

Socket 在SoC 设计中的重要性

本文介绍在现代片上系统(SoC) 设计中使用开放式内核协议(OCP) ，解释了为什么标准的工业套接口在富有竞争性的SoC设计很重要，说明了OCP 如何实现接口功能。讨论中说明了加速SoC设计以满足更短的上市时间的必要性，和复用IP 的优势。最后，本文讨论了三种不同的实现方法，阐明OCP 给半导体内核设计带来的灵活性。

问题

近年来，半导体工艺的改进和日益增长的市场压力使上市时间和设计重用成为半导体工业的热门话题。显然，减少SoC设计周期可以减少上市时间。设计重用表面上很简单 - 一次设计，多次使用。但是减少SoC设计时间和实现设计重用被证明是很困难的。

由Moore 定律知，每过18个月，随着工艺的改进集成电路密度增加一倍。这允许在给定大小的半导体面积上极大增加电路范围和功能，但由此引起的制造成本的增加确是微不足道的。比如，在过去的5年里，半导体门数从20万到50 万门增加到上千万门，甚至2千5百万门。门数提高了50倍，这也是SoC 设计得以实现的根本原因。

随着密度增加的还有性能的提高，设计者努力同时降低原型设计和后续改进设计的设计周期。这都是因为激烈的市场竞争压力，产品生命周期的缩短和功能的提高导致设计频繁修改，设计时间减半。见表1 。

	1997	1998	1999	2002	Delta
工艺技术	0.35m	0.25m	0.18m	0.13m	~7x
门数	200-500K	1-2M	4-6M	10-25M	~50x
设计周期(月)	12-18	10-12	8-10	6-8	~2x
后续设计周期(月)	6-8	4-6	2-4	2-3	~2x

表1不断增加的设计复杂度和不断减少的设计周期时间

总之，由简单一道数学题知，在一半的设计时间芯片，电路增加50倍等同于产率提高100倍。但通常这是很难实现的，尤其是在今天芯片设计的复杂度不断提高、设计周期不断缩短的情况下。其结果就是另外上市时间和内核的复用不断受挫、产品进度表和设计效率都受影响。

缩短SoC设计时间

为解决上市时间问题，首先想到的是并行设计单独的SoC内核和最终的SoC产品，设计企业可以在这方面减少设计时间因为设计的各个环节，包括SoC仿真(时序和性能分析等)都并行进行。

这可以将SoC设计时间减少为单个部件的最大设计时间。单个部件可以是单独的SoC内核，或是SoC集成。不管怎样，开发进度分险得到控制 - 允许在加速设计进度表的前提下保证SoC产品更高可靠性。这也允许进度表更可预测。因为所有的设计是受控的，所有的设计并行进行，问题不是被串行解决。这意味着问题可以更快被发现和解决，设计流程变得非常可预见。

然而，并行设计需要定义分割的每个内核和共享的SoC资源。这是因为内核只执行自身功能，而无系统信息。比如，PCI

接口内核或MPEG 解压内核只执行自身的功能，而不需了解SoC互连机制的任何信息。相似的，互连机制处理传输信息时，像进行判决控制、地址映射、数据传输时不需要了解内核提供任何信息。幸运的是，这种方法已经存在而且被研究好多年了，这种方法叫做分层法。

分层法已成功地被运用在网络上，在每一层定义了不同的功能和与相连层之间的接口。在软件上也一样，每个功能和任务都定义自身的功能和接口。分层法已经在不同的领域被证明是很有效的。

用分层法来解决问题

分层法自然地将系统处理的部件分离出来，部件可以是大型软件程序的一个软件模块，或SoC中的半导体内核，这对SoC设计者更为重要，两者的原理是相同的。分层法的优势有：

- 减少设计时间
- 更简单的验证
- 增加IP 复用

分层法使得设计团队能将设计努力分解成许多独立活动，能被同步处理，因为这些活动间相互依存度最低。缩小单个活动的时间能增加成功的几率，也更容易被验证。这可以大幅地加速最终产品的交付时间。

分层法使内核能重用在不同的系统内。采用分层法方法，其他的系统资源只需考虑其他的设计要求，而不需考虑单个内核的功能。如果用正确的内核接口设计，内核只在支持接口的子系统内不用改变的被重用。通过选择工业级标准接口，不需要增加额外的时间实现重用，因为所有内核都需要这样的接口。但是，什么是内核接口呢？这个问题的答案就是套接口socket 。

套接口

现在出现在SoC 设计者面前的问题曾经困扰了局域网(LAN) 设计几十年。最后，LAN 设计者建立了定义好的接口，包含了物理连接和物理连接上信息交换协议。这些行业惯例随后用在计算机工业，以能提供独立设计的和功能多样性的即插即用型产品，使商业公司能整合在一起定制LAN 的配置。这种方法已经被证明是可行的。

