验证通识4

1. 验证的周期

**需要**：验证周期的检查清单，对每一个项目节点的task、节点间的联系、以及不同节点在整个项目周期的作用，会有更全面的认识

保证每一位验证人员充分了解各个验证环节，保持信息通畅，团队共同承担压力，降低风险

验证里程碑：

RTL0：芯片框架和模块功能定义完成，制定验证的策略

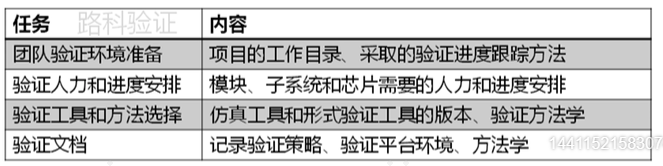


Figure 1 RTL0

RTL1：模块和子系统的功能信号定义完成，定制需要的储存模型

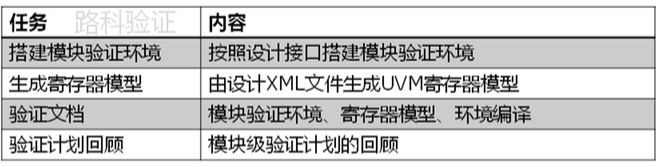


Figure 2 RTL1

RTL2: 完成所有模块的设计，以及80%以上的模块和子系统的验证，核心功能全部完成验证



Figure 3 RTL2

RTL3： 完成芯片系统的连线集成和验证，覆盖所有的功能验证点



Figure 4 RTL3

GLS: 完成门级网表的验证

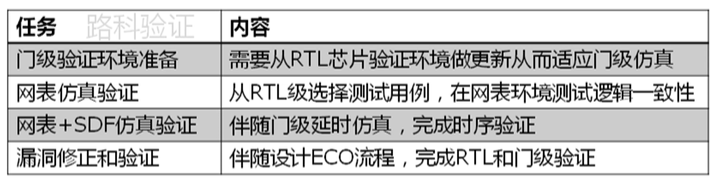


Figure 5 GLS

TO： 回顾验证的各项检查清单，最终流片

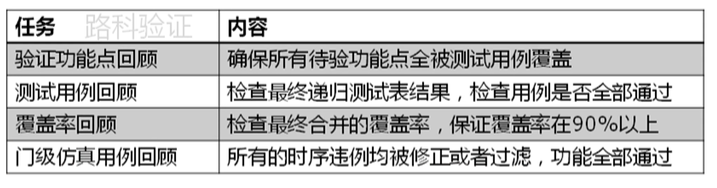


Figure 6 TO

1. 验证管理三要素

三要素：**时间**、**人力资源**、**任务**

**时间提前**：尽早参与前期定义环节，更早考虑选用什么**验证IP**、**验证方法**和相应的**工具**

在项目开始之前考虑更新**验证环境、流程、工具选择、方法学、技能训练、自主工具开发等**

将**验证环境搭建**工作和**测试用例创建**工作分开，少数人搭建维护验证环境（需要全面的系统知识和验证经验），绝大部分人专心创建测试用例，来提高效率

**减少依赖**：

减轻或者消除对于设计进度的依赖性

不因为未完成的设计而浪费时间

多个新老项目的协调，避免冲突

**大局观**：

所有的验证人员知道他们**共同的关键节点**，以及各个模块之间的**依赖性**

不止考虑方法或工具本身可以提高多少**仿真速度**或者**覆盖率**，同时也要考虑**人员的技能培训投入、学习曲线、新工具的整合、新环境的维护**等

**团队建设**：

验证成员需要不同的技术背景

考虑所具备的基本技能，和在某些技术领域拥有着丰富经验。如：软件编程、验证环境搭建、形式验证、硬件加速等。

团队中成员之间有重叠和差异，选派人员的时候有多种选择，共同工作时技能互补。

不同经验层次的梯队保证技术的传承和培养。将新任务给老员工（新技能培养），将老任务给新员工（快速适应环境）。

**技术管理**：需要技术良好的梯队，执行力，除了不同技术专长的验证人员还需要可以统观全局的验证经理

任务拆分和重组：在不确定的环境中找到确定因素，来安排验证进度、估算所需要的验证资源

**拆分**：将用时较长或者复杂的任务拆分为相对独立的小任务：

**重组**：统筹各个模块、不同验证节点之间的任务时，对不同任务进行合并、转接、排序，高效利用整体资源。

1. 验证的收敛

概述：随机验证的方式**回归**（regression）两种目的：

因为每次随机验证环境产生的**激励序列**不同，每次仿真均会对覆盖率做出贡献；

当设计**缺陷**被发现后，回归序列再次提交，保证功能测试点**无误**，设计缺陷也被**修复**。

回归测试指的是每次将**所有测试**用例提交到服务器上运行，并且检查**测试结果**。

需要考虑下面几个因素：

**回归流程**：模块级基本功能验证

**回归质量**：基本功能是否正常工作；（设计人员**修复**基本**缺陷**后，先通过**基本功能测试**，再验证是否通过之前的**错误场景**，同时创建专门**针对**该缺陷的基本**测试**）

**回归效率**：回归测试是一种**确保设计功能通过**的稳妥手段，而且方便操作管理，也可以用来**提升覆盖率**。但追求验证完备性的同时，**回归测试的效率**问题也越来越被重视。（切分测试用例；快速跳转至特定状态）

1. 问题追踪

追踪问题类型：

系统**功能定义**

硬件**设计**问题

芯片**验证环境**问题

**综合时序**问题

硅前**工具**问题

引用**库**和**IP**问题

追踪工具功能：

**记录**：需要记录的内容有问题标题，内容，出错场景，背景描述，发布版本，测试用例和相关文件等。

**分类**：归属于哪一个项目，哪一个环节（系统，设计，验证还是其他）哪一个模块，问题严重性（致命，重要，中级，改进）。

**派发**：在跟踪系统中，问题一旦提交，则由管理层指定接管人员完成其环节。

**查找**：查找该问题以前是否发生过，有无解决办法；另一方面，搜索该问题背景和进度。

**追溯**：问题从被提出到被派发，解决，验证和最终关闭，在一个项目中可能走完它的生命周期，但是不排除他可能会在下一个项目中复活。

**报告**：数据健全的内容报告

1. 团队建设

7个好习惯：

追求百分百

保持面向对象的开发习惯：Python，C++，等

合理复用

保持创新

高效沟通

突破责任边界

1. 验证的专业化