Intel Nehalem-EP 处理器首发深度评测

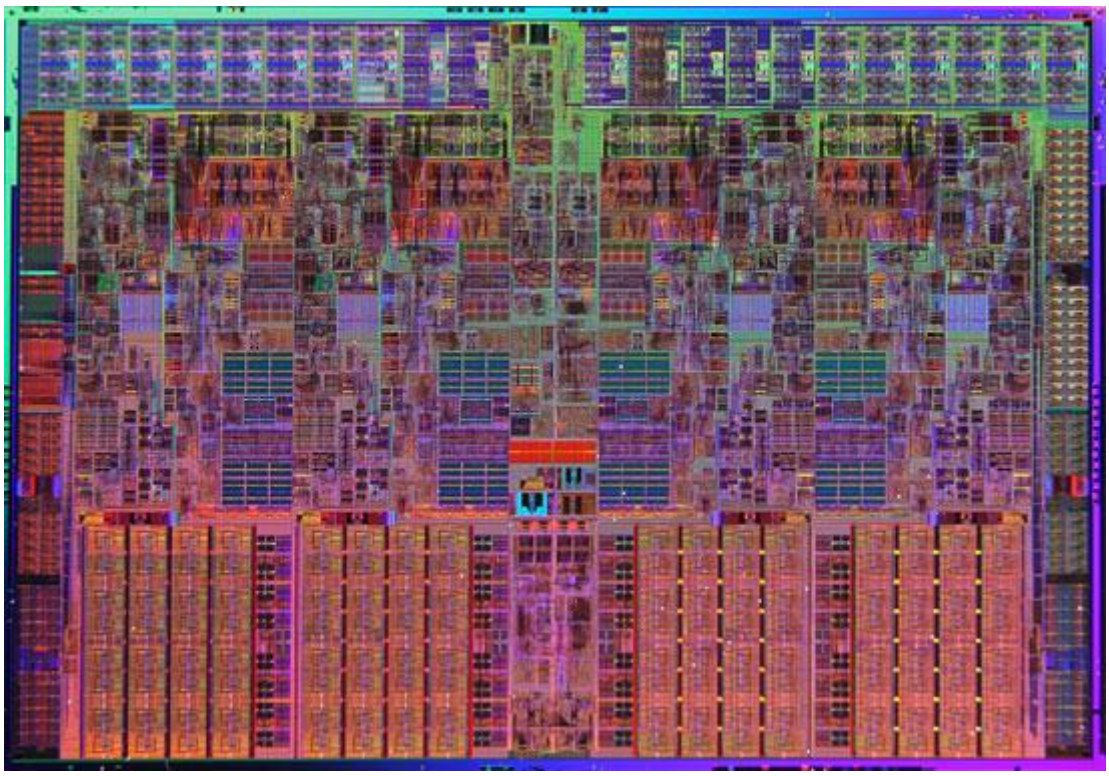
2009 年 03 月 31 日 00:00 IT168 网站原创 作者：IT168 评测中心 Lucifer 编辑：盘骏

盘骏：原文发表于 IT168 网站，有修订。

第 1 页：Intel Nehalem-EP 处理器发布

【IT168 评测中心】2009 年 3 月 31 号，春季的最后一天，在 Nehalem 处理器架构的桌面版本 Core i7（代号 Bloomfield）发布 134 日之后，其双路服务器版本 Nehalem-EP（代号 Gainestown）终于发布了。Nehalem-EP 处理器是 Nehalem 处理器架构的集中体现，在桌面版本乃至移动版本上看不到的多 QPI 总线等特性开始在 Nehalem-EP 上现身——我们早已经知道，不同于之前的 Core 处理器，Nehalem 架构是为企业应用而设计，因此，Nehalem 架构的精髓，只有在服务器版本 Nehalem-EP/Nehalem-EX 上才能完完全全地看到。

目前最高端的 Nehalem-EP 型号：Xeon X5570，主频 2.93GHz，QPI 频率 3.2GHz



4 核心 Nehalem-EP 处理器晶元图

Nehalem 架构仍然基于成熟的 Core 微架构并在其基础上进行改进，因此，桌面版本 Nehalem 使用了 Core i7 的系列名称（上一代则是 Core 2），服务器版本也不例外，仍然使用了 Xeon 的名称，不过处理器系列更新为 5500（上一代为 5400，上上一代为 5300）。

配合 Nehalem-EP 使用的 Intel Tylersburg-EP 芯片

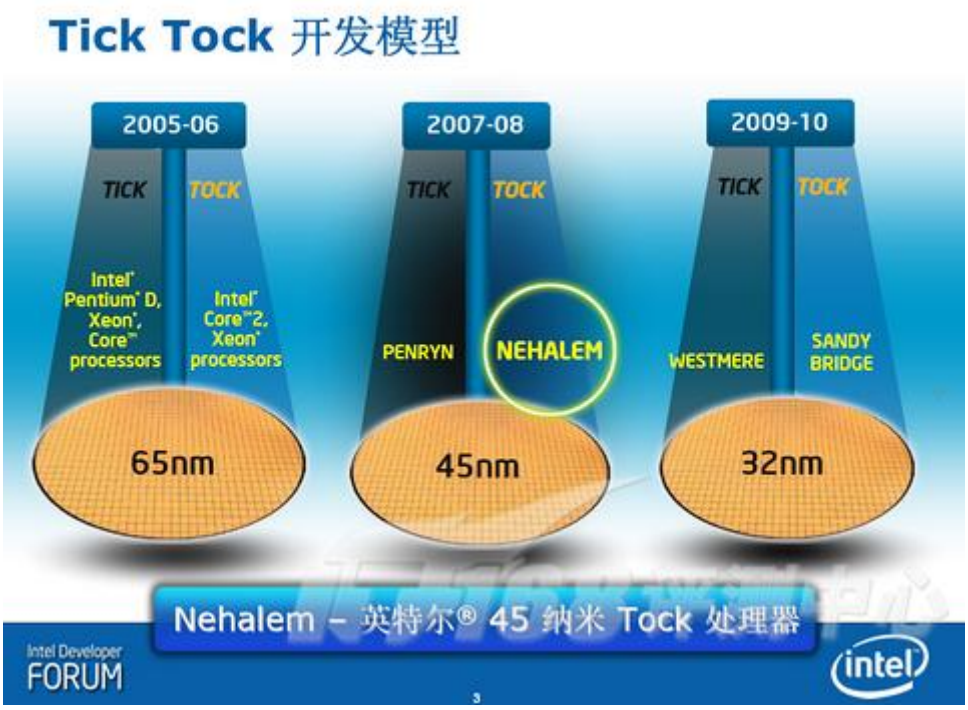
几天前，我们曝光了 Nehalem-EP 处理器 Xeon E5540 的实物（同时给出了基于 X58 主板的上机图），并发布了我们根据当前资料做出的架构分析，不过由于 NDA 保密协议的缘故，

我们不能给出其性能数据，现在随着 Nehalem-EP 的正式发布，禁令也相应消失，下面我们终于可以尽情地享受 Nehalem-EP 的极致性能了。由于篇幅较多，因此建议读者们使用下面的导航功能，直接打开到感兴趣的页面。

Intel Nehalem-EP 官方评测样机，配置了双路 Xeon X5570 处理器和 24GB DDR3 内存

我们 IT168 评测中心收到了 Intel 发出的第一批测试样机。由于样机是从 Intel 位于 Oregen 俄勒冈州的 Hillsboro 研发中心（也就是负责 Nehalem 架构开发工作团队的所在）发来，因此最终留给我们的时间不是很多，下面我们就来领略这台顶级 Nehalem-EP 服务器的威力。

第 2 页：Nehalem：Tick-Tock 战略的产物

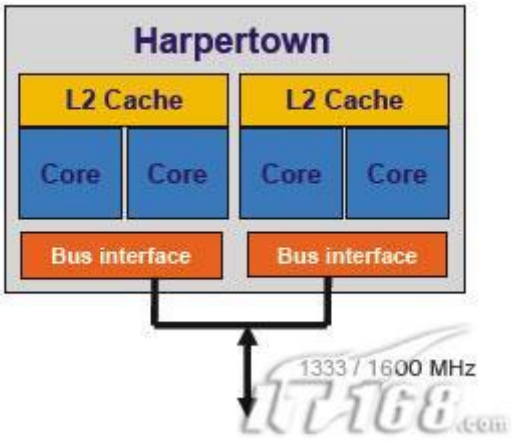


对于 Intel 的 Tick-Tock 战略已经是老生常谈了；从另一方面讲，这表明了 Tick-Tock 战略的成功之处，一个简单、明晰、有序和易于理解的发展计划，对合作厂商、用户和投资者都是极为有利的。Tick-Tock 战略简而言之就是 Intel 处理器在奇数年进行制程转换（Tick），例如 2005 年的 65nm 和 2007 年的 45nm，而在偶数年进行处理器的架构更新（Tock），Nehalem 架构发布的 2008 年轮换到了 Tock，也就是处理器的架构更新。

Nehalem 作为 Intel 用以取代 Penryn 微架构的新一代处理器架构，和 Penryn 相比，Nehalem 的微架构并非是全新的，不过，架构上则是一个很大的飞跃：Nehalem 采用了直联架构。除此之外，Nehalem 还具有一个鲜明的设计理念，就是采用了可扩展的模块化设计，它将处理器划分为两个部分：Core 核心和 Uncore 非核心（或者叫“核外”），所有产品线的 Nehalem 处理器，其 Core 核心部分都是一样的，只是 Uncore 部分可能不同，以满足 Intel 对其提出的动态可扩展的要求。Nehalem 满足了这个要求，它的内核具有可扩展的高可伸缩架构。

由于共处在一个 Tick-Tock 上，因此 Nehalem 和 Penryn 都同样属于 45nm 工艺，从 65nm 工艺转变到 45nm 工艺带来的巨大能耗降低已经无法再次重现，因此 Nehalem 就不再注重于能耗的降低，而是注重于性能的提升，这样的设计理念，带来了处理器架构的巨大变化，这些变化均面向性能的提高，也即是说，我们可以期望 Nehalem 具有着强大的性能。

第 3 页：Nehalem 设计思想的转变：基于企业应用

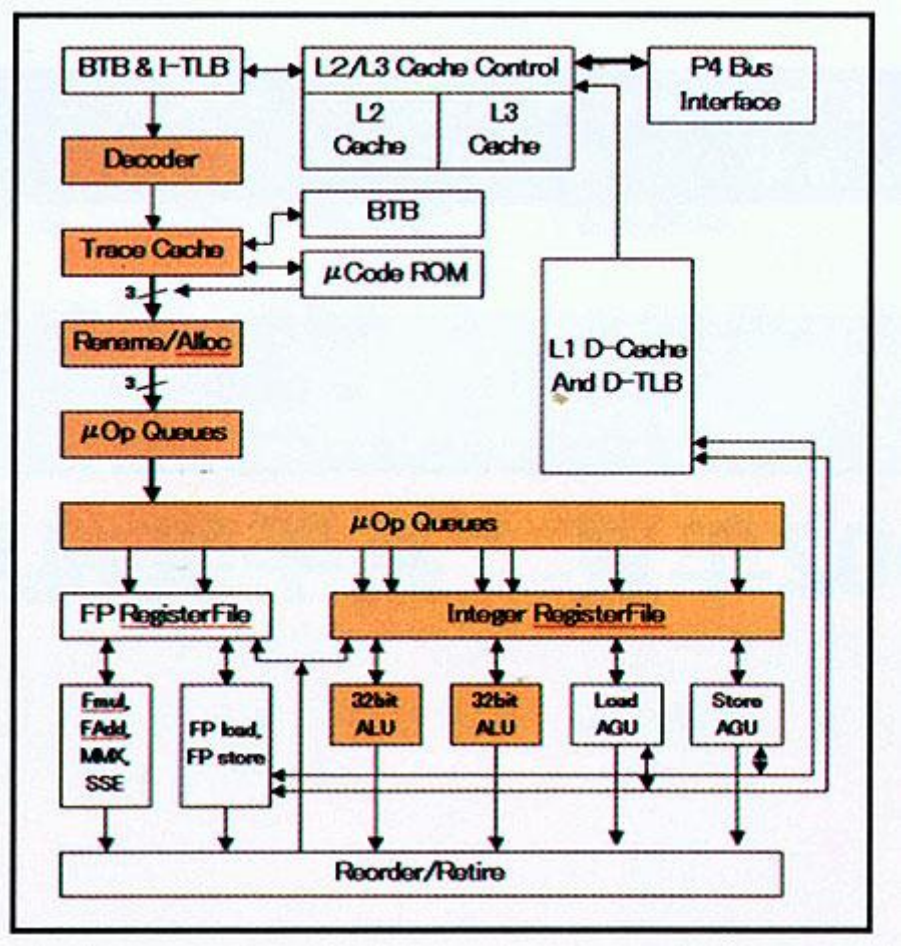


Harpertown 是基于 45nm Penryn 的四核 Xeon DP 处理器：两个“双核”粘结而成

可以说，Core 现在在所有产品线上都获得了成功——因为它强劲的性能，以及很好的功耗表现，然而，Core 确实具有一个移动计算的起源，它的原始架构都是围绕着这个中心来设计。例如，古老的双核设计：笔记本不需要太多的处理核心（即使是现在，也只有少量工作站级别的笔记本配置了四核），让目前的 Core 架构已经不太适合于服务器市场，虽然 6 核心的 7400 系列 Xeon 已经推出，不过它和当前的四核产品一样，都是多个“双核”粘结在一起的产物。不会有基于 Core 架构（不是 Core 微架构）的 8 核心产品了，那样做的代价太大，基于 FSB 方面的原因，性能将会很受限制（六核心的 Dunnington 是 Core 架构/FSB 总线的绝唱）。

六核心 45nm Penryn 至强 Dunnington 是 Core 架构/FSB 总线的绝唱，并融合了 Nehalem 的“原生单芯”设计

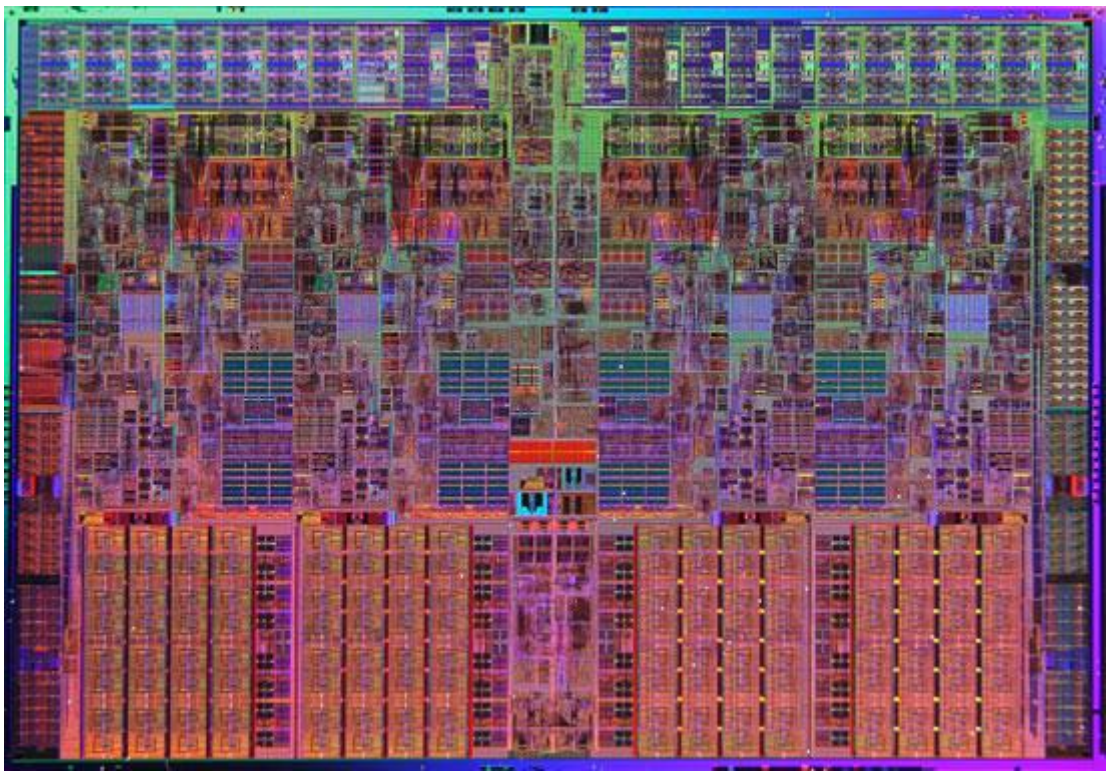
Core 架构具有一个移动计算的起源，它源自 Banias Pentium M 处理器，Pentium M 处理器是以色列（Israel）的海法（Haifa）研究中心专门针对笔记本电脑的产品，特点是高效、低耗。时值 2004 年，开发 NetBurst 架构的美国德克萨斯州（Texas）的奥斯丁（Austin）设计团队尚在设计 Tejas（Prescott 的下一代）。很快 NetBurst 失败，Core 架构被扶正，之后迅速地成为 Intel 的主要架构，产品开始扩展到桌面乃至服务器产品线（很可怕地，Austin 设计团队被分派去设计一个极低功耗的 CPU，就是后来的 Atom 凌动处理器）。



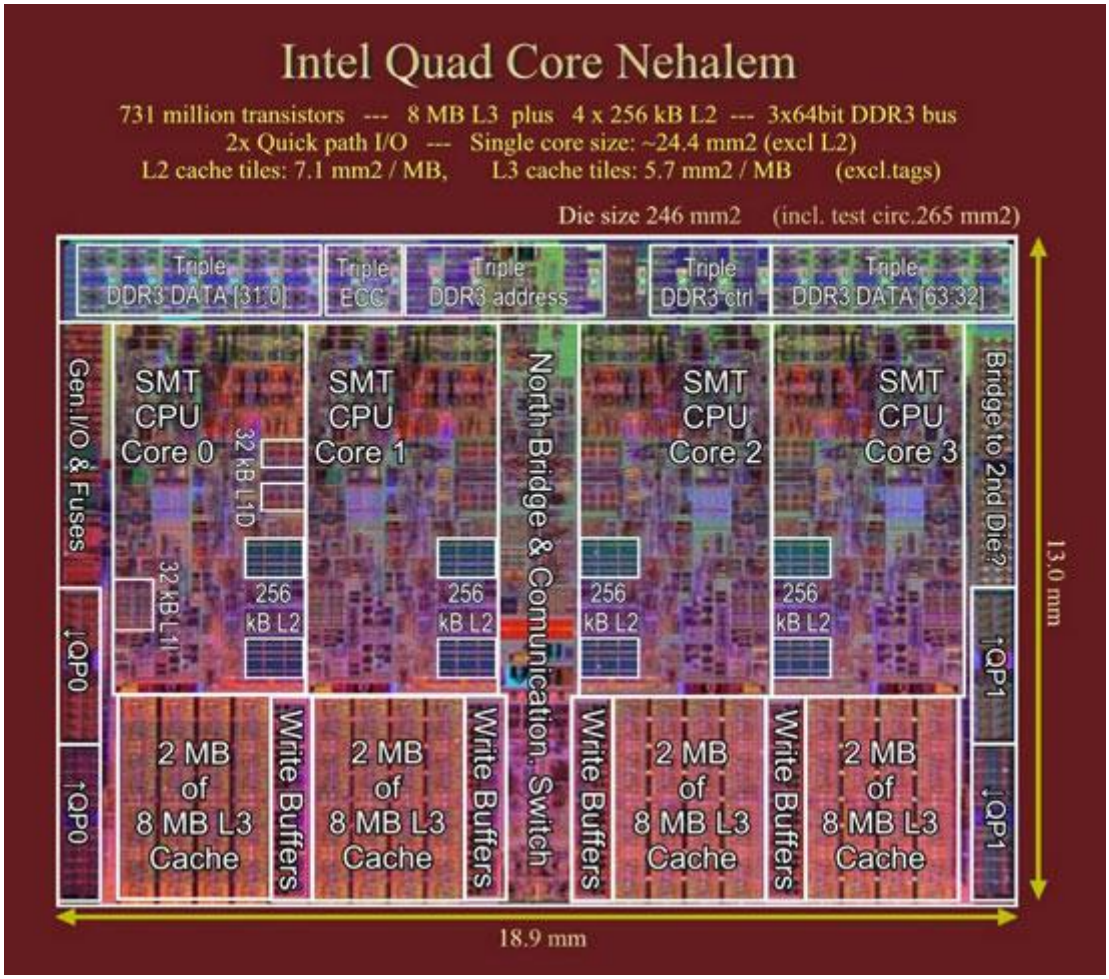
初代 Core 架构：Banias Pentium M

Core 架构的成功我们都已经看到了，然而随着时间的流逝，如古老的双核设计（4 核是两个双核粘在一起，6 核则是三个双核粘结）、过时的 FSB 总线以及没有充分为 64 位计算准备等等，让其无法获得很好的伸缩性，难以未来需要高性能多处理器需求的企业级市场。此外，在 NetBurst 架构上耗资了数亿美元的 HTT 超线程技术也没能得到体现，Intel 需要制作一款新的处理器产品来满足未来的需求。

Intel 对 Core 架构作出了改动，首先它将原来的架构扩展为原生 4 核（甚至 6 核、8 核）设计，并为多核的需要准备了新的总线 QPI 来满足巨大的带宽需求，结果就是 Nehalem 内核。Nehalem 内核还采用了集成内存控制器的设计，也是为了满足多核心巨大的带宽需求（从目前来看,Nehalem-EP 不会有 6 核、8 核的型号,这些产品会出现在 Nehalem-EX 上面）。



4 核心 Nehalem-EP 处理器晶元图



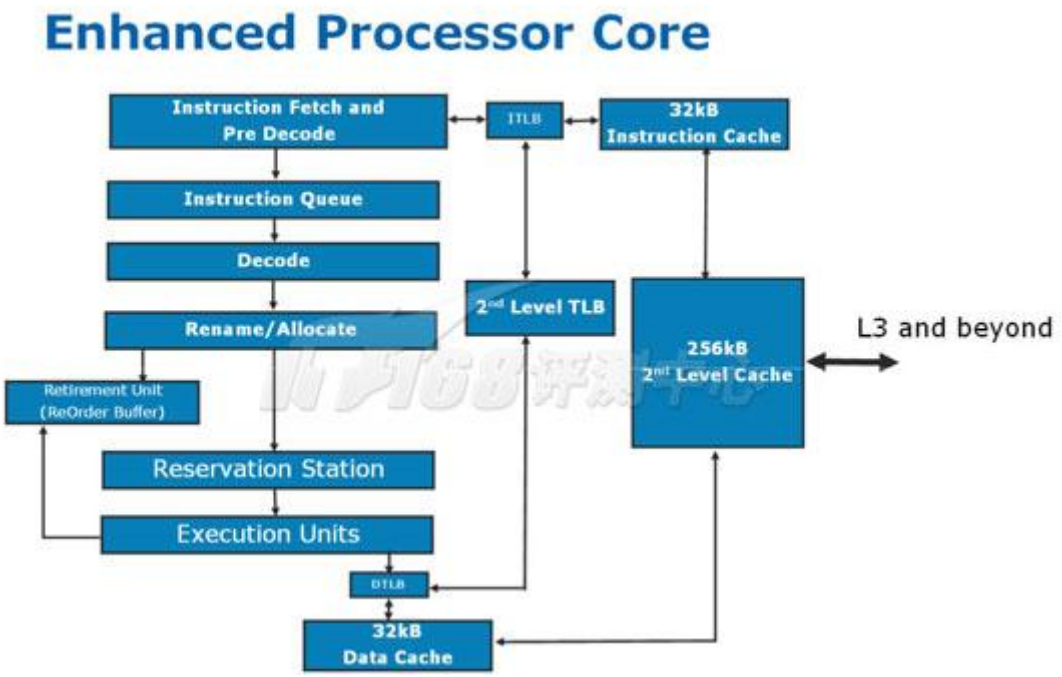
4 核心 Nehalem-EP 处理器的一些简要参数

了解Nehalem的设计思想之后，我们先来看看Nehalem微架构设计，和前面所说的一样，基于模块化设计，所有的 Nehalem 处理器的微架构都是一致的，因此接下来的内容适合于包括移动、桌面在内的 Nehalem 处理器。

The Core: Partition

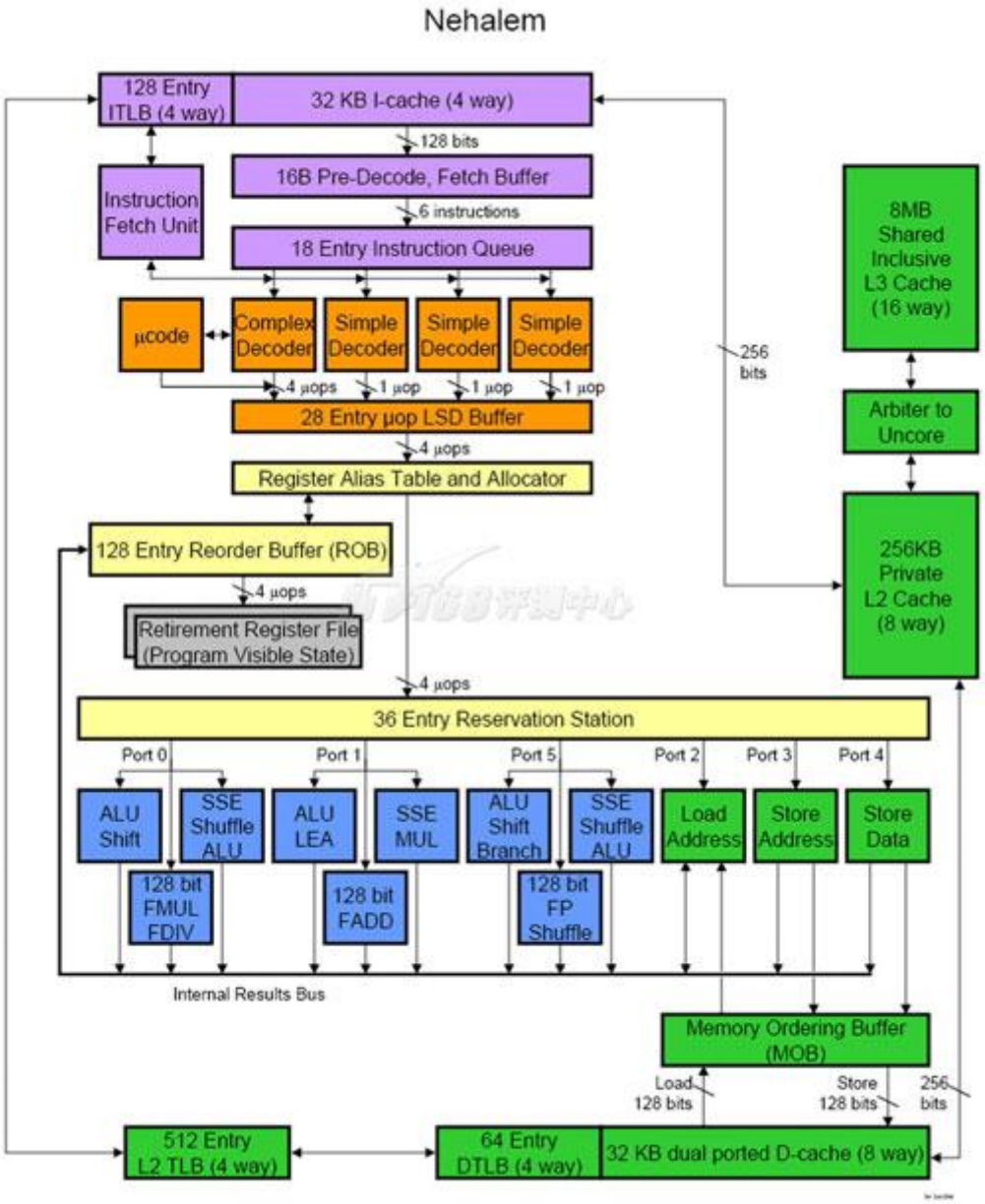
核心：功能区间划分

首先我们需要清楚地知道，Nehalem 是一款 OOOE（Out of Order Execute）乱序执行的 Superscaler 超标量 x86 处理器。x86 处理器上的超标量设计是从 Pentium 开始，乱序执行则是从 Pentium Pro 开始（Pentium 仍然是 IOE-In Order Execute 的）。现在的乱序执行处理器采用的流水线可能深度不一，但是它们都离不开取指（Instruction Fetch)、解码（Decode）、执行（Execute）、串行顺序回退（Retire）这四个阶段。



官方公布的简单的架构图

既然是乱序执行，那么四个阶段中，取指令、指令解码和回退阶段实际上仍然是属于 In-Order 顺序的。加上内存存取方面的内容，Nehalem 处理器可以按下面的颜色划分：



