安腾技术文集(修订版)

solution center

invent

Itanium 处理器系列的 EPIC 架构

前言

在计算机系统所有部件中，微处理器是最核心、也是性能增长速度最快的部件。20 世纪 80 年代 RISC 技术问世以来，处理器内部器件密度增加了 200 倍、时钟频率增加了 30 倍、而处理器的峰值速度却增加了 900 倍。由此可见，处理器速度的提高来自多方面的因素，其中最主要的是采用先进的架构。在处理器发展史上，架构每次革新都给处理器性能提高带来重大突破。根据摩尔定律，处理器的密度每 18–24 个月提高 1 倍。但是，新工艺并不能自动提高处理器性能。先进的架构能够利用工艺的发展提高主频、并行度和处理器性能。反之，落后的架构往往会阻碍充分利用新技术和新工艺来提高性能。最突出的例子是，80 年代初传统的复杂指令集(CISC)架构越来越不能适应技术发展和应用需求，于是，出现了精简指令集(RISC)架构，为处理器性能提高开辟了崭新的天地，促使高端计算机应用进入了 RISC 计算阶段，继而发展到 64 位 RISC 计算阶段。

今天，处理器技术的发展又进入了新的转折和变革时期。RISC 架构已经演变为成熟的传统技术，也面临非常多的基本挑战、很难继续跟上技术更新和应用需求的发展步伐，满足信息时代基于 Internet 电子商务和人类探索未知世界的高性能技术计算等高端应用的需求。尽管 RISC 架构有种种优点，但是它在发展过程中产生的一系列缺点又限制了快速提高处理器性能，其中最主要的是：

- **封闭性**：RISC 架构是厂商专属的封闭性架构，产品批量很难扩大；
- **有限的芯片资源**：为了保持高性价比，RISC 架构只能使用较少的芯片资源；
- **使用串行语义**：RISC 处理器没有必要的资源和机制来“表达”编译程序所发现的并行指令段。它所执行的目标程序是串行的、即“隐性并行”的；

这些都是 RISC 架构的本质特性。大多数 RISC 处理器不得不采用超标量、无序执行技术、以复杂的芯片逻辑从目标程序中重新(反复)发现可并行执行的指令段，使得 RISC 处理器设计难度越来越大、设计周期越来越长，制约了快速提高处理器性能。一方面许多重要的高端应用迫切要求使用性能更高的处理器，另一方面处理器制造工艺正在继续按照摩尔定律预示的速度提高，传统 RISC 架构开始应付不了面临的许多挑战、由推动芯片性能提高的动力逐步转化为制约进一步发展的因素。

Intel 和 HP 早就清楚地意识到为了推动处理器技术进一步发展，必须象桌面和个人计算机那样、走 IA–32 与 Windows 相结合开放性的道路。它们从 1994 年开始合作开发新型的 64 位芯片，选择了一个与大多数 RISC 微处理器大不相同的方向，推出了一种新的 64 位处理器架构 Intel Itanium EPIC 架构(有时也称为 IA–64 架构)。

EPIC 架构的设计思想基于如下的原则：

- **开放性**：硬件向所有 OEM 厂商开放、软件使用开放性的操作系统，力争成为工业标准、实现大批量生产；
- **丰富的芯片资源**：提供必要的芯片资源来记录有关编译程序所发现的并行指令段的信息并提高芯片的并行处理能力。通过扩大批量来实现高性价比；
- **使用并行语义**：允许编译程序把源程序转换成由一系列可并行执行指令段组成的序列、即使用并行语义来表达目标程序；

  EPIC 架构处理器允许编译程序在目标程序中明显地标识出可并行执行的指令段。相反，

RISC 使用串行(隐藏着并行)的目标程序，必须使用复杂的芯片逻辑来发现可并行执行的指令段。这成为 IA—64 架构与 RISC 架构之间最本质的差别。因此，HP 在发明这一架构时把它称为显性并行指令计算(EPIC)架构。EPIC 架构既不是 RISC 也不是 CISC 架构，它实质上是一种吸收了两者长处架构。EPIC 架构允许处理器使用简单的硬件逻辑、实现高度的并行处理，从而为利用新的工艺成就所提供的资源扩展潜力、简化芯片设计、缩短开发周期、加速发展步伐奠定了坚实的基础。

本文包含两章：第一章分析传统的 RISC 架构所面临的各种挑战，并指出这些挑战大多是根本性的，必须跳出 RISC 架构方能继续前进；第二章具体介绍 Intel Itanium 系列的 EPIC 架构以及它所拥有的一系列用以提高处理器性能的先进特性，并剖析对 EPIC 架构的疑虑。本文的要点是揭示处理器架构发展和变迁的必然性，指出当前 EPIC 架构是新兴的架构，为充分利用芯片工艺新发展、按照摩尔定律预示的速度提高处理器性能开辟了康壮大道，必将发展成为支持高端应用新的主流平台。

表 1 处理器架构和工艺发展历程

时期	1987 ——1996	1996 ——2001	2001——2002	2003 ——
芯片密度	2.0μm — —0.35μm	0.35 — —0.18μm	0.18μm——	0.13μm——
芯片资源	0.1M ——10 M	10M ——100M	25 M——214 M	更多的晶体管数
芯片面积	80 mm^2——200 mm^2	200mm^2——400 mm^2	300mm^2——450 mm^2	面积将更大
时代概貌	RISC 技术高速发展	RISC 技术发展缓慢	EPIC 技术兴起	EPIC 技术发展
设计思想	利用 RISC 相对于 CISC 的优势，简化处理器设计、提高主频和并行能力	不得不采用在有限资源下，通过增加处理器智能来提高性能的设计思想	提供足够的资源、发挥编译程序和处理器合力提高性能，大大简化处理器逻辑和缩短设计周期	在继续发挥 EPIC 体系结构优势和潜力基础上，吸收 RISC 技术的长处如 EV8 的 SMT 技术
主流技术	流水线和超流水线	超标量多路发送、无序执行、转移猜测，同时利用编译程序实现静态指令级并行	显性并行指令计算(EPIC) 使用先进编译技术优化程序、产生预测信息，提高并行度	并行度更高的 EPIC，使用更强的编译程序技术
市场状态	开创 64 位计算时代，占领高端应用市场	市场规模和批量较小	市场规模和批量大，从而提高性能价格比	进一步体现统一平台的市场优势
概述	当时的工艺条件下得在到充分发挥 RISC 优点，主频增加 40 倍，速度提高上千倍，符合摩尔定律	处理器逻辑越来越复杂，逐渐失去 RISC 特点 技术更新周期加长不能充分利用编译程序所产生的	技术和工艺发展为 EPIC 的兴起创造了条件 EPIC 开辟了采用新工艺和技术的坦途充分利用编译程	开创开放性系统和企业计算的新时代 提供更高的静态和独特指令级并行度 为摩尔定律注入新

		许多有用信息限制了采用最新工艺	序的优化功能和所产生的信息	的生命力

一、RISC 架构面临的挑战

本章分析传统的 RISC 架构对提高处理器性能的种种约束，指出这些约束大多是根本性的，必须跳出 RISC 架构，方能解除这些约束、找到满足应用需求和适应工艺更新速度的发展途径。

1.1 RISC 处理器指令级并行度太低

人们主要通过提高 IPC(每个周期执行的指令数)和主频来提高芯片的性能。为了提高 IPC，必须提高处理器的指令级并行度(ILP)。所谓 ILP 是指处理器同时执行多条指令的能力，即处理器在每个时钟周期内发送和执行指令的能力。这就要求处理器：(1)能够找到和标识程序中可以并行执行的指令段； (2)具有充分的资源在最短时间内发送和同时执行可并行执行的指令段。本节指出指令并行度太低是 RISC 处理器对性能提高的最大制约。

