保序模块的 formal fpv 验证

赵亚雪，植　玉，梁其锋，石义军
（深圳市中兴微电子技术有限公司，广东 深圳 518054）

摘　要： 与 simulation 验证相比，formal 验证方法可以在短时间内遍历所有可能的激励，大大提高验证的效率。保序模块与时序控制以及流水线控制密切相关，设计规模较大，逻辑复杂度较高。介绍了使用 formal fpv 验证保序模块的流程，并对 JasperGold debug 结果进行了分析，采用 formal fpv 验证能提高验证效率，加快验证收敛速度。

关键词： formal；fpv；保序模块；JasperGold
中图分类号： TN402　　**文献识别码：** A　　DOI：10.16157/j.issn.0258-7998.229801

中文引用格式： 赵亚雪，植玉，梁其锋，等. 保序模块的 formal fpv 验证[J].电子技术应用，2022，48(8)：38-41，45.
英文引用格式： Zhao Yaxue，Zhi Yu，Liang Qifeng，et al. Formal FPV verification of sequence preserving module[J]. Application of Electronic Technique，2022，48(8)：38-41，45.

Formal FPV verification of sequence preserving module

Zhao Yaxue，Zhi Yu，Liang Qifeng，Shi Yijun
(Shenzhen Sanechips Technology Co.，Ltd.，Shenzhen 518054，China)

Abstract： Compared with simulation verification，the formal verification method can traverse all possible incentives in a short time，which greatly improves the efficiency of verification. The sequence preserving module is closely related to timing control and pipeline control，with large design scale and high logic complexity. This paper introduces the process of verifying the sequence preserving module using formal FPV，and analyzes the JasperGold debug results. Formal FPV verification can improve the verification efficiency and accelerate the verification convergence speed.

Key words： formal；FPV；sequence preserving module；JasperGold

0 引言

芯片验证方向经过多年的探索和积累已经有一套较为完备的验证体系[1]。其中，simulation 验证和 formal 验证是两大常用的验证方法。从对测试点的覆盖程度上来说，两者的区别在于 simulation 着眼于测试空间中的单个点，而 formal 验证可以完全覆盖输入空间，从而能在约束条件下有效证明设计的准确度[2]，formal 验证方法能在短时间内遍历所有可能的激励，从而大大提高验证效率[3]，因此近年来 formal 验证方法得到了越来越多的关注。

formal 验证工具大体可以分为两类[4]，一类是 MFV (Mainstream Formal Verification)，其具有成熟的功能，能实现高度自动化验证。另一类是 FPV(Formal Property Verification)，需要手动开发验证环境，编写 property[5]。对于逻辑较为复杂、难以调用工具自带模型的模块更倾向于选择 FPV 工具来进行验证。

保序模块用于确保处理器内部读、写访问严格按照既定的顺序处理，其与时序控制以及流水线控制密切相关，设计规模较大，逻辑复杂度较高，采用 formal fpv 工具，本文按照验证对象介绍、Design Review、验证环境搭建、验证模型编写、JasperGold debug 的流程来展开介绍。

1 验证对象简介

保序模块是我司某存储器系统中用于保证读、写访问顺序的模块，基本框图如图 1 所示，主要包括 B 指令

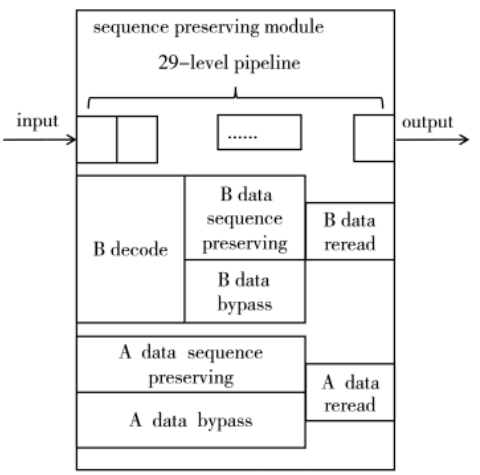


图 1　保序模块基本框图

译码、B 数据写访问缓存、B 数据保序、B 数据提前返回、A 数据保序、A 数据提前返回、重读等功能。

保序模块会对输入的 B 指令进行译码，译码后的 B 数据写访问会经过 28 级流水 buffer 缓存，在流水线上会对 B 数据写请求地址相关的访问进行保序处理，同时会判断 B 数据读与 B 数据写是否提前返回，以及重读指示信号是否产生。对于 A 数据访问来说，A 数据写访问也会经过 28 级流水 buffer 缓存，在流水线上会对 A 数据访问进行保序处理，也会判断 A 数据写数据是否提前返回。

