



电子产品架构设计、性能仿真分析 系统解决方案 - VisualSim®

Mirabilis Design公司是业界领先的复杂电子系统与嵌入式软件性能分析和架构设计工具提供商，其产品VisualSim®是一款基于加州伯克利大学Ptolemy仿真内核的软件环境，能够同时支持数字、模拟、DSP及嵌入式软件等系统部件的建模与仿真。VisualSim专有的快速虚拟原型可使设计团队在项目开发早期即对系统的架构、性能、成本、功耗等指标进行研究，并对设计方案进行评价。VisualSim软件能够发布系统模型的Applet版本，这些模型包含了设计规约（Specification），并能在普通的Web浏览器或Word文档中查看和执行。借助Architecture Generators 和SmartBlocks等VisualSim的革命性技术，能够快速创建系统模型，并执行系统层次软硬件体系结构的“*What if*”的分析。

VisualSim® 仿真器介绍

可执行的模型的运行通常可以称为仿真，特别是当该模型与其所描述的系统有明显区别时（例如用 C 语言模型来仿真一个机械运动过程）。但是对于很多电子系统来说，用于仿真的模型可能会逐渐变成为软件的实现。特别是对于嵌入式软件的仿真，这时系统实现本身和它的仿真模型之间的区别变得模糊不清。

可执行的模型是建立在计算模型基础上的，而计算模型是控制模型内各部件交互的物理法则的集合。尽管如此，更多时候模型是比较抽象的控制规则的集合，为设计者提供了一个建模的框架。控制部件交互的规则集合称为计算模型的语义。一个计算模型可能有多种语义，即对于一个模型的相同行为约束可以存在不同的规则集合。

计算模型的选择在很大程度上取决于待建模型的类型。在以往的仿真工具中，仿真不同类型的模型要使用不同仿真域的工具。但在 VisualSim 中，所有的计算模型都被集成到单一的系统级建模产品中：

- ◆ 对于将一个有限数据结构变换为另一个有限数据结构的纯计算性系统，使用严格强制语义的编程语言如 C, C++, Java 和 MATLAB 就足够了。VisualSim 中的表达式语言提供全面标准的数学、逻辑运算、信号处理和图像处理功能函数。VisualSim 的表达式语言支持用户扩展。
- ◆ 对于电子系统的建模，要求计算模型的语义能够处理并发、时序及事件排队。VHDL/Verilog

和 SystemC 所支持的离散事件计算模型是合适的。VisualSim 中的 DE 仿真器具有完善的离散域仿真功能并带有大量的模型库。

- ◆ 对于机械系统的建模,要求计算模型的语意支持并发及连续时间处理。MATLAB Simulink、Saber、ADS 和 VHDL-AMS 非常合适。VisualSim 提供的连续时间仿真器,即可以与有限状态机一起协同进行混合建模 (Hybrid Model),又可以与 DE 仿真器协同实现系统级的混合信号 (Mixed Signal) 仿真。
- ◆ 对于嵌入式软件和数字信号处理系统的建模,计算模型必须能够以同步状态对事件进行预先调度。SPW 所使用的同步数据流计算模型就非常合适。VisualSim 的 SDF 仿真器支持对嵌入式软件和 DSP 处理的建模,并带有功能全面的处理建模库。

以下对 VisualSim 的不同仿真器中执行的计算模型分别作简要介绍:

Continuous Time (CT)

在 CT (连续时间) 仿真器中,模块表示通过连续时间信号进行相互操作的部件。通常模块的输入和输出之间具有明确的代数或者微分关系。CT 仿真器的任务是找到一组能够满足所有关系的连续时间函数集。

CT 仿真器包含一系列可扩展的微分方程解算器,因此非常适合对那些能够以线性或非线型的代数/微分方程描述的物理系统进行建模,例如模拟电路和多数机械系统。CT 仿真器的计算模型与 MATLAB Simulink, Saber 和 VHDL-AMS 中所使用计算模型很相似,并且与 Spice 电路仿真器有非常密切的联系。

嵌入式系统中通常包含一些最好应用微分方程建模的部件,例如 MEMS 和其他的机械结构、模拟电路和微波电路。但是,这些部件通常会和系统中的一个数字电路子系统进行耦合,因为数字电路系统更适于作为控制器或是完成传感器数据的后续处理。将连续的模拟系统与离散的数字系统相结合,就是通常所说的混合信号 (Mixed Signal) 仿真。VisualSim 的 CT 仿真器完全支持与 VisualSim 的其他仿真器之间的相互操作,比如与其 DE (离散事件) 仿真器结合实现混合信号仿真建模。为了支持此类建模,CT 仿真器将离散事件均视为单位脉冲函数 (Dirac Delta Function),同时能够精确地对模拟信号进行阈值穿越检测,以产生离散事件。实际物理系统的建模通常可以分解为若干种简单模型,某种简单模型仅在某种特定的系统状态 (Regime) 下最适用作为系统模型。当系统在不同的状态之间转换时,系统建模所需要的不同简单模型之间的切换构成了形式模型 (Modal Model)。CT 仿真器与 FSM (有限状态机) 仿真器交互操作就可以创建形式模型。