串行性是 RISC 架构的本质特性

在应用开发过程中，程序员利用各种高级语言、基于串行执行语义(即按照程序规定的次序在计算机中“依次”执行)编写软件，由此决定执行指令的次序、造成指令间的潜在依赖性。尽管编译程序知道许多指令可以并行执行、或者可以通过适当调度(改变次序)提高执行效率，RISC 处理器并没有能够(也没有办法)充分利用这些有用的信息来提高指令并行度。编译时产生的许多宝贵信息都只能白白丢掉，所得到的仍然是一个基本上串行的目标程序。但是高性能需要并行执行，而并行执行要求指令级的独立性。RISC 处理器不得不在执行串行的目标程序过程中利用越来越复杂的芯片逻辑来发现(并且反复地重新发现)其中可并行执行的指令段。

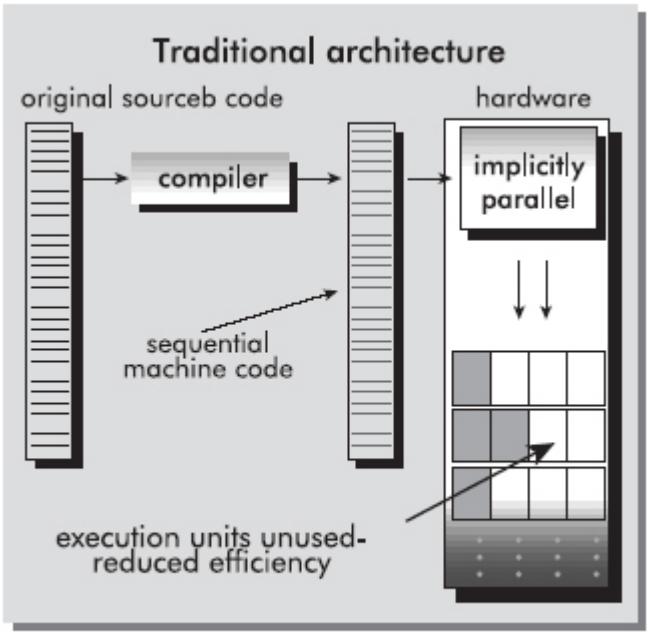

图 1 RISC 架构的隐性并行机制

当然，人们也可以通过优化 RISC 编译程序来改善目标程序的并行度。但由于 RISC 处理器一般只提供较少的芯片资源。RISC 处理器没有足够的芯片资源来存储编译时产生的有利于并行执行的信息，因此不可能充分利用处理器和编译程序的硬软件合力来提高性能。

编译程序知道可利用的并行指令段，但是没有“办法(词汇)”来表达它，处理器必须使用复杂的硬件(反复重新)发现隐藏在指令流中、编译时已知的并行指令段。目标程序的串行性和硬软件脱节(过分依赖于硬件智能而没有充分利用编译程序)是 RISC 架构的本质特性，也是它最本质的缺陷。

基本代码块中指令级并行度十分有限

人们一般把只有一个入口和一个出口的程序段、也就是其中不含条件转移和子程序调用的程序段称为基本代码块。由于编译时无法预测条件转移指令的转移方向，因此 RISC 处理器只能利用芯片逻辑、在程序执行过程中发现基本代码块中的指令级并行。但是大多数企业应用软件都频繁地使用条件转移指令、基本代码块的平均长度一般只有 3–5 条指令，在基本代码块内只能取得有限的指令级并行。

虽然，许多 RISC 处理器都试图通过增加芯片内处理部件的个数来提高性能(超标量处理器的设计思想)，但由于指令级并行度太低、处理器不能提供足够多的并行指令来满足多处理部件的工作需要。实际测试的结果表明：许多超标量 RISC 处理器的处理部件 50%左右时间都是空闲的。这说明在 RISC 架构下提高性能的努力已经碰壁，很难再通过增加部件来提高 RISC 处理器的性能。

转移预测的局限性

RISC 处理器需要跨过条件转移指令寻找指令级并行方能进一步提高 ILP。为此，需要预测转移指令方向，即按照预测的方向继续执行指令流水线、跨越基本块边界继续找寻指令级并行。如果预测正确，就好像消除了转移指令、扩大了基本块一样、确实能够提高 ILP。但

是，转移预测也有其局限：因为预测不一定正确。如果出错往往需要付出很大代价来弥补预测的错误，包括防止错误地执行的指令(如内存操作(装入)，浮点操作)产生意外的副作用。这将大大影响处理器的性能。如何消除企业应用中大量存在的条件转移指令影响成为进一步提高 RISC 处理器性能的一大障碍。

1.2 RISC 架构下的指令间相关性

计算机系统所有部件中处理器不仅速度最快，而且也是速度增加得最快的部件。实际测试结果表明，提高计算机系统性能的关键是确保内存系统能够及时供给处理器足够的数据、使得处理器觉察不到内存系统的延迟、即必须降低处理器的可觉察延迟。这一方面需要提高系统总线和内存子系统的带宽，另一方面必须改进处理器设计。后者往往是提高计算机系统性能、降低可觉察延迟的更有效途径。解决的方法之一是在保证结果正确的条件下尽量提前执行访问内存的指令，使得处理器与内存系统“并行地”操作，“提前”把数据取到高速缓存中供处理器使用。大多数 RISC 处理器都采用基于芯片逻辑的无序执行技术来实现这一方案，能够不按照访问内存指令在程序中位置、把它移动到前面执行。例如，Alpha 21264 处理器利用动态调度、寄存器换名、转移预测等技术来实现无序执行、取得了良好的效果。但由于程序中含有许多内存存储和条件转移指令，造成指令间的写——读相关性和过程相关性。例如，考虑如下的指令序列：

add r1,r2;　r1+r2 →r1

move r3,r1; r1 → r3

第 2 条指令虽然能够取出和译码，但是必须等待第 1 条指令执行完毕后才能执行，因为它需要使用前者的结果。这种情况称为写－读相关性、即不能把第 2 条指令移到前面执行。此外，条件转移(有发生或不发生两种情况)之后的指令有对转移的过程相关性，直到转移指令执行后才能确定是否在程序执行路线上、真正需要执行。因此，程序中大量存在的条件转移指令和存储指令往往造成指令间相关性、成为移动取数指令的壁垒。特别是，随着具有高可伸缩性的 NUMA 架构内存系统在高端计算机系统中更加广泛的使用，处理器访问远程内存的延迟比访问本地内存大得多，缩小内存延迟影响、成为提高计算机系统性能更加迫切的任务。RISC 处理器很难突破这些壁垒、提高 ILP。如何克服指令间的相关性、减少内存延迟影响，成为提高 RISC 处理器性能的又一障碍。

1.3 RISC 架构对芯片资源的限制

如上所述，RISC 处理器批量较小。为了提高性能价格比和竞争力，RISC 处理器设计师们不得不使用较少的芯片资源，使得 RISC 往往只能使用较小的寄存器空间、较少的浮点处理部件和芯片上缓存、要求指令共享许多芯片资源(如寄存器、条件标志等)。这不仅使本来独立的指令产生依赖性、造成指令间的依赖性(资源相关性)，影响处理器并行处理能力、降低高速缓存的命中率，而且也限制了编译程序“表达”并行的能力。

