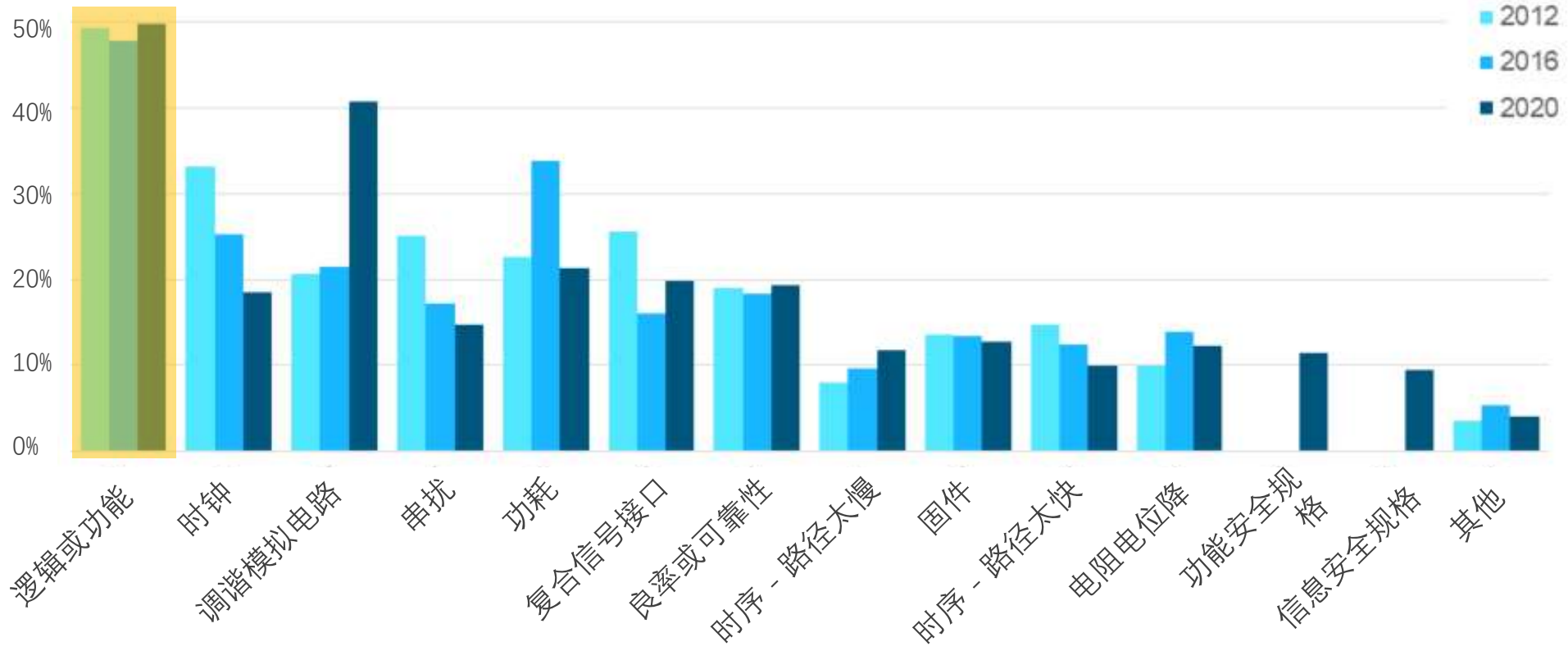
结合系统建模、架构设计与原型验证，快速且准确定义 SoC 规格

国微思尔芯

2022.3.18

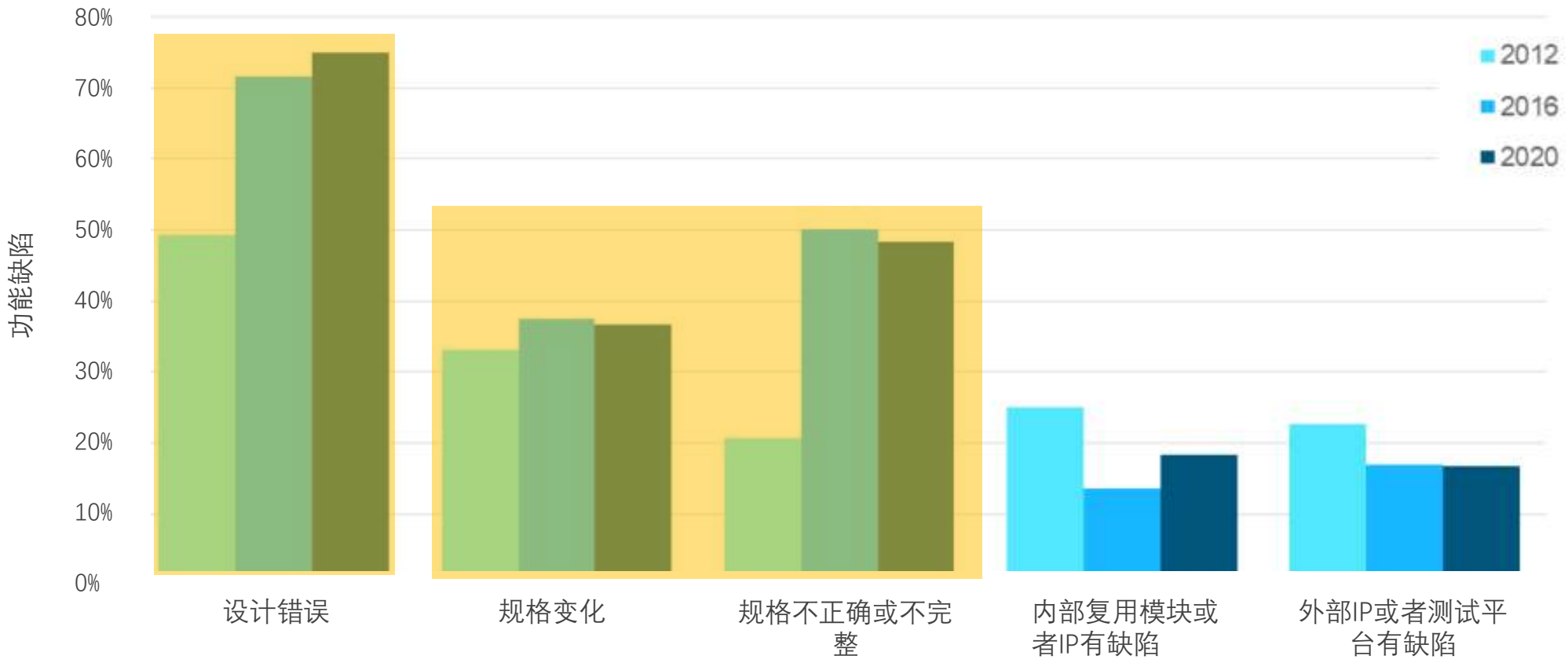
Cindy

➢流片失败主要因素



来源： Wilson Research Group and Mentor, A Siemens Business, 2020 Functional Verification Study

➢功能错误的主要因素



来源：Wilson Research Group and Mentor, A Siemens Business, 2020 Functional Verification Study