Discrete Event (DE)

在 DE 仿真器中维护着一个依时间顺序排列的事件队列,即每个事件由事件值和时间戳组成。DE 模块的功能可以是对发生事件的响应处理,也可以是击发新的事件。DE 域的计算模型在数字

硬件和电信系统仿真领域应用非常普遍，在很多仿真环境，仿真语言及硬件描述语言中被采用。

DE 模型在描述硬件并发机制上是非常出色的。VisualSim 的 DE 仿真器是一个复杂而完善的离散事件仿真器。其他的 DE 仿真器通常都需要维持一个依据时间排列的全局队列用于事件排队，因此在每次插入新事件到队列中时必须搜索确定正确的插入位置，当队列中事件众多时这一处理过程的开销很大。VisualSim 的 DE 仿真器的全局事件队列采用了一种日程队列数据结构。这个日程队列可以看作是量化时间作为散列函数的哈希表 (Hash Table)，因此向队列中插入事件或从队列中删除事件所需要的时间是固定的，与队列中事件数量的多少无关。

另外，VisualSim 的 DE 仿真器对同时发生的事件会给出了确定性的语义，这不同于其它大多数的离散事件仿真器产品。这就意味着对于任意两个具有同一时间戳的事件，仿真器对二者的处理顺序能够依据模型结构进行明确的判断。而之所以能够实现这一推断，是因为通过对模型的图形结构的分析可以直接推断出数据驱动的传播方向。与之相比，VDHL 则是采用 delta 时间来处理类似情况的。

Finite State Machine (FSM)

FSM 仿真器与 VisualSim 的其它仿真器有根本的区别。这个仿真器中的实体表示的不是模块，而是状态和代表状态转换的连接关系。模型将严格按照定义的状态转换顺序执行。FSM 仿真器利用 VisualSim 表达式语言生成保护信号量 (Guard)，以确定何时允许发生状态转换。

有限状态机模型在对嵌入式系统的控制逻辑的建模方面非常优秀。VisualSim 的 FSM 仿真器可以与其他仿真器进行层次化组合，构成相应域的状态图，例如与具有并发计算模型的仿真器一起构成支持并发语意的有限状态机。当 FSM 仿真器与 CT 仿真器结合时，形成用于混合建模 (Hybrid Model) 的形式模型。当 FSM 仿真器与 SDF 仿真器结合时则产生类似状态图的模型。

Synchronous Data Flow (SDF)

SDF 仿真器是处理数据流操作的计算模型，因此在信号处理领域此类模型应用非常广泛。数据流模型建立一组处理过程用于实现事件触发序列，而同步数据流模型则尤其适用于在执行前预先确定系统模型不存在死锁 (Deadlock) 及数据流速率边界不一致的情况。

在 SDF 仿真域中，各模块间的执行顺序在仿真执行之前已经确定下来，即事件触发调度、并行或顺序执行在运行时刻前已经计算确定下来，因此具有最小的仿真器执行开销。同步数据流模型非常适合于对不需要复杂控制流程的数据流进行建模，因此用于描述软件或硬件的数据流处理形式规范十分有效。

SystemC

SystemC 是事件驱动的仿真器，它在标准 C++ 基础上提供了面向硬件的模型结构及实现类

库。**SystemC** 可以将设计者对系统的设计与验证的概念转变为其实现。它提供了一个通用的建模平台，允许采用 **C++** 技术迅速开发系统级模型。它同时也提供一个开发系统级工具的平台。

SystemC 库除了明确定义用于软件编程的数据类型之外，同时提供一组数据类型用于表达系统建模时的多种数据形态，包括二值和四值信号表达、任意宽度信号矢量、定点数表达等。**SystemC** 定义的模块（**Module**）支持对结构、接口和数据通道的表达，并且库中内置一组广泛应用的建模要素（**Primitive**），如信号和 **FIFO** 等。

一个 **SystemC** 系统包括一个或多个模块，模块具有描述系统结构的能力。每个模块可以包含过程、端口、内部数据、通道以及其它模块的实例。所有的过程都是并发的，可以用于模块功能的建模。端口用于模块之间的通信。内部数据和通道用于过程间的通信和维持块状态。模块的例化支持层次化结构。接口、端口、通道等结构为 **SystemC** 的系统建模和模型优化提供了很好的灵活性。