RISC 处理器的资源限制是发展历史中形成的、根深蒂固的本质限制。64 位 RISC 芯片是从 32 位发展起来的。在 32 位时期，各厂商为了改进和发展原有的 CISC 产品，几乎都建立了自己的芯片设计队伍和生产机构。于是，RISC 从一开始就走上了各自为政、以邻为壑的发展道路。各主要厂商都拥有自己 64 位 RISC 微处理器产品系列、名为“开放”实际上不完全开放的 UNIX 操作系统、号称“兼容”其实并不完全兼容的中间件和应用编程界面(API)以及

一大堆引以自豪的专利。此外，由于基于多种 RISC 芯片的服务器和工作站分割市场，每一种 RISC 芯片的市场份额都不可能太大、产品的批量也较小。特别是，近年来基于 IA-32 的 x86 架构芯片的开放性工作站基本上夺取了基于 RISC 的 UNIX 工作站市场后，进一步限制了 RISC 产品批量的扩大。为了保持价格/性能的竞争力，RISC 芯片设计师们不得不控制芯片资源量。虽然，RISC 处理器也可以使用换名等技术部分地克服资源相关性，但是 RISC 架构的封闭性和小批量，使它不可能充分利用处理器工艺发展成果、采用大量的芯片资源。RISC 架构的芯片资源有限性也是进一步发展的另一个基本约束。

1.4 RISC 处理器设计周期长

由于条件的限制，传统的 RISC 设计师们不得不采用通过在芯片上增加更多的逻辑和智能（所谓“聪明的处理器”）同时又不必采用太高的工艺、增加太多的资源的途径，来提高指令并行度。虽然 RISC 处理器也利用经过编译程序加工、更加有利于发现独立指令段的目标程序，但是 RISC 处理器没有足够资源来存储编译程序所产生的信息，RISC 架构下的目标程序本质上是串行的。RISC 处理器主要依靠硬件即芯片逻辑在程序运行时发现和处理并行。大多数 RISC 处理器都还具有超标量、无序指令发送机制，使得处理器能够发送多条指令并根据程序的运行实际结果、改变指令发送和执行的次序，而不是阻塞处理器的运行。这种超标量无序执行技术的主要优点是能够在有限的工艺和资源条件下，大大提高指令并行度。最出色的例子是，Alpha EV6 采用无序执行技术实现了在基于与 EV56 相同的 0.35μ工艺条件下，把性能提高了 1 倍以上，使芯片不仅具有高性能，而且在批量不太大的条件下具有较高的性能价格比。

但是这种技术也有其缺点，最主要的是：超标量、无序执行技术要求处理器具有较高的智能和复杂的逻辑，使得芯片上控制逻辑面积不得不随着功能部件的增加而平方增加、其结构越来越复杂、设计难度也越来越大，使得许多 RISC 芯片的设计周期越来越长、经常不能按期上市，也使得 RISC 处理器很难充分利用现代工艺的新成果、按照摩尔定律预示的速度发展、满足应用发展的需要。

1.5 RISC 架构其他局限

除了上述的本质局限外，RISC 架构还有一系列其他局限。

过程调用开销太大

现代应用软件越来越多地采用面向对象、Java 等模块化方式编程，程序变得含有更多的过程调用。传统的 RISC 处理器寄存器空间不能由调用和被调用的程序所共享。调用/返回都需要保存/恢复寄存器，不必要地插入了附加的寄存器保存/恢复，增加了过程调用的开销、降低了应用软件的运行性能。

循环开销太大

各种应用软件中都包含许多循环执行的程序段。传统 RISC 处理器由于没有更加有效的机制，往往采用循环展开的方式来提供 ILP，大大增加了程序执行时代码长度。此外，RISC 处

理器上循环开始/循环结尾的开销，也往往进一步增加代码长度、影响运行性能。

缺少支持具体领域应用的指令

现代高端处理器大量应用于多媒体、流媒体、保密性很强的 Web 和电子商务、高精度浮点计算等领域。RISC 处理器往往缺少或很难设置面向许多重要应用领域的高性能专用指令，影响了基于 RISC 处理器计算机系统在支持许多具体领域应用时的性能。

二、 Intel Itanium EPIC 架构

Intel Itanium 架构是一种开放性的架构。它使用远比 RISC 架构处理器丰富的芯片资源和先进的设计思想、实现显性并行指令计算(EPIC)，即利用先进的编译程序明显地标识出目标程序中可并行执行的指令段，处理器只需使用简单的芯片逻辑来实现高度的并行。并行是 Itanium 架构的本质。它的设计思想和先进特性完全围绕同一个目标：打破 RISC 架构的局限，从根本上解决 RISC 架构面临的挑战，以新的技术和思路实现、增强、表达和利用并行，跟上摩尔定律预示的芯片工艺发展速度，满足实际应用的需要。

2.1 Intel Itanium 架构的设计思想

Intel Itanium 架构是一种面向并行处理的架构，它在吸收 RISC 架构经验教训基础上另辟蹊径，一开始就走开放性的道路，充分利用现代芯片制造工艺发展成果、提供足够的资源；同时，通过力争成为新的高端工业标准、扩大批量来降低成本。EPIC 是 Itanium 架构的基础，它的基本设计思想是：提供一种新的机制、充分利用硬软件协同能力来提高指令并行度：

(1) **用编译程序发现并行**：EPIC 架构要求充分发挥编译程序的作用，在程序员利用串行语义编写的源程序中寻找由可并行执行指令段，把它转换成由可并行执行指令段序列组成的目标程序。利用编译程序来发现并行比利用处理器硬件具有一系列优点：

- 更**广的视野**：编译程序能够观察更宽范围的程序段、从中发现可并行执行的指令段，而处理器硬件由于受到资源和时序限制，只能远比编译程序窄的程序段；

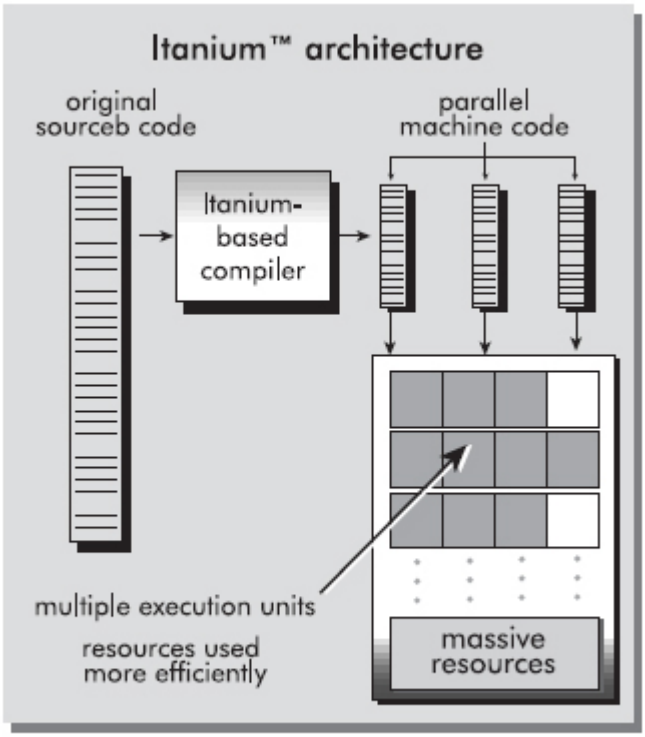

图 2 EPIC 架构的显性性并行机制

- **更加经济**：编译程序可以利用计算机资源和软件逻辑来发现并行并记录在目标程序中，一次发现可以反复使用；而处理器必须使用宝贵的芯片内部资源和逻辑来反复发现程序中可并行执行的指令段；
- **更加有效**：编译程序能够利用先进的编译技术和 IPF 处理器的特性来扩大可并行执行指令段的长度，如利用预测技术消除程序中的条件转移指令，基于猜测机制越过转移和存储指令界限调度指令的执行次序，从而提高指令级的并行度；