理想的SoC 套接口要求

理想情况下，SoC socket 使得内核设计者能集中精力内核功能的设计和内核之间的互连(如SoC 互连、USB 、802.11b 、SDRAM 等)。相似的，SoC 系统集成者可以集中于SoC 时序、内核服务带宽、延迟要求和独立于内核功能的物理布线设计。因此，Socket 能提供必要的物理和交换协议，以实现定义好的分层法。

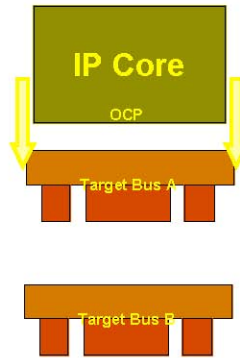
理想的SoC socket 必须是传输执行未知的。SoC 内核通过接口连接核内传输机制，但不需了解具体的传输机制(计算机总线、纵横交叉、可配置片上网络)。

这一点很重要。否则的话，进行内核设计时设计者需要了解传输知识，这不利于在使用不同的传输机制SoC 设计中进行内核重用。传输未知(transport-unaware) 方法,能保证设计的独立性，允许系统设计者根据系统需求选择最优的互连方式

最后，由于带宽要求的多样性，理想的接口应该允许设计者从不同的方面配置不同的接口方式。不同的方面包括接口数

据位宽、交换握手协议、交换反馈等等。这使得SoC 设计者精简SoC 内核设计，在满足内核和SoC 设计要求的前提下，使芯片复杂度和面积最小化。

- **OCP is a Core-Centric Protocol Interface:**
 - **Facilitates unrestricted delivery of ALL Core signals and features**
 - **Enables unconstrained interface bridge to ANY bus structure**



OCP 是一个面向内核的接口：

- 便于所有内核信号和功能不受限制的传递
- 能不受限制的接口桥接任何总线结构

SoC 套接口

据我们现在所知，使内核复用最大化的解决方案要求采用经精密构思的、专用的、以内核为中心的协议作为内核接口。通过选择工业标准，内核设计者不仅能保证内核复用于自己公司设计的内核，也可以通过IP 授权协议让内核复用在其他公司设计的内核中。最后，能最大化地和第三方IP 取得授权和合作进行SoC 设计。换句话说，他们获得了SoC 设计的快捷性和通过IP 授权取得收入的能力。

此外，严格的IP 内核接口规范，与最优化的系统互相结合，使得内核设计者能集中于设计内核的功能。这就消除了终端用户一些必需了解的复杂知识，终端用户可能使用内核和应用上的其他IP 内核。内核需要的接口能使内核从系统中剥离出来。接口具有socket 的属性 - 一种强有力的、精简的、已被工业界广泛应用的接口。

通过这种方法，系统集成者利用分层硬件达到分割芯片部件的目的，设计者不再需要在无数种的内核协议和核内传输策略徘徊。用标准IP 内核接口可以避免在每次SoC 集成时去适应每个内核，系统集成者可以集中解决SoC 设计上的问题。因为每个内核已从片上互连分离出来，交换内核来满足系统和市场的需求的努力将变得微不足道。

总之，对于真正内核复用，SoC 集成者将内核集成在SoC 时，不会修改被复用内核的信息。所以当总线位宽、总线频率和总线电气负载改变时不需要对内核进行修改。换句话说，一个真正的套接口使内核与SoC互连机制隔离开。Socket 支持工具、间接地支持协议、检查器、模型、测试向量和测试生成器。这允许独立的内核设计，设计出无需考虑内核互连的即插即用的模块，也使得能并行设计模块，这样能节省大量的设计时间。