其中，地址相关的保序需满足：读写同拍发生，认为读操作期望读取旧值；先写后读场景，认为读操作期望读取写入后的新值。

2 Design Review

常用的 formal sign-off flow 可以分为两种情况。

一种是传统 formal sign-off flow，如图 2 所示，特点是所有的断言都需要被证明。对于这种 sign-off flow，理想的 RTL 代码行数应在 1 500~3 000 范围内。在传统 Formal sign-off flow 中工具根据手动编写的断言自动提取生成 coverage，不需要再编写 cover，保序模块验证正是采用这种传统的方法。

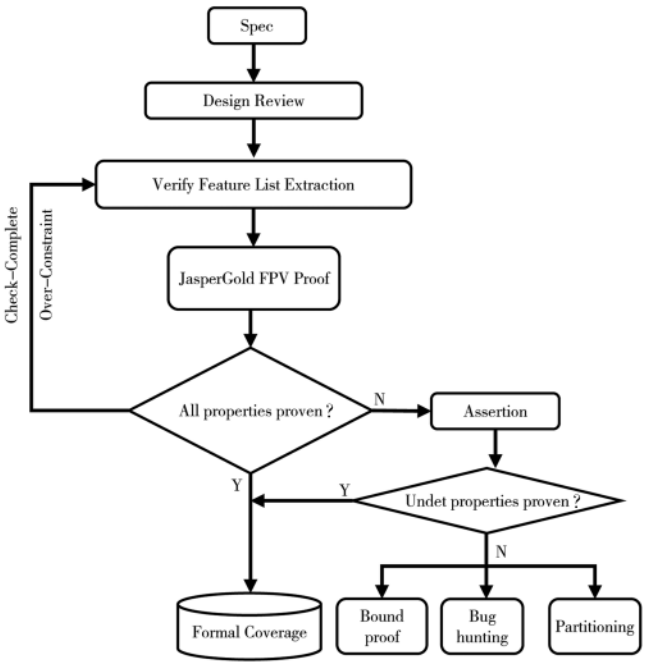


图 2 传统 formal sign-off flow

另一种是新型 formal sign-off flow，如图 3 所示，特点是允许有证不出来的断言，也就是说允许有处于 undetermined 的断言，对于证不出来的断言需要手动编写 function cover，对这种 sign-off flow 来说，理想的 RTL 代码行数应在 3 000~5 000 范围内。

通过 Design Review 可以梳理出 fpv 验证的大框架。保序模块涉及流水数较多有 29 级，且前级流水的信号

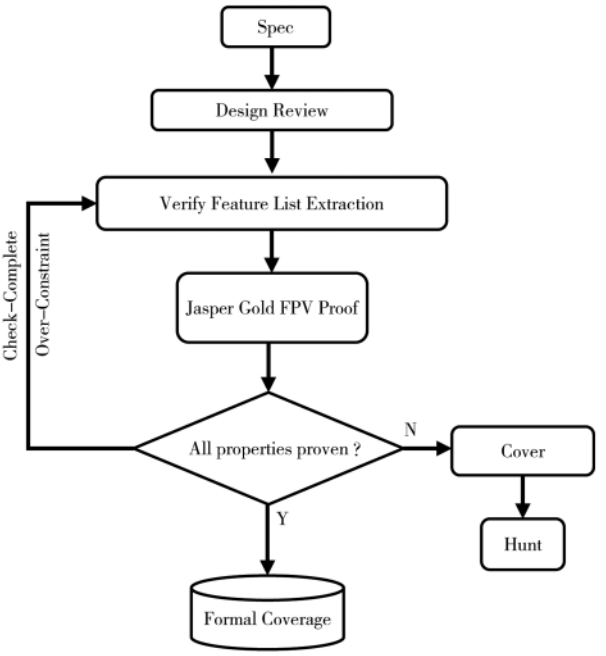


图 3 新型 formal sign-off flow

会对后级流水信号的变化产生影响，从 input-output 信号通路角度来考虑，将保序模块拆分成 6 条通路，分别对每条通路使用 formal fpv 进行验证。