➢芯片验证趋势和挑战

不仅要确定芯片设计正确

芯片设计成本日益高涨

- ➢ 芯片规模不断变大
- ➢ 软件内容持续增多
- ➢ 系统测试复制费时

还要确定设计规格正确的芯片

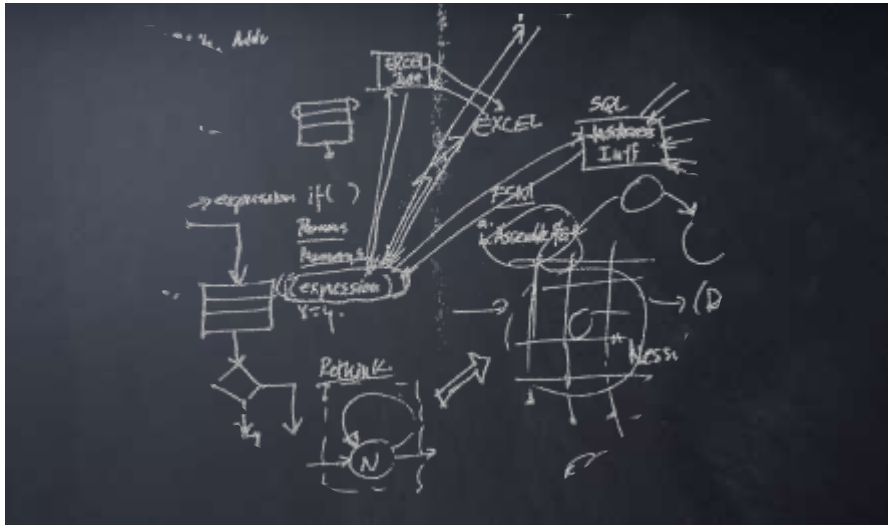
系统级芯片验证极其复杂

- ➢ 逻辑和功能设计错误是导致流片失败的第一因素
- ➢ 仿真和验证约占7成左右的研发时间

芯神匠 架构设计工具

市场需求：

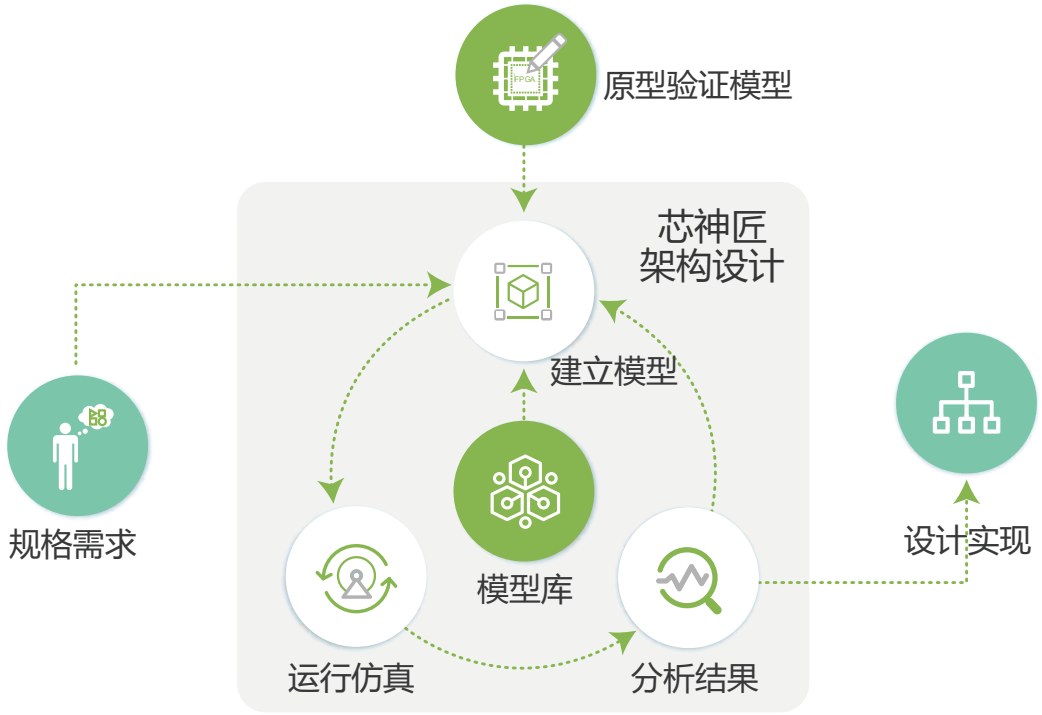
随着设计的规模不断增大，单个硬件与软件部件设计能力和可靠性的提高，系统失效更多是各子系统之间相互协调不好而导致；需要半导体与系统公司之间更有效地合作。



