理想的互连线或者基于线负载模型的互连线

​ 2.带有延迟和抖动估计值的理想时钟

在物理设计阶段，除了上述模型，STA还可采用以下模型：

​ 1. 具有近似估计值的全局布线的互连线、具有近似寄生参数提取值的实际布线的互连线、具有可以签收（signoff）精度寄生参数提取值的实际布线的互连线

​ 2.实际的时钟树

​ 3.包括串扰的影响或者不包括串扰的影响

**1.6 静态时序分析的局限性**

虽然时序分析和噪声分析在所有可能的情况下都可以很好地分析设计中的时序问题，但在最新的技术中仍然无法完全使用STA替代仿真， 这是因为时序验证的某些方面还无法完全被STA捕获并得到验证。

静态时序分析的局限性包括以下几点：

​ 1. 复位顺序：检查所有触发器在异步或同步复位后是否都复位为所需的逻辑值，这是无法使用静态时序分析来检查的。 芯片可能不会退出复位状态。 这是因为某些声明（例如信号的初始值）没有被综合，仅在仿真过程中被验证。

​ 2. 未知态X的处理：STA技术仅处理逻辑0和逻辑1（或高电平/低电平）的逻辑域，或者是上升沿和下降沿的逻辑域。 设计中的未知态X导致不确定的值在整个设计中传播，这也是无法使用STA进行检查。 即使STA内的噪声分析可以分析整个设计中的毛刺，但作为纳米级设计中基于仿真的时序验证的一部分，毛刺分析和传播的范围也与对未知态X的处理大为不同。

​ 3. PLL设置：PLL的配置可能未被正确加载或设置。

​ 4. 跨异步时钟域：STA不检查是否使用了正确的时钟同步器，需要其他工具来确保在任何跨异步时钟域的地方都有正确的时钟同步器。

​ 5. IO接口时序：可能仅根据STA约束无法规定IO接口要求。例如，设计人员可能使用SDRAM仿真模型为DDR接口选择详细的电路级仿真。仿真是为了确保可以以足够的余量读取和写入存储器，并且在必要时可以控制DLL（如果有）来对齐信号。

​ 6. 模拟模块和数字模块之间的接口：由于STA不处理模拟模块，因此验证方法需要确保这两种类型的模块之间的连接正确。

​ 7. 伪路径（false path）：静态时序分析会验证通过逻辑路径的时序是否满足所有约束，如果通过逻辑路径的时序不符合要求的规范，则标记违例。在许多情况下，即使逻辑可能永远无法传播通过该路径，STA也会将该逻辑路径标记为时序违例路径。 当系统应用程序从不使用此类路径时，或者在时序违例路径的敏感列表中使用了互斥的条件时，可能会发生这种情况。 这种时序路径被称为伪路径，因为这种时序路径实际上不会被执行。当在设计中指定了正确的时序约束（包括伪路径和多周期路径约束）时，STA结果的质量会更好。 在大多数情况下，设计人员可以利用设计的固有知识并指定约束条件，以便在STA期间消除伪路径。

​ 8. FIFO指针不同步：当两个预期要同步的有限状态机实际上不同步时，STA无法检测到该问题。在功能仿真过程中，两个有限状态机可能始终保持同步变化。但是，在考虑了延迟之后，一个有限状态机有可能与另一个就不同步了，这很可能是因为一个有限状态机比另一个更早退出复位状态，而STA无法检测到这种情况。

​ 9. 时钟同步逻辑：STA无法检测到时钟生成逻辑与时钟定义不匹配的问题。 STA会假设时钟生成器将提供时钟定义中指定的波形。 对时钟生成器逻辑的优化可能很糟糕，比如会导致在未适当约束的路径之一上插入较大的延迟，又或者，添加的逻辑改变了时钟的占空比。而STA无法检测到这些潜在情况中的任何一个。

​ 10. 跨时钟周期的功能行为：STA无法建模或仿真跨时钟周期变化的功能行为。

尽管存在诸如此类的问题，STA依然适合被广泛用于验证设计的时序，而时序仿真可作为备用方法来检查极端情况，并且能够更简单地验证设计的功能正确。

**1.7 功耗考虑**

功耗是设计实现中的重要考虑因素，大多数设计需要在电路板和系统的功耗预算内运行。若需要符合标准并且考虑到芯片运行在电路板和系统上的热预算，可能还会出现功耗方面的考虑。对总功率（total power）和待机功率（standby power）通常存在独立的限制，待机功率限制通常适用于手持式或电池供电的设备。

在大多数实际设计中，功耗和时序通常是密不可分的。设计人员希望使用更快（或更高速度）的单元来满足速度方面的考虑，但可能会受到功耗的限制。 在选择工艺技术和单元库时，功耗是一个重要的考虑因素。

**1.8 可靠性考虑**

设计实现必须满足可靠性要求。如1.4.1节中所述，金属互连走线具有寄生RC值，从而限制了设计的性能。除寄生效应外，在设计金属互连走线宽度时也应当要考虑可靠性因素。例如，高速时钟信号需要足够宽，以满足诸如电迁移之类的可靠性考虑。

**1.9 本书大纲**

尽管表面上静态时序分析似乎是一个非常简单的概念，但该分析背后有很多背景知识。基本概念的范围从准确地表示单元延迟到计算具有最小悲观度的最坏路径延迟。计算单元延迟、组合逻辑块的延时、时钟关系、多个时钟域和门控时钟的概念构成了静态时序分析的重要基础，为设计编写正确的SDC确实是一个挑战。

这本书是按照自底向上（bottom-up）的顺序编写的，即首先介绍简单的概念，在随后的章节中介绍更高级的主题。

​ ●本书首先介绍了准确计算单元延迟（第3章）。

​ ●估计或计算精确的互连延迟及其有效表示方法是第4章的主题。

​ ●在第5章中讨论如何计算由单元和互连线组成的路径延迟。

​ ●信号完整性（即信号在相邻网络上的相互影响）及其对路径延迟的影响是第6章的主题。

​ ●第7章介绍了时钟定义和路径例外来准确表示DUA的环境。

​ ●第8章介绍了在STA中执行时序检查的详细信息。

​ ●第9章将讨论跨各种接口的IO时序建模。

​ ●最后，第10章将介绍高级时序检查，如片上变化（on-chip variation）、时钟门控（clock gating）检查，电源管理和统计时序分析。

​ ●附录提供了SDC（用于表示时序约束），SDF（用于表示单元和网络延迟）和SPEF（用于表示寄生参数）的详细说明。

第7章至第10章介绍的是STA验证的核心，前面的章节为更好地了解STA打下了扎实的基础。