3 验证环境搭建

保序模块验证平台由三部分组成，分别是 rtl_dmss、signoff、sva，其中 rtl_dmss 用来存放设计 RTL 代码，signoff 用来存放 filelist、tcl 脚本以及编译仿真过程中产生的临时文件，sva 用来存放验证模型、验证平台的环境文件。

在验证平台环境中首先定义了模块端口上的所有信号，然后将待测设计 DUV 与验证模型连接起来作为激励入口，验证平台结构如图 4 所示。

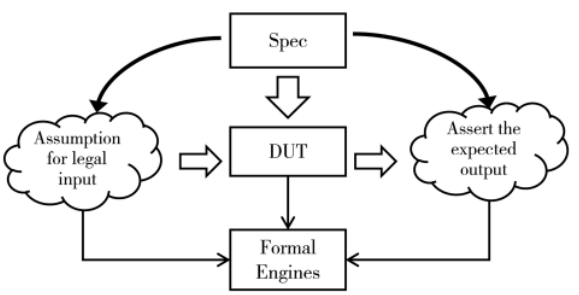


图 4 验证平台结构

在没有外部约束的情况下，formal 会穷举整个输入空间，所以为了避免出现不符合设计需求的场景，需要在验证模型中添加相应的约束。验证输出和待测设计输出的比对工作则放到了验证模型的 assertion 部分，在 assertion 部分会进行一致性比对和时序检查。

4 验证模型编写

由于保序模块涉及 29 级流水且逻辑较为复杂，fpv

工具自带的模型并不适合保序模块，需要手动搭建各条通路的验证模型。保序模块的验证模型包括模块功能模型、激励约束和断言三部分。

4.1 功能模型编写

功能模型用来模拟被测对象的功能，通过将功能模型的输出结果与待测对象的输出进行比对、检查，可以得知待测对象功能的正确性。功能模型使用 Verilog 语言来编写而不是 SystemVerilog，这是因为功能模型一定要可综合，而 SystemVerilog 有些语法不可综合。对于保序模块来说，功能模型以 cycle 为单位进行建模，描述了模块处于 29 级流水的工作情况。

4.2 激励约束编写

simulation 方法通过接口平台产生 transaction，再把 transaction 传输给参考模型和待测设计，而 formal 验证方法会对约束后的激励进行遍历，也就是说 formal 验证平台的激励来自约束条件，如图 5 所示，激励约束可以分为 legal 和 illegal 两种。

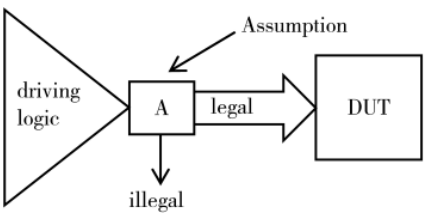


图 5 输入激励约束

工具会对所有输入信号进行全随机遍历，通过编写激励约束能保证输入信号满足待测设计的需求，而不会出现超出设计需求的场景。

值得注意的是，在编写激励约束时不要过约束，否则验证的完整性就会大打折扣。对于保序模块来说，调试初期可能存在过约束的场景，调试过程中再逐渐放开约束，保证验证的完整性和正确性。为了避免造成混乱，建议添加注释将过约束和正常约束加以区别，同时出于规范化考虑，可以给激励约束的名称添加“ASM_”前缀。

对保序模块的配置地址进行约束时，虽然配置地址可以是随机的，但在一次仿真中各个 cycle 的配置地址需要保持不变，所以也需要对配置地址加以约束。

4.3 断言编写

为了检查待测设计的准确性，需要把功能模型输出与待测设计输出进行比较，通过断言来检查两者是否匹配。断言检查流程如图 6 所示，当断言的所有状态都被分析证实后该条断言判断为 proved。

formal fpv 断言编写的原则之一是简单化。对于 bit 位较多的信号可以按一定的规则对信号进行拆分，例如在保序模块中检查输出读地址的正确性，由于读地址信号有 32×8 bit，包含 8 个通道每个通道 32 bit 地址，可以使用循环把读地址拆分成 8 份，编写断言来检查每一份读地址的正确性。