过去的系统设计形式

解决方案：

- 进行架构探索、优化、验证和交流决策
- 实现图形可视化项目的规格设计
- 提供可运行和仿真参数的规格参考

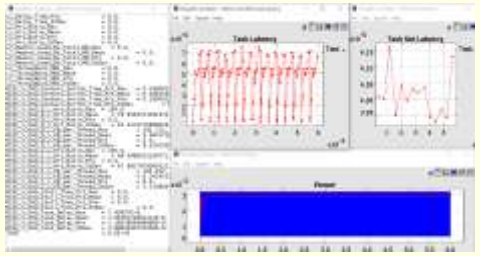


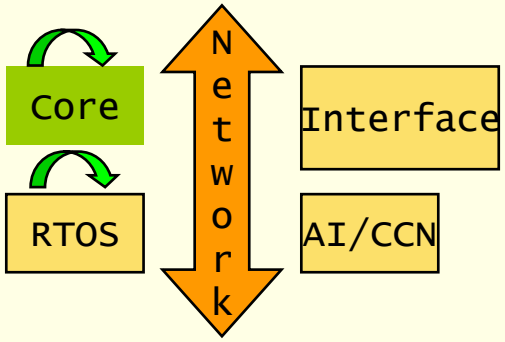
芯神匠架构探索领域

- 航空电子，飞行控制，惯导，通信，雷达
- 软件无线电，基于DSP通信系统
- 多种网络构建，Wifi定制协议，仲裁，流控制
- 自动驾驶，网络构建，功能安全，ECU选择
- 半导体，SoC, 处理器, FPGA, switch, TPU
- 算法类， SSD, HDD, HPC, NAS, 多媒体服务器

选择激励
系统负载
使用案例

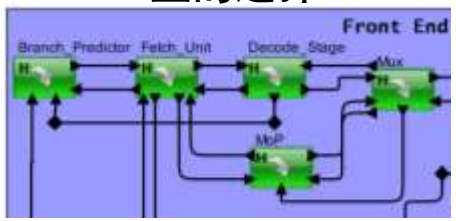
根据需求进行优化

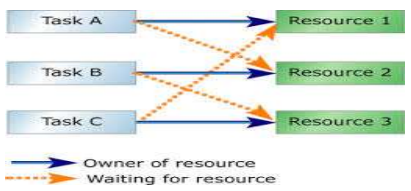




数量、大小等折中设计
不同工作模块分配
HW-SW 分配
参数调整

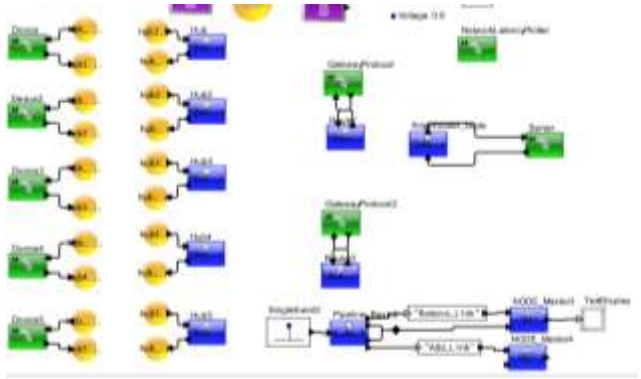
IP和内核的数量和类型的选择



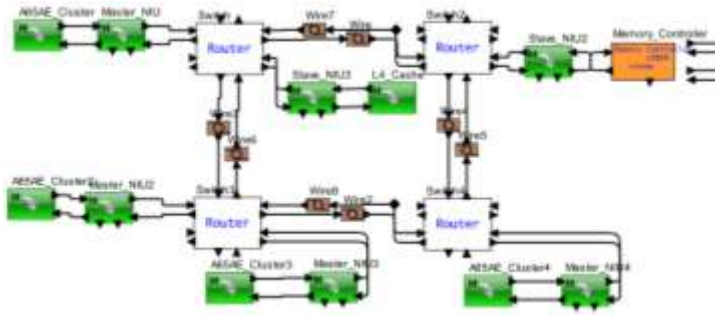


软件Performance Tuning

System Specification



SoC Specification

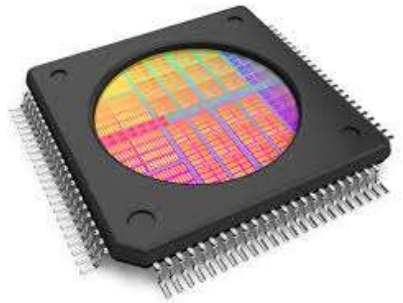


主要分析因素

- **硬件设计**
  - 功能正确性，速率，拓扑结构及仲裁
- **软件设计**
  - 设计质量评估、激励、配置及功耗对整个设计的影响
- **性能评估**
  - 吞吐率, 反应时间, 缓冲区大小和利用率
- **功耗测量**
  - 最大瞬时功耗和均值功耗，不同任务执行的消耗
- **功能安全分析**
  - 输出同ISO-26262 和RO-254 标准对标的分析结果

应用场景: SoC 设计中的分析

- **架构比对（Architecture comparison）**
  - 不同数量处理器内核的配置，总线类型，存储的选择及不同的软件搭配的性能对比
  - 不同场景下吞吐率和功耗的测量
- **功能安全分析（Functional safety analysis）**
  - 分析硬件、软件、网络、操作系统或者功耗失效后系统的反应
- **性能优化（Performance optimization）**
  - 使cache hit-ratio最大化并且缩短访问存储的时间为目标
  - 分析总线的通信，端到端延迟和系统的吞吐率的方式
- **虚拟原型性能（Virtual performance prototyping）**
  - 根据实际的软件代码来做性能优化
  - 功耗，软硬件partition，占用buffer等权衡