Nehalem Microarchitecture，经笔者整理

紫色部分属于取指令部分，橙色则属于解码部分。黄色部分是乱序执行的准备部分（灰色 Retirement Register File 属于乱序架构的 Retire 部分），蓝色方框是计算单元，绿色方框是内存子系统（包括紫色部分的指令缓存在内），计算单元和内存子系统的一部分（存取单元）一起成为乱序执行单元。绿色方框包含了 Core 和 Uncore 两部分，Uncore 的内容，不同系列的 Nehalem 的处理器是不同的，就上图标明和本文谈论到的，都适用于我们的主角：Nehalem-EP 处理器。

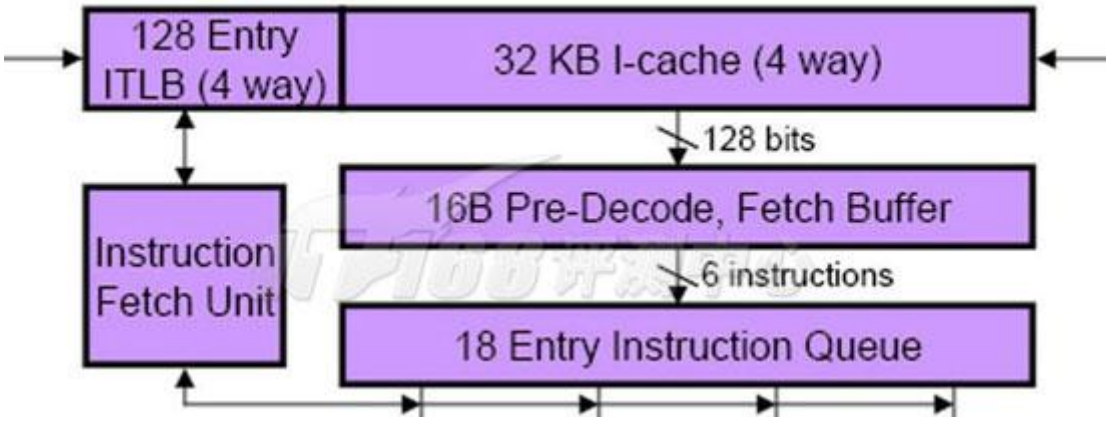
下面就大致从指令拾取开始介绍 Nehalem 的微架构，这些内容就经过了笔者的多方面查证以确保具有较高的准确性。然而由于内容太多，错漏难以避免，欢迎读者们一一指出。

The Core Front-End: Instruction Fetch

处理器核心前端：指令拾取

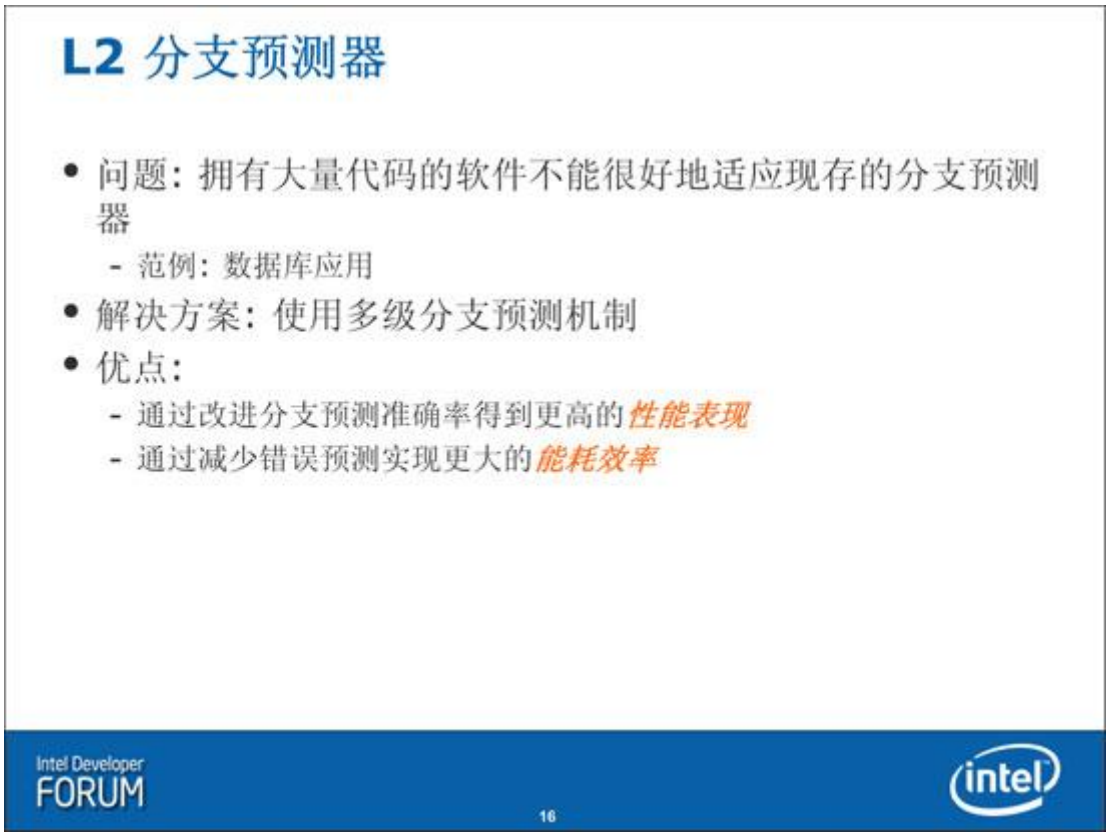
处理器在执行指令之前，必须先装载指令。指令会先保存在 L1 缓存的 I-cache（Instruction-cache）指令缓存当中，Nehalem 的指令拾取单元使用 128bit 带宽的通道从 I-cache 中读取指令。这个 I-cache 的大小为 32KB，采用了 4 路集合关联，在后面的存取单元介绍中我们可以得知这种比 Core 更少的集合关联数量是为了降低延迟。

为了适应超线程技术，RIP（Relative Instruction Point，相对指令指针）的数量也从一个增加到了两个，每个线程单独使用一个。



The Core Front-End: Instruction Fetch

指令拾取单元包含了分支预测器（Branch Predictor），分支预测是在 Pentium Pro 处理器开始加入的功能，预测如 if then 这样的语句的将来走向，提前读取相关的指令并执行的技术，可以明显地提升性能。指令拾取单元也包含了 Hardware Prefetcher，根据历史操作预先加载以后会用到的指令来提高性能，这会在后面得到详细的介绍。



两级分支预测机制

当分支预测器决定了走向一个分支之后，它使用 BTB（Branch Target Buffer，分支目标缓冲区）来保存预测指令的地址。Nehalem 从以前的一级 BTB 升级到了两个级别，这是为了适应很大体积的程序（数据库以及 ERP 等应用，跳转分支将会跨过很大的区域并具有很多的分支）。Intel 并没有提及 BTB 详细的结构。与 BTB 相对的 RSB（Return Stack Buffer，返回堆栈缓冲区）也得到了提升，RSB 用来保存一个函数或功能调用结束之后的返回地址，通过重命名的 RSB 来避免多次推测路径导致的入口 / 出口破坏。RSB 每个线程都有一个，一个核心就拥有两个，以适应超线程技术的存在。

RSB：重命名的返回堆栈缓冲器

指令拾取单元使用预测指令的地址来拾取指令，它通过访问 L1 ITLB 里的索引来继续访问 L1I Cache，128 条目的小页面 L1 ITLB 按照两个线程静态分区，每个线程可以获得 64 个条目，这个数目比 Core 2 的少。当关闭超线程时，单独的线程将可以获得全部的 TLB 资源。除了小页面 TLB 之外，Nehalem 还每个线程拥有 7 个条目的全关联(Full Associativity)大页面 ITLB，这些 TLB 用于访问 2M/4M 的大容量页面，每个线程独立，因此关闭超线程不会让你得到 14 个大页面 ITLB 条目。

指令拾取单元通过 128bit 的总线将指令从 L1I Cache 拾取到一个 16Bytes（刚好就是 128bit）的预解码拾取缓冲区。128 位的带宽让人有些迷惑不解，Opteron 一早就已经使用了 256bit 的指令拾取带宽。最重要的是，L1D 和 L1I 都是通过 256bit 的带宽连接到 L2 Cache 的。

由于一般的 CISC x86 指令都小于 4Bytes(32 位 x86 指令；x86 指令的特点就是不等长)，因此一次可以拾取 4 条以上的指令，而预解码拾取缓冲区的输出带宽是 6 指令每时钟周期，因此可以看出指令拾取带宽确实有些不协调，特别是考虑到 64 位应用下指令会长一些的情况下（解码器的输入输出能力是 4 指令每时钟周期，因此 32 位下问题不大）。

75%的 x86 指令小于 4Bytes。

指令拾取结束后会送到 18 个条目的指令队列，在 Core 架构，送到的是 LSD 循环流缓冲区，在后面可以看到，Nehalem 通过将 LSD 移动后更靠后的位置来提高性能。

The Core Front-End: Decode

处理器核心前端：解码

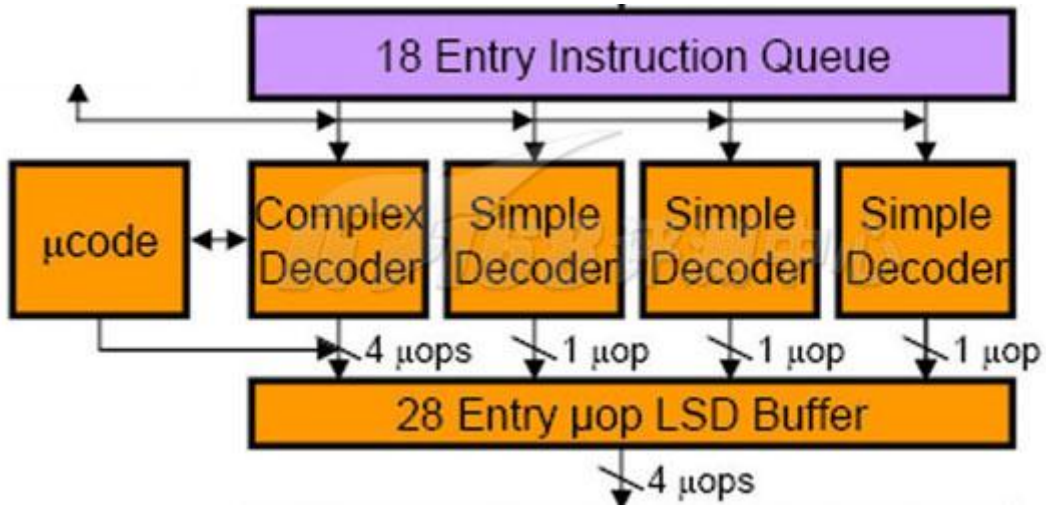
在将指令充填到可容纳 18 条目的指令队列之后，就可以进行解码工作了。解码是类 RISC（精简指令集或简单指令集）处理器导致的一项设计，从 Pentium Pro 开始在 IA 架构出现。处理器接受的是 x86 指令（CISC 指令，复杂指令集），而在执行引擎内部执行的却不是 x86 指令，而是一条一条的类 RISC 指令，Intel 称之为 Micro Operation——micro-op，或者写为 μ-op，一般用比较方便的写法来替代掉希腊字母：u-op 或者 uop。相对地，一条一条的 x86 指令就称之为 Macro Operation，或 macro-op。

RISC 架构的特点就是指令长度相等，执行时间恒定（通常为一个时钟周期），因此处理器设计起来就很简单，可以通过深长的流水线达到很高的频率（例如 31 级流水线的 Pentium 4……当然 Pentium 4 要超过 5GHz 的屏障需要付出巨大的功耗代价），IBM 的 Power6 就可以轻松地达到 4.7GHz 的起步频率。关于 Power6 的架构的非常简单的介绍可以看《机密揭露：Intel 超线程技术有多少种？》，我们继续说 Nehalem：和 RISC 相反，CISC 指令的长度不固定，执行时间也不固定，因此 Intel 的 RISC/CISC 混合处理器架构就要通过解码器将 x86 指令翻译为 uop，从而获得 RISC 架构的长处，提升内部执行效率。

和 Core 一样，Nehalem 的解码器也是 4 个：3 个简单解码器加 1 个复杂解码器。简单解码器可以将一条 x86 指令（包括大部分 SSE 指令在内）翻译为一条 uop，而复杂解码器则将一些特别的（单条）x86 指令翻译为 1~4 条 uops——在极少数的情况下，某些指令需要通过额外的可编程 microcode 解码器解码为更多的 uops（有些时候甚至可以达到几百个，因为一些 IA 指令很复杂，并且可以带有很多的前缀/修改量，当然这种情况很少见），下图 Complex Decoder 左方的 ucode 方块就是这个解码器，这个解码器可以通过一些途径进行升级或者扩展，实际上就是通过主板 Firmware 里面的 Microcode ROM 部分。

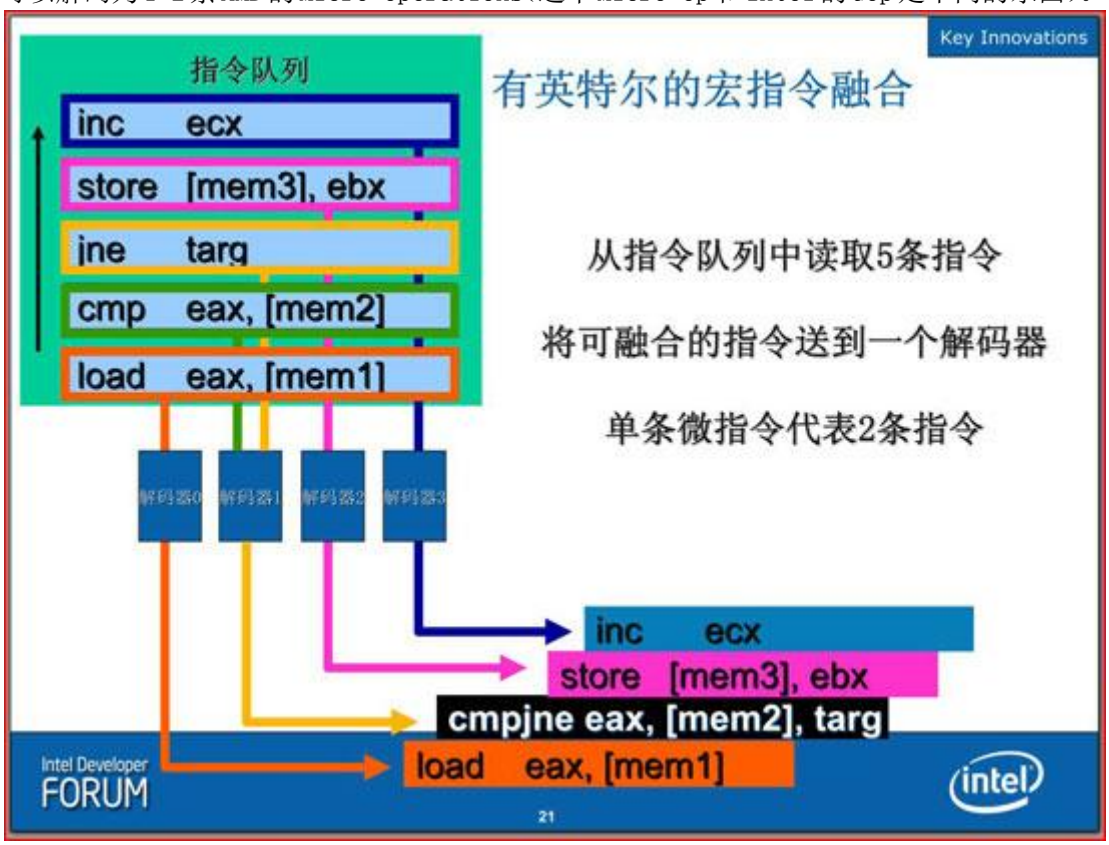
2006 年进行的一个研究当中表示，最常用的 20 条 x86 指令当中：
mov 占 35%（寄存器之间、寄存器与内存之间移动数据），push 占 10%（压入堆栈，也经常用来传递参数），call 占 6%，cmp 占 5%，add、pop、lea 占 4%（实际计算指令非常少）
75%的 x86 指令短于 4 bytes，也就是小于 32 bits。不过这些短指令只占代码大小的 53%——有一些指令非常长

之所以具有两种解码器，是因为仍然是关于 RISC/CISC 的一个事实：大部分情况下（90%）的时间内处理器都在运行少数的指令，其余的时间则运行各式各样的复杂指令（不幸的是，复杂就意味着较长的运行时间），RISC 就是将这些复杂的指令剔除掉，只留下最经常运行的指令（所谓的精简指令集），然而被剔除掉的那些指令虽然实现起来比较麻烦，却在某些领域确实有其价值，RISC 的做法就是将这些麻烦都交给软件，CISC 的做法则是像现在这样：由硬件设计完成。因此 RISC 指令集对编译器要求很高，而 CISC 则很简单。对编程人员的要求也类似。



The Core Front-End: Decode

作为对比，AMD 的处理器从 K8 以来就具有强大的三组复杂解码器，并且大部分指令都可以解码为 1~2 条 AMD 的 Micro-operations(这个 Micro-op 和 Intel 的 uop 是不同的东西)。



Macro Fusion 功能

在进行解码的时候，会碰到 Intel 在 Core 2 开始加入的技术：Macro Fusion。关于 Macro Fusion 可以看这里《64 位对决 32 位 SPEC CPU 运算效能测试》，这项技术可以将一些比较并跳转的分支 x86 指令集合（CMP+TEST/JCC）最终解码为单条 uop（CMP+JCC），从而提升了解码器的带宽、降低执行指令数量，让系统运行效率更高。和 Core 2 相比，Nehalem 现在支持更多的比较 / 跳转分支情况，如 JL/JNGE、JGE/JNL、JLE/JNG、JG/JNLE 等，此外，Nehalem 也开始更多地为服务器平台考虑，它的 Macro Fusion 开始支持 64 位模式，而不是

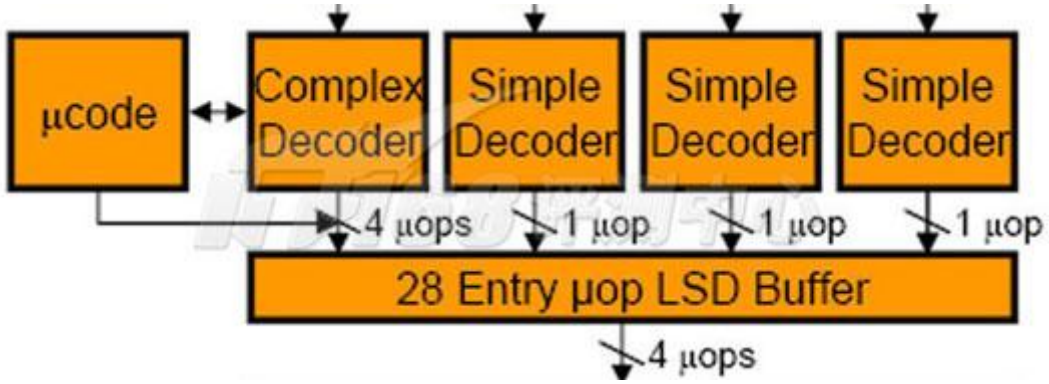
Core 2 的只支持 32 位模式。随着内存容量的日益增长，即使是在桌面平台及移动平台，64 位用户也在不断增加，因此这个改进对用户很有吸引力。

除了将多条 Macro Ops（就是 x86 指令）聚合之外，Nehalem 也支持将多条 uops 聚合，也就是 Micro Fusion 功能，用于降低 uop 的数量。这是一个从 Pentium M 开始存在的老功能了，它在顺序上是在比 Macro Fusion 后面的后面，也就是在 LSD 循环流监测器后方。据说，Micro Fusion 可以提升 5%的整数性能和 9%的浮点性能，而 Macro Fusion 则可以降低 10%的 uop 数量。

The Core Front-End: Loop Stream Detector

处理器核心前端：循环流检测

在解码为 uop 之后 Nehalem 会将它们都存放在一个叫做 uop LSD Buffer 的缓存区。在 Core 2 上，这个 LSD Buffer 是出现在解码器前方的，Nehalem 将其移动到解码器后方，并相对加大了缓冲区的条目。Core 2 的 LSD 缓存区可以保存 18 个 x86 指令而 Nehalem 可以保存 28 个 uop，从上一页可以知道，大部分 x86 指令都可以解码为一个 uop，少部分可以解码为 1~4 个 uop，因此 Nehalem 的 LSD 缓冲区基本上可以相当于保存 21~23 条 x86 指令，比 Core 2 要大上一些。



The Core Front-End: Loop Stream Detector

LSD 循环流监测器也算包含在解码部分，它的作用是：假如程序使用的循环段（如 for..do/do..while 等）少于 28 个 uops，那么 Nehalem 就可以将这个循环保存起来，不再需要重新通过取指单元、分支预测操作，以及解码器，Core 2 的 LSD 放在解码器前方，因此无法省下解码的工作。

Nehalem LSD 的工作比较像 NetBurst 架构的 Trace Cache，其也是保存 uops，作用也是部分地去掉一些严重的循环，不过由于 Trace Cache 还同时担当着类似于 Core/Nehalem 架构的 Reorder Buffer 乱序缓冲区的作用，容量比较大（可以保存 12k uops，准确的大小是 20KB），因此在 cache miss 的时候后果严重（特别是在 SMT 同步多线程之后，miss 率加倍的情况下），LSD 的小数目设计显然会好得多。不过笔者认为 28 个 uop 条目有些少，特别是考虑到 SMT 技术带来的两条线程都同时使用这个 LSD 的时候。

在 LSD 之后，Nehalem 将会进行 Micro-ops Fusion，这也是前端（The Front-End）的最后一个功能，在这些工作都做完之后，uops 就可以准备进入执行引擎了。

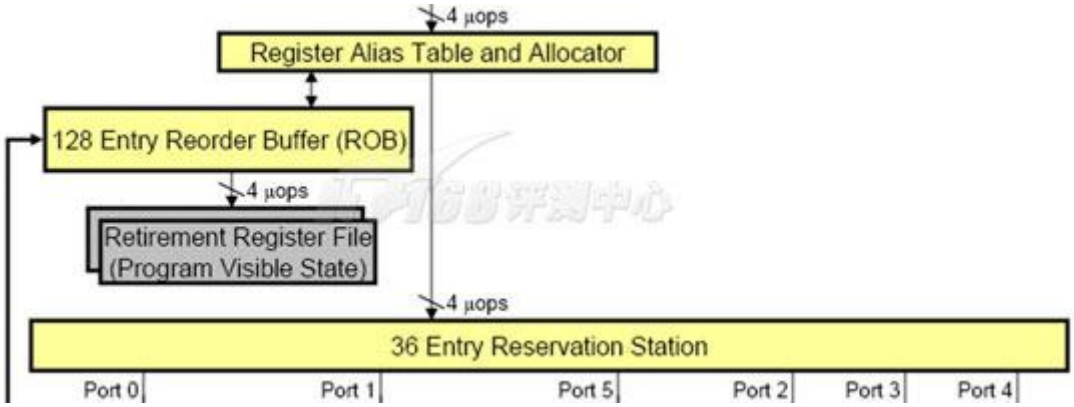
The Core Execution Engine: Out-of-Order Execution

处理器核心执行引擎：乱序执行

OOOE——Out-of-Order Execution 乱序执行也是在 Pentium Pro 开始引入的，它有些类似于多线程的概念，这些在《机密揭露：Intel 超线程技术有多少种？》里面可以看到相关的介绍。乱序执行是为了直接提升 ILP（Instruction Level Parallelism）指令级并行化的设计，在多个执行单元的超标量设计当中，一系列的执行单元可以同时运行一些没有数据关联性的若干指令，只有需要等待其他指令运算结果的数据会按照顺序执行，从而总体提升了运行效率。乱序执行引擎是一个很重要的部分，需要进行复杂的调度管理。

首先，在乱序执行架构中，不同的指令可能都会需要用到相同的通用寄存器（GPR，General Purpose Registers），特别是在指令需要改写该通用寄存器的情况下——为了让这些指令们能并行工作，处理器需要准备解决方法。一般的 RISC 架构准备了大量的 GPR，而 x86 架构天生就缺乏 GPR（x86 具有 8 个 GPR，x86-64 具有 16 个，一般 RISC 具有 32 个，IA64 则具有 128 个），为此 Intel 开始引入重命名寄存器（Rename Register），不同的指令可以通过具有名字相同但实际不同的寄存器来解决，这有点像魔兽世界当中的副本设计，当然其出现要比 WoW 早多了。

此外，为了 SMT 同步多线程，这些寄存器还要准备双份，每个线程具有独立的一份。



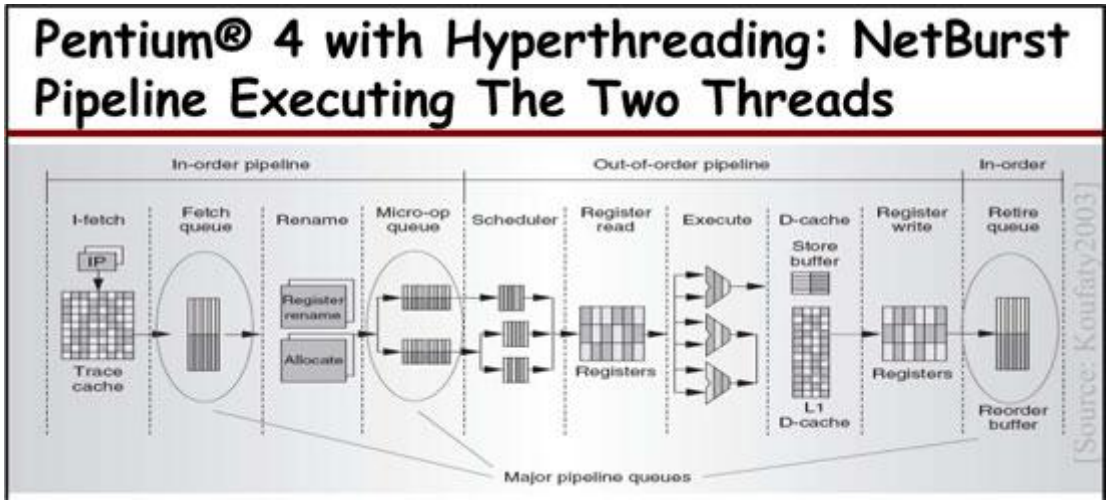
The Core Execution Engine: Out-of-Order Execution

乱序执行从 Allocator 定位器开始，Allocator 管理着 RAT（Register Alias Table，寄存器别名表）、ROB（Re-Order Buffer，重排序缓冲区）和 RRF（Retirement Register File，退回寄存器文件）。在 Allocator 之前，流水线都是顺序执行的，在 Allocator 之后，就可以进入乱序执行阶段了。在每一个线程方面，Nehalem 和 Core 2 架构相似，RAT 将重命名的、虚拟的寄存器（称为 Architectural Register 或 Logical Register）指向 ROB 或者 RRF。RAT 是一式两份，每个线程独立，每个 RAT 包含了 128 个重命名寄存器（Pentium 4 之前——Pentium Pro 到 Pentium III 的重命名寄存器基于 ROB，数量为 40 个；据说 Pentium 4 有 128 个）。RAT 指向在 ROB 里面的最近的执行寄存器状态，或者指向 RRF 保存的最终的提交状态。

ROB（Re-Order Buffer，重排序缓冲区）是一个非常重要的部件，它是将乱序执行完毕的指令们按照程序编程的原始顺序重新排序的一个队列，以保证所有的指令都能够逻辑上

实现正确的因果关系。打乱了次序的指令们依次插入这个队列，当一条指令通过 RAT 发往下一个阶段确实执行的时候这条指令（包括寄存器状态在内）将被加入 ROB 队列的一端，执行完毕的指令（包括寄存器状态）将从 ROB 队列的另一端移除（期间这些指令的数据可以被一些中间计算结果刷新），因为调度器是 In-Order 顺序的，这个队列也就是顺序的。从 ROB 中移出一条指令就意味着指令执行完毕了，这个阶段叫做 Retire 回退，相应地 ROB 往往也叫做 Retirement Unit（回退单元），并将其画为流水线的最后一部分。

在一些超标量设计中，Retire 阶段会将 ROB 的数据写入 L1D 缓存，而在另一些设计里，写入 L1D 缓存由另外的队列完成。例如，Core/Nehalem 的这个操作就由 MOB（Memory Order Buffer，内存重排序缓冲区）来完成。