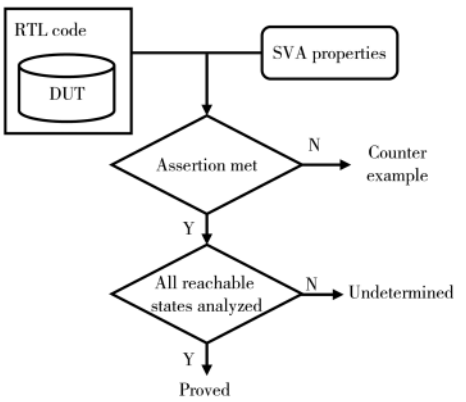


图 6 断言检查流程

出于规范化考虑，可以给断言的名称添加“AST_”前缀。为了避免断言调试出错，在复位信号有效时需要 disable 掉该断言，在保序模块中写作“disable iff(!core_sync_rst_n)”。

完备性是保序模块验证的重要衡量指标之一，通过给每条断言添加注释能方便地找出该条断言对应设计的哪些功能点，便于了解设计各个功能点是否都有断言覆盖。

5 JasperGold debug

5.1 JasperGold 工具介绍

运行 tcl 脚本启动 JasperGold 软件的 UI 界面，可以看到各条断言的仿真结果，如图 7 所示。

JasperGold 的配置、编译和仿真都是通过 tcl 命令来实现的，可以查阅相关命令的使用说明，如图 8 所示。

各条断言仿真结果可能有 prove、unreachable、undetermined 三种情况。图 7 中 assert 前打绿勾表示断言验证通过，打叉表示该断言出现反例，可以双击查看波形进一步分析。打勾和感叹号表示断言的前提条件无法满足，需要检查约束条件是否过约束，debug 分析是验证模型问题还是待测设计问题。

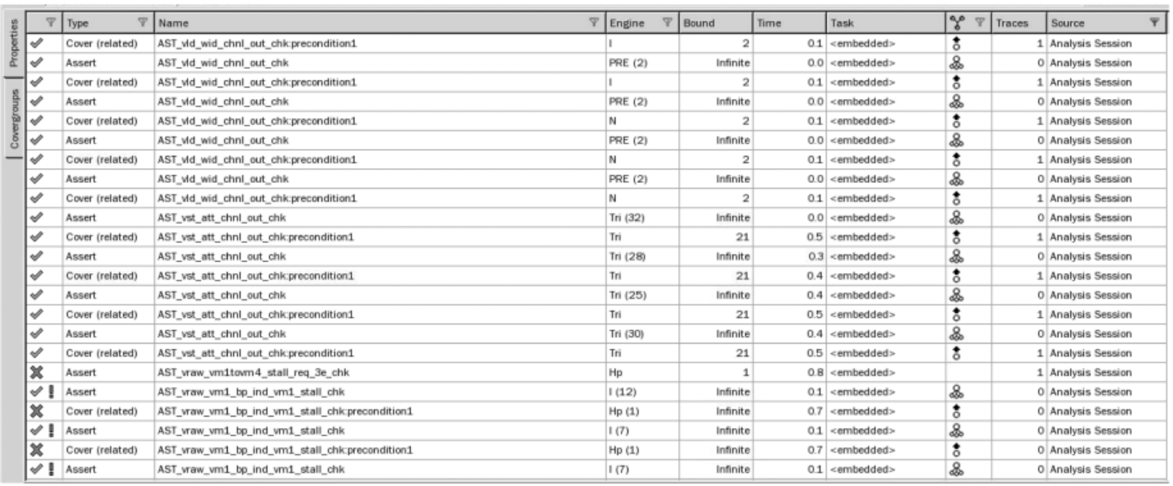
当断言出现反例时，双击失败的断言可以打开对应的波形，波形能精准定位到出现反例的时刻，如图 9 所示，深灰表示触发断言，浅灰色表示断言违例。

5.2 debug 结果分析

在验证保序模块验过程中发现了一些设计的缺陷，对这些缺陷加以整理汇总，主要有以下几类。

第一类缺陷是待测设计中某些信号定义错误，这类属于比较直观的缺陷。在断言检查时发现验证输出的 B 数据读地址与待测设计输出的 B 数据读地址不一致。定位问题发现是设计信号出现了位宽越界，养成良好的编码习惯能大大减少这种情况的出现。

第二类缺陷是待测设计某些通道的信号处理出错。保序模块读访问包含 8 个通道，断言检查时发现输出的地址有效指示信号出错。通过前向追溯问题发现待测设计某一通道的位宽处理出错，该缺陷在更上一层次的系



