

## 5 nm MSOA RapidPDK 及物理实现

贺恺华<sup>1</sup>, 丁学伟<sup>1,2</sup>

(1. 深圳市中兴微电子有限公司 IP 设计部, 广东 深圳 518055;

2. 移动网络和移动多媒体技术国家重点实验室, 广东 深圳 518055)

**摘要:** 随着当今电子行业的发展, 对 SoC 芯片, 尤其是数模混合芯片的要求越来越高。和传统的 DEF/GDS 数据交互方式相比, Mixed Signal Open Database(MSOA) RapidPDK 可以帮助设计人员通过相同的 PDK 更好地完成数字工具 Innovus 和模拟工具 Virtuoso 之间的数据传递。首先描述了 5 nm MSOA RapidPDK 生成方式, 其次使用生成的 PDK 实现 5 nm IP 物理实现, 同时验证 MSOA flow 对 5 nm 设计在版图完成和交付方面的速率提升。

**关键词:** MSOA RapidPDK; 5 nm; 物理实现; Virtuoso; Innovus

中图分类号: TN402

文献标识码: A

DOI: 10.16157/j.issn.0258-7998.219805

中文引用格式: 贺恺华, 丁学伟. 5 nm MSOA RapidPDK 及物理实现[J]. 电子技术应用, 2021, 47(8): 59-63, 67.

英文引用格式: He Kaihua, Ding Xuewei. 5 nm physical implementation with MSOA RapidPDK[J]. Application of Electronic Technique, 2021, 47(8): 59-63, 67.

## 5 nm physical implementation with MSOA RapidPDK

He Kaihua<sup>1</sup>, Ding Xuewei<sup>1,2</sup>

(1. IP Design Department of Sanechips Technology Co., Ltd., Shenzhen 518055, China;

2. State Key Laboratory of Mobile Network and Mobile Multimedia Technology, Shenzhen 518055, China)

**Abstract:** Success in today's electronics market place, most SoCs currently being developed have analog or mixed signal blocks, many so-called analog blocks actually have digital-control logic. Virtuoso-innovus flow with MSOA RapidPDK allows for all or parts of the physical hierarchy to pass back and forth between Virtuoso and Innovus easily without having to generate a DEF/GDS file. In this paper, we firstly describe the way to generate 5 nm RapidPDK, then use this PDK with digital flow to finish 5 nm IP design with custom lib. Additionally, this Virtuoso-innovus flow provides our analog designers with advanced techniques for 5 nm floorplanning, pin optimization, track generation and auto place & routing with RC extraction and timing analysis aware which will simplify the flow to enable virtuoso users to run 5 nm layout implementation, and also could help us to do full chip STA analysis.

**Key words:** MSOA RapidPDK; 5 nm; Virtuoso; Innovus

### 0 引言

随着芯片工艺的发展, 数模混合芯片中的数字和模拟模块(Block)已经没有明显的区分界限, 很多情况下, 模拟模块的内部会包含数字模块, 随之而来的问题是需要大量时间和精力解决在数字和模拟模块之间数据交互的问题和彼此 database 的兼容性。同时先进工艺下随着 double pattern 等特性的引入, 模拟版图的实现变得更加复杂, 而如何提高效率, 保证数据准确性也变得更加重要。

### 1 5 nm 工艺下数模芯片的挑战及 MSOA RapidPDK flow 的重要性

#### 1.1 5 nm 工艺下数模芯片的挑战

5 nm 工艺下模块之间数据交互难度更大, 需要的步骤和潜在的问题会越来越多。

##### 1.1.1 数据的传递和管理

(1) 5 nm 设计模块在物理实现时, 对模块的形状和 pin

的摆放有更多 DRC 等要求, 而且 GDS stream in 和 out 需要更长的时间。

(2) 由于 Virtuoso 和 Innovus 工具所需要的 techfile 文件不同, 同时 5 nm 存在 double pattern 等现象, 使得数模交付时使用的 LEF 和 DEF 文件在 import 和 export 的过程的复杂度和难度都有增加, 需要设计人员花费大量时间进行 debug。

(3) 即便是很小的改动, 交付数据的版本必须保留 Innovus 和 Virtuoso 两个版本, 而数模芯片多层次数字模拟 block 嵌套会导致版本追踪困难。