ReOrder Buffer 重排序缓冲区是一个 Retire Queue 回退队列，是 OOOE 乱序架构流水线的最后一段

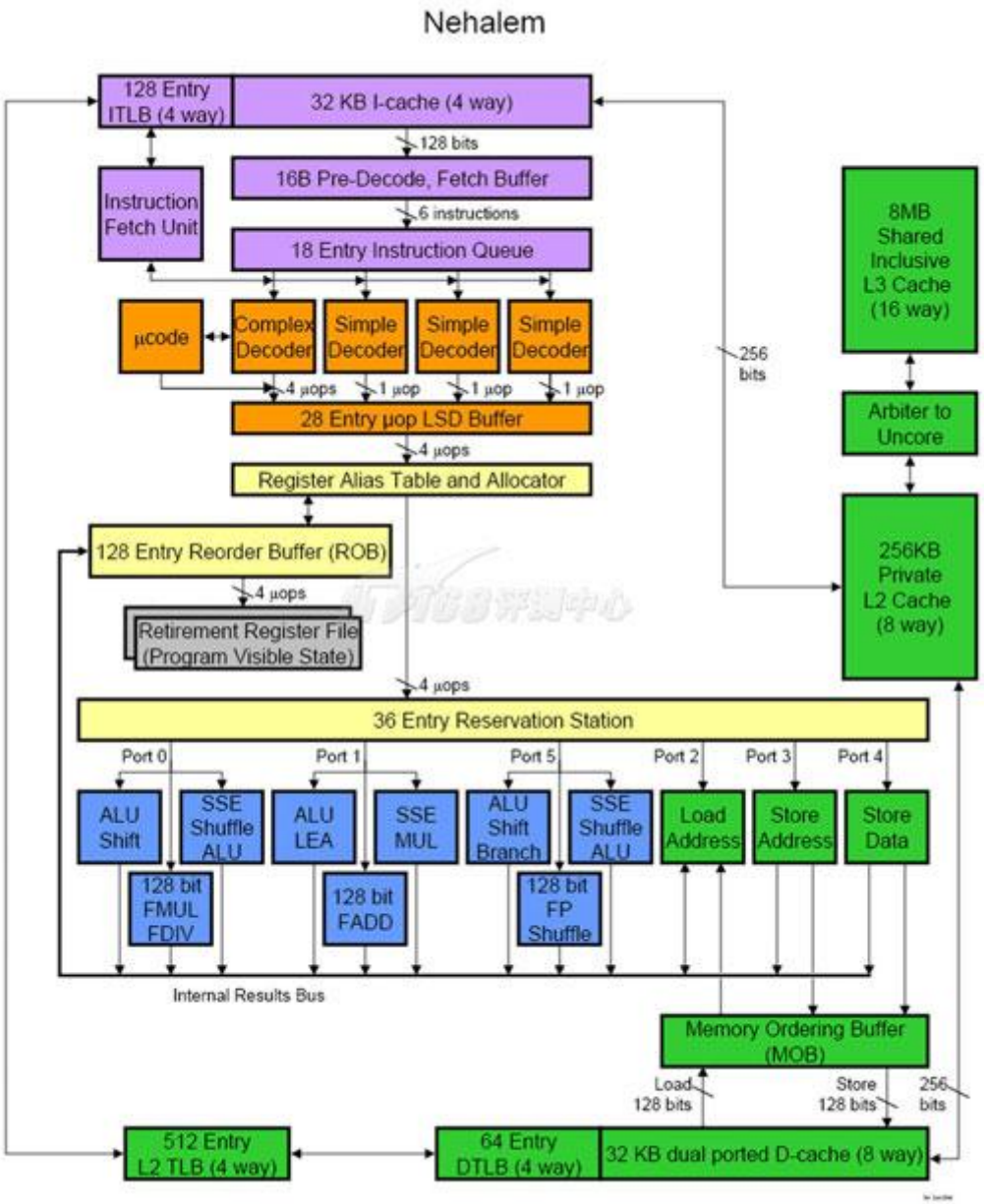
ROB 是乱序执行引擎架构中都存在的一个缓冲区，重新排序指令的目的是将指令们的寄存器状态依次提交到 RRF 退回寄存器文件当中，以确保具有因果关系的指令们在乱序执行中可以得到正确的数据。从执行单元返回的数据会将先前由调度器加入 ROB 的指令刷新数据部分并标志为结束（Finished），再经过其他检查通过后才能标志为完毕（Complete），一旦标志为完毕，它就可以提交数据并删除重命名项目并退出 ROB 了。提交状态的工作由 Retirement Unit（回退单元）完成，它将确实完毕的指令包含的数据写入 RRF（“确实”的意思是，非猜测执性、具备正确因果关系，程序可以见到的最终的寄存器状态）。和 RAT 一样，RRF 也同时具有两个，每个线程独立。Core/Nehalem 的 Retirement Unit 回退单元每时钟周期可以执行 4 个 uops 的寄存器文件写入，和 RAT 每时钟 4 个 uops 的重命名一致。

由于 ROB 里面保存的指令数目是如此之大（128 条目），因此一些人认为它的作用是用来从中挑选出不相关的指令来进入执行单元，这多少是受到一些文档中的 Out-of-Order Window 乱序窗口这个词的影响（后面会看到对 ROB 会和 MOB 一起被计入乱序窗口资源中）。

ROB 确实具有 RS 的一部分相似的作用，不过，ROB 里面的指令是调度器通过 RAT 发往 RS 的同时发往 ROB 的，也就是说，在“乱序”之前，ROB 的指令就已经确定了。指令并不是在 ROB 当中乱序挑选的（这在 RS 当中进行），ROB 担当的是流水线的最终阶段：一个指令的 Retire 回退单元；以及担当中间计算结果的缓冲区。

RS（Reservation Station，中继站）：等待源数据到来以进行 OOOE 乱序执行（没有数据的指令将在 RS 等待）

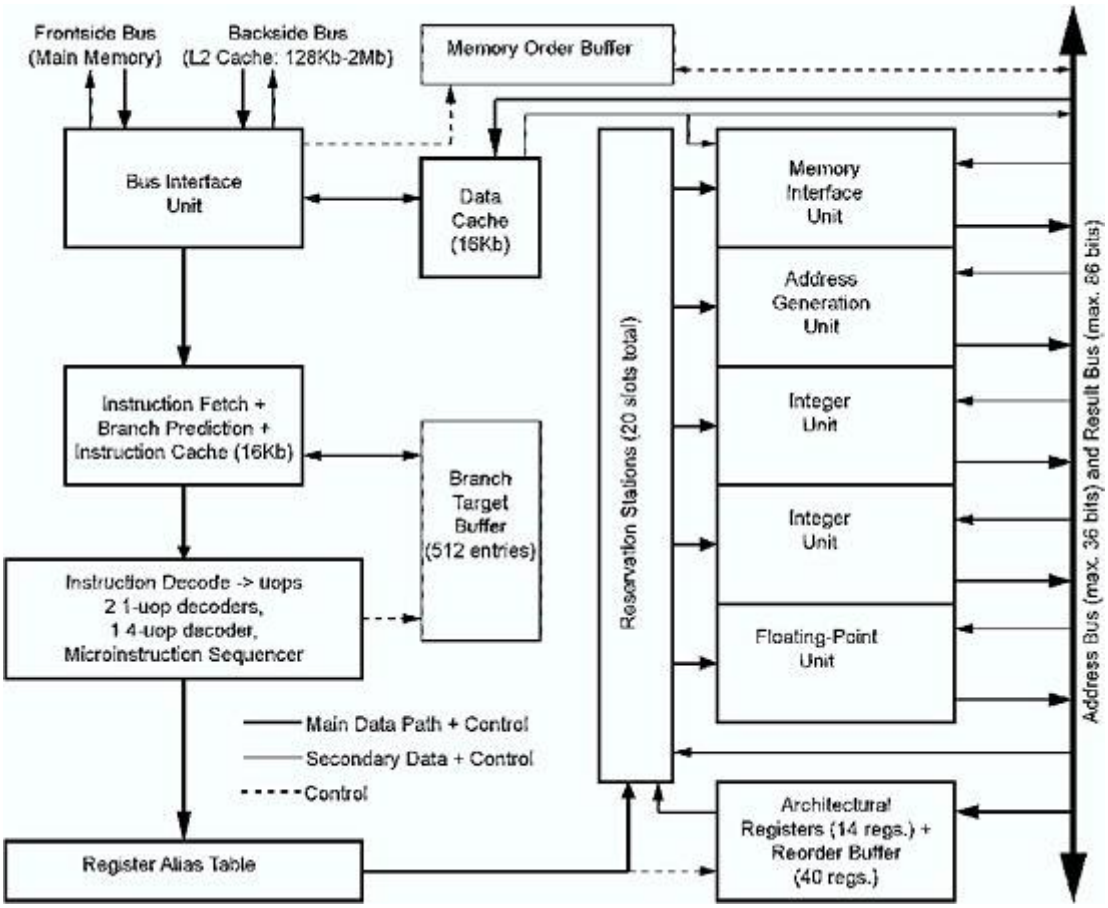
ROB（ReOrder Buffer，重排序缓冲区）：等待结果到达以进行 Retire 指令回退（没有结果的指令将在 ROB 等待）



The Core Execution Engine: Out-of-Order Execution

Nehalem 的 128 条目的 ROB 担当中间计算结果的缓冲区，它保存着猜测执行的指令及其数据（猜测执行也是在 Pentium Pro 开始引入的，Pentium Pro 真是一个划时代的产品），猜测执行允许预先执行方向未定的分支指令，Nehalem 的分支预测成功率没有被提及（只是含糊地说“业内最高”）。在大部分情况下，猜测执行工作良好——分支猜对了，因此其在 ROB 里产生的结果被标志为已结束，可以立即地被后继指令使用而不需要进行 L1 Data Cache 的 Load 操作（这也是 ROB 的另一个重要用处，典型的 x86 应用中 Load 操作是如此频繁，达到了几乎占 1/3 的地步，因此 ROB 可以避免大量的 Cache Load 操作，作用巨大）。在剩下

的不幸的情况下，分支未能按照如期的情况进行，这时猜测的分支指令段将被清除，相应指令们的流水线阶段清空，对应的寄存器状态也就全都无效了，这种无效的寄存器状态不会也不能出现在 RRF 里面。



Pentium Pro 架构，可见在后继型号在 OOOE 方面变化的基本只是一些数字变大了

重命名技术并不是没有代价的，在获得前面所说的众多的优点之后，它令指令在发射的时候需要扫描额外的地方来寻找到正确的寄存器状态，不过总体来说这种代价是非常值得的。RAT 可以在每一个时钟周期重命名 4 个 uops 的寄存器，经过重命名的指令在读取到正确的操作数并发射到统一的 RS（Reservation Station，中继站，Intel 文档翻译为保留站点）上。RS 中继站保存了所有等待执行的指令。

和 Core 2 相比，Nehalem 的 ROB 大小和 RS 大小都得到了提升，ROB 重排序缓冲区从 96 条目提升到 128 条目（鼻祖 Pentium Pro 具有 40 条），RS 中继站从 32 提升到 36（Pentium Pro 为 20），它们都在两个线程（超线程中的线程）内共享，不过采用了不同的策略：ROB 是采用了静态的分区方法，而 RS 则采用了动态共享，因为有时候会有一条线程内的指令因等待数据而停滞，这时另一个线程就可以获得更多的 RS 资源。停滞的指令不会发往 RS，但是仍然会占用 ROB 条目。由于 ROB 是静态分区，因此在开启 HTT 的情况下，每一个线程只能分到 64 条，不算多，在一些极少数的应用上，我们应该可以观察到一些应用开启 HTT 后会速度降低，尽管可能非常微小。

第 8 页：深入 Nehalem 微架构：乱序执行单元

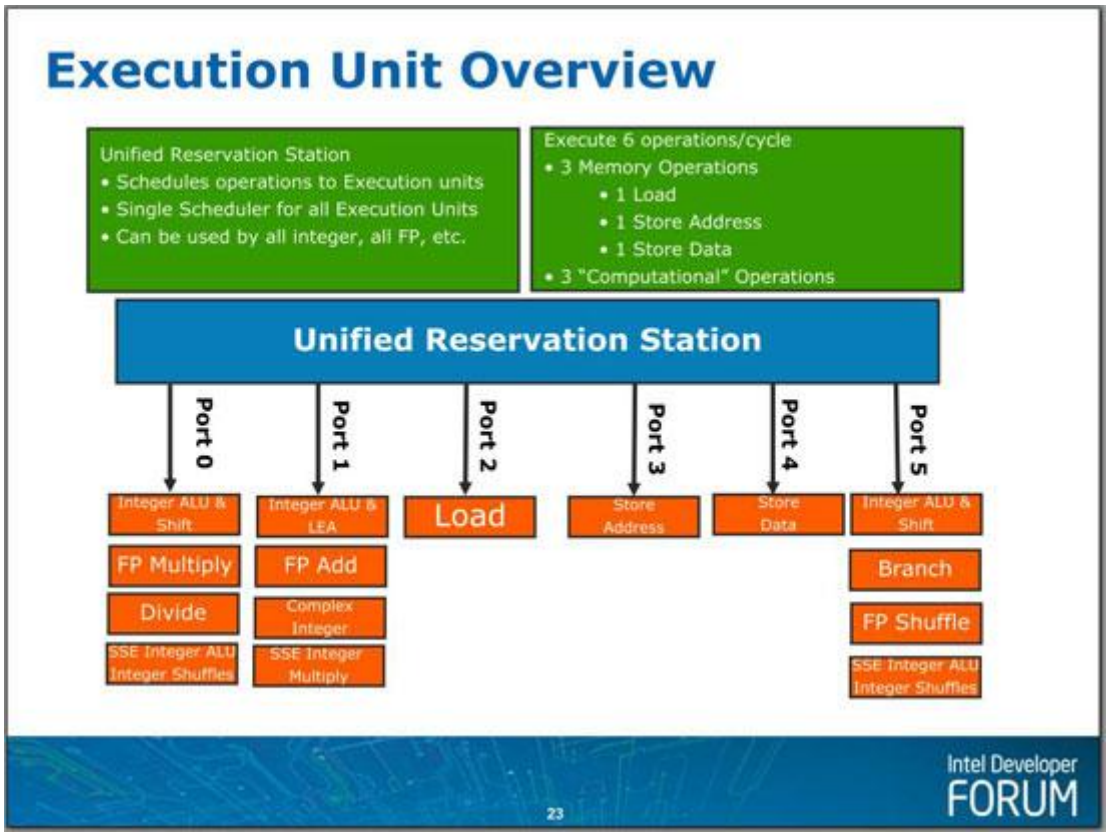
The Core Execution Engine: Execution Unit

处理器核心执行引擎：执行单元

在为 SMT 做好准备工作并打乱指令的执行顺序之后，uops 通过每时钟周期 4 条的速度进入 Reservation Station 中继站（保留站），总共 36 条目的中继站 uops 就开始等待超标量（Superscaler）执行引擎乱序执行了。自从 Pentium 开始，Intel 就开始在处理器里面采用了超标量设计（Pentium 是两路超标量处理器），超标量的意思就是多个执行单元，它可以同时执行多条没有相互依赖性的指令，从而达到提升 ILP 指令级并行化的目的。Nehalem 具备 6 个执行端口，每个执行端口具有多个不同的单元以执行不同的任务，然而同一时间只能有一条指令（uop）进入执行端口，因此也可以认为 Nehalem 有 6 个“执行单元”，在每个时钟周期内可以执行最多 6 个操作（或者说，6 条指令），和 Core 一样；令人意外的是，古老的 Pentium 4 每时钟周期也能执行最多 6 个指令，虽然它只有 4 个执行端口，然而其中两个执行端口的 ALU 单元是双倍速的（Double Pump，每时钟周期执行两条 ALU 指令）。

Nehalem:Superscale Execution Unit 超标量执行单元

中文版本可能反而不便于理解（如负载操作实际上是 Load 载入操作），下面是英文原版：

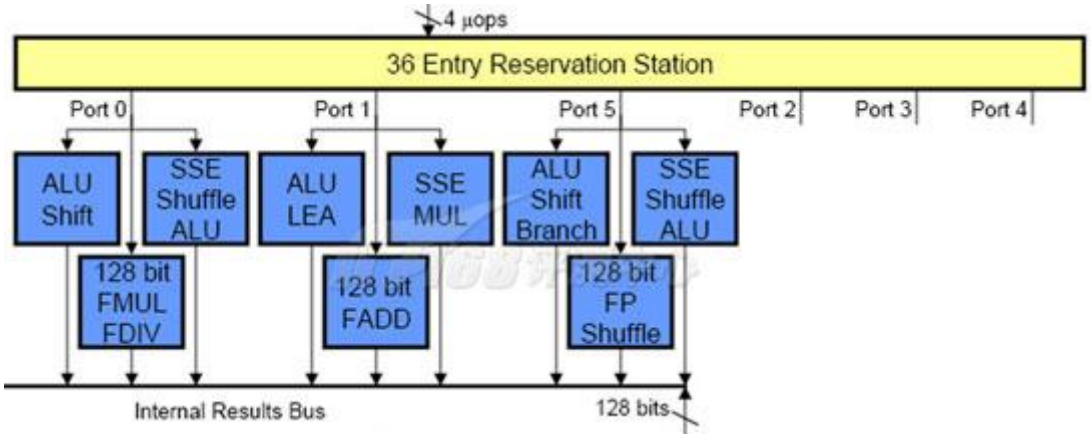


Nehalem:Superscale Execution Unit 超标量执行单元

36 条目的中继站指令在分发器的管理下，挑选出尽量多的可以同时执行的指令（也就是乱序执行的意思）——最多 6 条——发送到执行端口。

这些执行端口并不都是用于计算，实际上，有三个执行端口是专门用来执行内存相关的操作的，只有剩下的三个是计算端口，因此，在这一点上 Nehalem 实际上是跟 Core 架构一样的，这也可以解释为什么有些情况下，Nehalem 和 Core 相比没有什么性能提升。

计算操作分为两种：使用 ALU(Arithmetic Logic Unit，算术逻辑单元)的整数(Integer)运算和使用 FPU（Floating Point Unit，浮点运算单元）的浮点（Floating Point）运算。SSE 指令（包括 SSE1 到 SSE4）是一种特例，它虽然有整数也有浮点，然而它们使用的都是 128bit 浮点寄存器，使用的也大部分是 FPU 电路。在 Nehalem 中，三个计算端口都可以做整数运算（包括 MMX）或者 SSE 运算（浮点运算不太一样，只有两个端口可以进行浮点 ADD 和 MUL/DIV 运算，因此每时钟周期最多进行 2 个浮点计算，这也是目前 Intel 处理器浮点性能不如整数性能突出的原因），不过每一个执行端口都不是完全一致：只有端口 0 有浮点乘和除功能，只有端口 5 有分支能力（这个执行单元将会与分支预测单元连接），其他 FP/SSE 能力也不尽相同，这些不对称之处都由统一的分发器来理解，并进行指令的调度管理。没有采用完全对称的设计可能是基于统计学上的考虑。和 Core 一样，Nehalem 的也没有采用 Pentium 4 那样的 2 倍频的 ALU 设计（在 Pentium 4，ALU 的运算频率是 CPU 主频的两倍，因此整数性能明显要比浮点性能突出）。



The Core Execution Engine: "Computional" Unit

Nehalem 的 ALU 和 FP/SSE 单元都使用了相同的三个端口，相比之下，Barcelona Opteron 的 FP/SSE 单元和 ALU 单元具有不同的入口，因此每时钟周期可以同时执行最多 6 条计算指令。不过，Barcelona Opteron 的 3 存取操作使用和 ALU 单元一样的端口，因此其执行单元每时钟周期可以同时执行的指令仍然为 6 条。

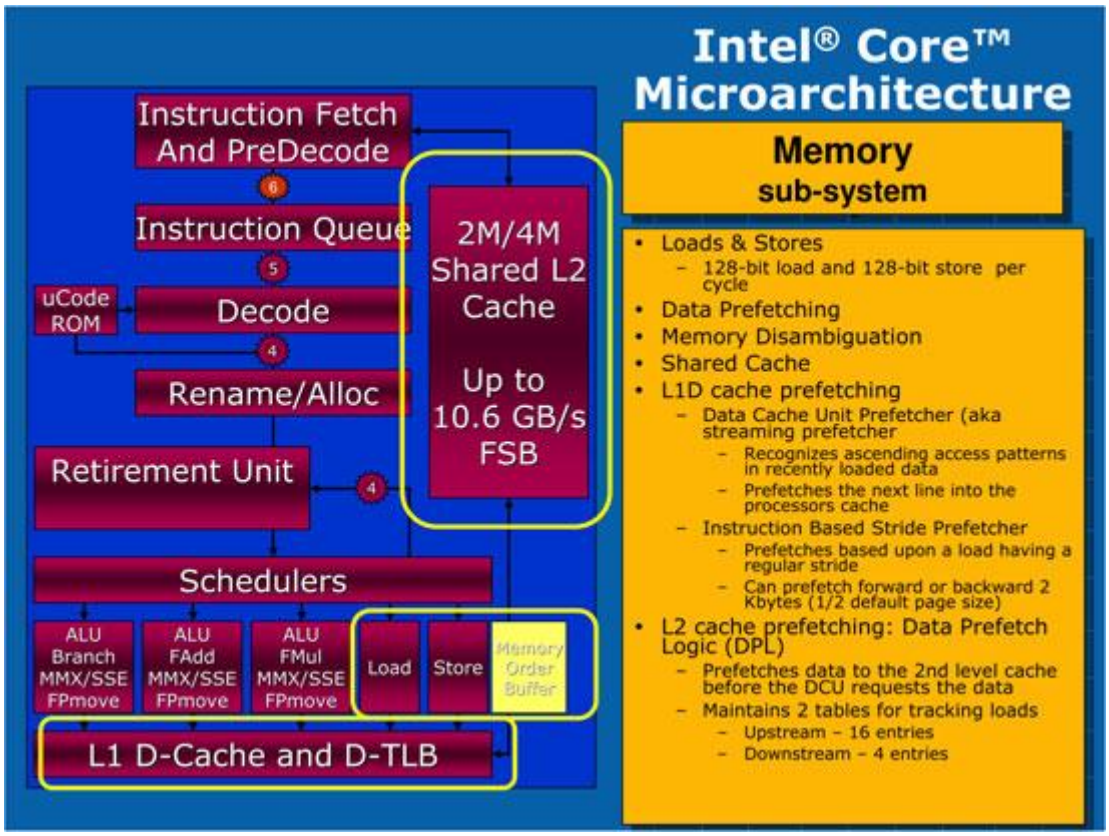
不幸的是，虽然可以同时执行的指令很多，然而在流水线架构当中运行速度并不是由最“宽”的单元来决定的，而是由最“窄”的单元来决定的。这就是木桶原理，Opteron 的解码器后端只能每时钟周期输出 3 条 uops，而 Nehalem/Core2 则能输出 4 条，因此它们的实际最大每时钟运行指令数是 3/4，而不是 6。同样地，多少路超标量在这些乱序架构处理器中也不再按照运算单元来划分，Core Duo 及之前（到 Pentium Pro 为止）均为三路超标量处理器，Core 2/Nehalem 则为四路超标量处理器。可见在微架构上，Nehalem/Core 显然是要比其他处理器快一些。顺便说一下，这也是 Intel 在超线程示意图中，使用 4 个宽度的方块来表示而不是 6 个方块的原因。

第 9 页：深入 Nehalem 微架构：乱序存取单元

The Core Execution Engine: Load/Store Unit

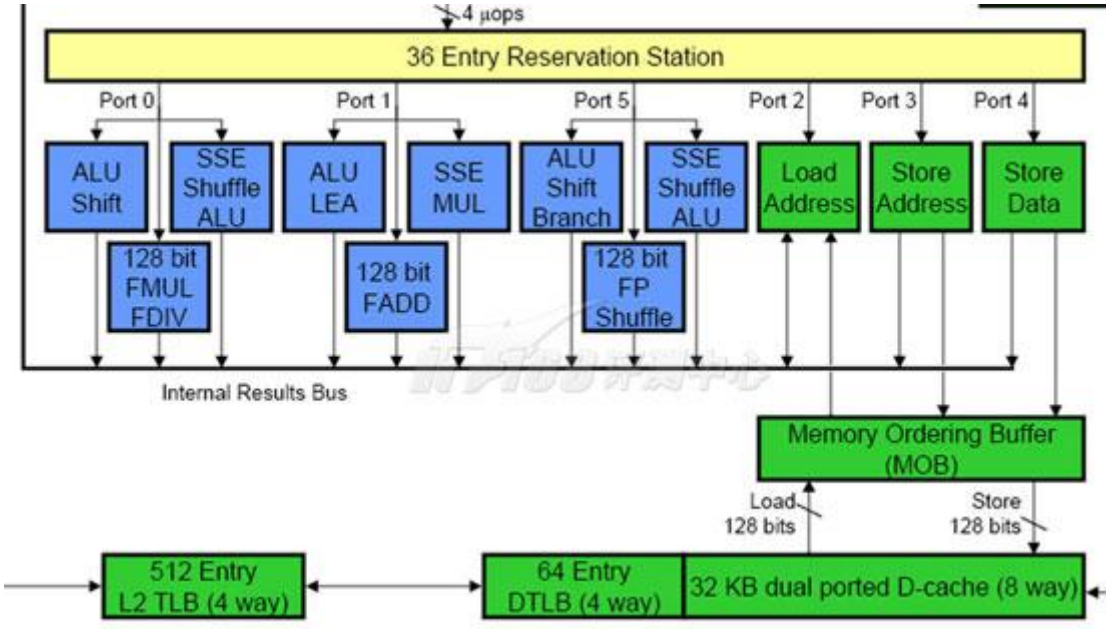
处理器核心执行引擎：存取单元

运算需要用到数据，也会生成数据，这些数据存取操作就是存取单元所做的事情，实际上，Nehalem 和 Core 的存取单元没什么变化，仍然是 3 个。



Nehalem 的 Load/Store 结构和 Core 架构一样

这三个存取单元中，一个用于所有的 Load 操作（地址和数据），一个用于 Store 地址，一个用于 Store 数据，前两个数据相关的单元带有 AGU（Address Generation Unit，地址生成单元）功能（NetBurst 架构使用快速 ALU 来进行地址生成）。



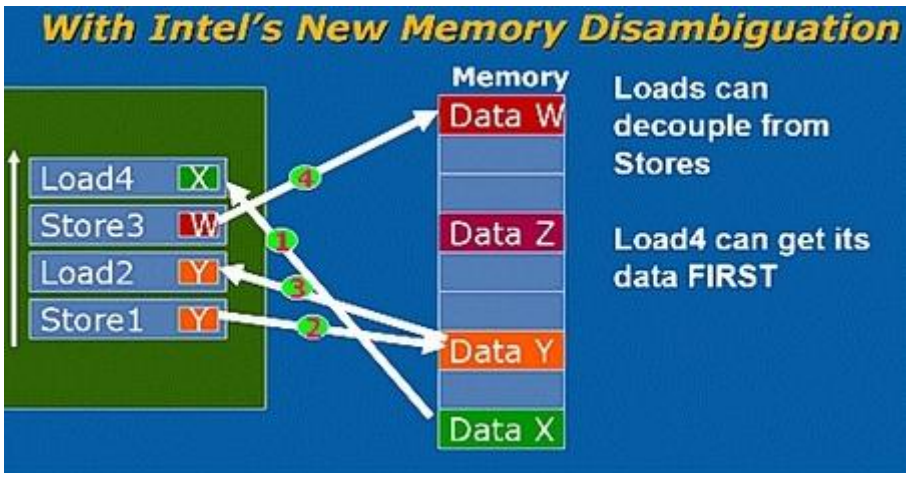
The Core Execution Engine: Load/Store Unit

在乱序架构中，存取操作也可以打乱进行。类似于指令预取一样，Load/Store 操作也可以提前进行以降低延迟的影响，提高性能。然而，由于 Store 操作会修改数据影响后继的

Load 操作，而指令却不会有这种问题（寄存器依赖性问题通过 ROB 解决），因此数据的乱序操作更为复杂。

数据乱序操作的困境：Load/Store 依赖性

如上图所示，第一条 ALU 指令的运算结果要 Store 在地址 Y（第二条指令），而第九条指令是从地址 Y Load 数据，显然在第二条指令执行完毕之前，无法移动第九条指令，否则将会产生错误的结果。同样，如果 CPU 也不知道第五条指令会使用什么地址，所以它也无法确定是否可以把第九条指令移动到第五条指令附近。