	Type	Name	Engine	Bound	Time	Task		Traces	Source
✓	Cover (related)	AST_vld_wid_chnl_out_chk:precondition1	I	2	0.1	<embedded>		1	Analysis Session
✓	Assert	AST_vld_wid_chnl_out_chk	PRE (2)	Infinite	0.0	<embedded>		0	Analysis Session
✓	Cover (related)	AST_vld_wid_chnl_out_chk:precondition1	I	2	0.1	<embedded>		1	Analysis Session
✓	Assert	AST_vld_wid_chnl_out_chk	PRE (2)	Infinite	0.0	<embedded>		0	Analysis Session
✓	Cover (related)	AST_vld_wid_chnl_out_chk:precondition1	N	2	0.1	<embedded>		1	Analysis Session
✓	Assert	AST_vld_wid_chnl_out_chk	PRE (2)	Infinite	0.0	<embedded>		0	Analysis Session
✓	Cover (related)	AST_vld_wid_chnl_out_chk:precondition1	N	2	0.1	<embedded>		1	Analysis Session
✓	Assert	AST_vld_wid_chnl_out_chk	PRE (2)	Infinite	0.0	<embedded>		0	Analysis Session
✓	Cover (related)	AST_vld_wid_chnl_out_chk:precondition1	N	2	0.1	<embedded>		1	Analysis Session
✓	Assert	AST_vst_att_chnl_out_chk	Tri (32)	Infinite	0.0	<embedded>		0	Analysis Session
✓	Cover (related)	AST_vst_att_chnl_out_chk:precondition1	Tri	21	0.5	<embedded>		1	Analysis Session
✓	Assert	AST_vst_att_chnl_out_chk	Tri (28)	Infinite	0.3	<embedded>		0	Analysis Session
✓	Cover (related)	AST_vst_att_chnl_out_chk:precondition1	Tri	21	0.4	<embedded>		1	Analysis Session
✓	Assert	AST_vst_att_chnl_out_chk	Tri (25)	Infinite	0.4	<embedded>		0	Analysis Session
✓	Cover (related)	AST_vst_att_chnl_out_chk:precondition1	Tri	21	0.5	<embedded>		1	Analysis Session
✓	Assert	AST_vst_att_chnl_out_chk	Tri (30)	Infinite	0.4	<embedded>		0	Analysis Session
✓	Cover (related)	AST_vst_att_chnl_out_chk:precondition1	Tri	21	0.5	<embedded>		1	Analysis Session
✗	Assert	AST_vraw_vm1tovm4_stall_req_3e_chk	Hp	1	0.8	<embedded>		1	Analysis Session
✓ !	Assert	AST_vraw_vm1_bp_ind_vm1_stall_chk	I (12)	Infinite	0.1	<embedded>		0	Analysis Session
✗	Cover (related)	AST_vraw_vm1_bp_ind_vm1_stall_chk:precondition1	Hp (1)	Infinite	0.7	<embedded>		0	Analysis Session
✓ !	Assert	AST_vraw_vm1_bp_ind_vm1_stall_chk	I (7)	Infinite	0.1	<embedded>		0	Analysis Session
✗	Cover (related)	AST_vraw_vm1_bp_ind_vm1_stall_chk:precondition1	Hp (1)	Infinite	0.7	<embedded>		0	Analysis Session
✓ !	Assert	AST_vraw_vm1_bp_ind_vm1_stall_chk	I (7)	Infinite	0.1	<embedded>		0	Analysis Session