##### 1.1.2 自定义数字模块集成

5 nm 模拟设计中需要用到越来越多的自定义数字模块, 同时 Boundary cell、Filler 和 Dcap 等 physical 种类繁多, 在使用时, 不同的物理位置需要填充不同的 physical cell, 因此需要版图设计工程师花费大量的时间在自定义数

字模块的集成以及 physical cell 添加等重复性问题上。

### 1.1.3 模拟顶层绕线

5 nm 数模芯片设计复杂性和芯片面积的增加都会导致模拟顶层有很多绕线需求,而使用传统的手动方式实现绕线会降低版图实现效率,加大工作难度。

## 1.2 传统 flow 和 MSOA RapidPDK flow 的区别

### 1.2.1 传统 flow

(1)数模工具所需的 TechFile 不同:Innovus 调用的是 foundry 提供的 tlef 文件,而 Virtuoso 调用的是 foundry 提供的 basicPDK。

(2)难度明显增加:和以往的工艺相比,模拟电路的 layout 实现需要更多的资源和精力。

(3)数模在前期设计阶段 RC 抽参方式不同:对于模拟 design,RC 抽参只能在 layout LVS Pass 后进行,数字 design 可以在 LVS pass 前,进行门级抽参。

### 1.2.2 MSOA flow

(1)相同的 TechFile:无论 Innovus 还是 Virtuoso 都使用 OAdb 进行数据保存,调用的 techfile 都是 MSOA RapidPDK。

(2)提高效率:Innovus engin 可以协助模拟工程师实现大规模低要求的绕线,同时通过脚本完成大量重复性工。

(3)模拟设计通过数字 Innovus 工具可以对版图进行门及 RC 抽参,此种情况下不需要 LVS Pass。

## 2 MSOA flow 的环境设置

### 2.1 模拟设计环境设置

添加 Innovus 相关设置:

```
setenv CDS_ENABLE_EXP_PCELL TRUE
setenv CDS_EXP_PCELL_DIR <dir path>
envSetVal("we" "mixedSignalWireEditingEnvironment"
boolean t )
```

### 2.2 数字环境设置

添加 Virtuoso 相关设置:

```
Innovus OA db
//打开/保存目录下必须包含 cds.lib 和.cdsinit 文件
setenv CDS_ENABLE_EXP_PCELL TRUE
setenv CDS_EXP_PCELL_DIR <dir path>
```

### 2.3 特别注意

相同 OAdb Innovus 和 Virtuoso 保存目录需要一致,使用相同的 library name、cell name 和 view name。

## 3 MSOA RapidPDK 生成以及验证

MSOA flow 需要 Innovus 和 Virtuoso 都调用 MSOA RapidPDK 进行设计和数据保存,因此 MSOA RapidPDK 需要同时包含 Techlef 和 Basic PDK。

### 3.1 MSOA RapidPDK 生成方式

(1)生成 RapidPDK

5 nm 工艺可以使用 Cadence lef2oa 工具快速将两者结合生成新的 MSOA RapidPDK。

```
unix >?bsub lef2oa -useFoundryInnovus -lef $lef -lib
$RapidPDK -techRefs $BasePDK
```

```
$lef: Innovus tech LEF from foundry
```

```
$RapidPDK: MSOA flow tech lib?
```

```
$BasePDK: Tech lib from foundry
```

(2)生成 MSOA 格式 stdcell 和 Macro abstract

MSOA flow 不仅需要调用 MSOAPDK,同时需要对 design 中所有用到的 lib cell 进行 OA 的转换。

```
unix>?bsub lef2oa -lef stdcell.lef -lib stdcelllib-techRefs
RapidPDK
unix>?bsub lef2oa -lef MACRO.lef -lib macrolib-techRefs
RapidPDK
```

(3)MSOA RapidPDK 生成过程中遇到的所有的 error 和 warning 都要留意,WARNING:(OALEFDEF-23174)已经和 Cadence 确认可以忽略。