应用场景: SOC/IP架构设计

微架构设计

Pipeline阶参数的选择

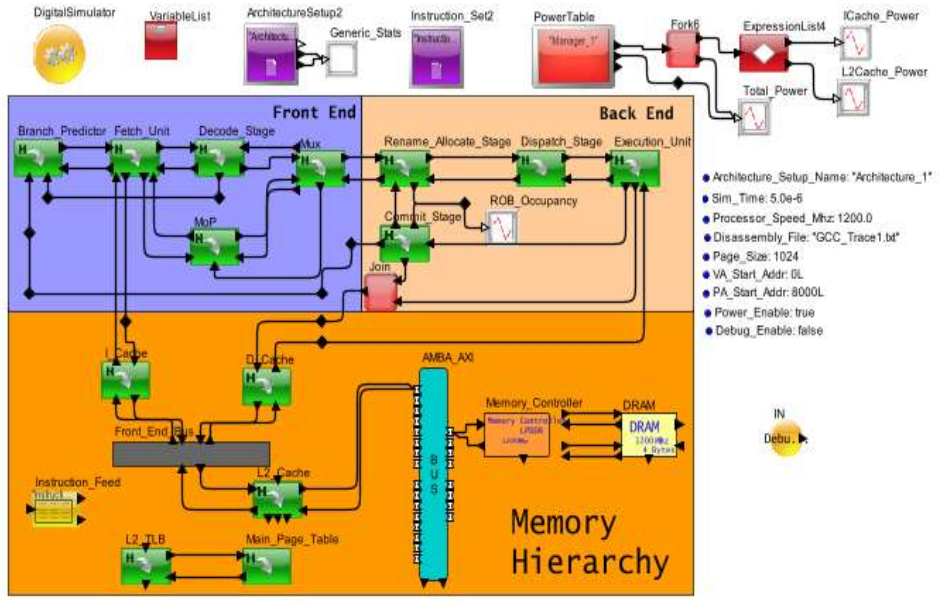
- 缓存大小，流控制算法，预取单元，线程调度，队列设置
- 不同参数对功耗和延迟的影响
- Cache, 总线及存储层级的设计

用软件来测试Pipeline的行为

- 从GEM5来获取trace文件

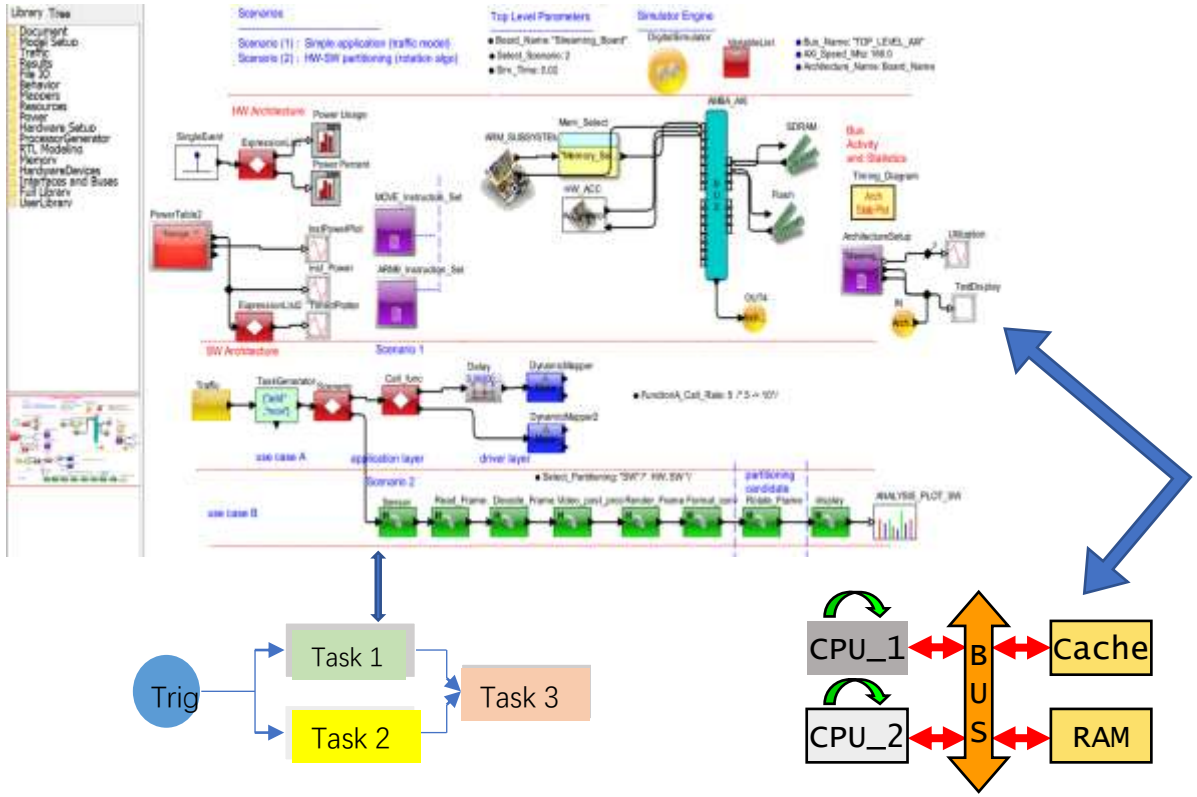
分析并输出结果

- 延迟
- 吞吐率
- 功耗 (平均&峰值)