图 7　断言仿真结果

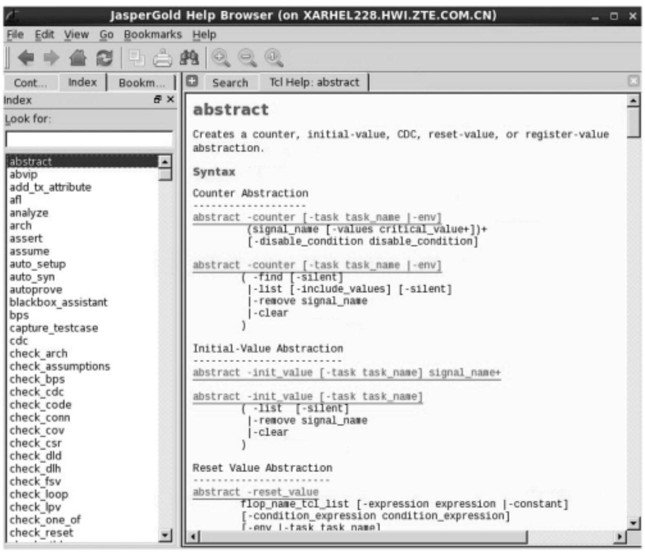


图 8　JasperGold 命令集

统验证中没有检查出来。这就要求在编写断言时，当遇到复杂的信号时可以将其拆分成多组，分别检查每组信号的时序，能迅速、精准地定位问题从而提高验证效率。

第三类缺陷是待测设计中循环处理出错。保序模块内信号的处理均受流水线控制，采用循环方法模拟流水线处理，在此过程中一些信号的赋值出错。通过断言检查发现验证输出与待测设计输出不一致，定位到写访问使能信号计算出错，进一步向前推算发现问题的源头是循环处理出错。对于这种问题链路过长的情况，如果从输出信号开始定位验证难度较大，可以通过添加辅助逻辑来缩短问题链路，从而降低问题难度。

6 结论

基于 formal fpv 的验证方法在保序模块验证中有着很不错的效果，验证共发现 8 单故障，其中一单故障发现了系统级验证遗漏的问题。formal fpv 验证能实现输入激励的全范围遍历，这给验证工作提供了极大的便利。但是，formal fpv 验证很大程度上依赖于断言编写的质量，且复杂的模块需要手动编写验证功能模型。

综合来看，对于设计相对简单的模块采用 formal fpv 验证能提高验证效率，加快验证收敛速度。

参考文献

[1] 张晓冬，巨鹏锦，濮晨，等.形式化验证在芯片研发中的应用[J].中国集成电路，2017，26(9)：38-42.