## 3.2 MSOA RapidPDK 验证

### 3.2.1 Innovus dumped techlef 验证

(1)在 Innovus 中分别对 OAdb 和传统的 LEF-DEF db 使用如下命令生成 tlef,并进行 tkdiff 比较验证。

```
write_lef_library -tech_only -ordered 1 xxx.tlef
```

下述几种 mismatch 已经和 cadence 确认可以忽略

①VIA 尺寸不同:如果比较时发现 VIA 的高度坐标不同,但 VIA shape 的高度值是相同的,则可以忽略,两者实际只是相对坐标不同而已。②USEVIA 不同:USEVIA 是 Innovus 工具自动生成的属性,只要后续 DRC 检查中 VIA 没有 violation 则可以忽略。③SPACING 缺失:MSOA 生成的 tlef 中会有 spacing 缺失,但在 tlef 的其他地方有对 SPACING 的定义,因此可以忽略。

### 3.2.2 Virtuoso QA 验证

使用 MSOA RapidPDK Re-attach design library,和传统方式下 attachPDK 的 library 进行 LVS 和 DRC 结果比较,发现两种方式结果一致。

### 3.2.3 Innovus QA 验证

(1)分别使用传统 LEF-DEF 方式和 MSOA flow 的方式进行 init 到 postroute 阶段的数字模块设计。(2)对两种方式生成的数据进行 DRC 和 LVS 验证,并比较 DRC 的类型和数量。(3)分析 DRC 结果发现,两种 flow 产生 DRC 的类型相同而数量不同,但深入了解 MSOA flow 和 LEF-DEF flow 可知两者在 place 和 route 阶段处理方式上存在差异会导致 DRC 的数量有不同。

3.2.4 Innovus 使用相同 database 分别对 OA 格式和 LEF-DEF 格式进行比较

将传统 LEF-DEF flow 生成的 database 进行 OA database 的转换,比较两者 LVS 和 DRC 的区别,发现两种方式结果一致。

## 4 5 nm MSOA flow 在项目中的实际应用评估

通过对实际 5 nm 项目中的模拟模块进行详细分析、评估,最终整理出针对模拟设计的 MSOA flow 相关规范和约束,通过搭建相关自动化 flow 提升模拟版图设计效率。

#### 4.1 前期分析

5 nm MSOA flow 评估主要针对如下两点:

##### (1) 5 nm 模拟设计效率提升

①评估 Innovus 布局和布线在模拟设计中的应用场景;  
②通过评估 MSOA flow 建立高效准确的数据管理和传递流程;③尝试通过脚本实现模拟设计自动化和标准化。

##### (2) MSOA flow 通用设计规范的建立

通过评估,建立5 nm MSOA RapidPDK flow 通用设计规范。

#### 4.2 具体设计评估过程

##### 4.2.1 基于网表的 Mix Signal Design

传统的 LEF-DEF flow 和 MSOA flow 在调用 Innovus 工具时,所需要的命令和存储方式主要区别见表 1。

表 1 LEF-DEF Flow 和 MSOA flow 的对比

LEF-DEF flow	MSOA flow
	init_oa_ref_lib
init_lef_file	init_abstract_view
	init_layout_view
saveDesign xxx.enc	saveDesign -cellview {...}
restoreDesign xxx.enc	restoreDesign -cellview {...}
数据存储在 invs_db 目录	数据存储在包含 cds.lib 的提交目录下
数据可以在任意目录 restore	数据只能在有 cds.lib 目录下 restore

传统 Lef-def flow 保存的 database 也可以用 OA 的方式打开以及存储,示例如下:

```
Innovus>?restoreDesign ${design}_${layout_step}.${view}.
enc.dat topDesign
Innovus> set init_oa_default_rule LEFDefaultRouteSpec
Innovus>?set init_oa_ref_lib { RapidPDK_lib ref_oa_lib }
Innovus>?set init_layout_view {layout}
Innovus>?set init_abstract_view {abstract}
Innovus>?set init_lef_file ""
Innovus>?createLib ${library} -attachTech{RapidPDK_lib}
Innovus>?set init_oa_lib { ${library} }
Innovus>?saveDesign -cellview { ${library} ${design}
${layout_step_view} }
```