ARM A77 微架构

应用场景: 硬件和软件任务的分配和协同



行为流->任务将如何执行的顺序　　硬件架构-> 设备的硬件架构是如何实现的

评估HW&SW 不同分配情况后的功耗：

- 所有的任务在软件内执行
- 分配一部分到硬件加速器后
- 增加功耗管理模块来减少功耗

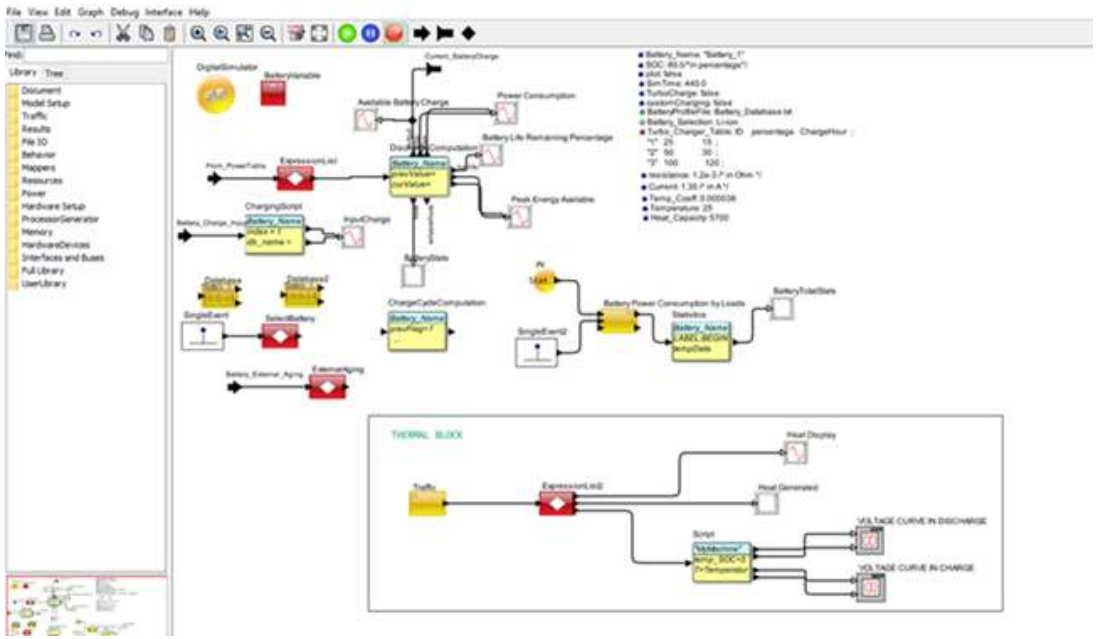
Soc/IP的层级

- 不同流水级设置
- 不同数量的执行单元，位宽，速率等

系统级

- 根据不同应用场景的架构搭建：IoT 设备，ECU 或者综合的平台等

应用场景: 功耗模块及电池模型

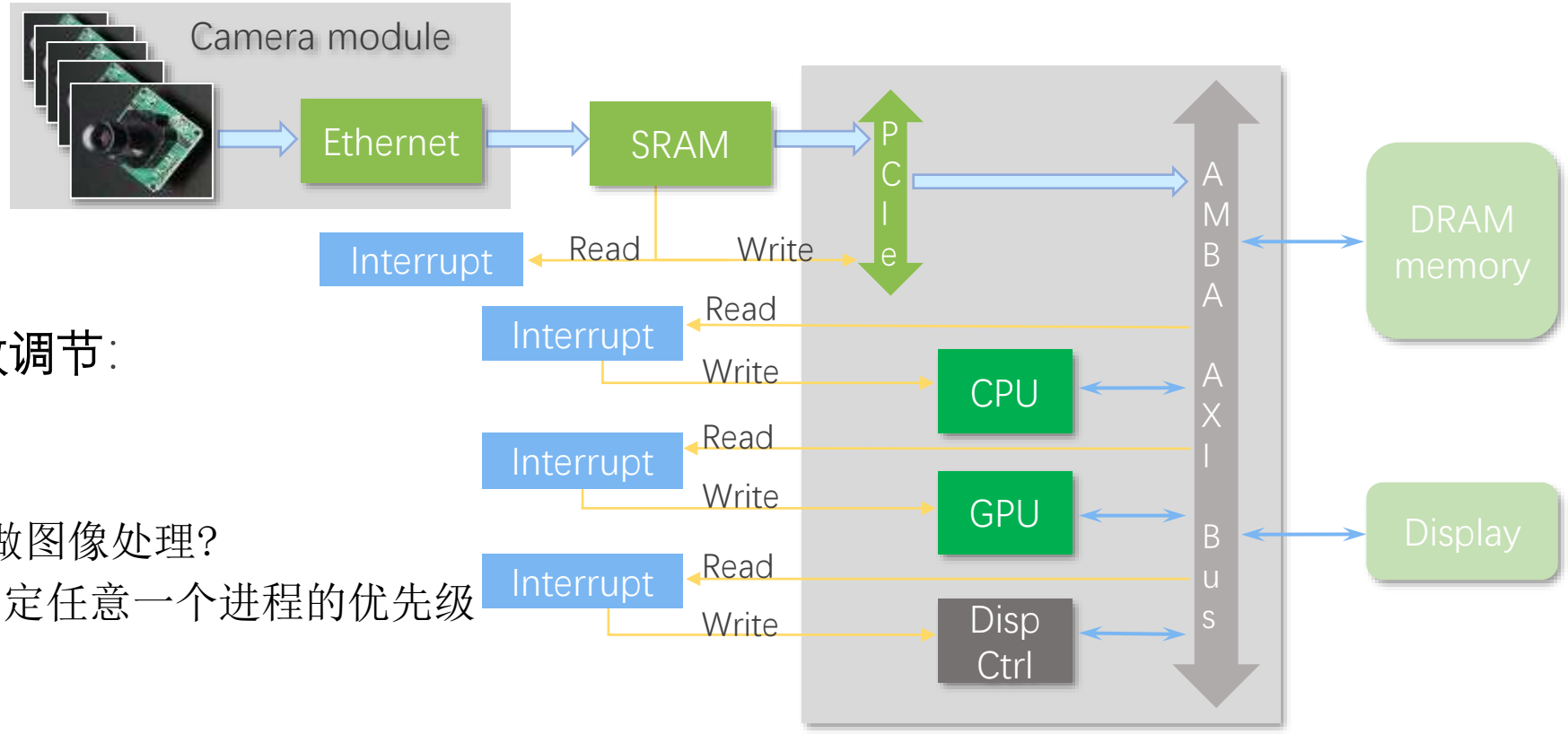


- **有硬件和事务库**
  - ✓ 无需要额外的建模工作
- **包含了硬件加速器等客制化模块**
  - ✓ 定义客户不同状态下的功耗
  - ✓ 用时间状态来定义不同状态的转换，并添加不同状态建的中间状态
  - ✓ 用功率表达式来描述状态或者修正功率耗散
- **模型运行的设置**
  - ✓ 建立各个设备瞬间电源状态周期表（电源管理）
  - ✓ 检查可用电池电量以做出决定(get Battery Level)
  - ✓ 增加电池使用寿命(通过增加电池电量耗散模型)