内存数据相依性预测功能（Memory Disambiguation）

内存数据相依性预测功能（Memory Disambiguation）可以预测哪些指令是具有依赖性的或者使用相关的地址(地址混淆，Alias)，从而决定哪些 Load/Store 指令是可以提前的，哪些是不可以提前的。可以提前的指令在其后继指令需要数据之前就开始执行、读取数据到 ROB 当中，这样后继指令就可以直接从中使用数据，从而避免访问了无法提前 Load/Store 时访问 L1 缓存带来的延迟（3~4 个时钟周期）。

不过，为了要判断一个 Load 指令所操作的地址没有问题，缓存系统需要检查处于 in-flight 状态（处理器流水线中所有未执行的指令）的 Store 操作，这是一个颇耗费资源的过程。在 NetBurst 微架构中，通过把一条 Store 指令分解为两个 uops——一个用于计算地址、一个用于真正的存储数据，这种方式可以提前预知 Store 指令所操作的地址，初步的解决了数据相依性问题。在 NetBurst 微架构中，Load/Store 乱序操作的算法遵循以下几条原则：

- 如果一个对于未知地址进行操作的 Store 指令处于 in-flight 状态，那么所有的 Load 指令都要被延迟

- 在操作相同地址的 Store 指令之前 Load 指令不能继续执行
- 一个 Store 指令不能移动到另外一个 Store 指令之前

这种原则下的问题也很明显，比如第一条原则会在一条处于等待状态的 Store 指令所操作的地址未确定之前，就延迟所有的 Load 操作，显然过于保守了。实际上，地址冲突问题是极少发生的。根据某些机构的研究，在一个 Alpha EV6 处理器中最多可以允许 512 条指令处于 in-flight 状态，但是其中的 97%以上的 Load 和 Store 指令都不会存在地址冲突问题。

基于这种理念，Core 微架构采用了大胆的做法，它令 Load 指令总是提前进行，除非新加入的动态混淆预测器（Dynamic Alias Predictor）预测到了该 Load 指令不能被移动到 Store 指令附近。这个预测是根据历史行为来进行的，据说准确率超过 90%。

在执行了预 Load 之后，一个冲突监测器会扫描 MOB 的 Store 队列，检查该是否有 Store 操作与该 Load 冲突。在很不幸的情况下（1%~2%），发现了冲突，那么该 Load 操作作废、流水线清除并重新进行 Load 操作。这样大约会损失 20 个时钟周期的时间，然而从整体上看，Core 微架构的激进 Load/Store 乱序策略确实很有效地提升了性能，因为 Load 操作占据了通常程序的 1/3 左右，并且 Load 操作可能会导致巨大的延迟（在命中的情况下，Core 的 L1D Cache 延迟为 3 个时钟周期，Nehalem 则为 4 个。L1 未命中时则会访问 L2 缓存，一般为 10~12 个时钟周期。访问 L3 通常需要 30~40 个时钟周期，访问主内存则可以达到最多约 100 个时钟周期）。Store 操作并不重要，什么时候写入到 L1 乃至主内存并不会影响到执行性能。

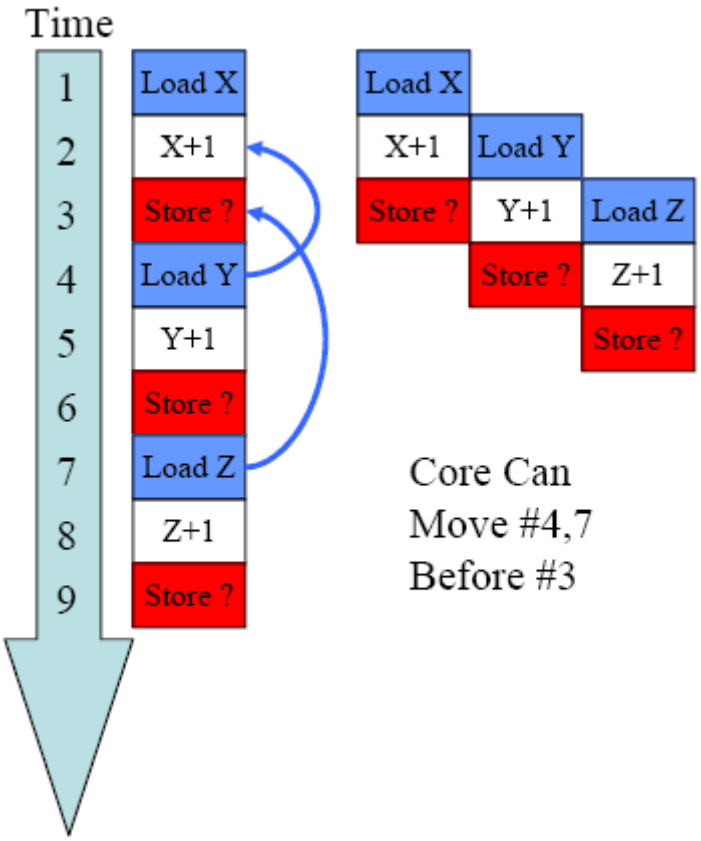


图 9：数据相依性预测机制的优势

如上图所示，我们需要载入地址 X 的数据，加 1 之后保存结果；载入地址 Y 的数据，加 1 之后保存结果；载入地址 Z 的数据，加 1 之后保存结果。如果根据 Netburst 的基本准则，在第三条指令未决定要存储在什么地址之前，处理器是不能移动第四条指令和第七条指令的。

实际上，它们之间并没有依赖性。因此，Core 微架构中则“大胆”的将第四条指令和第七条指令分别移动到第二和第三指令的并行位置，这种行为是基于一定的猜测的基础上的“投机”行为，如果猜测的对的话（几率在 90%以上），完成所有的运算只要 5 个周期，相比之前的 9 个周期几乎快了一倍。

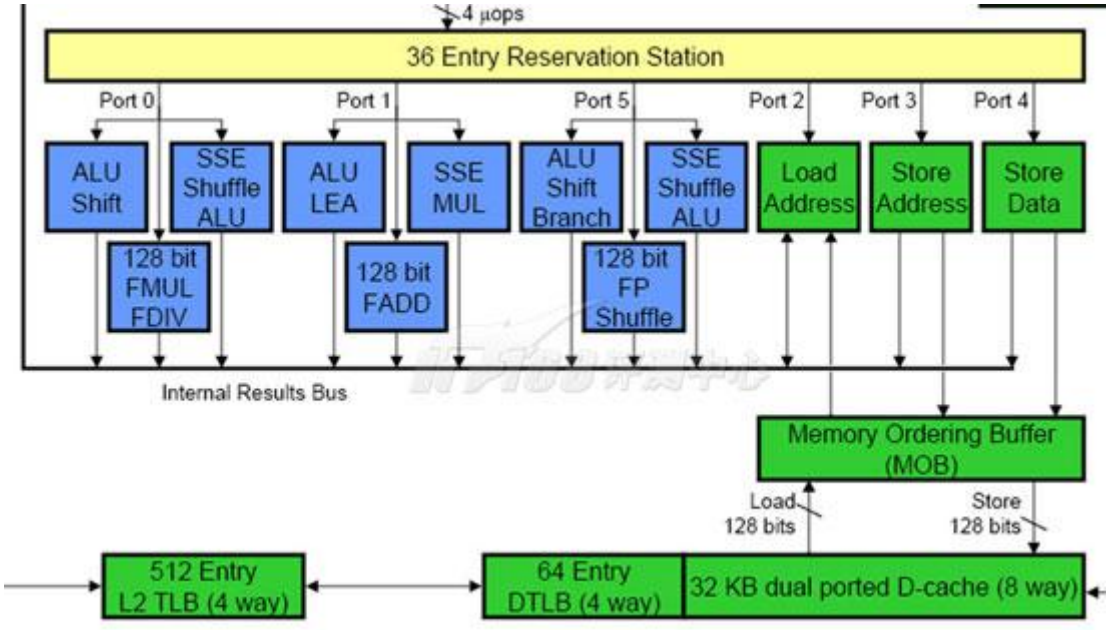
第 10 页：深入 Nehalem 微架构：乱序存取单元

The Core Execution Engine: Load/Store Unit

处理器核心执行引擎：存取单元

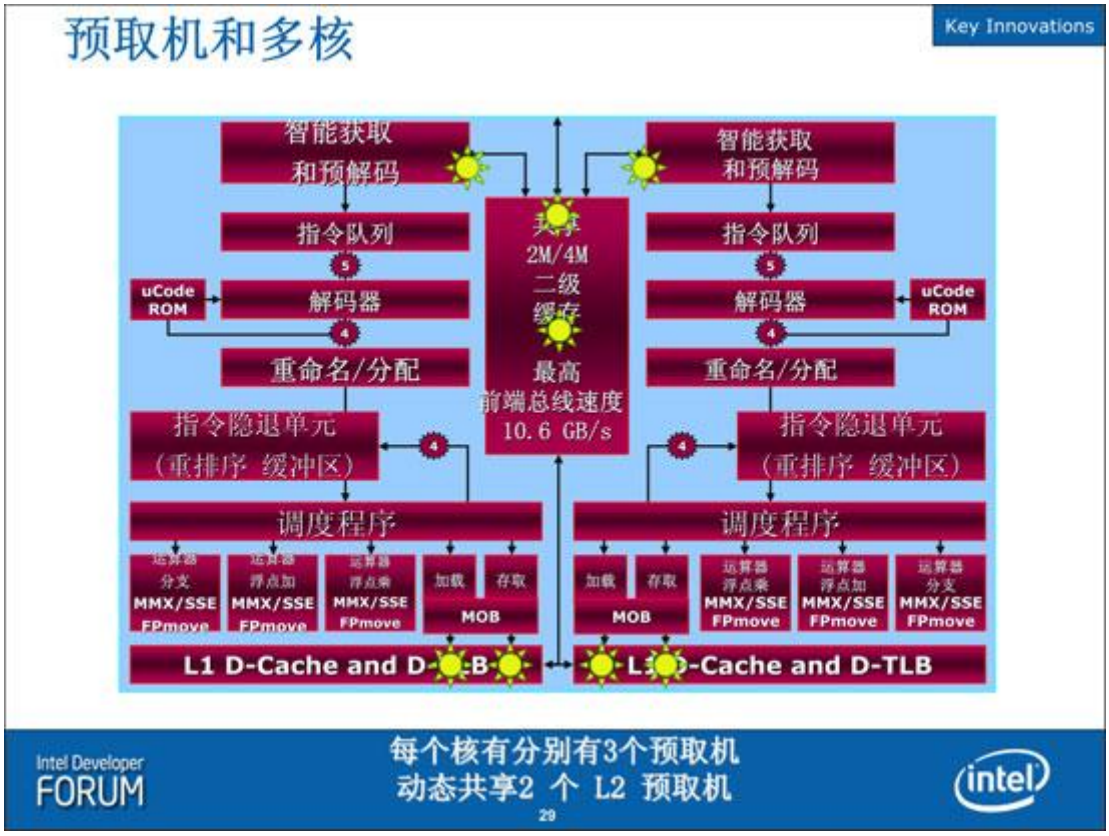
和为了顺序提交到寄存器而需要 ROB 重排序缓冲区的存在一样，在乱序架构中，多个打乱了顺序的 Load 操作和 Store 操作也需要按顺序提交到内存，**MOB（Memory Reorder Buffer，内存重排序缓冲区）**就是起到这样一个作用的重排序缓冲区（上图，介于 Load/Store 单元与 L1D Cache 之间的部件），所有的 Load/Store 操作都需要经过 MOB，MOB 通过一个 128bit 位宽的 Load 通道与一个 128bit 位宽的 Store 通道与双口 L1D Cache 通信。和 ROB 一样，MOB 的内容按照 Load/Store 指令分发（Dispatched）的顺序加入队列的一端，按照提交到 L1D Cache 的顺序从队列的另一端移除。ROB 和 MOB 一起实际上形成了一个分布式的 Order Buffer 结构，有些处理器上只存在 ROB，兼备了 MOB 的功能。

和 ROB 一样，Load/Store 单元的乱序存取操作会在 MOB 中按照原始程序顺序排列，以提供正确的数据，内存数据依赖性检测功能也在里面实现。Intel 没有给出 MOB 详细的结构——包括外部拓扑结构在内，在一些玩家制作的架构图当中，MOB 被放在 Load/Store 单元与 Internal Results Bus 之间并互相联结起来，意思是 MOB 的 Load/Store 操作结果也会直接反映到 ROB 当中。

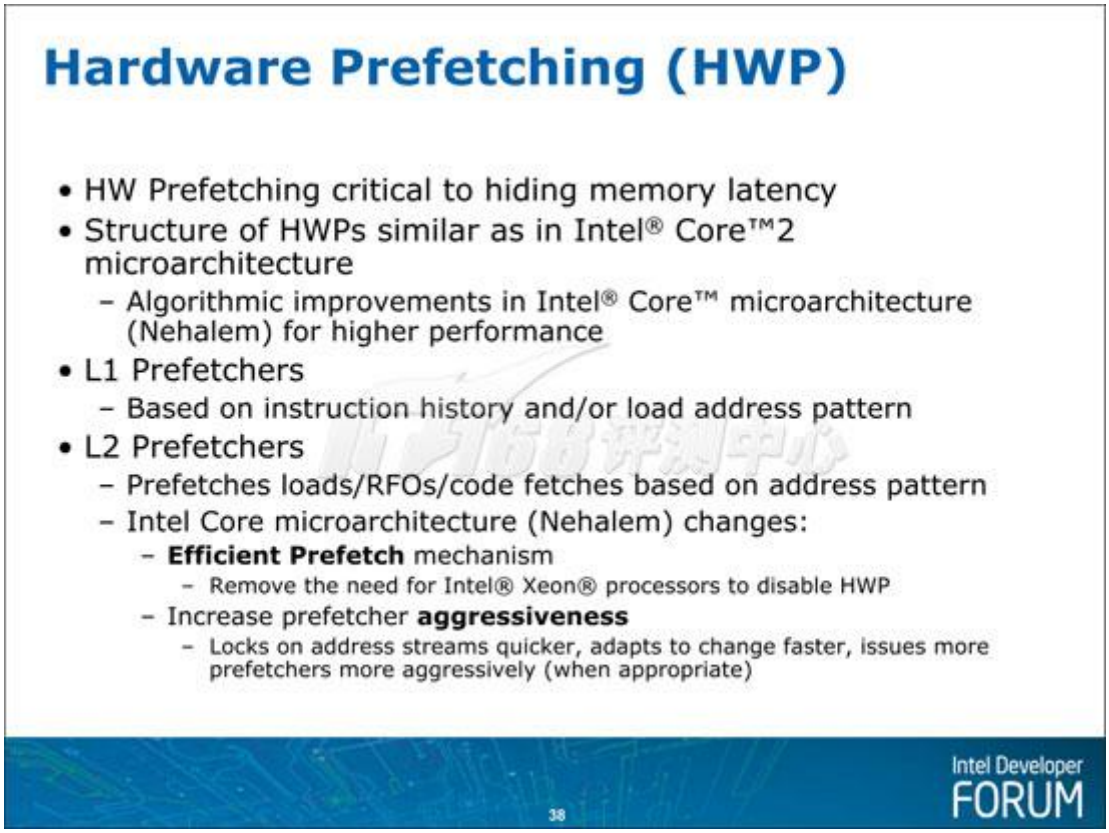


The Core Execution Engine: Load/Store Unit

然而基于以下的一个事实，笔者将其与 Internal Results Bus 进行了隔离：MOB 还附带了数据预取（Data Prefetch）功能，它会猜测未来指令会使用到的数据，并预先从 L1D Cache 缓存 Load 入 MOB 中(Data Prefetcher 也会对 L2 至系统内存的数据进行这样的操作)，这样 MOB 当中的数据有些在 ROB 中是不存在的(这有些像 ROB 当中的 Speculative Execution 猜测执行，MOB 当中也存在着“Speculative Load Execution 猜测载入”，只不过失败的猜测执行会导致管线停顿，而失败的猜测载入仅仅会影响到性能）。此外，MOB 与 L1D 之间是数据总线，不带有指令，经过 MOB 内部的乱序执行之后，ROB 并不知道进出的数据对应哪一条指令。最终笔者制作的架构图就如上方所示。

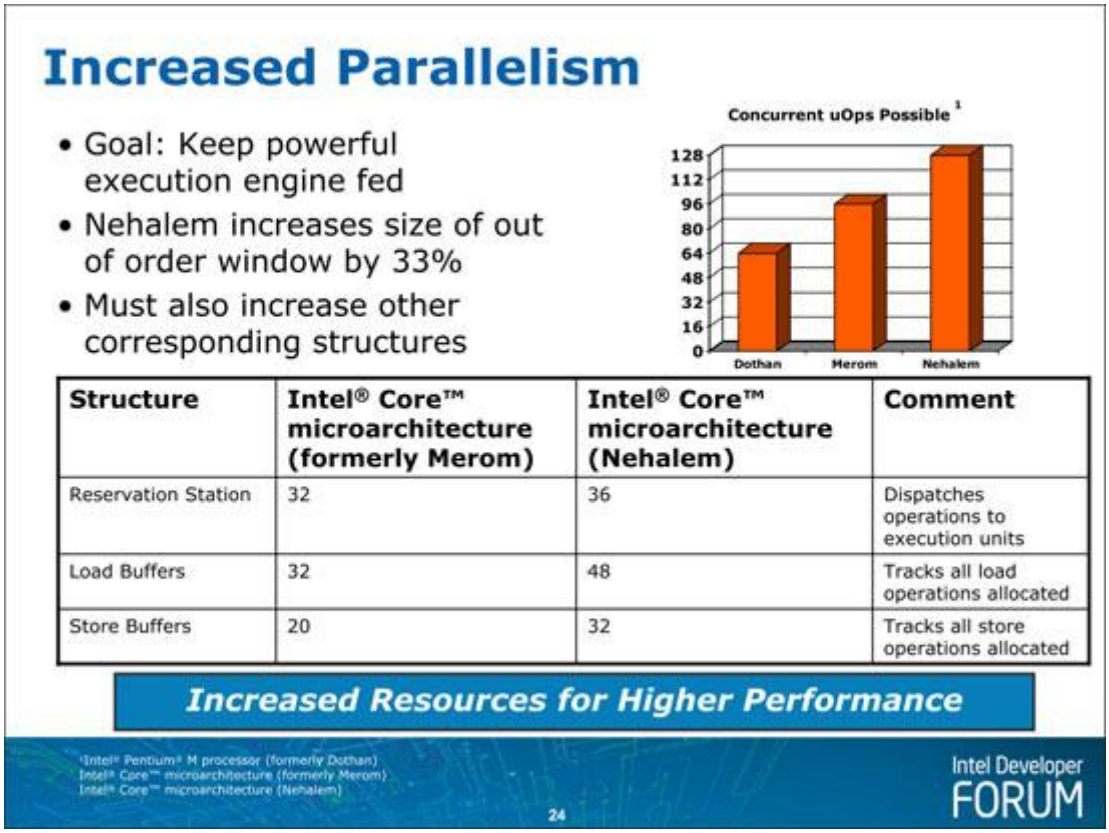


每个 Core 2 内核具有 3 个 Prefetcher（1 个指令，两个数据）；每两个核心共享两个 L2 Prefetcher



Nehalem 的 Hardware Prefetcher 功能，其中 L1 Prefetchers 基于指令历史以及载入地址参数

数据预取功能和指令预取功能一起，统称为 Hardware Prefetcher 硬件预取器。笔者在年少时对 BIOS 里面通常放在 CPU 特性那一页里面的 Hardware Prefetcher 迷惑不解（通常在一起的还有一个 Adjacent Cache Line Prefetch 相邻缓存行预取，据说这些选项不包含 L1 Prefetcher；亦尚不清楚是否包括 MOB 的预取功能），现在我们知道了这两个功能就是和这一页的内容相关的。很不幸，在以往的 CPU 中，失败的预取将会白白浪费掉 L1/L2/L3/Memory 的带宽，而在服务器应用上通常会进行跨度很大的 Load 操作，因此 Hardware Prefetcher 经常会起到降低性能的作用。对于这种情况，处理器厂商们除了在 BIOS 里面给出一个设置选项就没有更好的方法了（这些选项在桌面应用上工作良好）。糟糕的是，很多用户都不知道这些选项是干什么用的。据说 Nehalem 上这个情况得到了好转，用户可以简单地设置为 Enable 而不用担心性能下降。或许以后我们 IT168 评测中心会进行相关的测试检验是否是这样，不过我们可以想象，内存带宽得到巨大提升的 Nehalem 已经具有足够的资本来开启这些选项了。



提高并行度：扩大 RS 和 MOB 的容量（MOB 包括了 Load Buffers 和 Store Buffers），所谓的乱序窗口资源

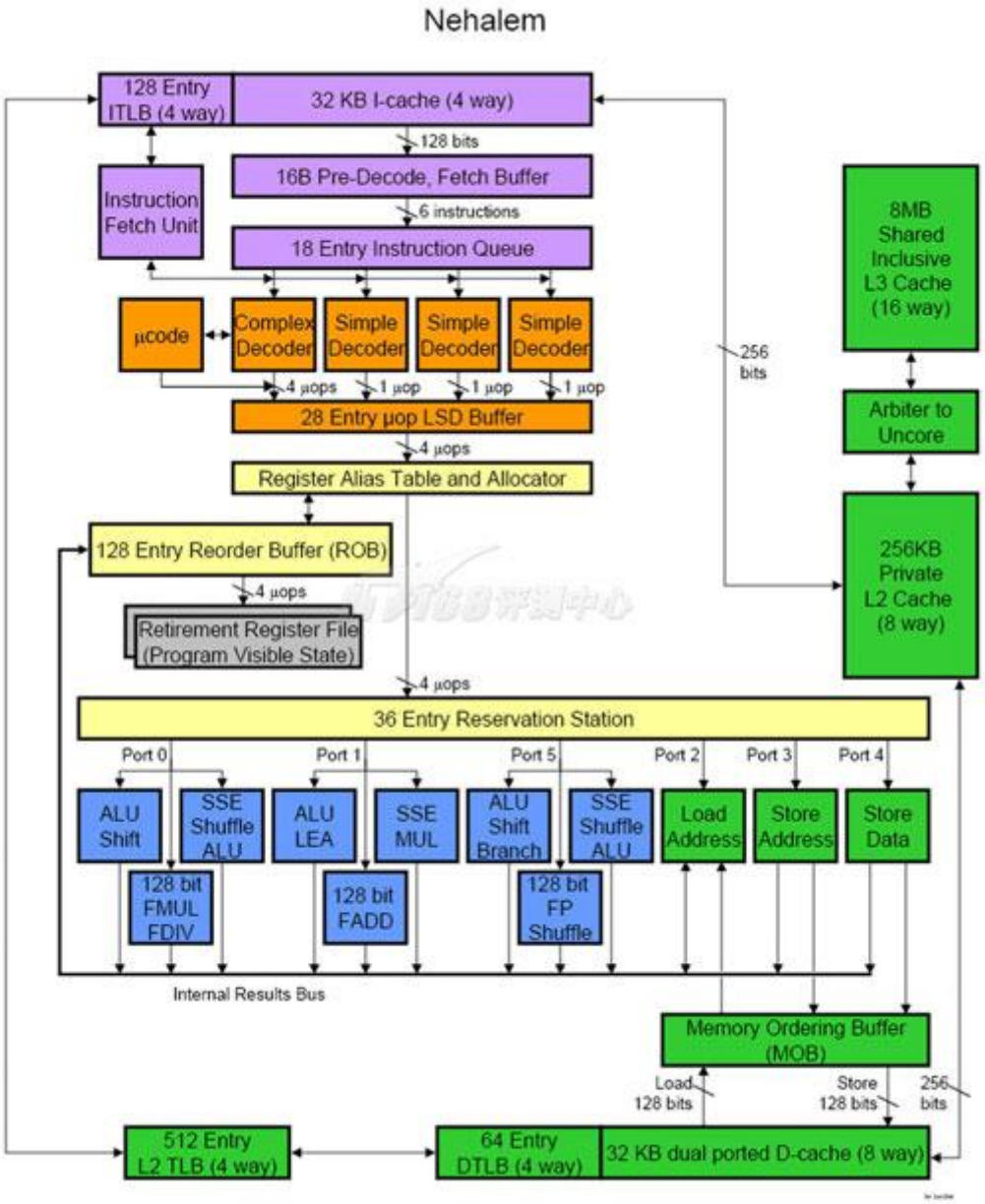
乱序执行中我们可以看到很多缓冲区性质的东西：RAT 寄存器别名表、ROB 重排序缓冲区、RS 中继站、MOB 内存重排序缓冲区（包括 LB 载入缓冲和 SB 存储缓冲）。在超线程的作用下，RAT 是一式两份，包含了 128 个重命名寄存器；128 条目的 ROB、48 条目的 LB 和 32 条目的 SB 都静态划分为两个分区：每个线程 64 个 ROB、24 个 LB 和 16 个 SB；RS 则是在两个线程中动态共享。可见，虽然整体数量增加了，然而就单个线程而言，获得的资源并没有提升。这会影响到 HTT 下单线程下的性能。

第 11 页：深入 Nehalem 微架构：缓存子系统

The Memory sub-System: Cache

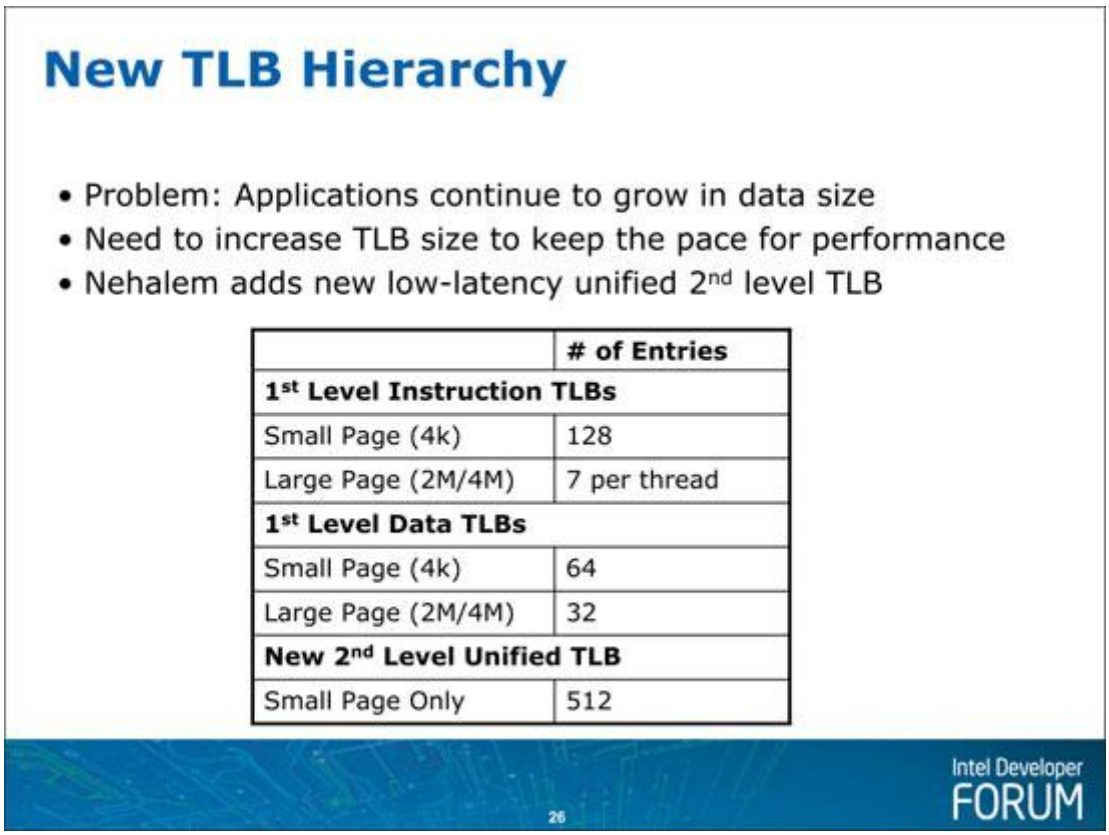
内存子系统：缓存

MOB 通过两条 128 位宽的 Load/Store 通道与 L1D Cache 连接，L1D Cache 同时通过 256 位宽的总线与 L2 连接：L1D Cache 是双口（Dual Ported）的。在缓存方面，Nehalem 和 Core 相比具有了一些变化。