##### 4.2.2 基于 Schematic 的 Mix Signal Design

MSOA flow 和传统模拟设计 flow 主要区别是 attach PDK 不同,见表 2。

表 2 传统 flow 和 MSOA flow 的对比

	Basic PDK Design	RapidPDK OA design
Attach PDK	Basic PDK	Rapid PDK

传统 Virtuoso flow 保存的 layout 也可以使用 OA 的方式通过 Innovus 工具打开以及存储,示例如下:

```
Virtuoso > create new library with RapidPDK tech lib
Virtuoso > copy cell to new library
Innovus > set init_design_netlisttype {OA}
###netlist-driven use "init_verilog"
```

```
Innovus > set init_oa_design_lib ${library}
Innovus > set init_oa_design_cell ${design}
Innovus > set init_oa_design_view ${layout_view}
Innovus > set init_oa_ref_lib {RapidPDK_lib stdcell_lib
macro_lib customized_lib }
Innovus > set init_abstract_view { abstract }
Innovus > set init_mmmc_file {}
Innovus > init_design
```

##### 4.2.3 基于 Schematic 的 custom digital block 设计

在 5 nm 设计中,很多模拟模块是通过需要调用自定义的数字 cell 进行设计,这类模块可以统一为 custom digital block,评估过程中将其内部调用的 custom digital cell 参考数字 stdcell 的方式进行规范化处理,使 Innovus 可以对这类 block 进行自动布局布线,处理方式如下:

(1)使用 abstract generator 对所有 custom digital cell 提取 lef 文件;

(2)由于 5 nm 是 double patten,因此 LEF 中一定要有 FIXEDMASK 属性,对底层金属 color lock;

(3)创建 Innovus 自动布局布线时所需要的 site 定义,举例如下:

```
SITE coreW57H448
CLASS CORE;
SYMMETRY Y;
SIZE 0.114 BY 0.448;
END coreW57H448
```

规范化处理 custom digital cell 后,使用 innovus 对 custom digital block 进行自动布局布线,包括使用 TCL 脚本的方式添加 physical cell、电源地线等,输出结果如图 1~图 3 所示。

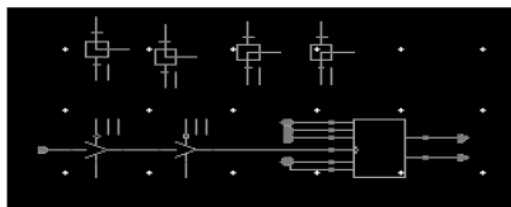


图 1 Custom digital block Schematic View

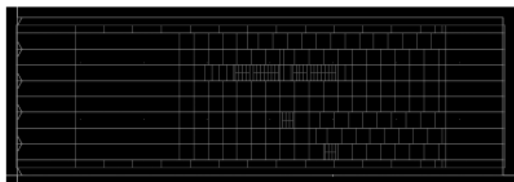


图 2 custom digital block open in Innovus

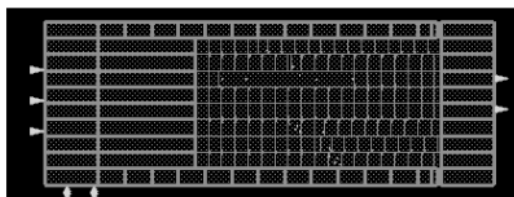


图 3 custom digital block open in Virtuoso



#### 4.2.4 基于 Schematic 的 custom stdcell 设计

5 nm custom digital block 评估过程中调用的 custom stdcell 的创建也可以用 MSOA flow 通过调用 Innovus 实现自动布线,可以通过添加 routing 相关 constraint 对工具绕线方式进行约束,满足 designer 需求,如图 4、图 5 所示。