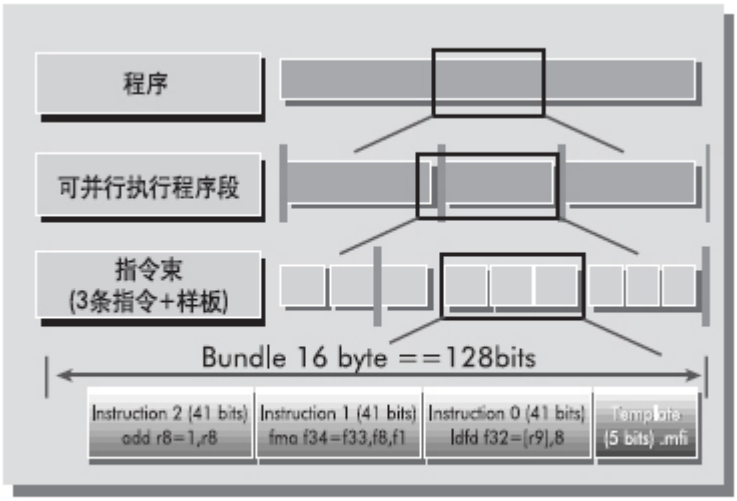

图 3 EPIC 架构的显性性并行指令编码

(2) **显性并行指令编码**：EPIC 架构要求处理器具有足够资源来标识编译程序所发现的程序中可并行执行指令段之间的划分信息(停止位)，实现 EPIC 架构独特的显性并行指令编码。在 EPIC 架构的处理器中，编译程序所产生的目标程序由一系列指令段组成，指令段由没有相关性的可并行执行指令组成、各段之间用显性的停止位划分开来；指令段内的指令以集束方式存在，每个指令束长度位 128 位，包含长度为 41 位的三条指令(共 123 位)和长度为 5 位的一个样板。

样板位的作用是：

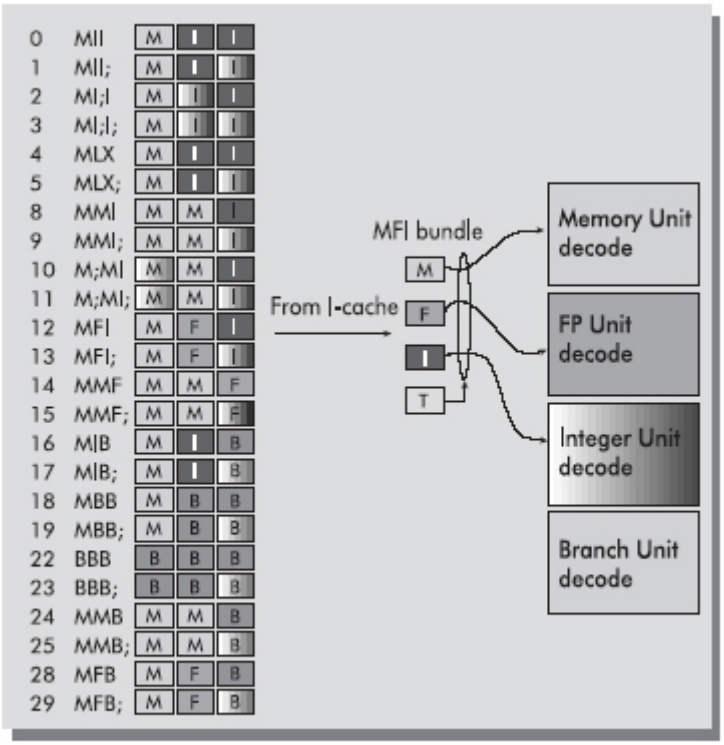

图 4 样板定义图示

- 标识束中指令的类型，Itanium 中指令的类型可为整数运算(I)、访问内存(M)、转移(B)、浮点运算(F)和扩展型(X)等领；
- 标识相关操作，即停止位所在位置(或没有停止位)；
- 定义用于执行本束指令的执行部件；
- 编译程序可以调度功能部件、避免冲突；

长度为 5 位的样板最多可以定义 32 个模式，目前仅使用 24 个。图 4 表示样板模式的含义，其中“;”表示停止位在指令束中的位置。在该图中，样板模式 12 定义一个指令束：包含 1 条 M 型、1 条 F 型和 1 条 I 型指令，不含停止位。这 3 条指令分别进行内存、浮点和整数执行部件译码、找出对应的执行部件；

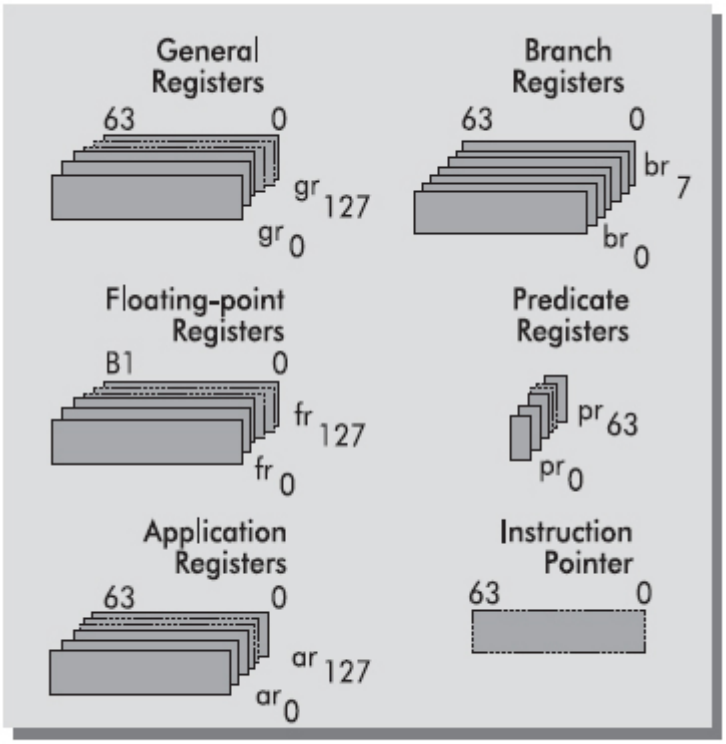

图 5 EPIC 架构处理器的寄存器空间

因此对 EPIC 架构处理器来说，所执行的目标程序中可并行执行的指令段已经由编译程序显性地标识出来了、指令间的数据相关性和过程相关性也已经由编译程序消除了，使得处理器只需使用相对简单的逻辑来反演目标程序、逐个执行其中的可并行指令段，从而实现很高的指令级并行度(ILP)。发现并行工作已经由编译程序完成，无需利用硬件逻辑来反复发现程序中可并行执行的程序段；

(3) **丰富的芯片资源**：EPIC 架构处理器提供远远超过 RISC 处理器的芯片资源、提高处理器反演目标程序的速度，包括：

- 大寄存器空间：EPIC 架构处理器提供大大超过传统 RISC 出来取得寄存器空间，为编译程序通过寄存器换名等技术消除相关性、产生尽可能在寄存器中进行运算的优质目标程序提供充分的余地；IPF 处理器允许程序访问的寄存器包括：

128 个 通用寄存器(用于整数运算及其他操作)；
128 个浮点寄存器；
64 个预测寄存器；
8 个转移寄存器；
128 个应用寄存器；
指令指针(IP)寄存器；

- **具有足够长度的指令**：便于在指令中访问大寄存器空间中多个寄存器和消除程序中转移指令。例如，为了访问 128 个寄存器需要 7 位；
- **足够的指令发送端口和派送网络**：用于提高每个时钟周期发送到指令数和提高发送到速度和效率；
- **多种类型的新型寄存器**：允许编译程序越过条件转移和存储指令的界限调度目标程序中访问内存指令的位置、消除相关性，并弥补调整位置所产生的错误；
- **更多的浮点执行部件**：有利于处理器实现高并行度的浮点计算；