应用场景: 车载娱乐系统设计开发

系统各个组成模块的参数调节:

- ✓ Bus 、DDR 带宽设置
- ✓ SRAM 容量多少合适?
- ✓ 在GPU还是Camera模块做图像处理?
- ✓ 设计仿真激励，如何确定任意一个进程的优先级
- ✓ 修改bus/DDR 特性
- ✓ 采用不同CPU/配置
- ✓ 插入更多的摄像头模块？多少比较合适？
- ✓ AXI总线拥塞后的处理



车载娱乐系统架构模块

应用场景: AUTOSAR 硬件和软件任务配置

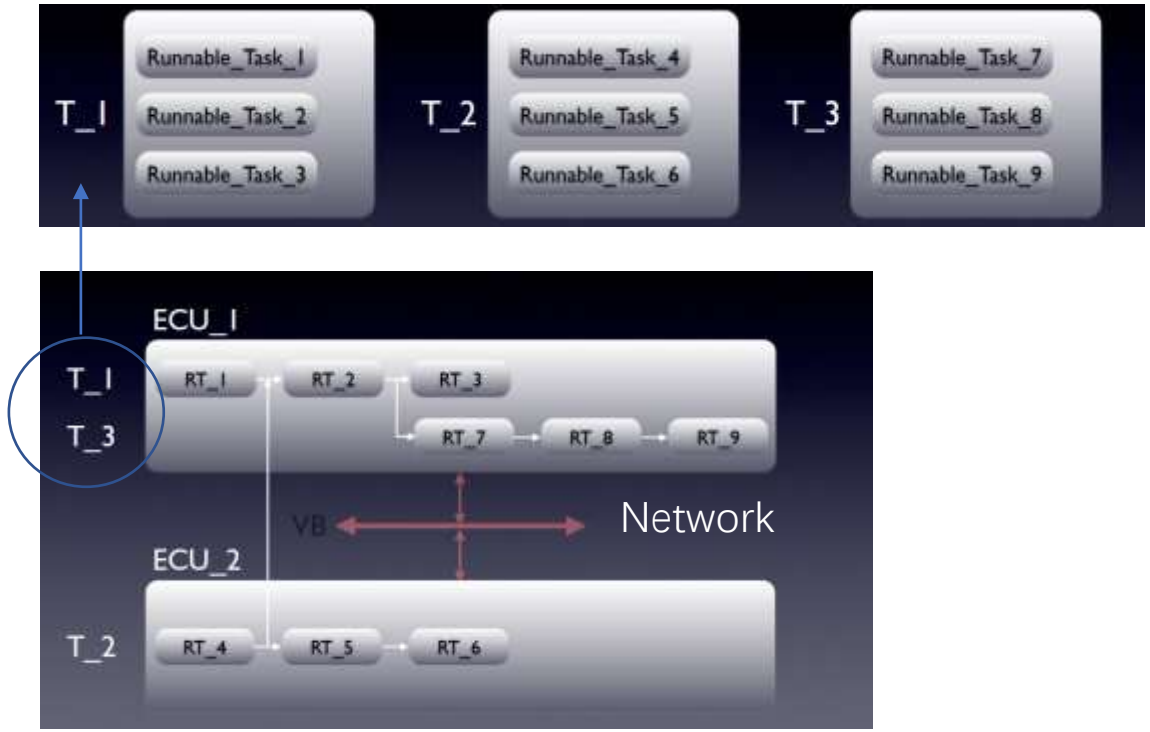
AUTOSAR 功能在架构构建时可定义和验证

- 软硬件的系统协同

Genesis' AUTOSAR library

- 探索不同的bus拓扑

  CAN, FlexRay, Ethernet, …
- 定义 runnable tasks
- 在各个 ECU安排 runnable tasks
- 完成 HW/SW partition