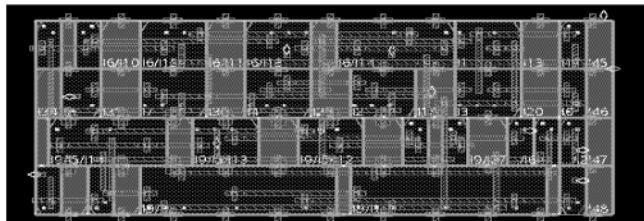


图 4 custom stdcell open in Innovus

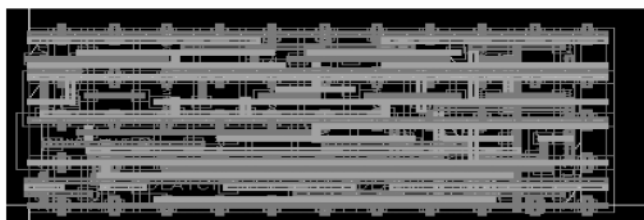


图 5 custom stdcell open in Virtuoso

#### 4.2.5 基于 Schematic 的 Innovus 绕线和效率提升

5 nm 工艺下模拟设计的 block 在复杂度、net 数量和面积方面都有大幅度提升,如果一味地要求版图工程师使用手动或者 VSR 的方式在 Virtuoso 中绕线,会使整个 block 设计所需的人力和精力相较之前的工艺都有所增加,而 Innovus NanoRoute 引擎对于大规模绕线有较好的处理方式,因此在模拟设计中对走线精度要求较低的大规模芯片可以采用 Innovus 进行自动化绕线大幅度提高速率,节省人力,同时也可以添加相关 constraint 约束 Innovus 绕线的方式。

Innovus NanoRoute 目前支持的绕线方式有: NDR-routing, shielding, differential pair match length, resistance match, layer match, bus routing, length control。相关约束命令举例如下:

```
setIntegRouteConstraint -type netClass -net {A B} -rule
NDR_rule -topLayer Metal5 -bottomLayer Metal1 -shield-
Net VSS -connectSupply anyPoint -shieldType side -toler-
ance 20.0
```

#### 4.2.6 基于 Schematic design 的数据集成

(1)schematic-driven 和 netlist-driven design 通过 OA db 可以代替 GDS 文件实现 database 的传递,这样无论数字还是模拟工程师在打开彼此数据时的速度都会有所提升。

(2)schematic-driven 的 block 可以通过 Innovus 中提取 LEF/DEF 的相关命令来生成数字所需要的 LEF/DEF 文件,同时生成的文件在一致性和准确度方面都有所提升,文件中也不会包含纯模拟相关的数据,影响数字工具使用,相关命令如下:

```
Innovus> write_lef_abstract xxxx
```

```
Innovus> defOut -floorplan -routing -netlist xxxxx
```

(3)Innovus 中在 5 nm 工艺下对 double patten 有 add\_track 的定义,通过参考 Innovus 中对 track 的定义,可以在 Virtuoso 中添加对应的 WSP rule,避免了数模之间交互时遇到的很多 DRC 问题。

#### 4.3 模拟版图设计使用 MSOA flow 时的相关规范

下述模拟设计的规范是在 5 nm MSOA flow 评估过程中整理出来的相关约束条件,可以帮助 MSOA flow 在 5 nm 设计中得到更广泛的应用。

##### 4.3.1 Schematic 规范

(1)Power 和 Ground pin 必须是 Power Ground 属性,不能定义为 signal。

(2)对于 signal pin 必须使用 tie cell 进行 VDD/VSS 的连接,不可直连。

##### 4.3.2 版图规范

(1)使用 generate from source 创建 layout,保留 Schematic 的连接关系。

(2)避免使用 pcell 的方式创建 VIA,Innovus 会将 pcell 识别为 block,而丢失 net 连接关系。

(3)必须使用 create->P&R objects 的方式创建 pr boundary,且 pr boundary 的原点在(0 0)。

(4)Block type design 的 pin shape 必须在边界。

(5)使用 create->pin 来创建 pin,且 pin 的金属层次需要在 M0 或 M0 以上。

(6)Power nets 使用 geometric nets 即 create->shape->path 创建,确保"part of a route"为 false。

(7)signal nets 使用 symbolic type route,即 create->wiring->bus/wire。

(8)确保 Schematic 和 layout 之间关系是 XL compliance。

##### 4.3.3 Abstract 规范

Abstract View 必须确保 color lock,对应的 LEF 中有 FIXEDMASK 属性。

#### 4.4 MSOA flow 相关脚本输出

(1)创建自动化 flow 相关的 TCL 脚本,将重复性操作统一脚本化,优化整个 flow,让模拟工程师通过简洁的操作界面调取相关的 Innovus/Virtuoso 脚本。