- **更多的整数和多媒体运算执行部件**：有利于处理器实现高并行度的整数和多媒体信息处理；
- **更多的芯片上高速缓存**：有利于提高缓存命中率、降低内存延迟；

相反，传统的 RISC 处理器没有足够的资源来纪录编译程序所产生许多有用的信息，也没有充分利用现代编译程序强大的对程序执行过程的调度能力。虽然在 RISC 架构下，设计师们也使用编译程序来优化目标程序、为处理器提高 ILP 创造条件。但是，处理器所执行的基本上是隐性并行和包含大量相关性的目标程序，发现并行、线路预测、动态调度等提高 ILP 的无序执行技术、完全是由处理器硬件利用复杂的芯片逻辑来实现的。尽管两种架构之间存在着许多差别，但是执行显性并行或者隐性并行目标程序是其中最主要的本质差别。所以最初发明这种新架构的 HP 把它称为显性并行指令计算(EPIC)架构。

表 2 IPF 处理器与 RISC 处理器芯片上资源比较

处理器类型	Intel Itanium	Intel Itanium2	Intel Itanium2 6M (2003 年中)	HP PA—8700+	Sun UltraSparcIII Cu	IBM Power4+
操作系统	HP—UX Windows Linux	HP—UX Windows Linux	HP—UX Windows Linux	HP—UX	Solaris v9	AIX 5L
时钟速率	800 MHz	1 GHz	1.5 GHz	875 MHz	1.45 GHz	1.7 GHz
L1 缓存 (1+时钟周期延迟)	32 KB	32 KB	32 KB	750 KB (i) 1500 KB (d)	32 KB (i) 64 KB (d)	32 KB (i) 64 KB (d)
L2 缓存 (5+时钟周期延迟)	96 KB	256 KB	256 KB+	N/A	8 MB(芯片外，15＋时钟周期延迟)	1.5 MB(共享)
L3 缓存 (12+ 时钟周期延迟)	4 MB(芯片外)	3 MB (芯片上)	6 MB (芯片上)	N/A	N/A	N/A
发送端口	9	11	11	4	4	8
执行部件	9	11	11	4	4	8
持续发送速率	6	8	8	4	4	5
流水线长度	10	8	8	7	14	12
寄存器	264*	264*	264*	32	96	72

2.2 Intel Itanium EPIC 架构的特性

Intel Itanium 显性并行的 EPIC 架构是为了允许编译程序通过显性地控制处理器的执行资源、最大限度提高指令级并行度(ILP)设计的。Intel Itanium 架构不仅具有 64 位寻址能力而且提供一系列使得编译程序能够最大限度地提高指令级并行度(ILP)的关键特性，以提高处

理器的持续性能，为实现 EPIC 利用编译程序和芯片资源提高 ILP 和性能并简化芯片逻辑、缩短设计周期的基本思想开辟了广阔的前景。

预测特性

传统的处理器架构使用超标量技术(如 Alpha，PA—RISC)或者超流水线技术(如 Pentium)来增加处理器中并行执行的流水线数和流水线各阶段的并行度，从而提高处理器的性能。这些技术的主要缺点之一是转移预测错误造成很大的延迟。虽然能够使用硬件逻辑来提高预测的准确度，但仍然会有 10%—20%难以预测的转移会造成预测错误。随着处理器性能的提高，每次预测错误都可能会造成巨大的性能惩罚：如果预测错误，以后在执行流水线上发现这些错误，将使得处理器重新“转向”正确的转移路径。这样的重新转向造成流水线中待处理的许多指令失效，不得不花费很多执行资源重新填充流水线。流水线内容的无效倒换的延迟成为处理器性能下降的一个主要原因。在过去 10 年中，随着流水线长度的增加(超流水线)和并行功能部件的增加，预测错误对性能的影响变得越来越严重。

EPIC 的转移预测特性，允许使用编译程序消除预测错误转移的影响。编译程序将产生多个转移指令。当处理器执行被预测的转移时，只有正确的转移“猜测”被使用(预测寄存器置位的指令)，其余的未使用路径将被丢掉。执行将毫不停顿地沿着预测的目标转移继续下去(“0”周期转移延迟)。这一编译技术称为条件语句转换技术(“if—conversion”)。EPIC 架构提供63 个可寻址预测位,这些预测位控制几乎所有EPIC指令。零时钟周期预测是Intel Itanium 架构的关键优点之一。随着计算机企业应用的普及，应用软件中转移指令比例越来越大，EPIC 架构预测特性的作用也越来越大。EPIC 架构引入零时钟周期预测机制来消除转移指令使得处理器不必花额外的时钟周期来处理转移：

首先，所有 Itanium 指令都包含一个预测寄存器作为附加的输入，指令仅当预测正确时才被执行。因此，Itanium 的指令实际上是 “If(预测寄存器) 指令操作”形式的，仅当指令所引用的预测寄存器为真时指令操作才被执行；

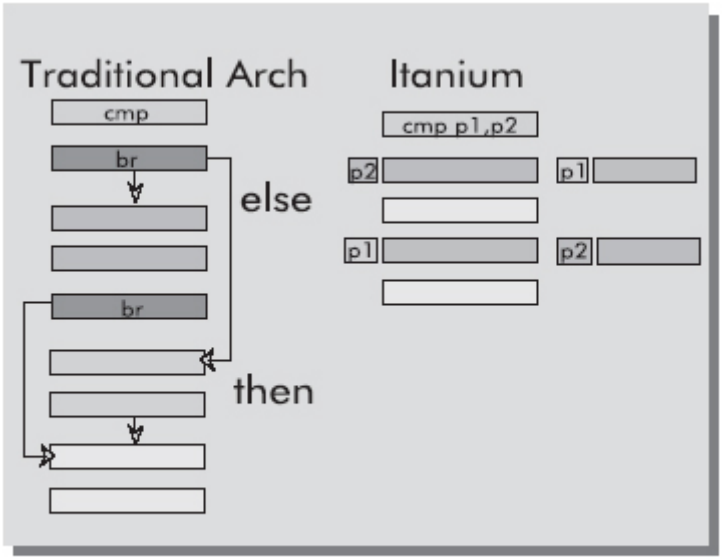

图 6 EPIC 架构的预测特性

第二，为了支持预测机制，Itanium 设置一条功能强大的比较指令来产生预测结果。该指令可以简化如下：

pT,pF CMP(crel r2,r3)

这条比较指令使用 crel 给出比较规则(例如大于)比较 r1 和 r2。比较的结果一般写入预测寄存器 pT，它的相反状态写入预测寄存器 pF。这给出两个预测来控制 if—then—else 语句的两边；

第三，Itanium 的预测机制允许在编译时对程序作优化，消除转移、提高效率。例如，假定程序中原有如下的语句：

if（a>b）then c=c+1 else d=d*e+f

通过编译的优化，可以消除条件语句中的转移指令，把它转化成预测执行：

pT，pF = CMP(a > b)

if（pT）C=C+1

if（pF）d=d*e+f

于是，成功地实现了通过预测机制不增加任何附加开销即消除了转移，把控制流转化为数据流。此外，编译程序还可以把 pT 和 pF 后的指令调度成让处理器并行地执行它们，然后视 pT 和 pF 的状态，采用一边的结果。