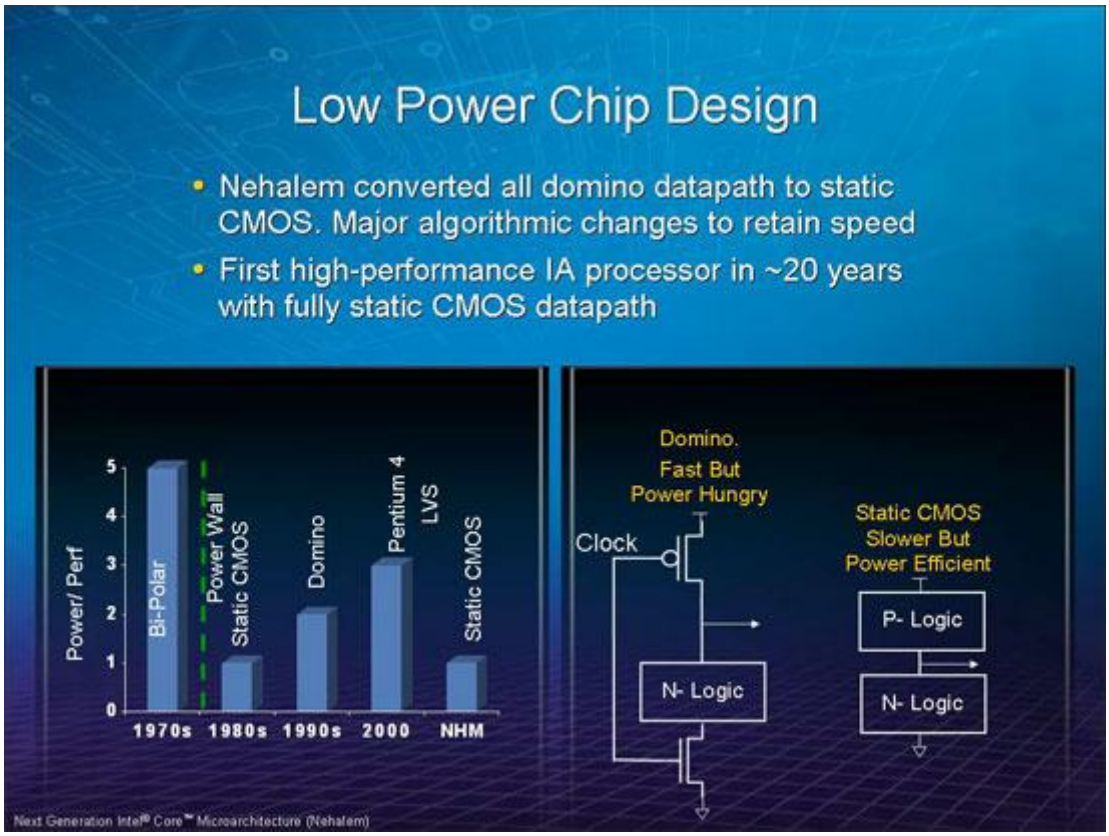
绿色部分都属于缓存相关部分

Nehalem/Core 的 L1I Cache（L1 指令缓存）和 L1D Cache（L1 数据缓存）都是 32KB，不过 Nehalem 的 L1I Cache 从以往的 8 路集合关联降低到了 4 路集合关联，L1 DTLB 也从以往的 256 条目降低到 64 条目（64 个小页面 TLB，32 个大页面 TLB），并且 L1 DTLB 是在两个多线程之间动态共享的（L1 ITLB 的小页面部分则是静态分区，也就是 64 条目每线程，是 Core 2 每线程 128 条目的一半；每个线程还具有 7 个大页面 L1I TLB）。



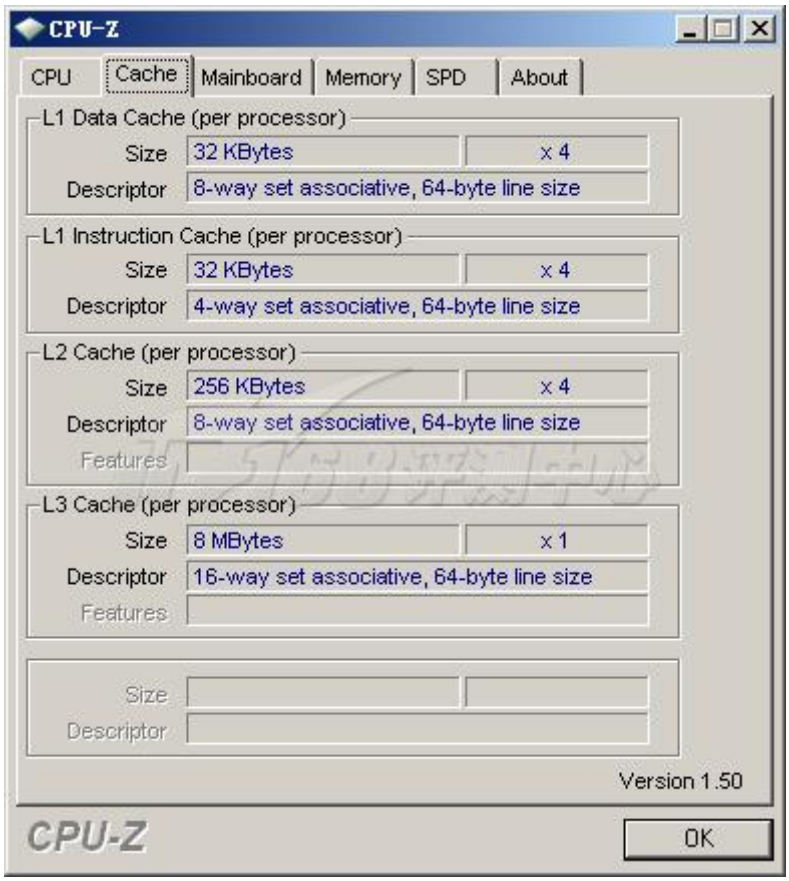
Nehalem TLB 架构

为什么 L1I Cache 的集合关联降低了呢？这都是为了降低延迟的缘故。随着现代应用程序对数据容量的要求在加大，需要提升 TLB 的大小来相应满足（TLB：Translation Lookaside Buffer，旁路转换缓冲，或称为页表缓冲；里面存放的是虚拟地址到物理地址的转换表，供处理器以及具备分页机构的操作系统用来快速定位内存页面；大概很多人知道 TLB 是因为 AMD 的处理器 TLB Bug 事件）。Nehalem 采用了较小的 L1 TLB 附加一层较大的 L2 TLB 的方法来解决这个问题（512 个条目以覆盖足够大的内存区域，它仅用于较小的页面，指令和数据共用，两个线程共享）。



为了降低能耗，Nehalem 架构将以往应用的 Domino 线路更换为 Static CMOS 线路，并大规模使用了长沟道晶体管技术，速度有所降低，但是能源效率提升了

虽然如此，Nehalem L1D Cache 的延迟仍然从 Core 2 的 3 个时钟周期上升到了 4 个时钟周期，这是由于线路架构改变的缘故（从 Domino 更换成 Static CMOS，大量使用长沟到晶体管）。类似地 L1I Cache 乃至 L2、L3 的延迟都相应地会上升，然而指令缓冲的延迟对性能的影响要比数据严重；每一次取指令都会受到延迟影响，而缓存的延迟则可以通过乱序执行和猜测载入来解决。因此 Intel 将 L1I Cache 的集合关联从 8 路降低到 4 路，以维持延迟仍然在 3 个时钟周期。



Nehalem-EP Xeon X5570 的缓存结构：64KB L1，256KBL2，8MB 共享 L3

第 12 页：深入 Nehalem 微架构：缓存子系统

The Memory sub-System: Cache

内存子系统：缓存

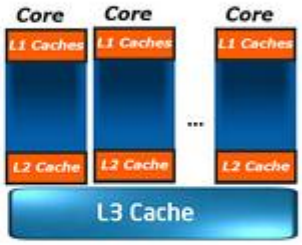
与 Core 2 相比，Nehalem 新增加了一层 L3 缓存，这是为了多个核心共享数据的需要（Nehalem-EX 具有 8 个核心），也因此这个 L3 的容量很大。出于消除多核心共享数据的压力，前面的缓存不能让太多的缓存请求到达 L3，而且 L3 的延迟（约 30~40 个时钟周期）和 L1 的延迟（3~4 个时钟周期）相差太大，因此 L2 是很有必要的。Nehalem 简单地在很小的 L1 和大尺寸的 L3 之间插入 256KB 的 L2 来起到中继的作用——中继具有两个方面的含义：容量和延迟。256KB 不算大，可以维持约低于 10 个时钟周期的延迟。Nehalem 的 L2 和 L1 不是包含也不是非包含的关系。

英特尔智能高速缓存– 内核的缓存

- 新的3级高速缓存构架
- 第1级高速缓存
  - 32kB 指令缓存
  - 32kB 数据缓存
    - 比酷睿 2 在并行模式下支持更多的1级缓存错过
- 第2级高速缓存
  - 在Nehalem中引入的新缓存
  - 统一化 (保持代码和数据)
  - 每个内核256 kB
  - *性能*: 非常低的延迟
  - *可扩展性*: 由于内核数量的增长，减小共享缓存的压力

英特尔智能缓存 – 第3级高速缓存

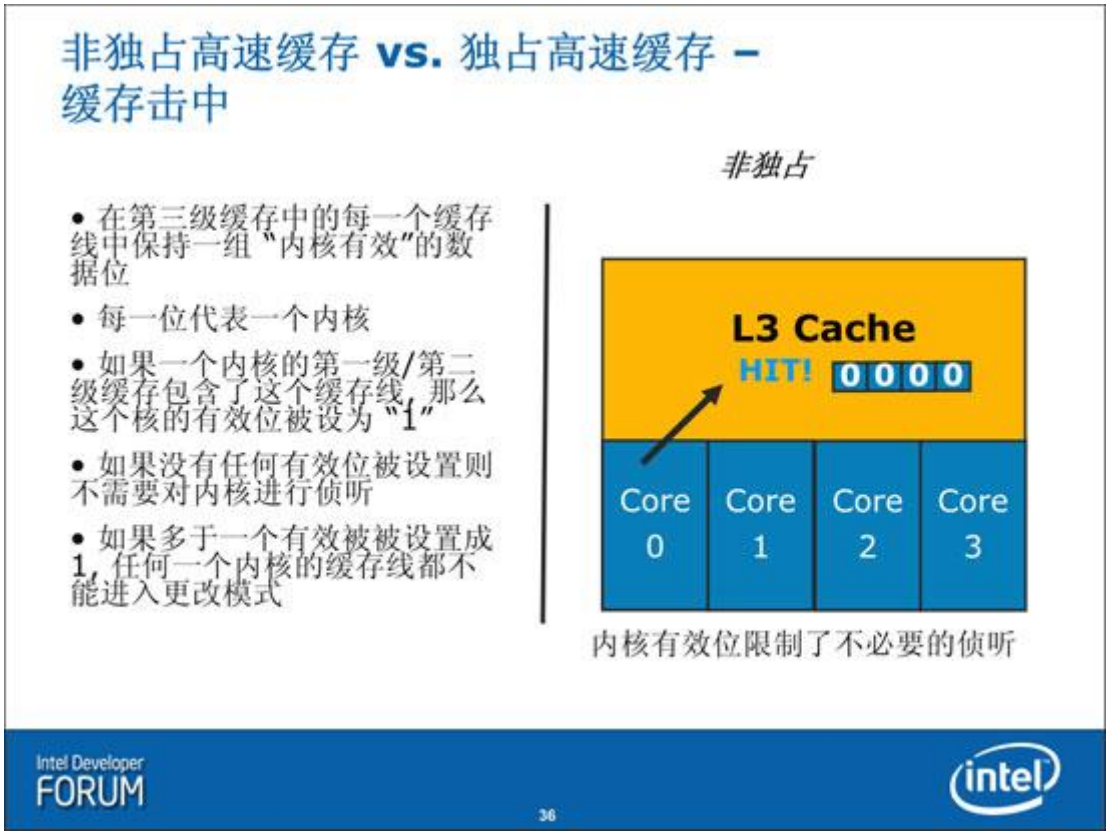
- 新的第3级高速缓存
- 所有内核共享
- 根据内核数量决定缓存大小
  - 4核: 最大 8MB
  - *可扩展性*:
    - 根据不同的内核数量采用不同大小的缓存
    - 在未来的产品中使第3级缓存的容量增加变得更容易
- 为了提供最优的*性能*采用非独占的第3级高速缓存
  - L1/L2级缓存中贮存的数据*必须*在第3级高速缓存中存在



通常缓存具有两种设计：非独占和独占，Nehalem 处理器的 L3 采用了非独占高速缓存设计（或者说“包含式”，L3 包含了 L1/L2 的内容），这种方式在 Cache Miss 的时候比独占式具有更好的性能，而在缓存命中的时候需要检查不同的核心的缓存一致性。Nehalem 并采用了“内核有效”数据位的额外设计，降低了这种检查带来的性能影响。随着核心数目的逐渐增多（多线程的加入也会增加 Cache Miss 率），对缓存的压力也会继续增大，因此这种方式会比较符合未来的趋势。在后面可以看到，这种设计也是考虑到了多处理器协作的情

况（此时 Miss 率会很容易地增加）。这可以看作是 Nehalem 与以往架构的基础不同：之前的架构都是来源于移动处理设计，而 Nehalem 则同时为企业、桌面和移动考虑而设计。

在 L3 缓存命中的时候（单处理器上是最通常的情况，多处理器下则不然），处理器检查内核有效位看看是否其他内核也有请求的缓存页面内容，决定是否需要对内核进行侦听：



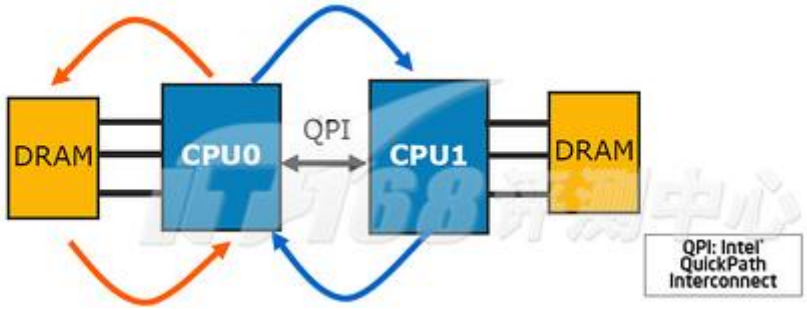
笔者相信这一点是不对的：假如一个 L3 页面被多个内核共享（多于一个有效被设置为 1），那么这个处理器的该页面就不能进入 Modified 状态

基于后面的 NUMA 章节的内容，多个处理器中的同一个缓存页面必定在其中一个处理器中属于 F 状态（可以修改的状态），这个页面在这个处理器中没有理由不可以多核心共享（可以多核心共享就意味着这个能进入修改状态的页面的多个有效位被设置为一）。笔者相信 MESIF 协议应该是工作在核心（L1+L2）层面而不是处理器（L3）层面，这样同一处理器里多个核心共享的页面，只有其中一个是出于 F 状态（可以修改的状态）。见后面对 NUMA 和 MESIF 的解析。（20110113：L1/L2/L3 的同步应该是不需要 MESIF 的同步机制）

在 L3 缓存未命中的时候（多处理器下会频繁发生），处理器决定进行内存存取，按照页面的物理位置，它分为近端内存存取（本地内存空间）和远端内存存取（地址在其他处理器的内存的空间）：

近段内存存取

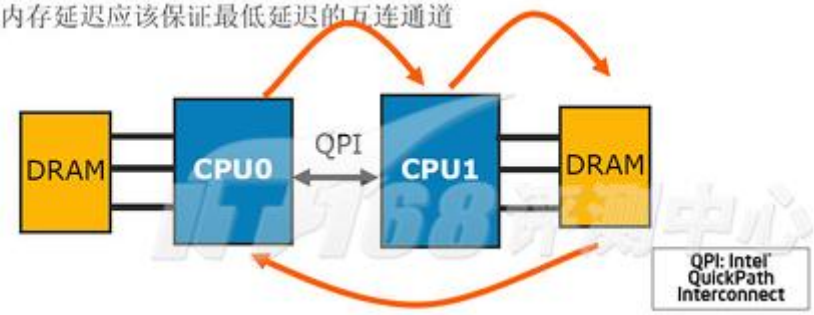
- CPU0 发送请求缓存线 X, 如果不存在任何 CPU0 的缓存中
  - CPU0 从它的DRAM中请求数据
  - CPU0 通过侦听 CPU1 来检查数据是否存在
- 第2步:
  - DRAM 返回数据
  - CPU1 返回侦测响应
- 近端内存延迟是取这两个响应时间中的最大值
- Nehalem 经过优化可以使各个延迟之间差距较小



Cache Miss 时而页面地址为本地的时候，处理器进行近端内存访问延迟取本地内存访问和远程 CPU Cache Hit 的延迟的最大值

远端内存存取

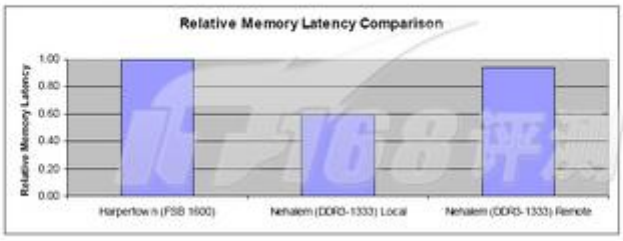
- CPU0 发送请求缓存线 X, 如果不存在
  - CPU0 从 CPU1获取数据
  - 通过 QPI 向 CPU1发送请求
  - CPU1 的IMC 向它的 DRAM 发送数据请求
  - CPU1 侦听内部缓存
  - 数据通过QPI 返回到 CPU0
- 远端内存延迟应该保证最低延迟的互连通道



Cache Miss 时而页面地址为远程的时候，处理器进行远端内存访问延迟取远程内存访问和远程 CPU Cache Hit 的延迟的最大值

内存延迟比较

- *低内存延迟* 对高性能的实现至关重要
- 为了减小延迟设计了集成的内存控制器
- 需要优化近端和远端的内存延迟
- Nehalem 提供了
  - 近端内存延迟大大减小
  - 即使是远端内存延迟也比较快速
- 根据不同应用程序/操作系统分别实现有效的内存延迟降低
  - 近端存取 vs. 远端存取的比例
  - 即使是混合模式NHM 也有更低的延迟



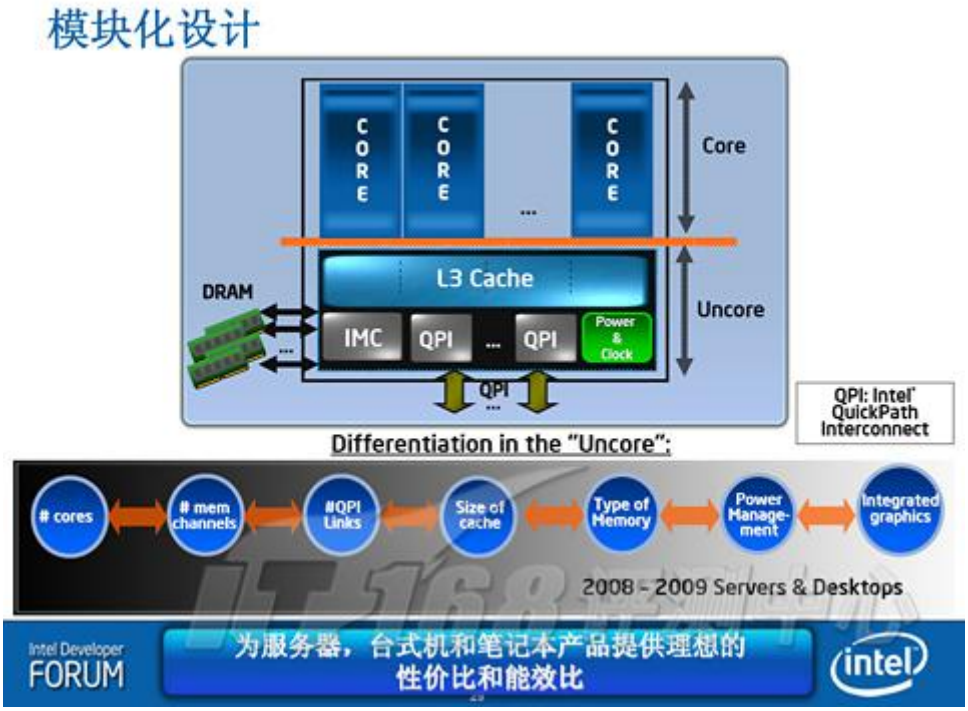
近端访问约 60 个时钟周期，远端访问约 90 个时钟周期(据说仍然比 Harptertown Xeon 快)，本地 L3 Cache Hit 则为 30 个时钟周期

第 13 页：深入 Nehalem 微架构：核外系统/IMC

The Uncore: IMC

核外系统：集成内存控制器

从形式上来看，L3 缓存、集成的内存控制器乃至 QPI 总线都属于 Uncore 核外部分，从 L2、L1 一直到执行单元都属于 Core 核内部分。由于 Nehalem 首次采用了这种核心内外的相对独立设计思路，因此核心之外的设计相对于 Core 架构来说显得新颖许多，这就是 Nehalem 的模块式设计。



模块式设计可以提供灵活的产品给用户，现在 4 核心、三通 IMC 和单 QPI 的桌面 Nehalem 已经面市，预计明年 3 月将会出现 4 核心、4 通道 IMC 和双 QPI 的企业级 Nehalem 产品。包含了 PCIE 控制器乃至集成显卡的产品也已经在路线上了。

继续回到处理器架构：我们都知道，Nehalem 和 Intel 以往处理器相比最大的特点就是直联架构——包括两个方面：处理器直联以及内存直联，前者就是依靠 QPI 总线的实现，后者则是由于处理器内置了内存控制器（IMC，Integrated Memory Controller）。当处理器在 L3 Cache 未找到所要内容（L3 Cache Miss）的时候，它将会继续通过 IMC 集成内存控制器往系统内存索取，同时通过 QPI 总线询问其他处理器（如果是多处理器平台）。

为什么直联架构可以很明显地提升性能？这要先从 x86 架构的存储体系说起。在很久很久以前，在一个记忆体短缺的时代——不仅仅处理器外面记忆体很少，处理器里面也是。使用了 CISC 架构的 x86 处理器里面只有 8 个 GPR 通用寄存器（一般的 RISC 处理器有 32 个以上的通用寄存器，现在的 x86-64 有 16 个通用寄存器），由于通用寄存器数量上的短缺，因此不像 RISC 处理器那样，CISC 的 x86 处理器使用了堆叠运算指令。堆叠运算也就是将运算结果保存在源寄存器上的，如 ADD AX, BX 指令会将 AX 寄存器与 BX 寄存器的内容相加，并将结果保存到AX上——这样对比于使用三个寄存器做同一运算的非堆叠指令RISC架构就节约了一个寄存器，然而相应地源寄存器的内存就销毁了。x86 架构需要执行大量的 Load/Store 微指令（Pentium Pro 开始具备）来进行寄存器-寄存器或寄存器-内存之间的数据搬运操作。RISC 处理器当中，Load/Store 操作也很频繁。（20110113：寄存器间操作不需要 Load/Store 参与）

如前面所述，最常用的 20 条 x86 指令当中：

mov 占 35%（寄存器之间、寄存器与内存之间移动数据），push 占 10%（压入堆栈，也经常用来传递参数），call 占 6%，cmp 占 5%，add、pop、lea 占 4%（实际计算指令非常少）

mov、push、pop 都是和 load/store 直接相关的，add、cmp 等则间接相关

顺便：

75%的 x86 指令短于 4 bytes，也就是小于 32 bits。不过这些短指令只占代码大小的 53%——有一些指令非常长

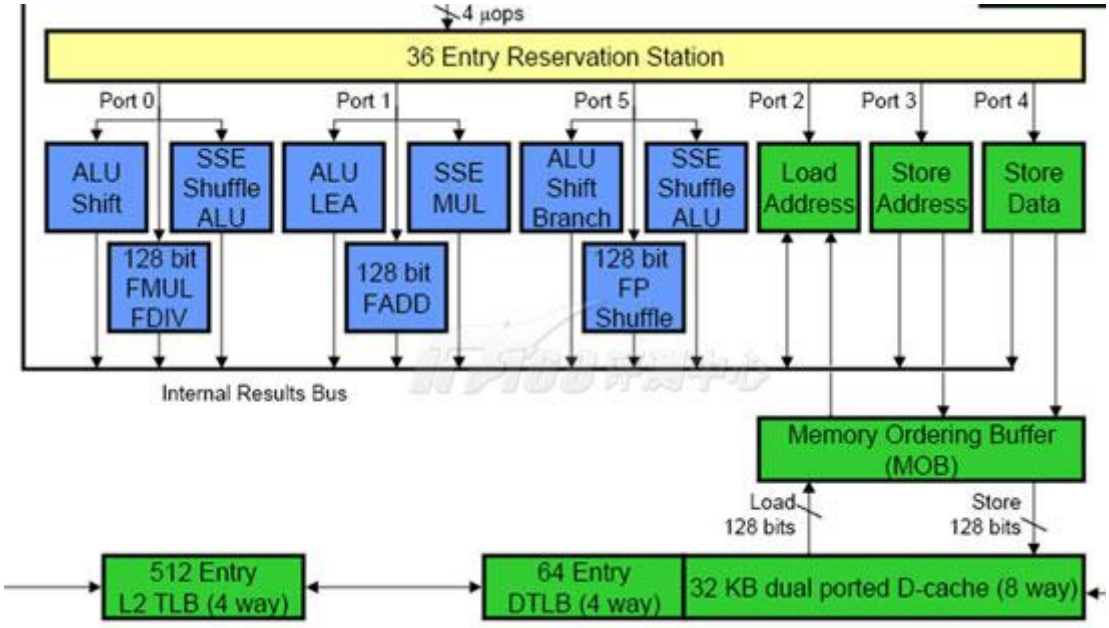
单操作数指令占 37%，双操作数指令占 60%

双操作数指令中，直接数操作 20%，寄存器操作数 56%，绝对寻址操作数 1%，间接寻址操作数 23%

Load 操作占据了 x86 uops 当中的约 30%

大量的 Load/Store 操作已经通过 ROB/MOB 降低到一定程度，不过，在多核心／超线程的情况下，对缓存/内存子系统仍然具有很大的压力

现在来看这样的设计简直是无法想象，不过这样脑残的设计不仅仅用到了今天，而且还加速到了一个不可思议的境界……在与各种 RISC 架构处理器的交锋也不落下风……回到架构上，由于 x86 架构实际上是通过耗费寄存器带宽及缓存-内存带宽来节约处理器内部寄存器数量，大量的 Load/Store 操作（Load 操作占据了 x86 uops 当中的约 30%），对缓存乃至内存的性能非常依赖。



Nehalem 具有三个 Load/Store 单元以及一个 MOB 架构，并支持内存数据相依性预测功能，缓存性能非常出色

缘此，x86 架构在缓存－内存上的提升是不遗余力，不提 2008 年度评测报告：深入 Nehalem 微架构中说到的内存数据相依性预测功能(Memory Disambiguation)，对于 Nehalem 而言，这方面最大的改进就是直联架构带来的 IMC 集成内存控制器，它使 CPU 到内存的路径更短，大幅度降低了内存的延迟，同时每一个 CPU 都具有自己专有的内存带宽。这一点在数据库应用中表现非常显著，数据库应用对存储器的延迟很敏感。

第 14 页：深入 Nehalem 微架构：核外系统/QPI

The Uncore:QPI

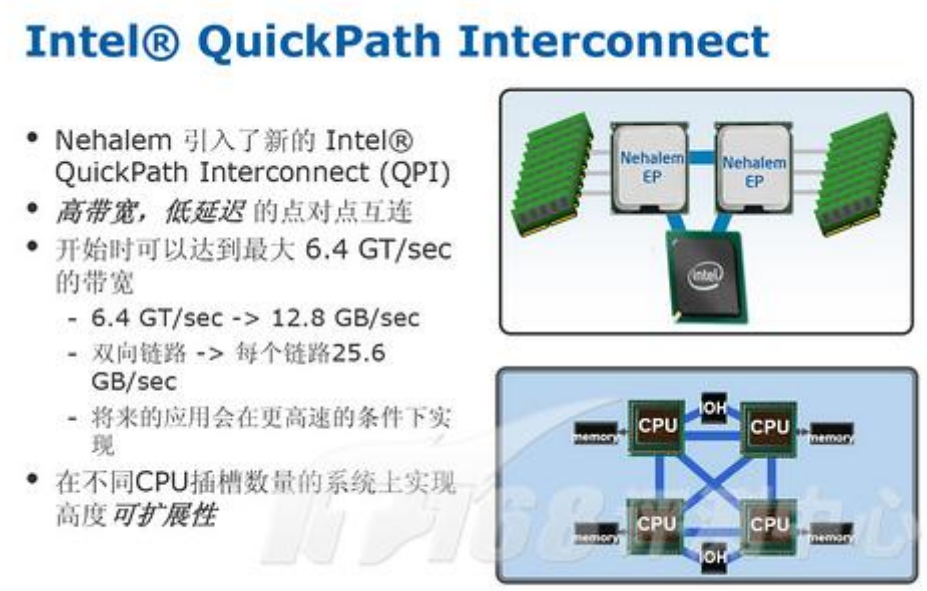
核外系统：QPI

直联架构不仅仅意味着处理器与内存直接相连，还让处理器之间也直接联系起来。Hyper-Transport 总线的使用让 Opteron 进入了高性能计算市场，QPI 所作的事情是一样的。通过 QPI 总线，处理器之间可以直接相连，不再需要经过拥挤、低带宽的 FSB 共享总线，多处理器系统运行效率大为提升。对于多处理器系统而言，QPI 提供的巨大带宽对性能提升很有作用。