(2)使用 TCL/PERL 脚本实现模拟工程师在 Innovus 中自动生成 power mesh、endcap、welltap、filler 等 physical cell,为模拟设计师提供流程上更多选择性,相关脚本参考:

```
set step [set Mx_step]; set w [set Mx_width]
set space [expr $step/2]
set dir [dbget [dbget head.layers.name Mx -p].direction]
addStripe -layer Mx -direction $dir -nets "$pwr $gnd"
-width $w -set_to_set_distance $step -spacing $space
-area $area -snap_wire_center_to_grid Grid
```

(3)通过参考 Innovus 中对 track 的定义,在 Virtuoso 中使用 skill 语言规定 WSP rule,帮助项目中模拟工程师在

绕线时避免 DRC violation, 相关脚本参考:

```
wspDeleteCellViewAllWSPData(getEditCellView())
cv = getEditCellView()
wspCreateWSP(cv "M0_WSP" ?allowedRepeatMode "
any" ?defaultRepeatMode "stepped" 0.03 0.08 "color:1" )
wspCreateWSPDef(cv "M0_H_H280" "M0" "vertical"
0.28 ?defaultActive "M0_H280" ?allowedPatterns '
("M0_H280"))
```

#### 4.5 Virtuoso 中通过 VDI 启动 Innovus

对于数字工程师来说, 通常直接使用 bsub 的方式启动 Innovus 就可以对数字 block 进行编辑, 而模拟工程师对 Virtuoso GUI 界面的操作更加熟练, 使用 Virtuoso 中自带的 VDI 也可以直接启动 Innovus, load Innovus 相关脚本命令, 后续使用方式和直接启动 Innovus 相同, 如图6~图9所示。

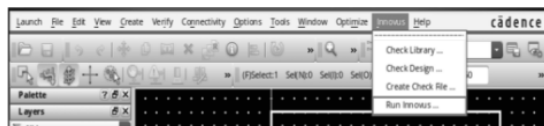


图6 Open VDI in Virtuoso

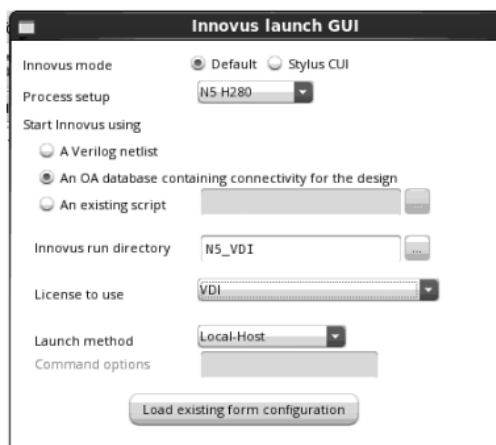


图7 MSOA flow VDI 相关配置(1)

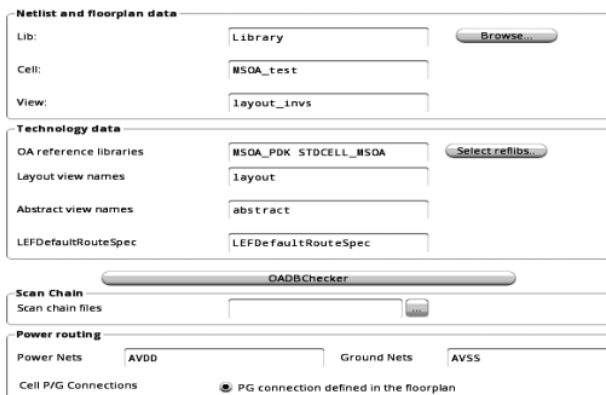


图8 MSOA flow VDI 相关配置(2)