为了增加指令执行的并行度，EPIC 架构还提供创新的转移提示指令和转移寄存器来提高转移指令取指令的缓存命中率，从而提高处理器的性能。

转移提示指令能够提供一个要执行的转移指令的目标地址，允许硬件提前从目标地址预取指令到指令缓存。优质的编译程序能够提供所有上述的提示，使得硬件能够减少转移预测错误和更加有效地执行指令预取。

间接转移可能会造成性能问题，因为只有在执行转移指令后才能知道转移的目标地址。Itanium 架构定义 8 个 64 位的转移寄存器，允许编译程序在到达转移指令前规定转移的目标地址。这给处理器提供预取指令的提示，允许处理器隐藏间接转移的潜在指令缓存不命中所引起的部分或全部延迟。

EPIC 架构的高效执行转移指令的特性所产生的效果是通过优化编译程序和硬件设计，平滑和加速指令流通过流水线的过程、提高处理器的性能。

猜测特性

由于处理器的速度远远超过内存的速度，为了最大限度地降低内存延迟到影响，Itanium 提供两类猜测特性：线路猜测和数据猜测，通过提前发送读取内存操作、从关键路径中消除它的延迟，使处理器能够提高指令级并行度、充分提高功能部件的工作效率(防止由于内存延迟而闲置)，实现最高的性能。

控制猜测

控制猜测是指编译程序把指令移动到转移指令的前面执行。这允许提前执行程序内不命中缓存的装入指令等延迟长的操作、提高程序的执行效率。但是，当把指令移动到转移前时，可能会执行本来不应执行的指令，编译程序必须避免由此产生的副作用。为了解决这一问题，Itanium 引入了两条新的指令：一条是猜测装入(sload)，另一条是猜测检查(scheck)。当出现一个意外条件时，猜测装入将把检查寄存器的第 65 位置位，并对意外置之不理。猜测检查指令检查寄存器的第 65 位，如果置位，则发出意外信号。这允许把意外条件延迟到控制到 达装入指令原来所在块时才加以处理，如果控制不到达该块，则永远不加以处理，从而避免执行不该执行的指令。例如，对如下的条件语句：

if（a>b） load(ld_addr1,target1)

else load(ld_addr2,target2)

由于编译时不可能知道 a 与 b 那个大，如果提前执行两条 load 指令(即执行线路猜测)，虽然能够得到减少延迟的好处，但也可能会产生副作用。为此，编译程序按照上述的原理，对目标作如下的调度：

```
/* off critical path */
sload(ld_addr1,target1)
sload(ld_addr2,target2)

/* other operation including uses of target1
/target2 */
if (a>b)  scheck(target1,recovery_addr1)
scheck(target2,recovery_addr2)
```

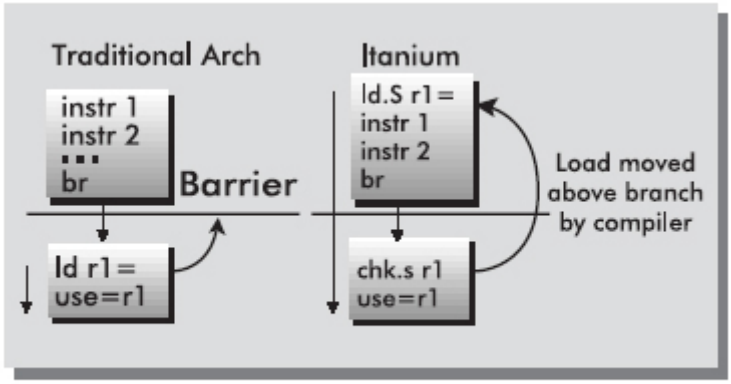

图 7 EPIC 架构的控制猜测特性

数据猜测

数据猜测是指编译程序把从内存中把数据读入寄存器的指令(load 指令)移动到把数据从寄存器存储到内存的指令(store 指令)前面执行，从而提前从内存中读出数据、减少内存延迟的影响。

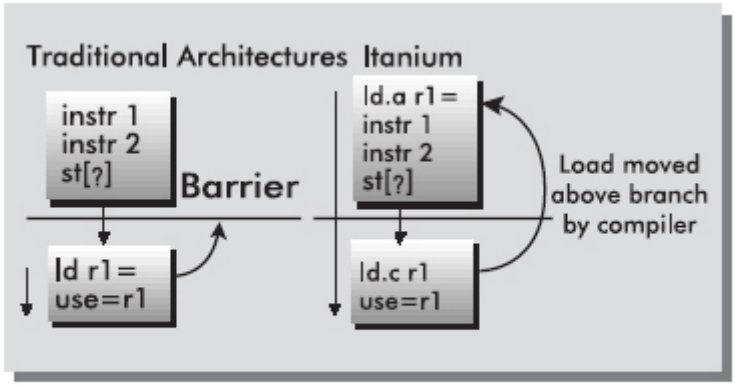

图 8 EPIC 架构的数据猜测特性

例如，编译程序把下例中的 load 指令调度到 store 指令前面：

```
store(st_addr,data)      load (ld_addr,target)
load (ld_addr,target)          store(st_addr,data)
use (target)  use (target)
```

如果在程序执行时 ld_addr 与 st_arrd 不一致，那么该程序将能够正确地享受到猜测的好处；但是，如果在执行时两个内存地址重迭，则必须采取必要的补救措施，否则就会产生错误的结果。为了防止数据猜测带来副作用，编译程序在 load 指令原来位置上放置一条检查指令，检查两个内存地址是否重迭。如果重迭，则转移到一段恢复程序、消除猜测所带来的副作用。于是上面的指令段变成：

```
/*  off critical path */
aload (ld_addr, target)

/*  other operations including uses of target */
store (st_addr,data)acheck (target,
recovery_addr)
use (target)
```

高效的过程调用

大多数 RISC 处理器的函数调用需要卸出和重新装入寄存器，开销很大。Itanium 增加了一个通用寄存器窗来支持高效的函数调用。这个 128 项的通用寄存器窗被分为一个 32 项的全程存储器和一个 96 项的堆栈存储器。Itanium 允许编译程序在被调用的函数过程入口，设置一条 ALLOC 指令创立一个最多包含 96 项的新寄存器堆栈；在返回时，恢复调用程序的寄存器堆栈帧。对编译程序来说似乎有长度无限的物理寄存器堆栈，从而降低了函数调用的开支、提高了效率；如果在调用和返回时，没有足够的寄存器可供使用(堆栈溢出)，那么处理器将被阻塞，等待卸出和装入寄存器，直到有足够的寄存器为止。

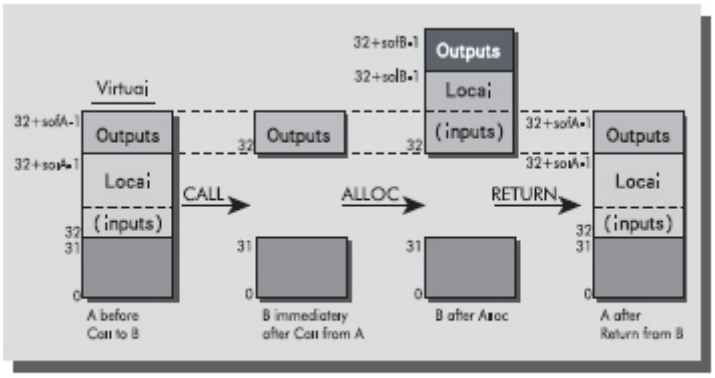

图 9 EPIC 架构的寄存器堆栈和引擎

软件流水线

循环步之间相互独立的循环可以象硬件流水线一样执行，即下一个循环步可以在上一个循环步结束前开始执行。这也可以称为软件流水线。传统的架构在同时执行多个循环步时，需要把循环拆开和软件重新命名寄存器。Itanium 引入了两个新特性：旋转寄存器存储器和蕴含预测来支持软件水线。Itanium 能够通过旋转寄存器机制为每个循环步提供自己的寄存器，并且不需要把循环拆开，使得软件流水线能够适用于更加广泛范围的循环，包括小的和大的循环，大大减少循环的附加开销；