QPI 总线 vs FSB 总线		
QPI vs. FSB		
名称	Intel FSB(Front Side Bus)	Intel QuickPath Interconnect(QPI)
拓扑	共享总线	点对点连接
物理总线宽度(bits)	64	20 x 2(双向)
数据总线宽度(bits)	64	16 x 2(双向)
传输速率	333MHz 1.333GT/s 10.6GB/s	3.2GHz 6.4GT/s 12.8GB/s(单向) 25.6GB/s(双向)
需要边带信号	是	否
引脚数	150	84

时钟数	1	1
集成时钟	否	否
总线传输方向	双向	单向

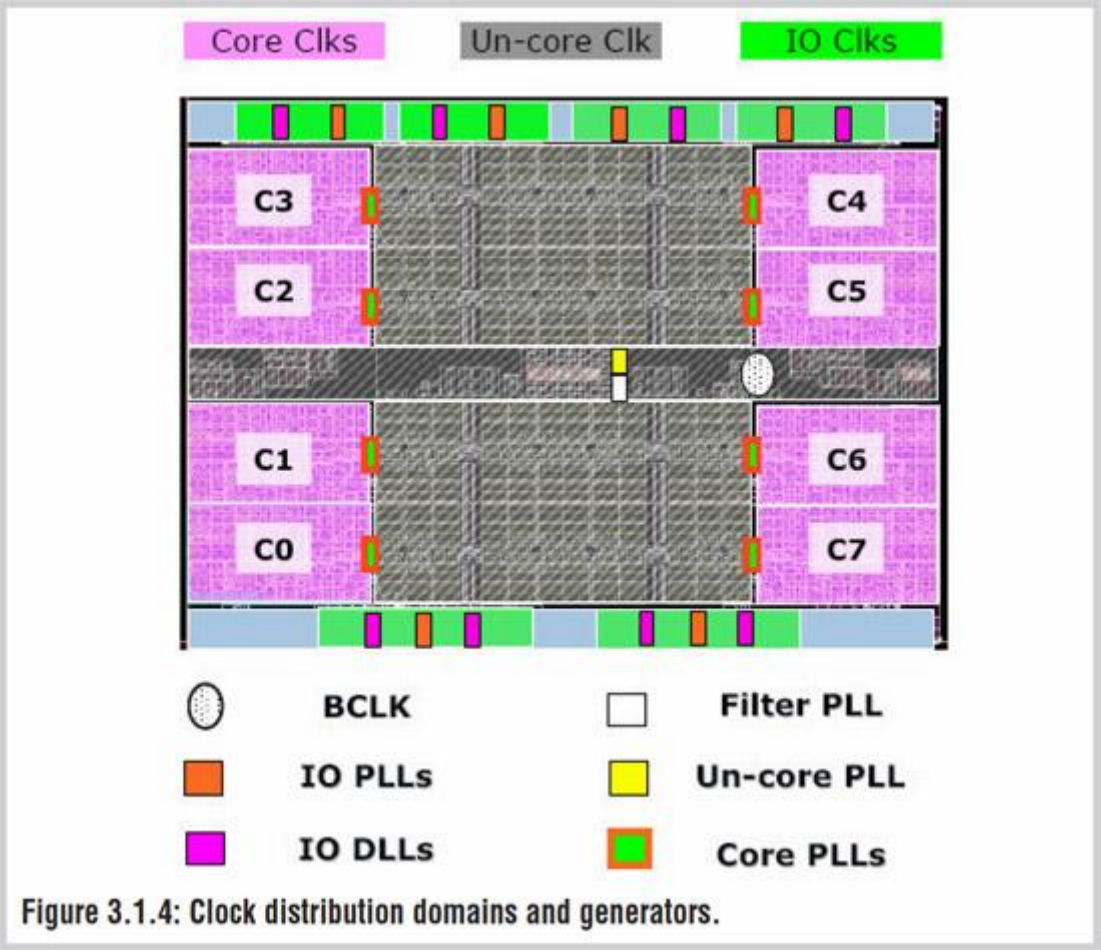
使用高频率 DDR3 内存，访问本地内存的延迟大约为 60 个时钟周期，而通过 QPI 总线访问远端的处理器并返回数据大约需要 90 个时钟周期（如上一页所述）。QPI 的就是 Core 架构为了使用服务器市场而做出的进化，它可以建立一个庞大的可扩展的解决方案。



上：Nehalem-EP，两条 QPI 总线

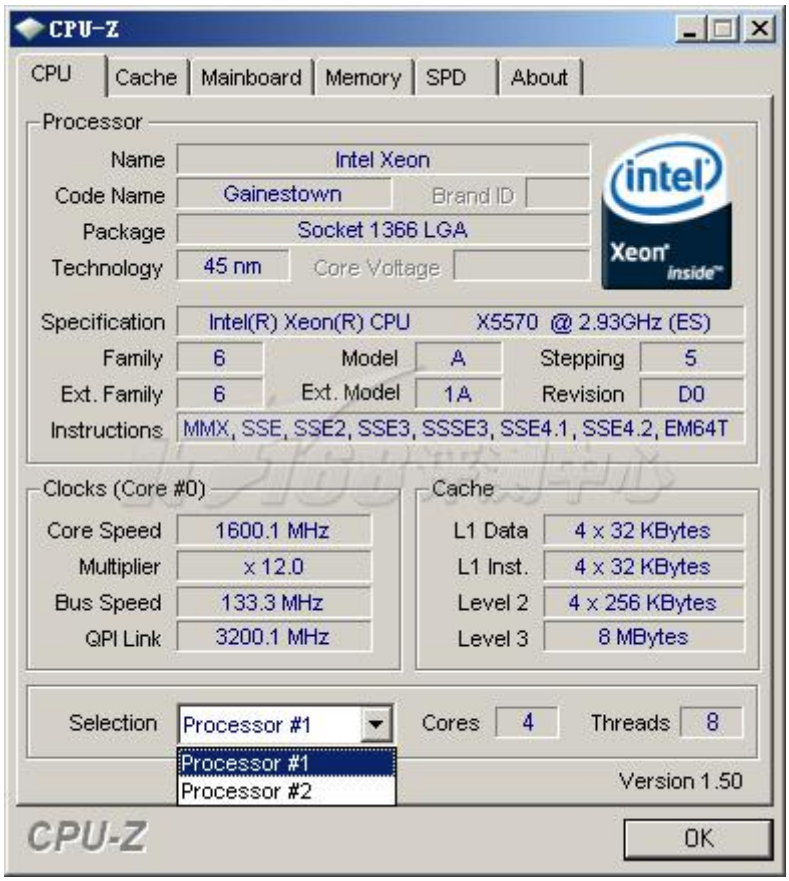
下：Nehalem-EX，四条 QPI 总线

除了提供更高的带宽（每链路 25.6GB 双向带宽）之外，QPI 总线还让多处理器系统更有效率：处理器之间可以直接连接。如上图，每个 CPU 都可以直接和其他三个 CPU 通信。假如放宽些要求，不需要对角线处理器直接相连，那么 Nehalem-EX 还可以直接实现 8 路相连，而不需要加入额外的芯片。



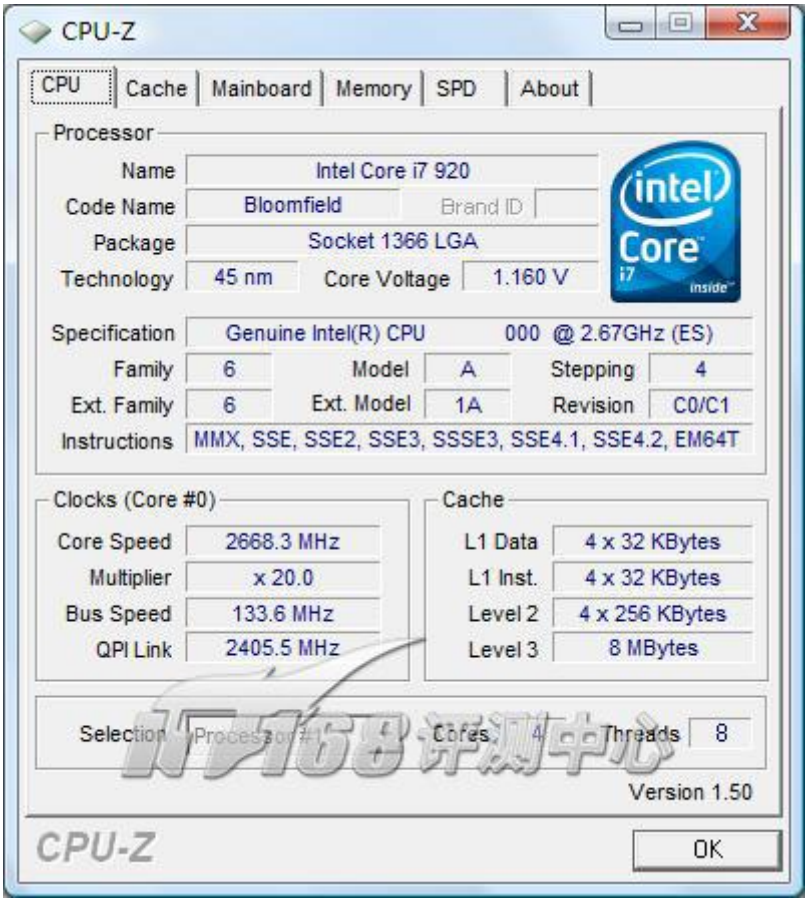
Nehalem-EX 的时钟架构

所有的 Nehalem 都按时钟分为三个部分：核心、核外（L3 和系统逻辑）和 IO（QPI 和 IMC），这三个部分的频率通常互不相同。由于 L3 缓存属于核外部分，因此它的频率和核心频率通常是不同的，在以往，CPU 内的高速缓存通常都是全速的，只有 Pentium II 的 L2 缓存是半速的（它和 CPU 内核不在同一个晶圆上，虽然在同一个 CPU 封装内），而 K6 之前的 L2 缓存都是放在主板上面的，速度极低。现在，Nehalem 架构下，L3 缓存的时钟频率也不再是全速，而是要较低一些，例如，Core i7 920 的 L3 频率应该是 2.133GHz，Xeon X5570 的 L3 频率应该是 2.667GHz。



Nehalem-EP/Gainestown Xeon X5570 处理器，主频 2.93GHz，QPI 总线频率高达 3.2GHz，比主频还要高

QPI 总线频率一般和 L3 频率也不同，不过它们具有一些联系。对于桌面处理器来说，QPI 总线只有一条，简单地连接处理器与 IOH，然而对于服务器处理器来说，除了连接 IOH 之外，处理器与处理器之间也需要通过 QPI 总线，因此服务器的处理器都具有很高的 QPI 频率，有些时候甚至高于处理器主频率，如 Xeon X5570 处理器。



桌面 Nehalem：Core i7 920 的主频是 2.67GHz，而 QPI 总线频率只有 2.4GHz

一些主板允许单独设置这些不同的频率以方便超频。在这里，笔者可以回答很多用户关心的 UCLK 频率（一些主板上具有的 Uncore Clock 设置选项）的问题：L3 缓存频率和 IMC 集成内存控制器的频率是不同的，也就是 UCLK 和内存频率是不同的，不过它们具有一些内在关系。此外，由于 UCLK 关系的 Uncore 部分关系到了整个处理器的中枢部分：系统逻辑（包括中央路由器和集线器），因此它的频率设定可以很大地影响到整个处理器的运行效能。

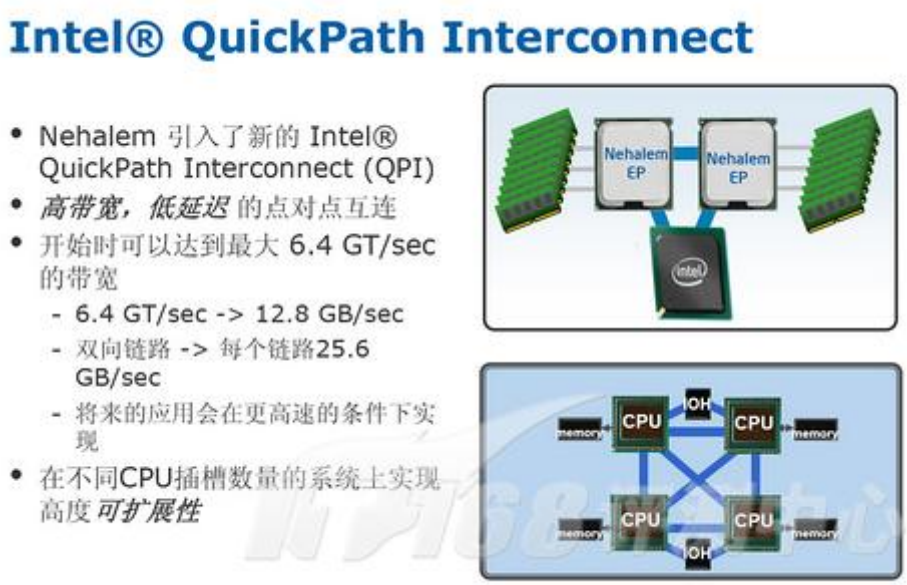
第 15 页：深入 Nehalem 微架构：ccNUMA 架构

Architecture: ccNUMA Architecture

架构：缓存一致的非一致性内存访问架构

由于 IMC 和 QPI 的存在，从形式上来看，多路 Nehalem 处理器系统将会组成一个 NUMA（Non-Uniform Memory Access 或者 Non-Uniform Memory Architecture，非一致性内存访问或非一致性内存架构）系统，NUMA 系统是多处理器系统的一种形式，以往的通过单条 FSB 连接多个 Xeon 处理器的系统叫做 UMA（Uniform Memory Architecture，内存一致架构）系统——传统地，按照处理器架构来分的话属于 SMP（Symmetric MultiProssecor，对称多处理器）系统。NUMA 的特点是访问内存不同的区域具有着不一致的延迟，NUMA 和 UMA 的共同

点是所有内存是硬件共享的，操作系统只看到一片单一的内存区域，比起 MPP 大型并行处理系统在编程方面更为简便。



多个 Nehalem 处理器之间使用 MESIF 协议来保持缓存一致性

按照缓存页面同步的形式，NUMA 可以分为两种：Cache Coherent 缓存一致和 Non-Cache Coherent 缓存非一致性，由于编程上的艰难，因此后一种形式的实际产品几乎不存在，所以 NUMA 几乎就是 ccNUMA（Cache Coherent NUMA）的代名词。多路 Nehalem 处理器就是一个典型的 ccNUMA 架构。ccNUMA 的特点是多个处理器之间进行共享、传输的是缓存页面（缓存页面所对应的内存页面则固定地保留在某一个处理器连接的内存上）。

Nehalem 通过 MESIF 协议来维护缓存页面的一致性（也就是 Cache Coherent 缓存一致的含义），而使用 HT 总线的 AMD Opteron（多 Opteron 也组成一个 ccNUMA 架构）则使用的是 MOESI 协议，老的 Xeon 则使用 MESI 协议。MESIF 的意思就是 M(Modified)E(Exclusive)S(Shared)I(Invalid)F(Forward)，MOESI 则是 M(Modified)O(Owner)E(Exclusive)S(Shared)I(Invalid)，这些词分别代表了一个缓存页面的状态，Nehalem 多了一个 F 状态，而 Opteron 则多了一个 O 状态。

Cache Coherent Protocol 缓存同步协议						
	干净/脏	唯一	可写	转发	可安静地转化成的状态	说明
MESIF over QPI/CSI(Intel Nehalem)						
M(Modified) 修改	Dirty 脏	是	是	是		被请求时需要先写入内存 并转化为 F 状态
E(Exclusive) 独占	Clean 干净	是	是	是	M、S、I、F	被写入时转化为 M 状态
S(Shared) 共享	Clean 干净	否	否	否	I	主副本被写入时转为无效

I(Invalid) 无效	-	-	-	-	-	
F(Forward) 转发	Clean 干净	是	否	是	S、I	主副本 被写入时转换为 M 状态 并使其他 S 副本无效
MOESI over HTT(AMD Opteron)						
M(Modified) 修改	Dirty 脏	是	是	是	O	被请求时不需要写入内存 而仅仅转化为 O 状态
O(Owner) 拥有者	Dirty 脏	是	是	是		主副本 转换为其他状态需要先写入内存
E(Exclusive) 独占	Clean 干净	是	是	是	M、S、I	被写入时转化为 M 状态
S(Shared) 共享	干净或脏	否	否	否	I	可以同时为干净或者脏 主副本被写入时转为无效
I(Invalid) 无效	-	-	-	-	-	
MESI over FSB(Intel Xeon)						
M(Modified) 修改	Dirty 脏	是	是	是		被请求时需先写入内存
E(Exclusive) 独占	Clean 干净	是	是	是	M、S、I	被写入时转化为 M 状态
S(Shared) 共享	Clean 干净	否	否	是	I	可以转发
I(Invalid) 无效	-	-	-	-	-	

三种缓存同步协议对比：Nehalem MESIF、Opteron MOESI、Xeon MESI

MESIF 可以说是 Intel 在多 Xeon 使用的 MESI 协议的扩充，增加了一个 F 状态（同时修改了 S 状态让其无法转发以避免进行过多的传输）。F 状态就是这样一个状态：在一个多处理器之间共享的缓存页面中，只有其中一个处理器的该页面处于 F 状态，另外所有处理器的该页面均处于 S 状态，F 状态负责响应其他没有该页面的处理器的读请求，而 S 状态则不响应并且不允许将缓存页面发给他人（或许 S 用 Silent 来代表更合适）。

当一个新处理器需求读取这个 F 页面时，原有的 F 页面则转为 S 状态，新的处理器获得的页面总是保持为 F 状态。在一群相同的页面中总有并且只有一个页面是处于 F 状态，其他的 S 副本则以 F 副本为中心。这种流动性让传输压力得以分散到各个处理器上，而不是总维持在原始页面上。

不会改变的页面的共享很好处理，关键的是对 Dirty 页面的对待（Dirty 页面是指一个内容被修改了的缓存页面，需要更新到内存里面去），显然，一堆页面的副本中同一时间内

只能有其中一个可以被写入。MESIF 中，只具有一个副本的 E 状态在被写入的时候只需要简单地转化为 M 状态；而 F 状态被写入时则会导致其所有的 S 副本都被置为无效（通过一个广播完成）；S 副本是“沉默”的，不允许转发，也不允许被写入，这些副本所在的处理器要再次使用这个副本时，需要再次向原始 F 副本请求，F 副本现在已经转化为 M 副本，被请求状态下 M 副本会写入内存并重新转化为 F 状态，不被请求时则可以保持在 M 状态，并可以不那么快地写入内存以降低对内存带宽的占用。

MESIF 实际上只允许一堆共享副本当中的中央副本（F 状态）被写入，在多个处理器均需要写入一个缓存页面的时候，会引起“弹跳”现象，F 副本在各个处理器之间不停传输——这有点像令牌环——会降低性能，特别是 F 副本不在其所在的原始内存空间的时候。

Opteron 的 MOSEI 协议不需要被写入的 M 状态写入内存就可以进行共享（这时 M 状态会转变为 O 状态，共享后的 Dirty 副本被标记为 S 状态），这避免了一次写入内存，节约了一些开销，尚不清楚为什么 Intel 没有在新生的总线上采用这种更为优化的协议；当再次写入 O 状态副本时，其他的 S 副本同样会被设置为无效。MOSEI 也只允许一堆共享副本当中的中央副本（O 状态）被写入，也存在着弹跳现象。

不过，在一个方面 Nehalem 具有优势：包含式（或者非独占式）L3 缓存，当一个处理器被请求一个页面，或者被通知一个页面要被设置为无效的时候，它只需要检查 L3 就可以知道该如何操作。在 L3 缓存没有这个页面的时候，不需要像非包含式 L3 设计那样，再检查 L1、L2 页面。

第 16 页：深入 Nehalem 微架构：超线程技术

Architecture: Hyper Threading

架构：超线程

HTT 超线程技术出自 Intel 位于 Oregen 俄勒冈州的 Hillsboro 研发中心。Pentium Pro、Pentium 4、Nehalem 架构都是出自这个 Hillsboro 研发中心。Pentium 4 和 Nehalem 搭载的 HTT 超线程都是同一个东西，都是让处理器可以同时运行多条指令，实际上，它们属于多线程技术中的一个分类：SMT 同步多线程。起先，Intel 在资料中使用 SMT 来称呼 Nehalem 的 HT 技术，然而 SMT 实是一个专有名词，并不仅仅 Nehalem 有采用，于是 Intel 又改变了主意，又将其称作为 HTT 超线程。各种典故可以看这里：**机密揭露：Intel 超线程技术有多少种？**。

Nehalem 的超线程技术就是 NetBurst 超线程技术的升级版本，和 Atom 和 Itanium 的超线程技术都不同

并不是所有的 Nehalem 处理器都提供了超线程技术，在 Nehalem-EP 当中，只有末尾是 0 的型号才具有，是其他数字的就不具备 HTT。如 L5502 是一款双核的、不搭载超线程技术的 Nehalem-EP 处理器，千颗售价$188，非常便宜。当然值不值得又是另外一回事了。

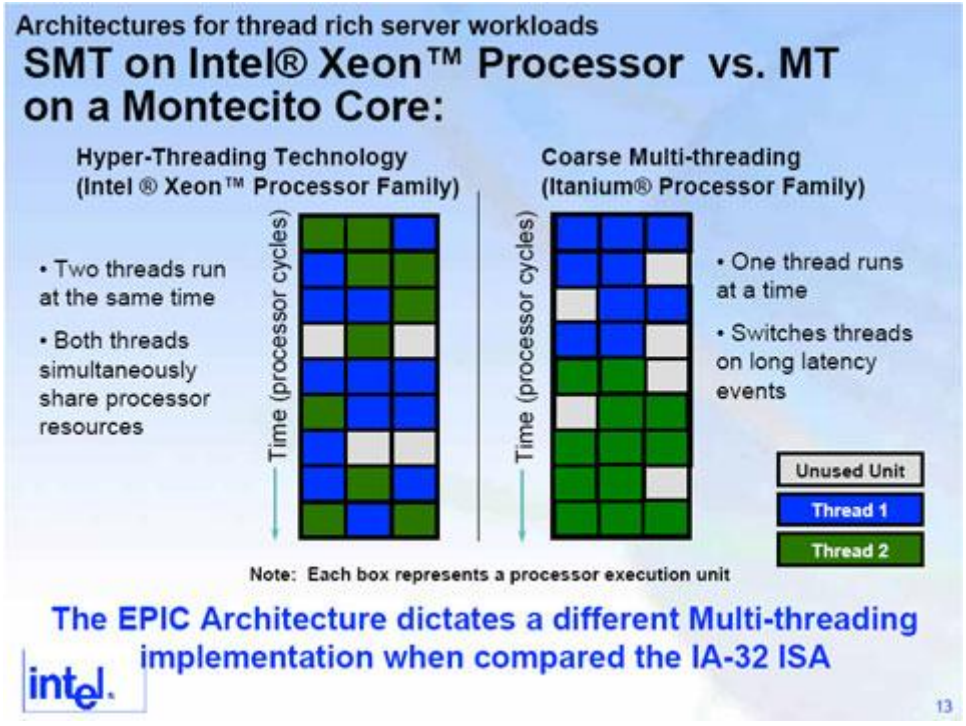
Nehalem 超线程技术的实施细节

超线程技术可以通过很少的代价提升并行应用的性能，特别是在服务器领域，因此 Nehalem 在服务器领域的能力将会再一次得到提升。AMD 目前并没有类似的技术，因此在未来的对阵当中，Nehalem 更被看好些。

SMT 属于 MTT 的一种，下面是 MTT——MultiThreading 多线程技术的主要分类，MultiThreading 多线程就是在一个单个的处理核心内同时运行多个工作线程的技术，和 CMP（Chip MultiProcessing，芯片多处理）不同，后者是通过集成多个处理内核的方式来让系统的处理能力提升——也就是现在常见的多核技术。现在主流的处理器都使用了 CMP 技术。主流的MultiThreading具有着三种形式，差别在于线程间共享的资源以及线程切换的机制：

多线程架构异同			
多线程技术	线程间共享资源	线程切换机制	资源利用率
粗粒度多线程 Coarse-Grained MultiThreading	除取指令缓冲、寄存器、控制逻辑外	流水线停顿时	提升单个执行单元利用率
细粒度多线程 Fine-Grained MultiThreading	除寄存器、控制逻辑外	每时钟周期	提升单个执行单元利用率
同步多线程 SimultaneousMultiThreading	除取指令缓冲、返回地址堆栈、寄存器、控制逻辑、重排序缓冲、Store 队列外	所有线程同时活动，无切换	提升多个执行单元利用率

其中 CMT 和 FMT 都是在单个执行单元下的技术，不同的线程在指令级别上并不是真正的“并行”，而 SMT 则具有多个执行单元，同一时间内可以同时执行多个指令，因此前两者有时先归类为 TMT（Temporal MultiThreading，时间多线程），以和 SMT 相区分。



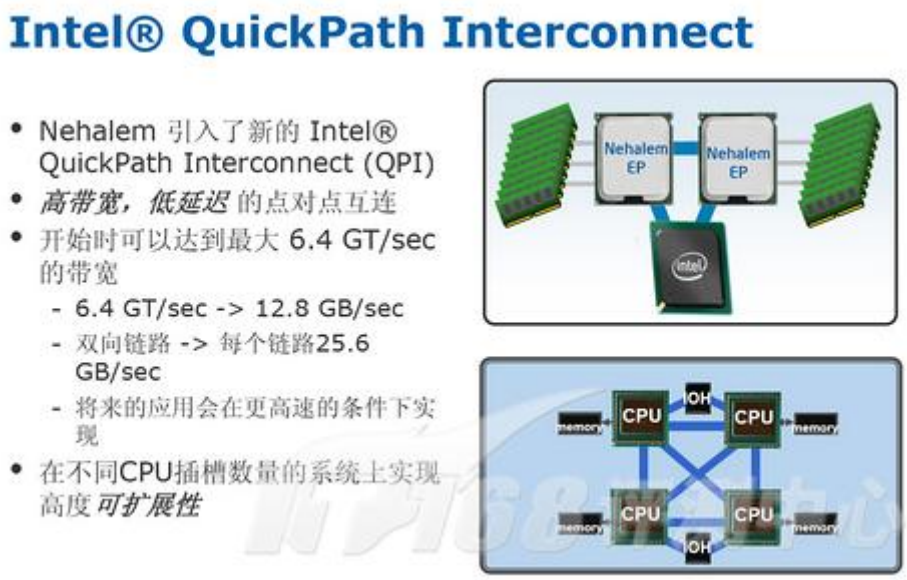
Itanium 2 Montecito 也具有超线程技术，不过，和 Pentium 4/Nehalem 不同，它属于 CMT 粗粒度多线程技术

第 17 页：深入 Nehalem 微架构：ccNUMA、SMT 与 OS

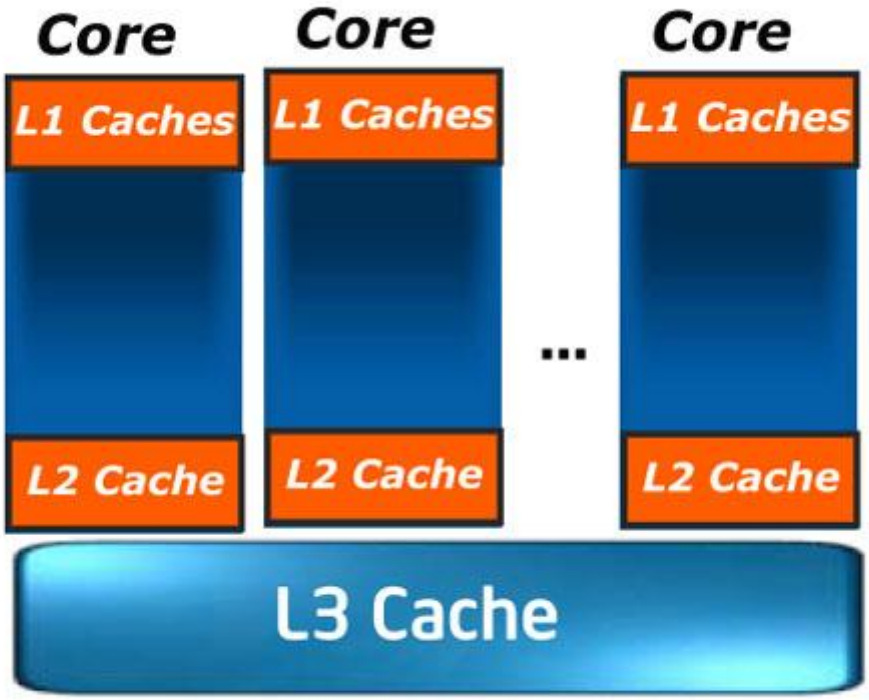
Nehalem: ccNUMA & SMT & OS

Nehalem：ccNUMA 与 SMT 与操作系统

我们已经知道多路 Nehalem 会形成一个 ccNUMA 架构，在 NUMA 系统中，由于本地内存的访存延迟低于远程内存的访存延迟，因此将进程分配到本地内存附近的处理器上可极大优化应用程序的性能。这就需要操作系统支持并智能地进行这样的分配。