接口解决方案的要求

内核接口设计的要求是多样化的，没有一个单一专用的设计方法能满足所有的要求。一个标准内核接口规范需满足：

- 不同等级的一系列要求
- 允许设计者能应不同的要求配置接口实例(总线位宽、数据握手等)
- 支持更多的数据流信号

- 误差
- 中断
- 标志和软件流控制
- 控制和标志
- 测试
- 捕捉所有的内核和系统之间的信号

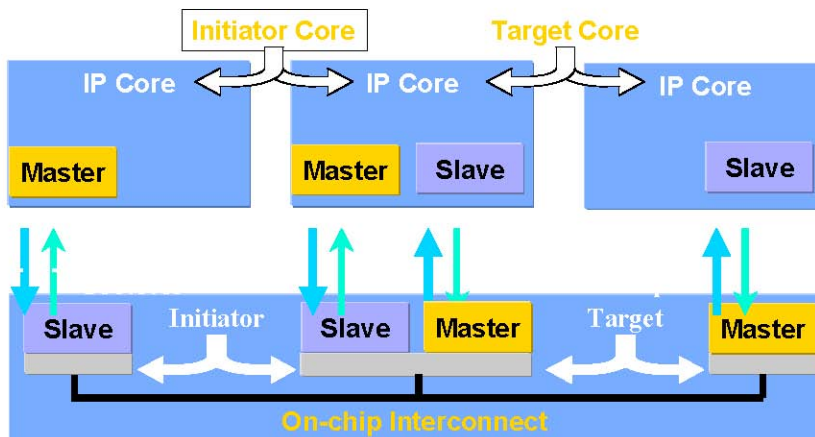
OCP 介绍

OCP 是免费使用的、与总线独立的协议，它满足上面的以内核为中心的所有协议要求。特别的是，它能完全满足IP 内核通信机制的所有要求。作为一可配置的接口，OCP 有一系列能共享公用定义的协议组成。

OCP 通过对基本OCP 数据集的可选择扩展能支持边带信号。这些边带信号包括:重启(reset) 、中断(interrupt) 、误差(error) 、控制/状态信息(control/status information) 等。另外，类标志总线能满足任何内核发信号的要求。可选择的OCP 测试接口扩展在集成SoC 时支持扫描、JTAG 、时钟控制、内核调试、制造测试。

系统设计者能配置OCP 以很好地满足内核需求。通过简单的配置程序，OCP 用简单且精简的OCP 接口能支持简单、低性能内核，也用复杂的接口支持复杂的、高性能的内核。

IP 设计者通过OCP 接口能实现IP 内核设计，设计者不需要解除OCP 外的终端应用知识，允许全球设计团队的人员互相独立的设计。系统集成者可自由选择片上互连方式，然后有效的转换成OCP 接口连接内核。



OCP-IP 成员接受CoreCreator® 工具为OCP 协议达标环境，并封装成一起以有效的IP 内核复用。这对所有的OCP-IP 会员免费开放使用。

这些实例说明了三个不同功能的内核如何使用OCP 接口，这些实例有:

- 1 总线桥(bus bridge)
- 2 处理器接口(processor interface)
- 3 内存接口(memory interface)

第一次是孤立地讨论每个内核的OCP 接口，之后提出了一些附加的公共信号.信号名称是OCP 协议的一部分，关于完整

的信号定义读者可以参考开放式内核协议参考手册。OCP 规范可在网站免费获得www.ocpip.org。

在一个nutshell, OCP 互连有主机实体(Master entity)和从机实体(Slave entity).一些简单的OCP 术语和协议:

- 主机发出的信号以首字母M开始
- 从机发出的信号以首字母S开始
- 协议内有简单的握手信号
- 主从机可同时进行信号流控制
- 所有的传输和信号与OCP 时钟的上升沿同步

例1 - 总线桥

总线桥可以将PCI、USB 和其他总线标准与OCP 互连。控制器有外部的接口像PCI 或USB 接口，内部的SoC 接口就是OCP。

总线桥在内部SoC 互连机制上扮演的是主从机的角色，主机发出信号给目的地址，而从机向总线桥或内部控制状态寄存器读写。

从机包含简单的OCP 接口和一些边带信号，从机内核的OCP 信号集主要有：

- MCmd 主机命令信号(像读/写)
- MAddr 主机地址信号(可达32 bits)
- MData 主机数据(写数据; 8,16,32,64,128 bits wide)
- SCmdAccept 从机命令接受信号
- SResp 从机响应信号
- SData 从机数据(读数据, 数据大小与MData 一致)
- SError 从机误差信号, 误差来自桥
- SInterrupt 从机中断信号, 中断来自桥
- Control 总线桥的控制位
- Clk 时钟信号
- Reset_N Reset 信号

这个实例的接口只用一个中断，就是从机中断。如果中断超过一个，从机标志位(SFlag) 能提供8个额外的中断。

总线桥主机的信号有:

- MCmd 主机命令信号
- MAddr 主机地址信号(可达32 bits)
- MData 主机数据(写数据; 8,16,32,64,128 bits wide)
- MBurst 主机Burst
- SCmdAccept 从机命令接受信号
- SResp 从机响应信号
- SData 从机数据(读数据, 数据大小同MData 一致)
- Clk 时钟信号

- Reset_N Reset 信号

可选择的OCP 线程信号与PCI 结合能增强接口的通信能力，支持接口并行和乱序处理传输信号。可选择的OCP 扩展信号支持多线程。不同线程间的事务没有排序限制，可以乱序执行。但在一个线程里的数据流，所有的OCP 传输应保持顺序。PCI 接口的线程可用于不同的memory 和I/O 。