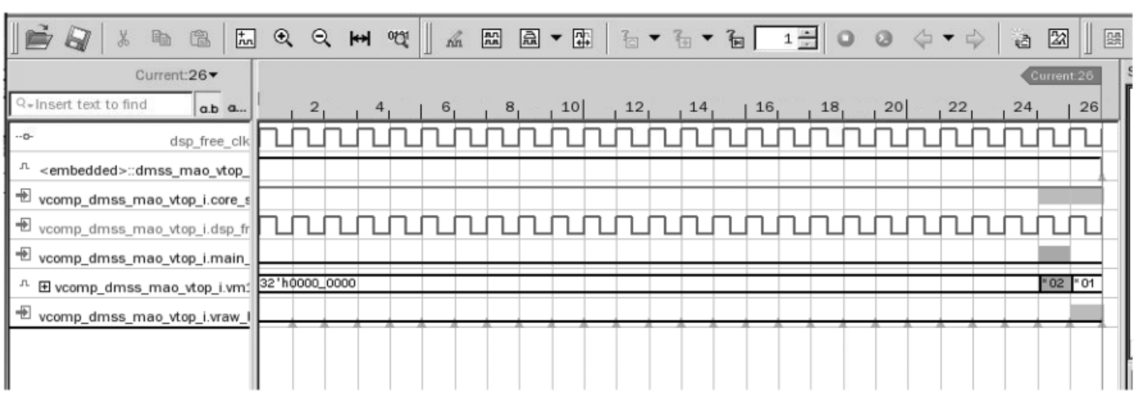


图 9　反例断言仿真波形

（下转第 45 页）

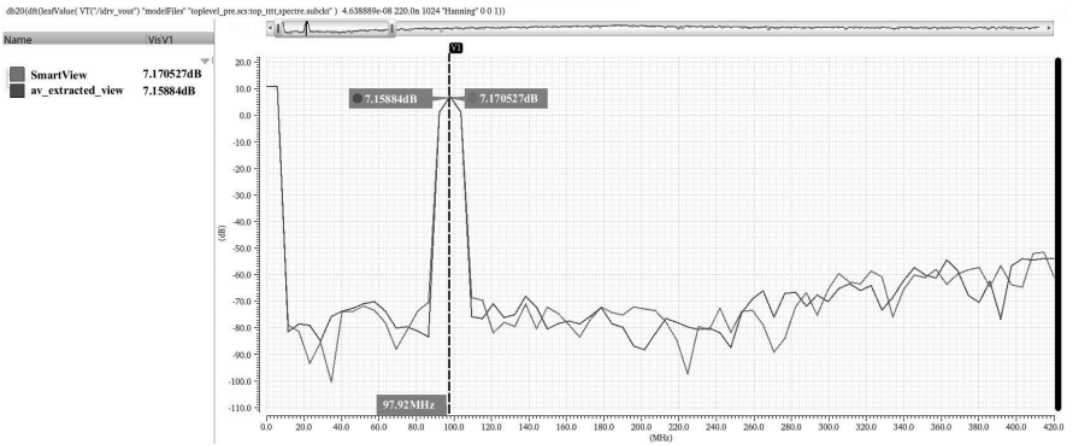


图 2 adc 电路两种网表的波形及频谱对比

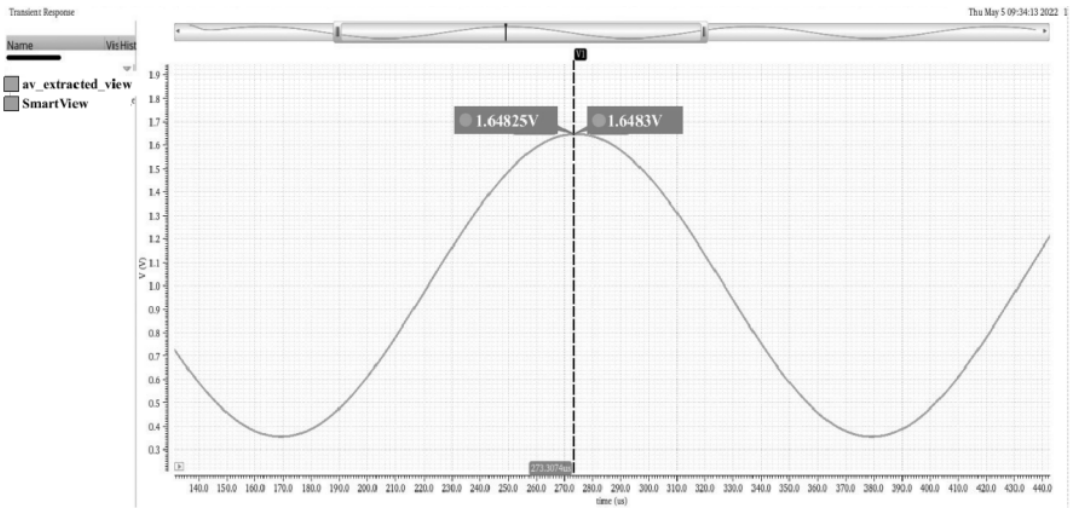


图 3 filter 电路两种网表的波形及频谱对比

smart-parasitics，不能完整进行描述。ADE Assembler 支持的 config sweep 功能可以实现同时扫描 design cellview，比如前后仿真可以同时进行，可以很方便地观察前后仿结果差异。

参考文献

[1] Quantus user guide version 21.2[Z].2022.
[2] ADE Assembler user guide version ICADVM20.1[Z].2022.

（收稿日期：2022-06-20）

作者简介：

陈思雨(1997-)，女，硕士，模拟 IC 设计工程师，主要研究方向：ADC/DAC 模拟电路设计。

黄亚平(1990-)，男，硕士，资深高级模拟 IC 设计工程师，主要研究方向：高性能射频和数模混合 IC。

胡劼(1984-)，男，硕士，资深模拟 IC 设计专家，主要研究方向：高性能射频和数模混合 IC。

扫码下载电子文档

（上接第 41 页）

[2] 郭炜，魏继增，郭筝，等.SoC 设计方法与实现[M].北京：电子工业出版社，2011.
[3] 朱健，胡凯，张伯钧.智能合约的形式化验证方法研究综述[J].电子学报，2021，49(4)：792-804.
[4] 谢小赋，曾梦岐，庞飞.面向计算机并发程序的形式化验证方法设计[J].信息安全与通信保密，2022(3)：54-62.
[5] 朱夕辉.核电厂仪控系统中 FPGA 的形式验证方法及 V&V 技术[J].工业控制计算机，2017，30(10)：84-85，87.

（收稿日期：2022-06-20）

作者简介：

赵亚雪(1994-)，女，硕士，IC 验证工程师，主要研究方向：无线通信基带芯片系统设计、通用处理器芯片设计。

植玉 (1981-)，男，硕士，IC 验证经理，主要研究方向：无线通信基带芯片系统设计、通用处理器芯片设计。

梁其锋(1985-)，男，硕士，IC 验证专家，主要研究方向：无线通信基带芯片系统设计、通用处理器芯片设计。

扫码下载电子文档

版权声明