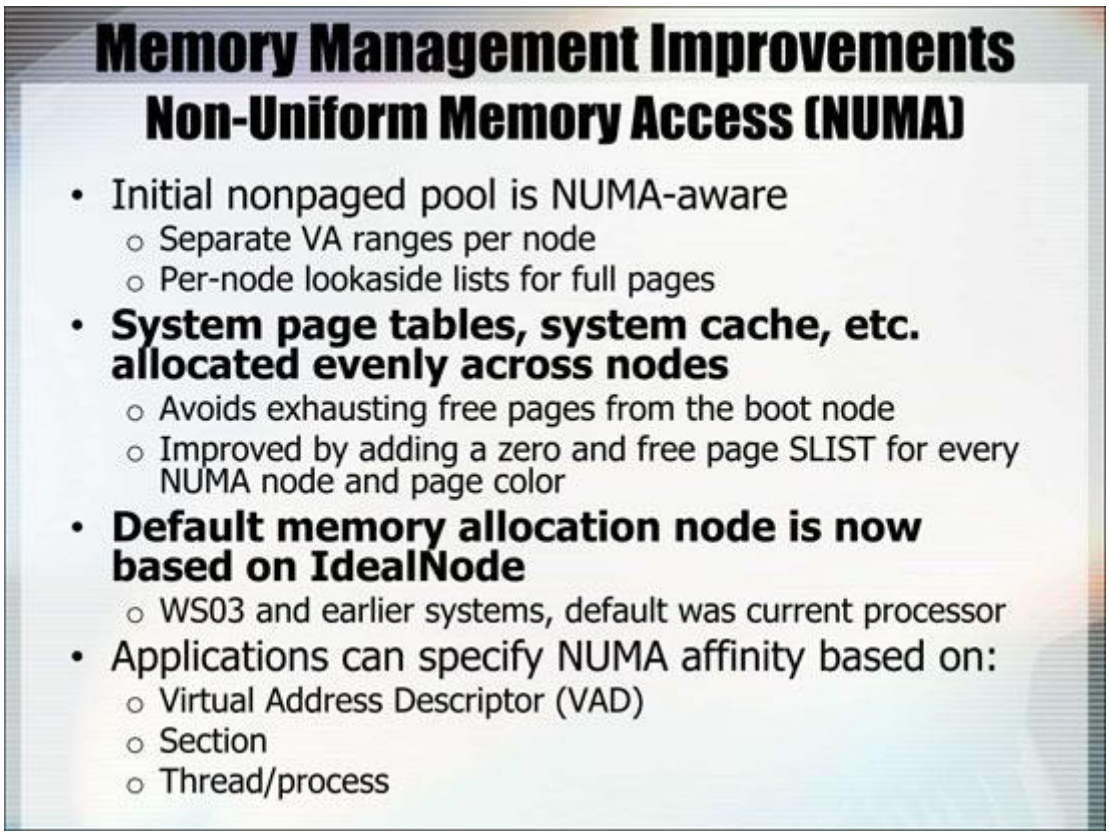
多个 Nehalem 处理器之间使用 MESIF 协议来保持缓存一致性



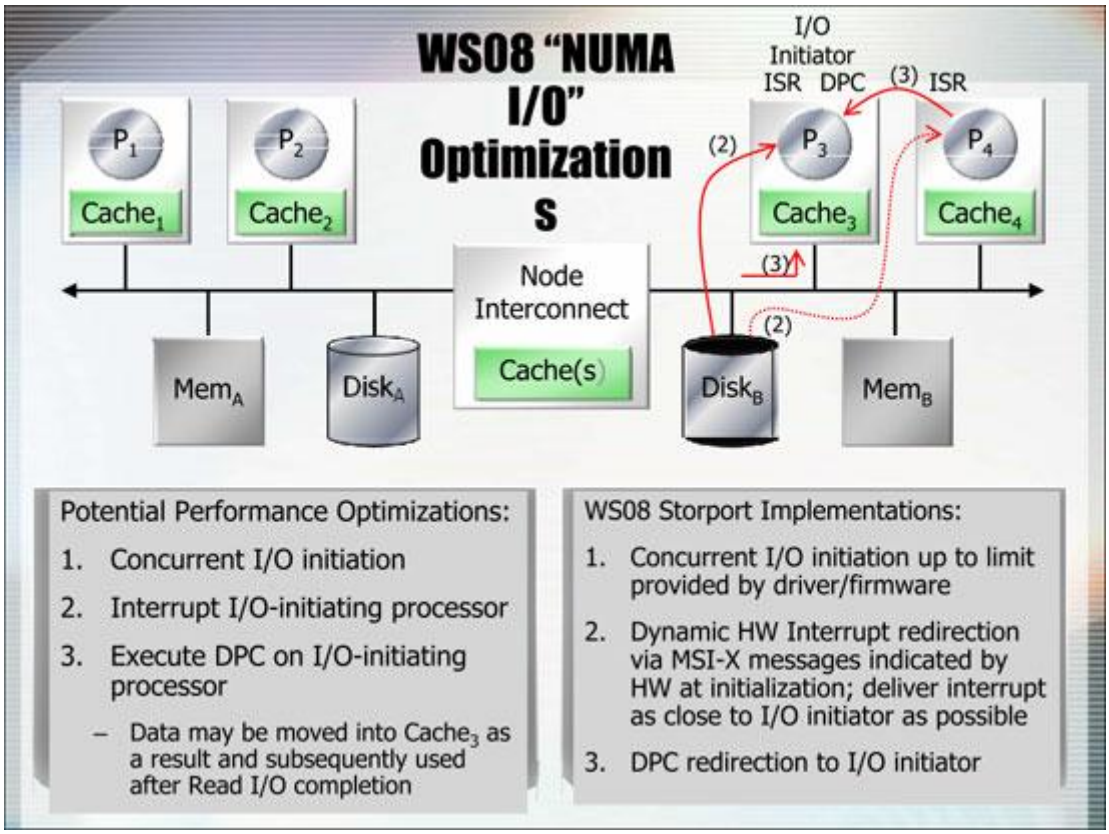
多个核心之间是否也使用 MESIF 协议来保持缓存一致性呢？

除了 NUMA 架构的要求外，Nehalem 的 SMT 技术（超线程技术）也要求操作系统的支持，这是基于这样的一个事实：线程调度时在两个逻辑 CPU 之间进行线程迁移的开销远远小于物理 CPU 之间的迁移开销以及逻辑 CPU 共享 Cache 等资源的特性。这一点和 NUMA 上同一个 CPU 的不同核心之间进行线程迁移的开销远远小于多个 CPU 之间的迁移开销以及同核心的 CPU 共享 Cache 等资源的特性是一样的，要系统发挥最大的性能，操作系统必须对 NUMA 以及超线程这样的实质上比较类似 NUMA 的这些架构作出优化。

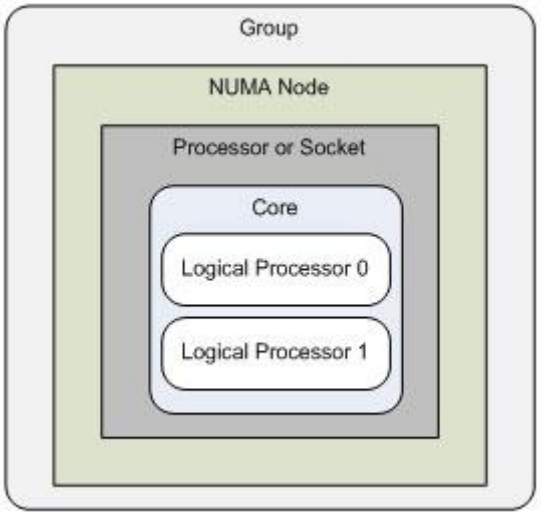
传统的基于 NT 核心的 Windows 都可以支持 SMP 对称多处理器技术，然而它们并没有很好地为 NUMA 和超线程优化（这也是当初 Pentium 4 HT 推荐使用 Windows XP 而不是 Windows 2000 操作系统的原因），在购买到 Nehalem 系统之后，你需要采用最新的操作系统：



Windows Server 2008 内核对 NUMA 的优化



Windows Server 2008 内核对 NUMA IO 的优化



Windows Server 2008 对逻辑处理器们的划分（Group——Processor Group 是 Windows Server 2008 R2 / Windows 7 加入的功能）

经过多次升级的 Windows Server 2003 可以较好地支持 NUMA 技术（为了支持广泛应用的 Opteron——典型的 NUMA 架构），Windows XP 也为超线程技术做了优化，然而它们都不够 Windows Server 2008 深入。2008 为 NUMA 做出了包括内存管理方面的多种优化：分布式的非分页池、系统页表、系统缓存以及内存分配策略，同时还更好地支持 NUMA I/O。在使用多 Nehalem 或者多 Opteron 这样的处理器时，你应该使用 Windows Server 2008 操作系统或者 Windows Vista 操作系统(2008 和 Vista 使用了相同的内核,区别只是一些小的特性)。甚至在使用单 Nehalem 的时候，你也应该使用 Vista，因为超线程的缘故。

Linux 2.4 内核中的调度器由于只设计了一个运行队列，可扩展性较差，在 SMP 平台表现一直不理想。后来在 2.5 内核开发时出现一个多队列调度器（Ingo Molnar），称为 O(1)，每个处理器具有一个运行队列，从 2.5.2 开始集成。由于 O(1)调度器不能较好地感知 NUMA 系统中结点这层结构,从而不能保证在调度后该进程仍运行在同一个结点上,为此 Linux 2.6 内核出现了结点亲和的 NUMA 调度器（Eirch Focht），建立在 Ingo Molnar 的 O(1)调度器基础上的（这个调度器后来也向后移植到 2.4.X 内核中），因此现在的 Linux 2.6 核心可以较好地支持 NUMA 和超线程。

FreeBSD 的 SMP 功能直到 7.0 版本才算大为完善，就目前来看，FreeBSD 对 NUMA 的支持还比较原始。

第 18 页：深入 Nehalem 微架构：虚拟化

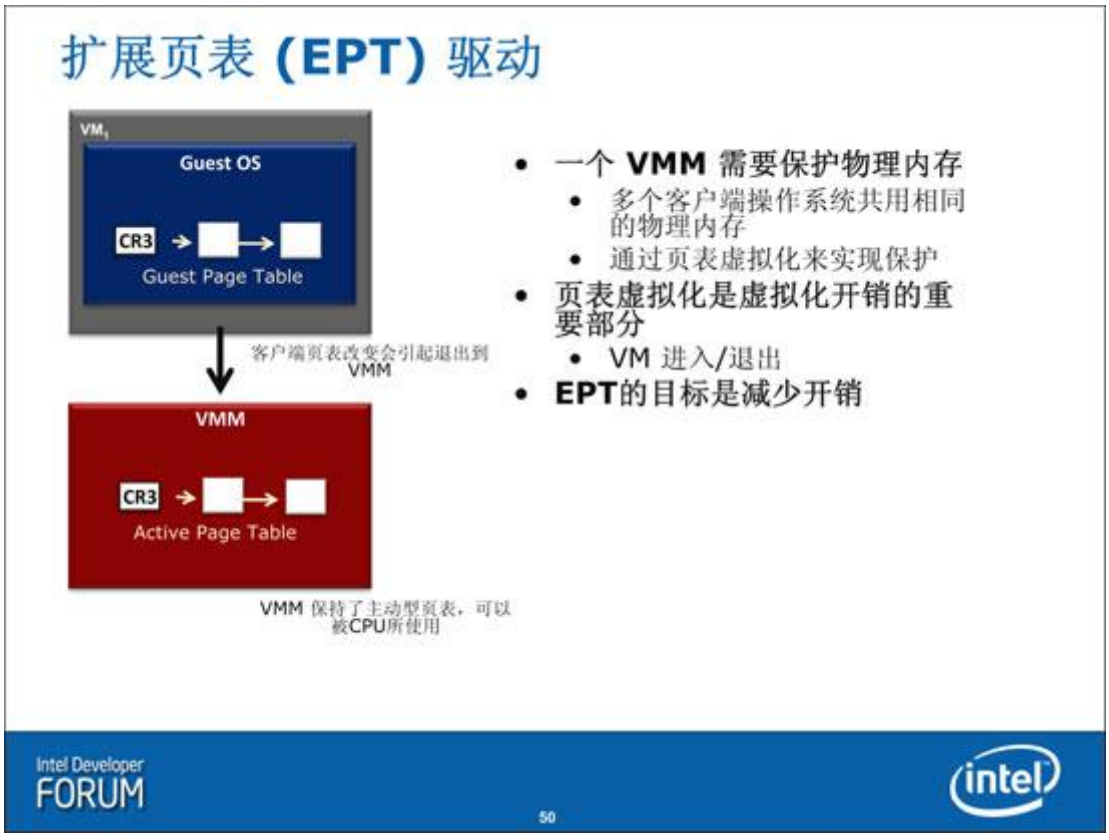
Nehalem: Virtualization

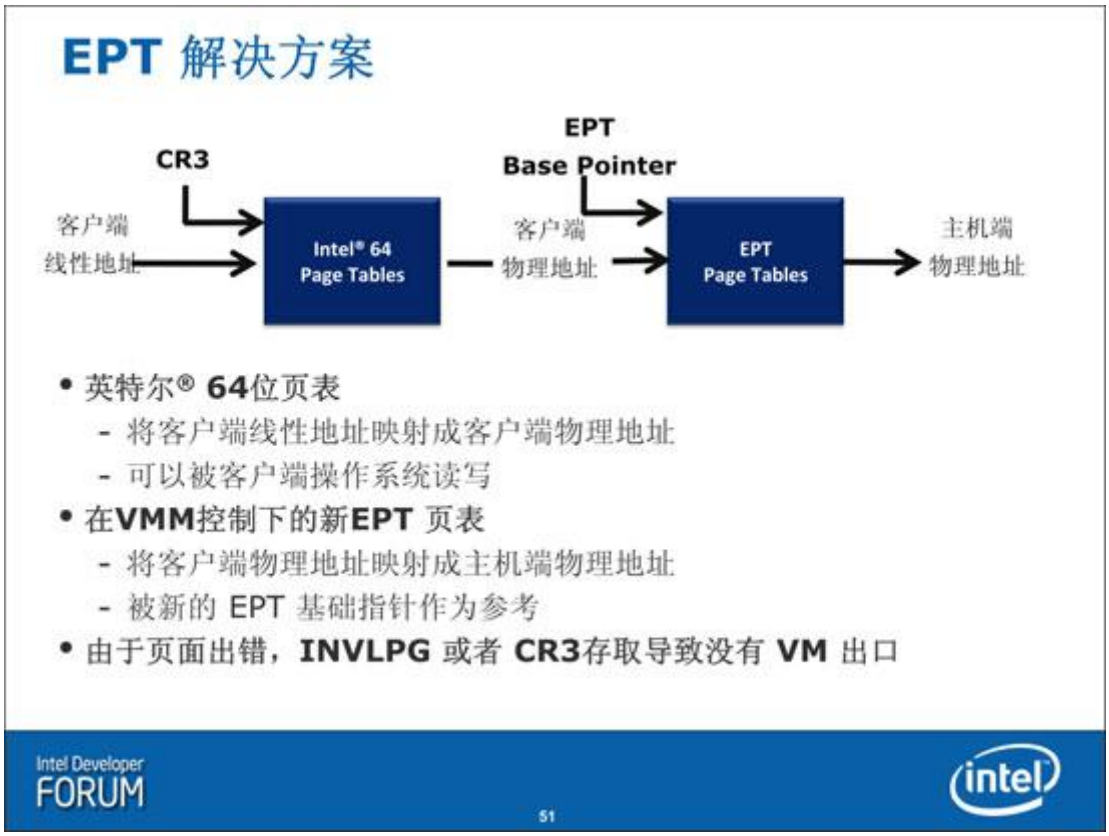
Nehalem：虚拟化

虚拟化作为 Intel 架构的重点，一直是 Intel 处理器的重要特性，每次处理器架构的更新，都会得到更多的支持。Nehalem 也不例外，改进的地方虽然不多，然而这些改动大大提高了虚拟化性能。这些改动包括了两个部分：EPT 扩展页表和 VPID 虚拟处理器 ID，其中前者消灭了当前存在的虚拟机内存操作中存在的大量内存地址转换（以前使用软件来模拟 EPT

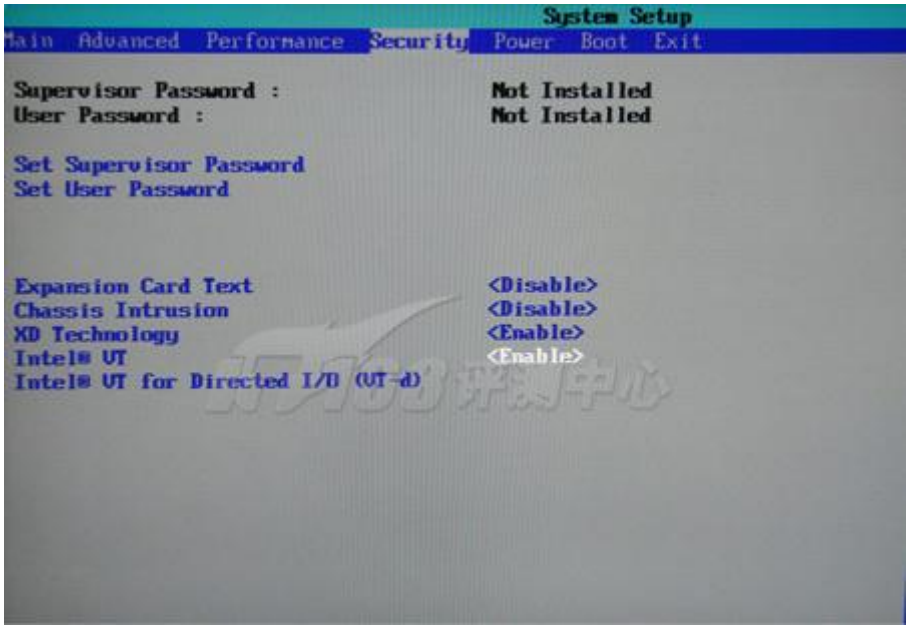
的功能，现在用硬件实现了，据说虚拟化延迟比 Penryn 降低了 33%），后者则减少了对 TLB 的无效操作，这些都明显提升了虚拟机的性能。

EPT 扩展页表和 VPID 虚拟处理器 ID

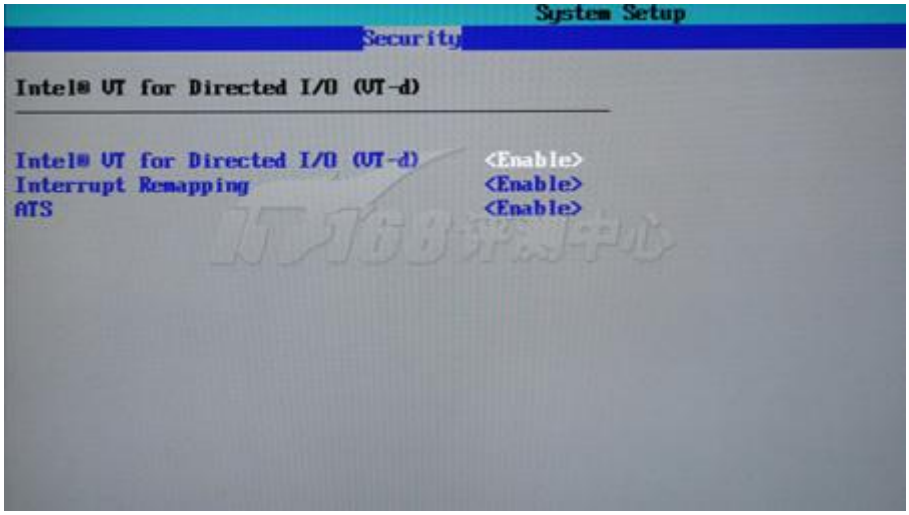




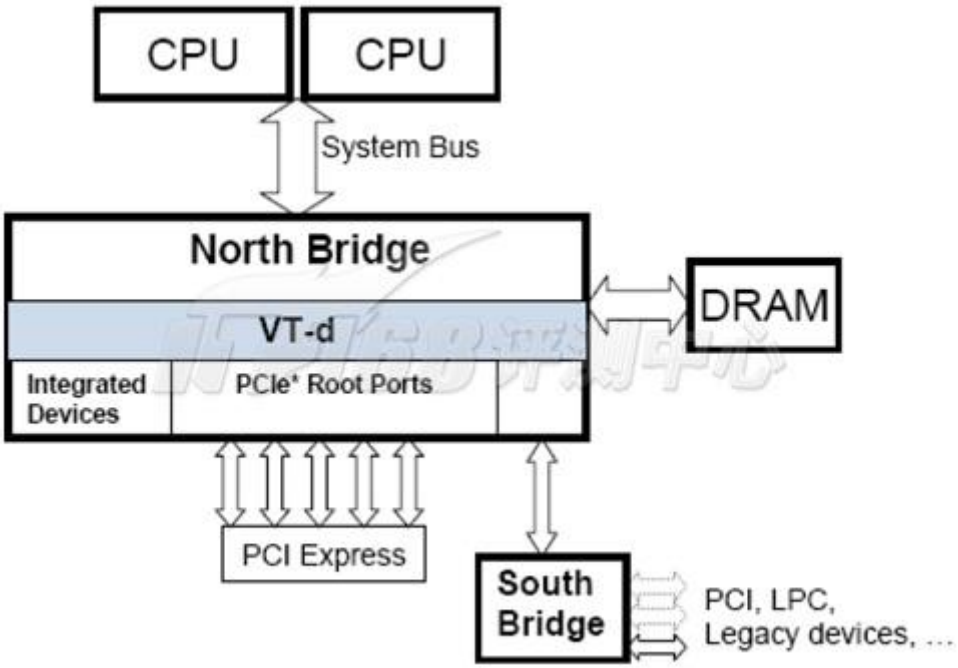
I/O 虚拟化的关键在于解决 I/O 设备与虚拟机数据交换的问题，而这部分主要相关的是 DMA 直接内存存取，以及 IRQ 中断请求，只要解决好这两个方面的隔离、保护以及性能问题，就是成功的 I/O 虚拟化。在以前，Intel 提供的设备虚拟化技术（VT-d，VT 是 Virtualization Technology 虚拟化技术，d 是 device 设备的意思）多出现在服务器芯片组上，现在随着 Nehalem 的出现，VT-d 技术也开始流入桌面 / 移动市场（Core i7 主板上已经可以见到 VT-d 功能）。



Core i7 主板：Intel X58SO 主板 - VT-d 设置界面



Core i7 主板：Intel X58SO 主板 - VT-d 设置界面



以往 VT-d 技术集成在北桥 MCH 内，和内存控制器的关系非浅

Intel 的虚拟化平台包含了三个部分，除了 EPT/VPID 属于的 VT-x 虚拟化之外，还有关键的 I/O 虚拟化 VT-d，用于解决 I/O 设备与虚拟机数据交换的问题，而这部分主要相关的是 DMA 直接内存存取，以及 IRQ 中断请求。在以前，Intel 提供的设备虚拟化技术（VT-d，VT 是 Virtualization Technology 虚拟化技术，d 是 device 设备的意思）是集成在 MCH 芯片上面的，现在 Nehalem 集成了内存控制器，因此其部分功能也就相应地进驻处理器当中——剩下一部分则仍然留在了新的 Tylersburg 芯片组当中，并且得到了进一步的提升。

Intel 82576EB 千兆网络芯片，支持 VMDq，支持 VT-c

Intel 虚拟化平台策略的第三个部分是连接虚拟化 VT-c（c 是 connetive 的意思），在 Nehalem-EP + Tylersburg 平台上，这一点也得到了体现。关于 VT-c 技术，将另有专文介绍（计划中）。

最后，Nehalem-EP 的超线程也是和虚拟化紧密相关的部分：多了一倍的逻辑处理器，可以支持更多的虚拟客户机数，而且，硬件实现的逻辑处理器，要比虚拟机软件虚拟出来的效果要好的多了。

第 19 页：深入 Nehalem 微架构：SSE4.2 指令集

Nehalem: SSE4.2 Instruction Set

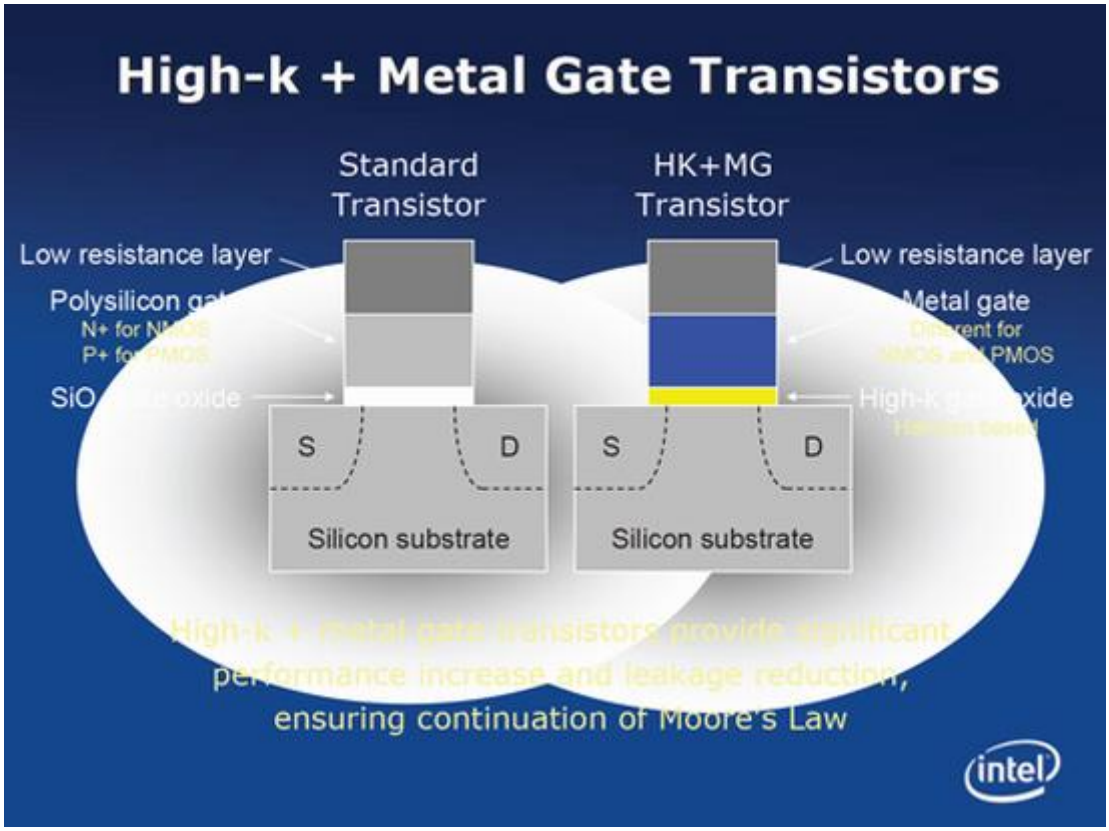
Nehalem：SSE4.2 指令集

由于现在的 x86 CPU 都可以通过解码器将 x86 指令解码为 uops 再执行，而这个解码阶段是可以通过 Microcode ROM 控制的，添加新的解码方法很容易，因此现在 Intel 处理器的指令集扩充显得有些任意：指令数目不停地增多。当然，从编程的角度来看，使用预先给出的硬件指令到底是会比自己进行繁琐的程序设计要方便得多，而性能也获多或少总有些提升：