应用场景: 功能安全——故障注入

硬件失效（Hardware Failure）

- 处理器内核的缺失/工作异常，存储容量不够，内存缩减或者减少，总线过载/错误信息传输

软件失效（Software failure）

- 资源受限, deadlocks, 数据冗余

网络失效（Network failure）

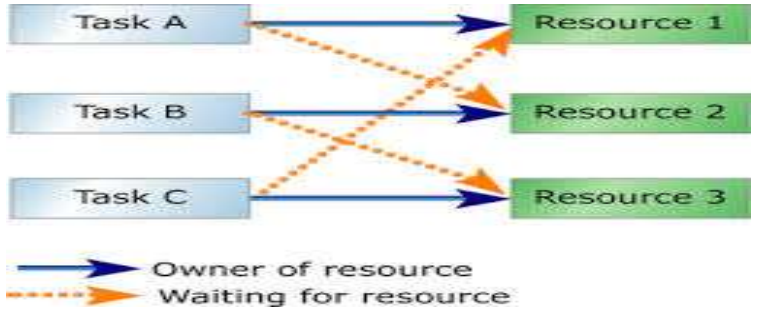
- 网络拥塞,错误配置，链路丢失和网络错误

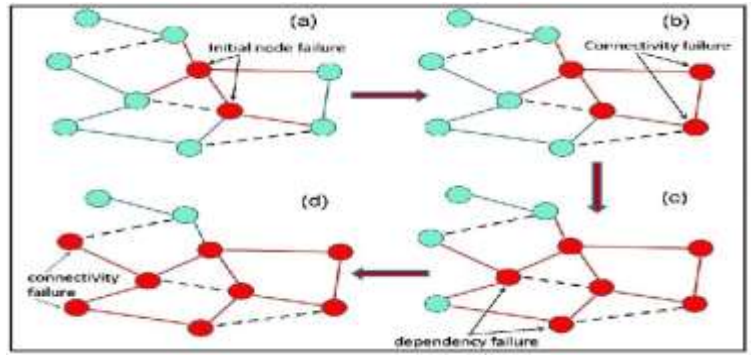
RTOS失效 （RTOS failure）

- 无法达到实时截止,恶意篡改进度表，超过时间设定执行

功耗失效（Power Failure）

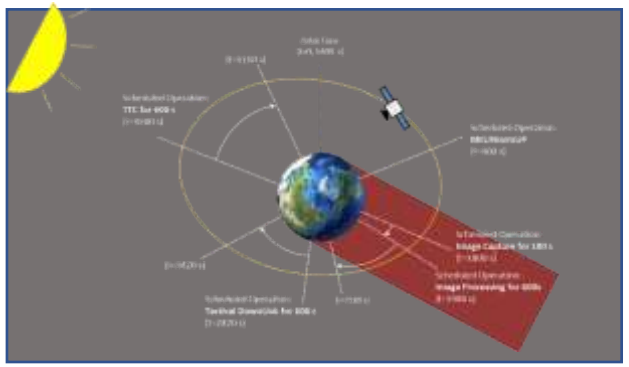
- 功耗消耗过大或者功率不够，导致处理速度缓慢减少资源的有效使用





应用场景: 航空航天及国防解决方案

- Mission-Level 网络分析
- 导航系统Cyber security
- 太阳能系统
- 航电综合模块
- 测绘雷达在硬件上的应用
- 接口数据采集SoC



Orbital-Level

Network-Level

涵盖了各个领域的IP Library库

Traffic
- Distribution
- Sequence
- Trace file
- Instruction profile

Custom Creator
- Script language
- 600 RegEx fn
- Task graph
- Tracer
- C/C++/Java
- Python

Reports
- Timing and Buffer
- Throughput/Util
- Ave/peak power
- Statistics

Support
- Listener and Trace
- Debuggers
- Assertions

RTOS
- Template
- ARINC 653
- AUTOSAR

Software
- GEM5
- Software code integration
- Instruction trace
- Statistical software model
- Task graph

ARM
- M-, R-, 7TDMI
- A8, A53, A55, A72, A76, A77

Processors
- GPU, DSP, mP and mC
- RISC-V
- Nvidia- Drive-PX
- PowerPC
- X86- Intel and AMD
- DSP- TI and ADI
- MIPS, Tensilica, SH

Stochastic
- FIFO/LIFO Queue
- Time Queue
- Quantity Queue
- System Resource
- Schedulers
- Cyber Security

Power
- State power table
- Power management
- Energy harvesters
- Battery
- RegEx operators

SoC Buses
- AMBA and Corelink
- AHB, AB, AXI, ACE,CHI,CMN600
- Network-on-Chip
- TileLink

Memory
- Memory Controller
- DDR DRAM 2,3,4, 5
- LPDDR 2, 3, 4
- HBM, HMC
- SDR, QDR, RDRAM

System Bus
- PCI / PCI-X / PCIe
- Rapid IO
- AFDX
- OpenVPX
- VME
- SPI 3.0
- 1553B

FPGA
- Xilinx- Zynq, Virtex, Kintex
- Intel-Stratix, Arria
- Microsemi-Smartfusion
- Programmable logic template
- Interface traffic generator

RTL-Like
- Clock, Wire-Delay
- Registers, Latches
- Flip-flop
- ALU and FSM
- Mux, DeMux
- Lookup table

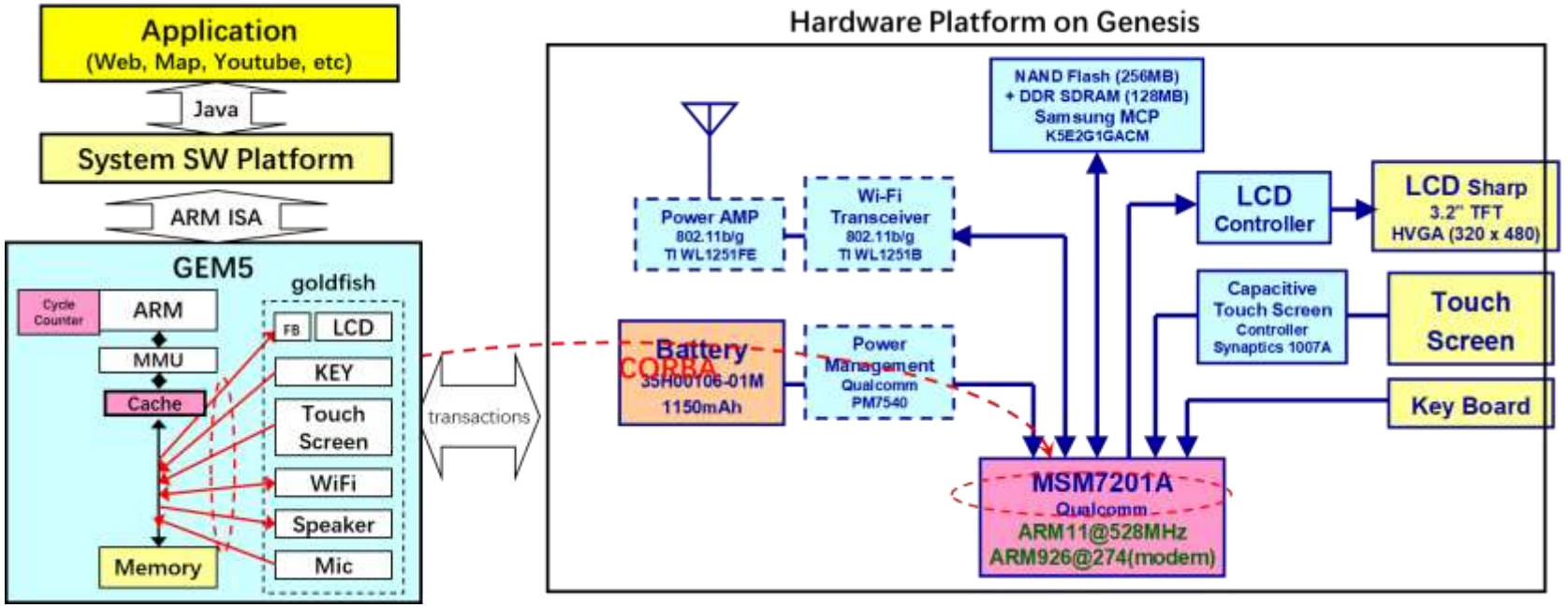
Interfaces
- Virtual Channel
- DMA
- Crossbar
- Serial Switch
- Bridge