### 5 5 nm MSOA flow 在模拟 block 中效率提升总结

#### 5.1 custom digital block

针对 custom digital block, 分别尝试使用传统 Virtuoso

```
setDesignNode -process 5
setNanoRouteNode -dbProcessNode N5
##### Design Netlist Input #####
set init_design_netlist_type {00}
set init_oa_design_lib {Library}
set init_oa_design_cell {M0A_test}
set init_oa_design_view {layout_invs}
##### Technology and Cell Abstract Information #####
set init_oa_ref_lib {M0A_PDK STDCELL_M0A}
set init_abstract_view {abstract}
set init_layout_view {layout}
set init_oa_default_route {LEFDefaultRouteSpec}
##### Top Level PG Nets #####
# Ensure that the sigtype for the power and ground nets in OA is set correctly to supply and ground
respectively,
set init_per_net { AVDD }
set init_gnd_net { AVSS }
##### Design Initialization #####
setGenerateViaNode -auto true
init_design
#####Route Design#####
source route_design.tcl
setdbNode -saveHashDataLocked true
saveDesign -cellview "Library M0A_test layout_route"
```

图9 MSOA flow VDI 调用的相关脚本

flow 和 MSOA Innovus flow 完成版图物理实现, 结果如表3所示。尝试了布局布线, 以及 physical cell 摆放等全流程, 发现 MSOA flow 在 physical cell 摆放时通过自动化脚本可以大幅缩减摆放时间, 提升效率。

表3 custom digital block 评估结果

N5 project 评估类型	传统 Virtuoso flow		Rapid MSOA flow		效率 提升	应用 场景
	Sub-flow	所需 时间/d	Sub-flow	所需 时间/h		
Case1: 小型模块	Virtuoso	1	Innoovus	2	75%	大部分 场景
Case2: 大型模块	P&R	2~3	P&R	4	75%~ 83%	

#### 5.2 custom stdcell

该阶段主要尝试了传统 flow Virtuoso 布局布线和使用的 MSOA Innovus flow 进行布局布线的效率比较, 评估结果如表4所示。发现在绕线阶段, 如果对 custom stdcell 的绕线要求较低, 没有定制化需求则使用 MSOA flow 方式会有大幅效率提升, 但同时由于 MSOA flow 有相关规范要求, 因此在初期需要话较多时间满足 MSOA 规范要求。

表4 custom stdcell 评估结果

N5 project 评估类型	传统 virtuoso flow		Rapid MSOA flow		效率 提升	应用 场景
	Sub-flow	所需 时间/d	Sub-flow	所需 时间/h		
Case1: 中等规模	Virtuoso	1	Virtuoso VCP		87%	有一 定 限 制
Case2: 较大规模	P&R	2	Placement & Innovus NanoRoute	1	93%	

#### 5.3 数据传递

该阶段主要尝试使用 Innovs 和 Virtuoso 分别开启模拟和数字的 OA database, 和传统使用 GDS 方式比较速率提升, 同时尝试使用 Innovus 的方式输出模拟 OA db 的 LEF 和 DEF 文件交付给数字后端进行调用, 比较 LEF 和 DEF 文件输出的速率提升。数据传递评估结果如表5所示。

(下转第 67 页)

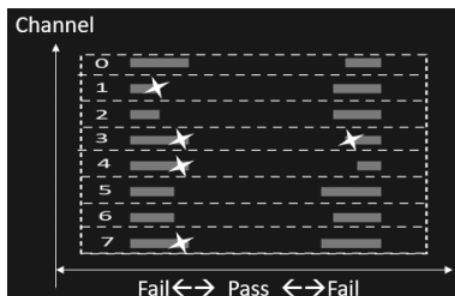


图 10 HBM 模块 8 通道启动眼宽损失分布(PRBS 码)

大约 1 280 根,从图 10 和图 11 的扫描结果,可以看到单个通道随着全通道的启动,受到了不同的影响。其中 PRBS 码测试下,通道 3 受到了 2 个眼宽单位的损失,这里,每个单位实验室设定为 0.78%眼宽。可以看出,PRBS 下通道 3 最大损失了 1.56%眼宽,1010 码型下通道 3 损失了 2.34%眼宽。

## 5 结论

本文讨论了在 2.5D 和 3D 封装形式下,基于信号完整性与电源完整性的考量,介绍了如何仿真优化 HBM 系统高速信号通道的设计,特别提出了 CCCS 预测式电源噪声的仿真方法,进行了全仿与预测仿的对比,并与实验室芯片实测数据进行对比,验证了仿真数据的合理性和一致性。