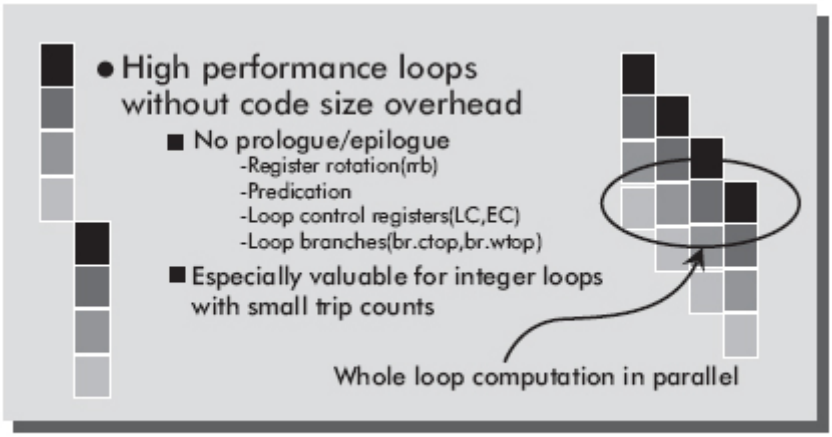

图 10 EPIC 架构的软件流水线

其他特性

Itanium EPIC 架构处理器还提供功能强大的浮点处理、多媒体和加密／解密指令、更好地满足高性能技术计算、多媒体、流媒体、高度保密的 Web 和电子商务等应用领域的需要。

2.3 Intel Itanium EPIC 架构释疑

新兴的 EPIC 与传统的 RISC 孰优孰劣？EPIC 是不是一种新思想？EPIC 在与 RISC 和其他架构竞争过程中能不能取得预期的成功？不少人还心存疑虑。在处理器技术历史上，对所出现任何一种新技术的评价都离不开当时的技术发展和市场需求的背景、都离不开实践的检验、也不能持过于绝对化的观点。从这一观点出发、结合 EPIC 架构 IPF 处理器发展现状，我们将能够清楚看到对 EPIC 的批评和疑虑虽然有其值得关注的一面、但是大多是非本质、支流和可以在实践中解决的问题。基本的主流是：EPIC 确实开辟了提高处理器性能的新路径，EPIC 具有雄厚的物质基础， EPIC 必将发展成为支持高端应用的主流平台。

EPIC 开辟了新的发展途径

如上所述：EPIC 的基本设计思想是：充分利用编译程序来发现并行(信息并行)；提供必要的资源实现并行，从而提高处理器性能。简化处理器逻辑。与传统的 RISC 和在 x86 基础上改进的 x86—64 架构对比，EPIC 是一种革命性的新架构。它在技术上是否可行、有没有竞争力和发展前景？基于 EPIC 架构的 IPF 处理器系列的实践证明了 EPIC 的可行性和先进性。

表 3 中 Itanium 处理器系列自身和对比数据说明：IPF 系列性能的确能够通过改进编译程序和增加资源以超过摩尔定律 3.5 倍的速度提高、并且领先于其他处理器。

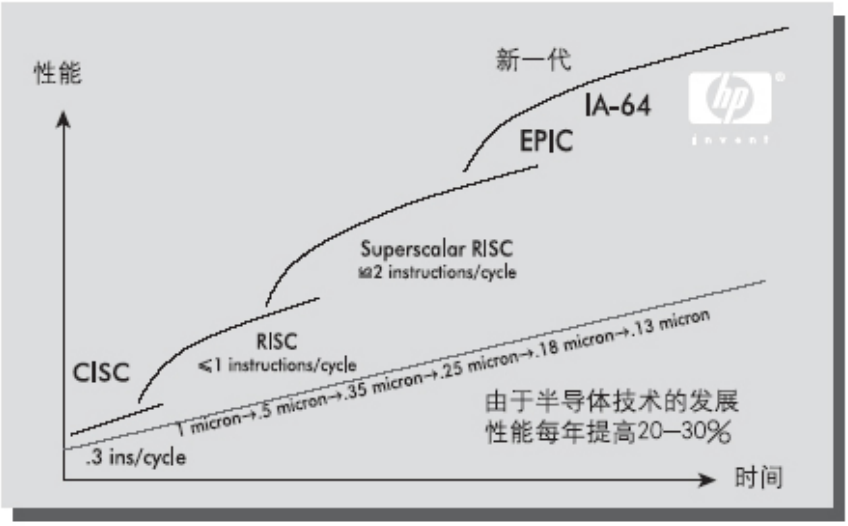

图 11
提高指令级并行度是提高处理器性能的关键

图 11　说明 EPIC 的显性并行机制和并行特性能够实现更高的指令级并行度(ILP)。主频 1.5 GHz 的 Itanium2 性能领先于主频更高的 Power4+,Xeon 和 AMD Opteron 等处理器也充分说明这一点。

实践证明，EPIC 架构能够通过硬软件两条腿走路方式向前发展，允许 IHV 和 ISV 厂商基于显性并行的特点、通过优化编译程序和应用软件独立地提高性能。在今后几年中，人们将看到为 Intel Itanium 架构开发的编译程序在提高性能方面将能够比基于不能充分发挥硬软件合力的 RISC 旧架构(如 POWER、UltraSPARC 、PA—RISC 等)和 x86—64 开发的编译程序发挥更大的作用。这将使得基于 Itanium 系统的性能随着后续处理器和操作系统版本推出、得到更大幅度的提高。例如，HP 预期在今后几年中单单是通过改善编译程序质量就能够促使应用性能产生 2 位数字的提高。将来，新一代的应用也将针对 EPIC 特点编写成能够充分利用 EPIC 架构的指令级并行、实现更高的性能。

EPIC 领先的特性

有人认为：EPIC 在注意发挥编译程序作用的同时忽视了提高处理器智能和充分利用运行时信息，使得处理器成为“愚笨的处理器”。超标量、无序执行的 RISC 处理器号称是“聪明的处理器”，实质上也有其不足之处：由于其隐性并行而不得不利用较复杂的逻辑和较高的智能来提高性能。IPF 是一种逻辑简单的处理器，它的聪明之处是善于利用编译程序产生的信息和丰富的芯片资源来提高性能，也还提供一系列能够提高 ILP 的特性创造了条件。这些特性都是 RISC 和 x86—64 架构所不可能提供的。EPIC 能够借助硬软件的合力、利用较简单的芯片逻辑实现更高的智能，这就是 EPIC 作为一种新兴架构优越之处。将来，EPIC 架构也能够采用 Alpha 等 RISC 处理器同时多线程(SMT)或芯片上多处理器技术,利用线程级并行(TLP)的智能进一步全面提高未来 IPF 系列产品的性能。