Networking
- Ethernet & GiE
- Audio-Video Bridging
- 802.11 and Bluetooth
- 5G
- Spacewire
- CAN-FD
- TTEthernet
- FlexRay
- TSN & IEEE802.1Q
- ARINC 664
- AFDX

Storage
- Flash & NVMe
- Storage Array
- Disk and SATA
- Fibre Channel
- FireWire

芯神匠 & GEM5

- 芯神匠集成了GEM5, 用于在处理器模型上运行软件代码
- 用户可以将GEM5与系统的其他部分集成，以研究缓存命中率、内存吞吐量和延迟



➢芯神瞳 原型验证

协同仿真套件
ProtoBridge

自动原型编译软件
PPro-CT

原型验证系统
FPGA Prototyping Platform

实时控制软件
PPro-RT

深度调试套件
PPro-DT

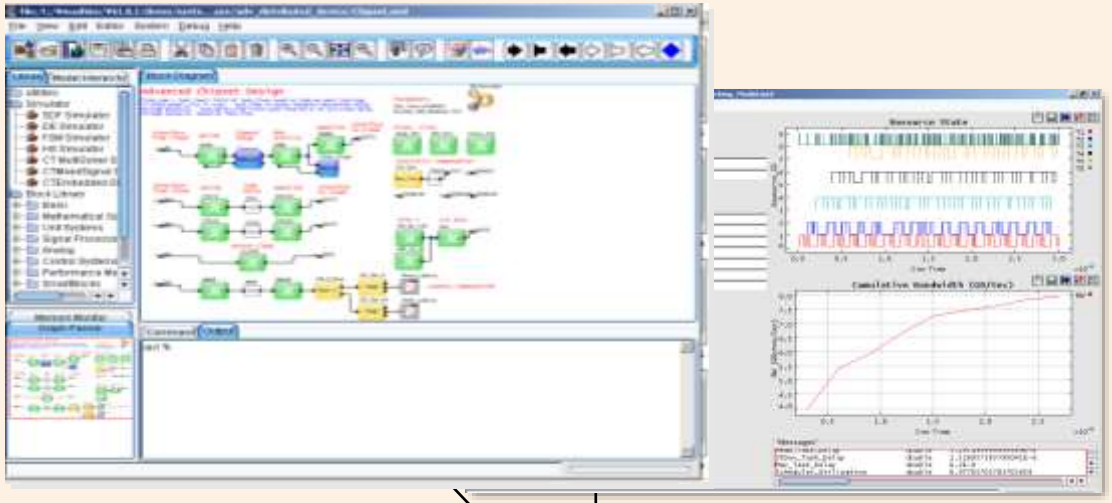
外置应用库
Prototype Ready IP

云管理软件
Neuro

芯神匠+芯神瞳 联合验证方案

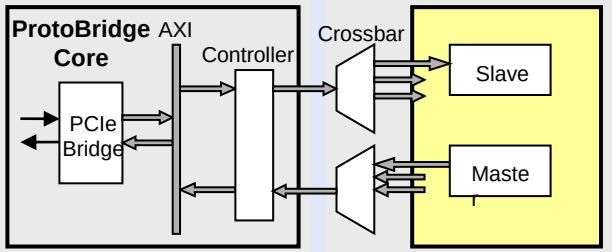
利用现有模型库
- 大幅缩减建模时间

Genesis Environment



Sub-model

myDesign.v



Custom RTL code and peripherals

Synthesize

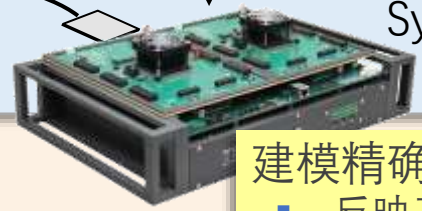
FPGA P&R

Bit Stream

PCIe Cable

Host Computer

Prodigy Logic Systems



快速仿真分析
- 以硬件运行的速度进行仿真

建模精确
- 反映了RTL设计的真实行为

芯神匠+芯神瞳 助力系统开发

芯神匠

- 借助芯神匠快速虚拟原型开发技术，改变传统建模方法来加速概念工程，将依附经验积累的规约转变为实际可展示、可运行、可仿真的模型架构。保证了产品从需求分析到设计实现的连贯性，有效缩减总体项目开发时间，并为后续系统方案的扩展升级或新方案设计提供科学的、可重用的参考依据。

芯神瞳

- 缩短设计映射到FPGA的时间，以最灵活与可扩展的架构体系，以满足不同设计容量、应用程序和设计阶段的需求。

- 透过异构验证方法学混合仿真来覆盖多种验证场景，缩短芯片验证周期，加速客服软件开发

谢谢观看