- MThreadID 主机线程ID (可达16 个不同的线程)
- SThreadID 从机线程ID (可达16 个不同的线程)

例2 - 处理器接口

处理器接口只需一个OCP 主机，信号与总线桥主机相似，但对较少的单字传输增加了位使能信号。这个实例的内核OCP 信号集可能有：

- MCmd 主机命令信号
- MAddr 主机地址信号(可达32 bits)
- MData 主机数据(写数据; 8,16,32,64,128 bits wide)
- MBurst 主机Burst
- MByteEn 主机位使能信号
- SCmdAccept 从机命令接受信号
- SResp 从机响应信号
- SData 从机数据(读数据, 大小与MData 一致)
- SError 从机误差信号, 输入处理器
- SInterrupt 从机中断信号, 通常是NMI pin
- SFlag 从机标志位信号, 其他给处理器的中断输入(可达8 flags)
- Clk 时钟信号
- Reset_N Reset 信号

最新可用的处理器支持指令并行和数据cache-miss 预取，OCP 线程支持这个。因此，用以提高这些处理器的并发内存操作的信号有：

- MThreadID 主机线程ID (可达16 不同的线程)
- SThreadID 从机线程ID (可达16 不同的线程)
- MThreadBusy 主机线程忙信号
- SThreadBusy 从机线程忙信号

主从机线程忙信号用以控制每个线程流。虽然处理器可能有好几种取指等待出来，但没有资源同时处理所有信号。所以线程忙信号使OCP 主机处理器或者目标从机能控制传输流。

例3 - 内存子系统

与内存子系统互接口的有DRAM, DDR, SRAM 或FLASH 。OCP 信号需要复杂的OCP 扩展来充分利用带宽。内存子系统可以是支持很多个memory bank 的多线程。内存控制器也做一些简单的扩展，如burst 和位使能，以有效地服务请求。

这个实例的内核OCP 信号集可能有：

- MCmd 主机命令信号
- MAddr 主机地址信号(可达32 bits)
- MData 主机数据(写数据; 8,16,32,64,128 bits wide)
- MBurst 主机Burst
- MByteEn 主机位使能信号
- SCmdAccept 从机命令接受信号
- SResp 从机响应信号
- SData 从机数据(读数据, 大小与MData 一致)
- MThreadID 主机线程ID (可达16 不同的线程)
- SThreadID 从机线程ID (可达16 不同的线程)
- MThreadBusy 主机线程忙信号
- SThreadBusy 从机线程忙信号
- Clk 时钟信号
- Reset_N Reset 信号

内存子系统在主机数据和从机数据使用的数据位宽比连接到内存的数据位宽要大一些。这使得每个系统互连传输更有效。

每个例子的共同信号

先前的实例都有扫描信号和JTAG 信号, 这些测试和扫描信号对每个内核都相同, 但扫描链的数目是不同。在OCP 中测试结构不是孤立的实体, 而作为接口的一部分。这完善了套接口, 不仅仅只作寻址数据流方面的考虑。有扫描和JTAG 的内核需要的信号有:

- ScanCtl 扫描控制信号
- ScanIn 扫描输入(可达256 扫描端口或扫描链)
- ScanOut 扫描输出(可达256 扫描端口或扫描链)
- ClkByp Clock Bypass, 同测试时钟代替正常时钟
- TestClk 测试时钟
- TCK JTAG 测试时钟, 按照IEEE 1149.1 定义所有信号
- TDI JTAG 测试输入
- TDO JTAG 测试输出
- TMS JTAG 测试模式选择
- TRST_N JTAG Reset

结论

标准的套接口内核协议对SoC 设计社区非常重要。OCP 是唯一的, 完整的、全面支持的和已被证明的套接口。采用OCP 能避免互相不兼容或私有权引起的问题, 也有利于扩展IP 内核商业化交易市场。

完整全面支持的以核为中心的OCP 相比以前的以总线为中心的协议有很多重大的优势。OCP 是以内核为中心、公开授权、免版权的内核接口协议。它不会限制和干涉内在内核的信息。OCP 是可升级的、可配置的, 以满足不同内核和SoC 设计需要的不同的通信要求。

拥有OCP 接口和OCP 互连系统使内核真正的模块化，集成“即插即用”，允许设计者自由选择内核以最优应用互连系统。设计内核和设计系统能并行进行，可以缩短设计时间。另外，因为内核没有系统逻辑，使得内核能被复用和重新被设计。

最后在写OCP 规范中，验证和测试在多重设计中完全是可移植的，不需要对某个特定的接口桥哪怕做最小的调整详细的OCP 规范可在网站免费获得www.ocpip.org