先进的工艺是实施 EPIC 的基础

EPIC 使用大量芯片资源来支持并行和存储信息，必然加大芯片面积、增加芯片的成本，

这在技术上和经济上是否可行、是否会使芯片价格太高而难以推广？是否能够进一步采用多核、多线程等先进技术？

EPIC 的确需要使用大量芯片资源，这在 VLSI 技术发展初期的确是技术上和经济上致命的缺点，也许也是 80 年代 Multiflow 和 Cydrome 设计的超长指令字计算机失败的原因。即使在 0.35μm 和 0.25μm 线宽时代，要想象 Itanium2 (Madison)那样在芯片内置入 4 亿多个晶体管也是一个很大的负担。但是，技术的进步使得 IPF 系列能够全面使用 0.13μm 线宽的 CMOS 技术，今后还将使用 90 和 65 纳米技术。在这样的技术背景下，增加芯片资源不仅在技术上可行，而且经济上负担也不大，反而能够为充分利用现代 VLSI 技术创造条件，使 EPIC 具有强大的生命力。事实上，芯片面积缩小与线宽变窄是平方关系。当采用 90 纳米工艺时，Intel 将推出双核的 Montecito；65 纳米的工艺，将能够支持 16 个核的 Tukwila、并吸收 Alpha EV8 同时多线程技术的精华，使得 IPF 能随着工艺的发展而发展，始终保持领先地位。

芯片上资源多当然有可能会增加成本。这对于批量较小的 RISC 处理器(例如 Alpha)来说，确实较为严重。这也是 Alpha 的设计师们非常注意通过改进芯片逻辑设计来提高性能，十分注意避免使用过多的芯片资源的原因。但是，当前 64 位计算市场需求很旺、应用也已成熟，基于 EPIC 架构的 IPF 系列是面向广阔市场的通用和开放性产品，不但面向科学计算、而且面向企业应用，不但面向服务器、而且面向工作站。因此，IPF 将主要通过批量和规模生产来降低成本，而这正是 Intel 的特长之一。

结束语

处理器架构是处理器系列发展的基础。当前，人们越来越清楚地看到 RISC 架构的封闭性、芯片资源的有限性和串行语义(隐性并行)机制，限制了处理器性能紧跟芯片制造工艺的发展步伐快速提高。应用需求和工艺发展不仅已经推动支持高端应用的处理器架构全面向 64 位过渡，而且将孕育新的架构为充分利用工艺水平和编译技术的发展提高处理器性能开辟新的天地。Intel Itanium 系列的 EPIC 架构是对传统 RISC 架构的重大革新，许多方面突破了 RISC 架构的局限、发展了 RISC 架构。

在处理器技术发展长河中，随着应用需求、工艺和硬软件技术水平的提高、研究和发展新架构的过程是永无穷尽的。各种架构在存在和发展期间都有其长处，也终将由于其短处而被淘汰或演变、继承和发展。当前， EPIC 架构不仅仅提供 64 位处理能力而且以其开放性、丰富的芯片资源和显性并行机制、一系列创新特性，为利用新工艺、制造和编译技术、按照摩尔定律预示的速度提高处理器性能提供了可靠的保证，成为处理器发展史上的一个新的制高点。

时也！势也！离开了具体的背景和立场，很难评价任何架构，批评也好、表扬也好都是相对的。但是，从技术发展的现状和未来趋势来看，EPIC 架构将更加有利于利用 VLSI 工艺和技术发展的新成就如高密度 CMOS 技术、先进的互联和散热技术等以及大规模批量生产的市场优势。EPIC 架构是一种面向未来的架构，基于它的 IPF 是一个将有 25 年以上设计寿命的产品系列，是一个发展中的新生事物。它将随着 VLSI、芯片设计、软件和系统集成技术的发展而不断发展。从实践来看，Itanium 的上市证明了 EPIC 设计思想的可行性；Itanium 2 的成功证明了 IPF 的发展潜力。Intel 和 HP 合作还将继续发展 EPIC 技术并吸收 RISC 架构的长处，推出 Itanium 处理器系列(IPF)的第 3 代、第 4 代、第 5 代以至更后的产品，把 IPF 系列推向新高峰。人们必将越来越清楚地看到：EPIC 是代表 64 位芯片发展未来的新技术，它将能够跟上摩尔定律预示的速度，满足实际应用的需求，发展成为支持高端应用主流平台。

表 3　IPF 系列提供不断提高及其领先地位						
处理器类型	Intel Itanium	Intel Itanium2	Intel Itanium2 6M （2003 年中）	HP PA—8700+	Sun UltraSparc III Cu	IBM Power4+
SPECint2000 (Base)	379	810	1000+	642	537 (1.05 GHz)	909 (1 CPU)
SPECfp2000 (Base)	715	1,427	1,600+	464	701 (1.05 GHz)	1,221 (1 CPU)
SPECint_Rate (Base)	19	37	64.4	23.2	21.4	35.8

表 3　EPIC 架构的提高 ILP 的特性以及与 RISC 架构的对比		
EPIC 架构特性	作用	RISC 架构限制
预测	把控制流转变成数据流，避免预测错误的惩罚，增加基本代码块的长度	基本代码块平均长度为 5 条指令，只有有限的 ILP 可供利用，转移使得析出 ILP 更加困难
控制猜测	通过编译程序把装入指令移动到转移指令的前面，打破了转移的障碍，解决了内存延迟问题，提高了 ILP	条件转移指令造成的过程相关性进一步限制了 ILP 的提高、扩大了内存延迟的影响
数据猜测	通过编译程序把装入指令移动到存储指令的前面，打破了存储指令的障碍，解决了内存延迟问题，提高了 ILP	存储指令造成的写－读相关性进一步限制了 ILP 的提高、扩大了内存延迟的影响
寄存器堆栈和寄存器堆栈引擎（RSE）	降低了过程调用的开支，更好地支持先进的模块化编程方式	过程调用开支较大
旋转寄存器和软件流水线	降低了循环的开销	循环的开销较大
提供一系列面向具体应用的指令	提高了支持 HPTC、Web 和多媒体应用的性能	——

表 4 工艺发展的实际情况和未来预测							
	实际			预测			
工艺名称	P586	P858	Px60	P1262	P1264	P1266	P1268
首次采用时间	1997	1999	2001	2004	2005	2007	2009

线宽	0．25μm	0.18 μm	130 纳米	90 纳米	65 纳米	45 纳米	32 纳米
门长度	0.20 μm	0.13 μm	65 纳米	45 纳米	32 纳米	22 纳米	16 纳米
状态	实际生产使用			开发中		研究中	

表 5 EPIC 架构对 RISC 架构的主要发展

架构	EPIC	RISC
本质特性	并行是 EPIC 架构的本质特性	顺序性是传统架构的本质特性
目标程序语义	**显性并行指令语义** 目标程序 ＝ 并行指令组序列规定指令组的执行次序组内指令间没有依赖性(完全的独立性)	**串行(隐性并行)指令语义** 目标程序＝串行指令序列决定了指令执行的次序指令间存在各种依赖性(相关性)
市场特性	开放性，标准化，大批量	专利和封闭性，非标准，小批量
芯片资状况	允许使用丰富的芯片资源	芯片资源受到批量和性价比的限制
并行特性	编译程序知道可利用的并行指令段，并且有必要的资源来显性表达它	编译程序虽然知道可利用的并行指令段，但没有相应的资源来显性表达它，许多有用信息都白白丢掉了
实现并行机制	硬件容易利用编译程序所标识的并行，因而大大减轻硬件并行执行的任务	需要复杂的硬件逻辑来重新(反复)从目标程序析出可并行执行的指令段
设计周期	大大简化了处理器设计、缩短了设计周期	设计任务越来越复杂、设计周期越来越长
结果	能够符合摩尔定律预示的速度，能够满足实际应用的需要	很难符合摩尔定律预示的速度，越来越不面对实际应用发展的挑战

HP创建*动成长企业*

中国惠普有限公司
北京市朝阳区建国路112号中国惠普大厦
电话：010-65643888
传真：010-65643999
邮编：100022
欲查询更多相关信息：请访问HP网站：
http://www.hp.com.cn
中国惠普客户互动中心：800-820-2255
售后服务支持热线：800-810-5959
010-68687980

invent

印制日期:2004年2月北京印刷