## 参考文献

[1] Shi Wenjun, Zhou Yaping, SUDHAKARAN S. Power delivery

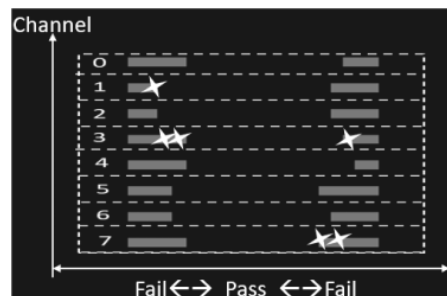


图 11 HBM 模块 8 通道启动眼宽损失分布(1010 码)

network design and modeling for high bandwidth memory (HBM)[C]//2016 IEEE 25th Conference on Electrical Performance of Electronic Packaging and Systems(EPEPS),2016: 3-6.

[2] CHO J, KIM H, FAN J, et al. Investigation of statistical eye-diagram estimation method for HBM including ISI, X-talk, and power noise[C]//2017 IEEE International Symposium on Electromagnetic Compatibility & Signal/Power Integrity (EMCSI), 2017: 411-415.

[3] THOMAS H, NA N, WONG A, et al. Top-Down jitter specification approach for HBM system optimization[C]//Design Con 2019, 2019.

[4] 李玉山. 信号完整性与电源完整性分析(第二版)[M]. 北京: 电子工业出版社, 2015.

(下转第 71 页)

(上接第 63 页)

表 5 数据传递评估结果

N5 project 评估类型	传统 virtuoso flow		Rapid MSOA flow		效率 提升	应用 场景
	Sub-flow	所需 时间	Sub-flow	所需 时间		
模拟-> 数字	Virtuoso LEF/DEF 文件输出	2 天	Innovus OA restore & Innovus LEF/DEF 文件输出	1 小时	>90%	大部分 场景
数字-> 模拟	Virtuoso GDS stream in & out	2~4 小时	Innovus OA save & Virtuoso open	<8 分钟	>93%	

## 5.4 power mesh 和 routing

对于 power mesh 和重复性的 routing 脚本, MSOA 使用有着一定的局限性, 在前几个评估的 case 中可见, 对于大尺寸、绕线要求低、但绕线多的情况下, MSOA 对 design 速率有很大提升, 但如果设计定制化要求高, Virtuoso 和 Innovus 所需要的时间会比较接近, MSOA flow 对于这种情况下只是向 designer 多提供了一种选择方式, 具体如

何使用由 designer 决定, Power mesh 也是同理。power mesh 和 routing 评估结果如表 6 所示。

表 6 power mesh 和 routing 评估结果

N5 project 评估类型	传统 virtuoso flow		Rapid MSOA flow	
	Sub-flow	所需 时间/h	Sub-flow	所需 时间/h
Power mesh or highly repetitive routing unit	Virtuoso Power Planning/Group/ Skill/WSP	2	TCL script for special nets in Innovus	2

## 参考文献

[1] Cadence Design Systems, Inc. Mixed Signal (MS) interoperability guide[Z]. 2021.

[2] Cadence Design Systems, Inc. Mixed Signal design challenges and cadence MSOA rapid flow overview[Z]. 2021.

(收稿日期: 2021-06-23)

## 作者简介:

贺恺华(1989-), 女, 硕士, 工程师, 主要研究方向: 数字后端、模拟后端。

丁学伟(1982-), 男, 硕士, 工程师, 主要研究方向: 模拟 IP。



扫码下载电子文档

## 版权声明

经作者授权，本论文版权和信息网络传播权归属于《电子技术应用》杂志，凡未经本刊书面同意任何机构、组织和个人不得擅自复印、汇编、翻译和进行信息网络传播。未经本刊书面同意，禁止一切互联网论文资源平台非法上传、收录本论文。

截至目前，本论文已经授权被中国期刊全文数据库（CNKI）、万方数据知识服务平台、中文科技期刊数据库（维普网）、DOAJ、美国《乌利希期刊指南》、JST 日本科技技术振兴机构数据库等数据库全文收录。

对于违反上述禁止行为并违法使用本论文的机构、组织和个人，本刊将采取一切必要法律行动来维护正当权益。

特此声明！

《电子技术应用》编辑部

中国电子信息产业集团有限公司第六研究所