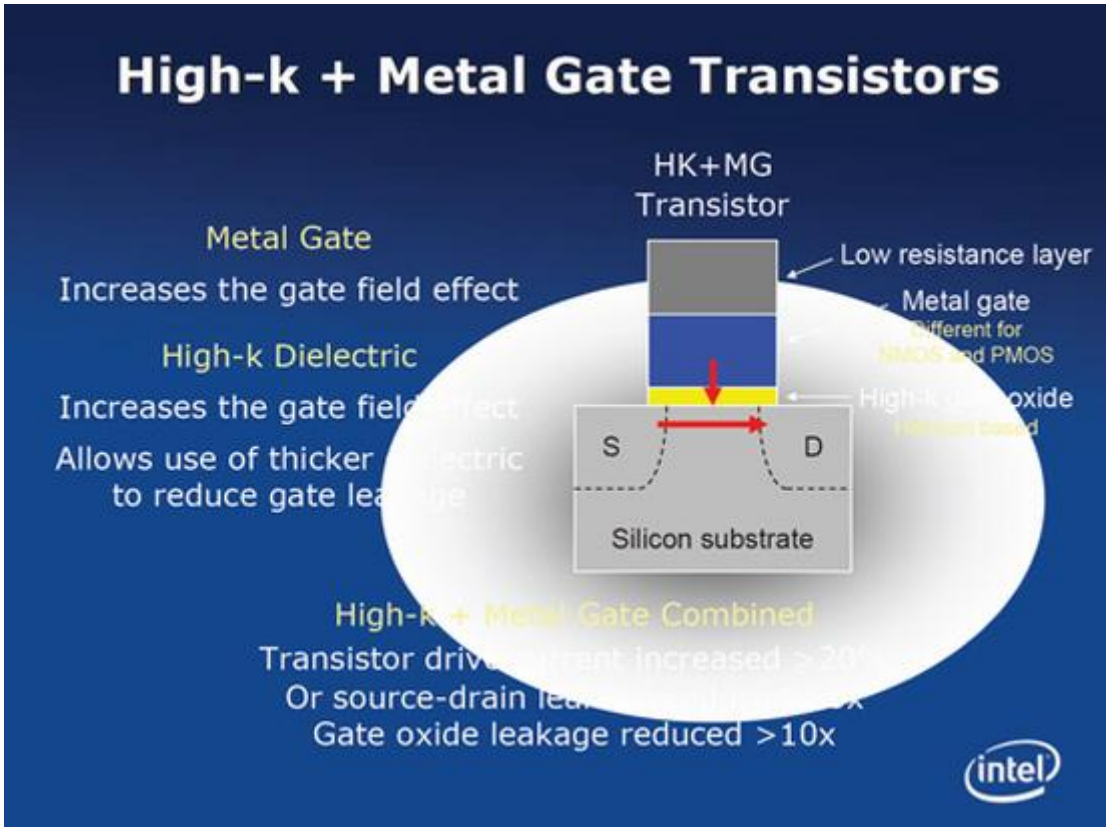
Nehalem 附带了 SSE4.2 指令集，共 7 条指令

SSE4 指令集是自 SSE 以来最大的一次指令集扩展，它实际上分成了三个阶段来更新：提前发布的SSSE3、Penryn 中出现的 SSE4.1 和 Nehalem 中出现的 SSE4.2，其中成熟的 Penryn 中集成的 SSE4.1 占据了大部分的指令，因此 Nehalem 中的 SSE4 指令集更新很少。要发挥新指令集的功能，需要在程序设计方面的支持，Intel 自己的编译工具自然有所提供——从 10.0 版本开始。

第 20 页：深入 Nehalem 微架构：制造工艺

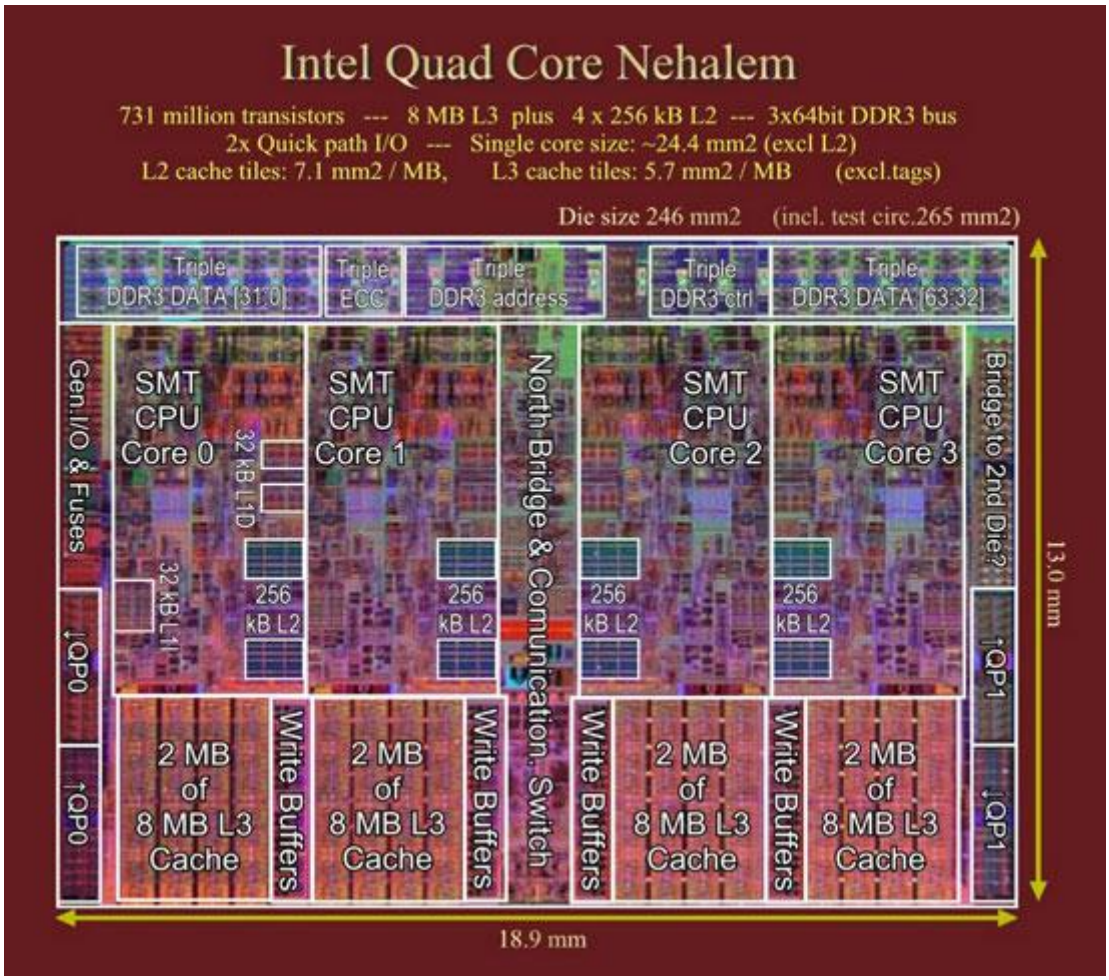


Intel High-k Metal Gate 晶体管



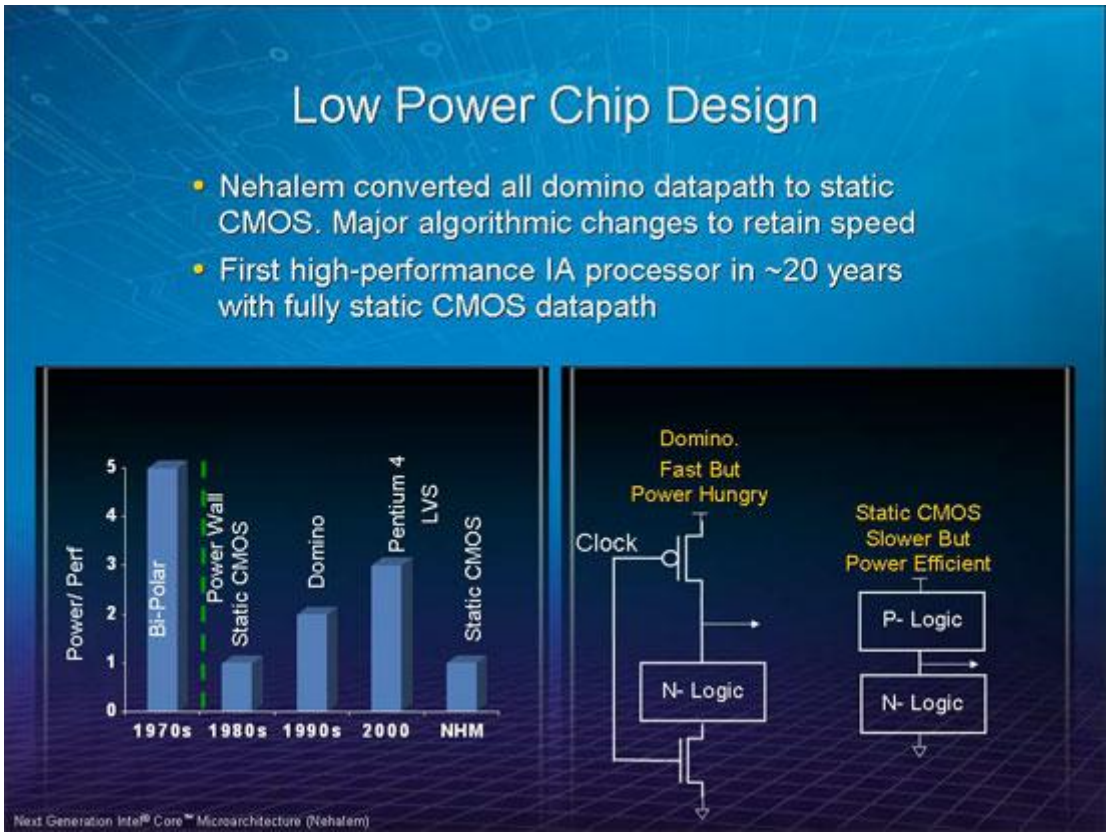
Intel High-k Metal Gate 晶体管

和 Penryn 一样，Nehalem 的生产工艺，都是 45nm CMOS 工艺，采用了金属栅极 High-K 电介质晶体管以及 9 层铜互联技术，总晶体管数量则为 0.781 Billion——7.81 亿，比 Bloomfield 要多一些，因为 Gainestown 要比 Bloomfield 多了一个 QPI 总线。



Nehalem-EP 晶圆下部的左边是 QPI0，右边是 QPI1

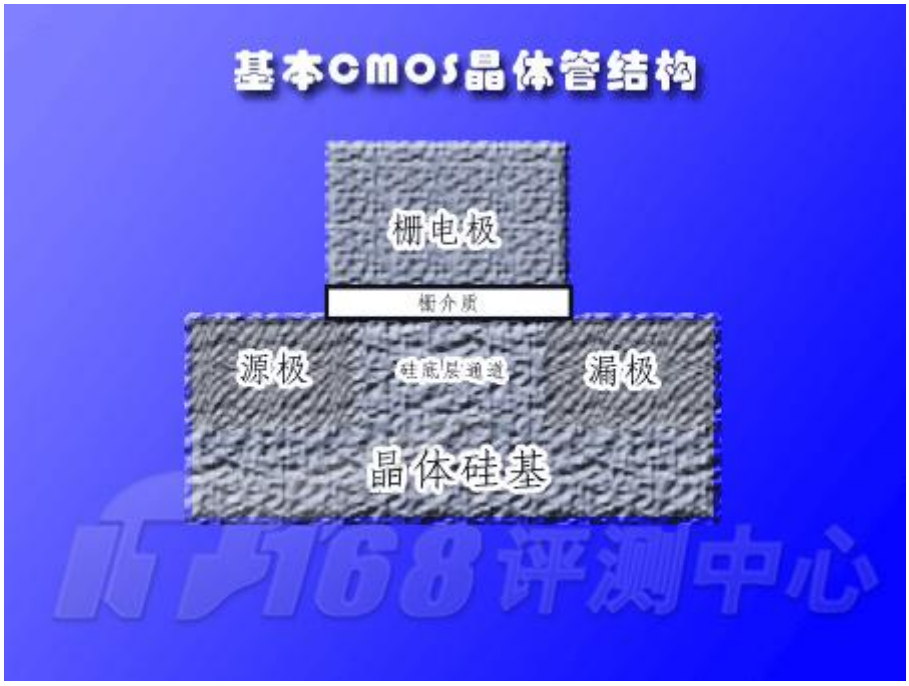
在《2008 年度评测报告：深入 Nehalem 微架构》中，笔者简单地提到了为了降低功耗，Nehalem 将以前使用的 Domino 线路更换为了 Static CMOS 线路：



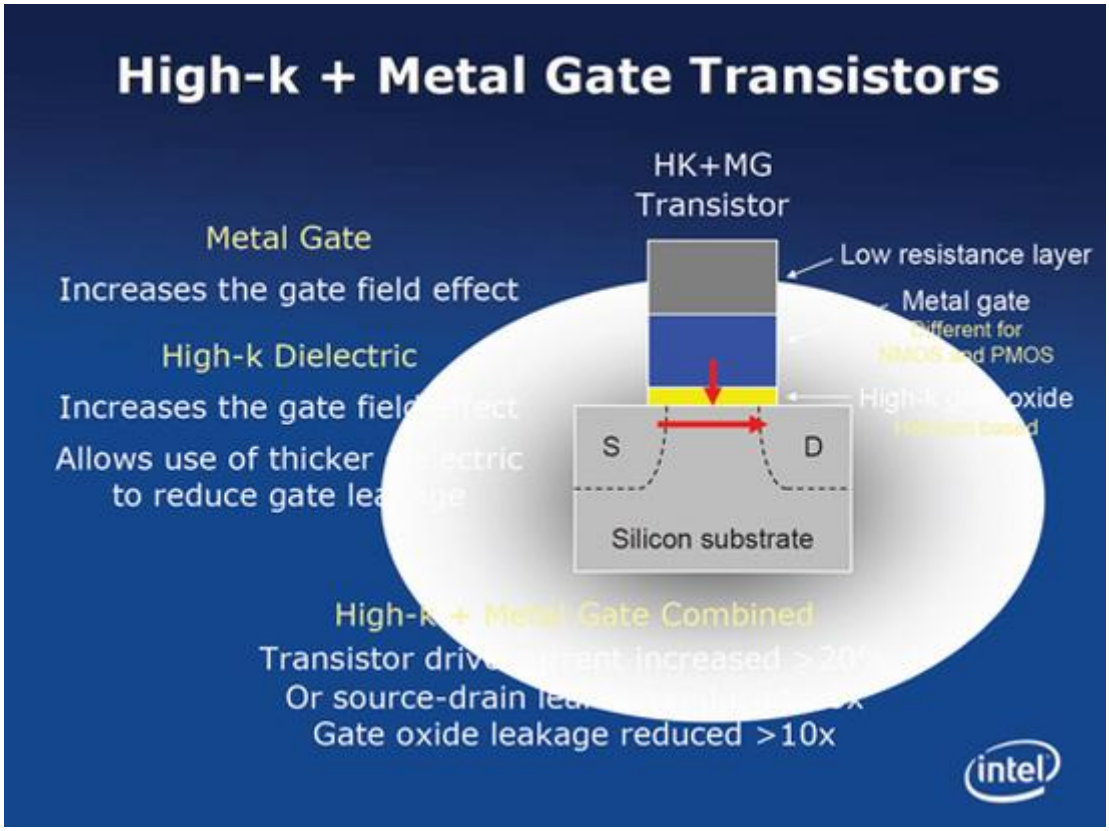
为了降低能耗，Nehalem 架构将以往应用的 Domino 线路更换为 Static CMOS 线路，速度有所降低，但是能源效率提升了

第 21 页：深入 Nehalem 微架构：长沟道晶体管技术

除了线路类型的变更之外，Nehalem 的晶体管也进行了变化：

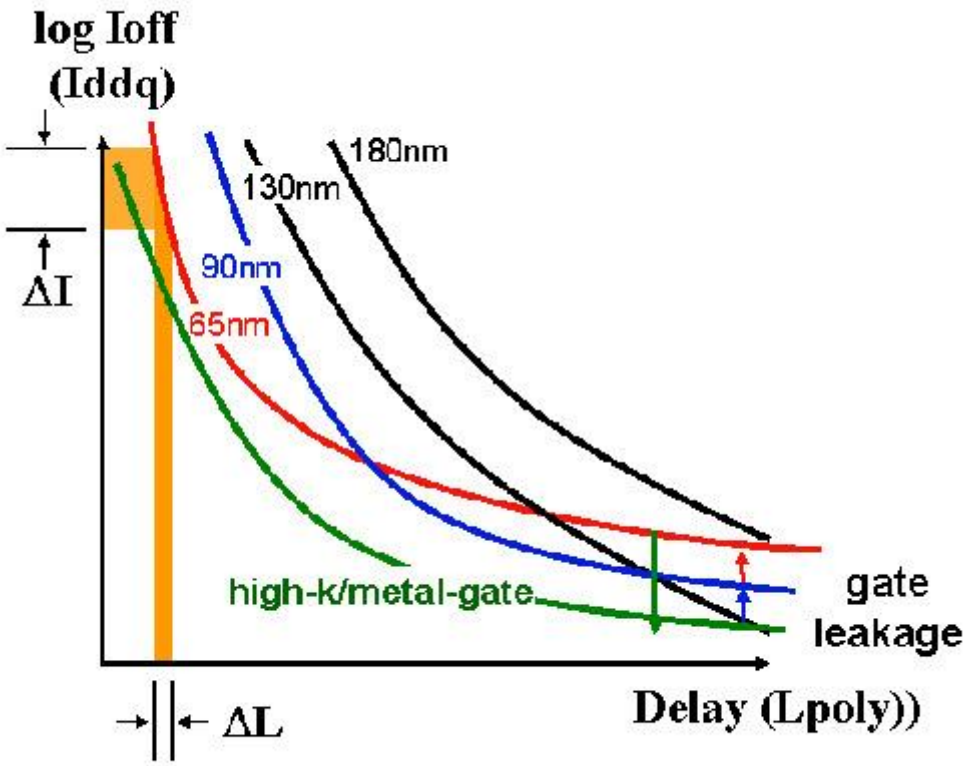


基本 CMOS MOSFET 晶体管结构，channel 沟道存在于图上的“通道”区域



Intel High-k Metal Gate 晶体管，S 极到 D 极的红色箭头就是“channel 沟道”，也就是耗尽区所在

在同一个线路中，使用的晶体管不同，耗电也是不同的，MOSFET 元件按沟道长度可以分为长沟道 Long Channel 和短沟道 Short Channel，短沟道具有较好的性能，不过其漏电流也相应更大（耗尽区宽度不足而与源极合并而导致大量漏电电流）。

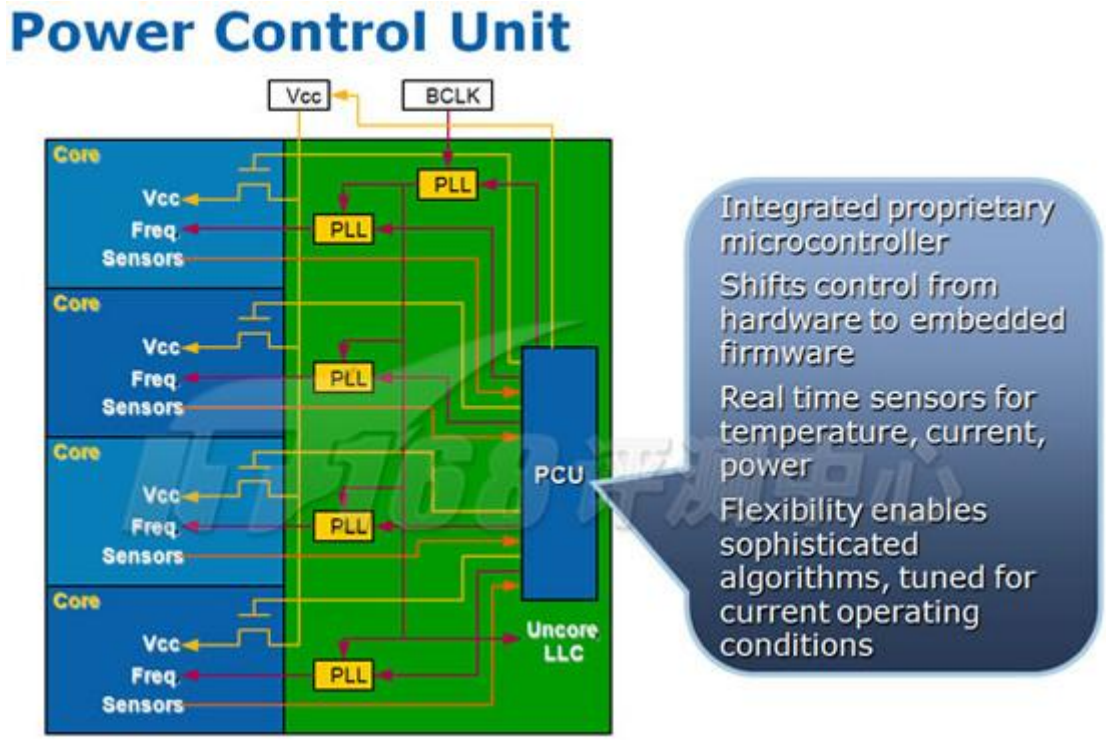


这个图可不容易明白：沟道长度与漏电的关系，请自行理解（越低的延迟，越高的漏电电流）

在 IC 设计当中通常需要根据不同的情况使用不同沟道长度的晶体管，对于 Nehalem 而言，非时序关键（non-timing-critical）的线路可以使用性能略差的长沟道 MOSFET 晶体管以减少亚阈值漏电（subthreshold leakage，MOSFET 的 subthreshold 亚阈值特性被广泛利用在低电压线路上），实际上 Intel 用的是"longer-channel"——“更长沟道”的 MOSFET。Nehalem 核心部分的 58%和核外部分（不包括缓存阵列）的 85%都使用了更长沟道晶体管，最后，漏电功率被控制到总功耗的 16%。代价是 Nehalem 的 L1-D 延迟由上一代的 3 时钟周期上升到 4 时钟周期。

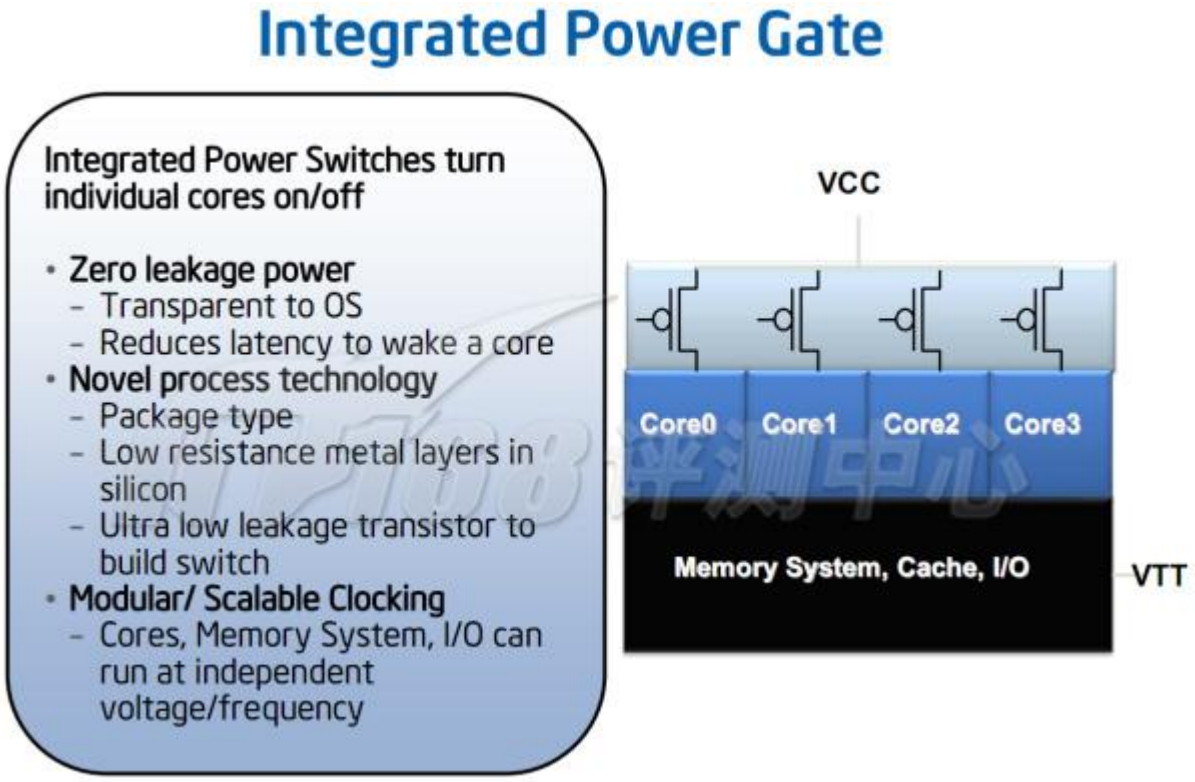
第 22 页：深入 Nehalem 微架构：能耗比控制

在 Nehalem 处理器当中，除了大规模使用长沟道晶体管技术来降低总漏电之外，还搭载了一个新的单元，来管理所有的核心的工作状态，包括电压、频率等，这个单元的名字就叫作 Power Control Unit 电源管理单元。它也负责处理器参数的实时监测。空闲的核心和缓存将会被降低供应电压，并降低工作频率，以达到降低功耗、节约能源的目的。



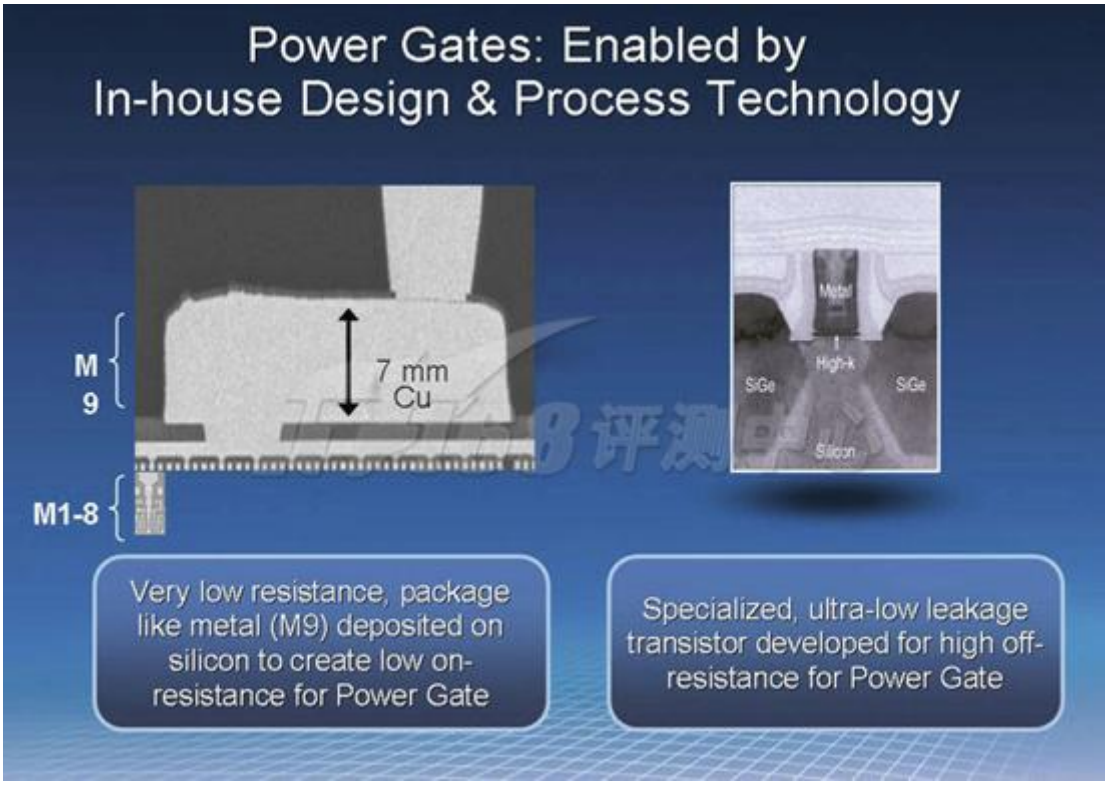
PCU 是 Nehalem 处理器的电源总管

在需要的情况下，空闲的核心和缓存可以设置为关闭模式以降低耗电。彻底避免这些线路用电是不太可能的，在关闭模式下，SRAM 的供电将从 0.90V 降低到 0.36V，提供 83%的漏电功耗节约，作为比较，睡眠电压是 0.75V，节约为 35%。关闭模式是由 Power Gate 电源阀来实现的：



Power Gate 是 PCU 的实际执行者之一

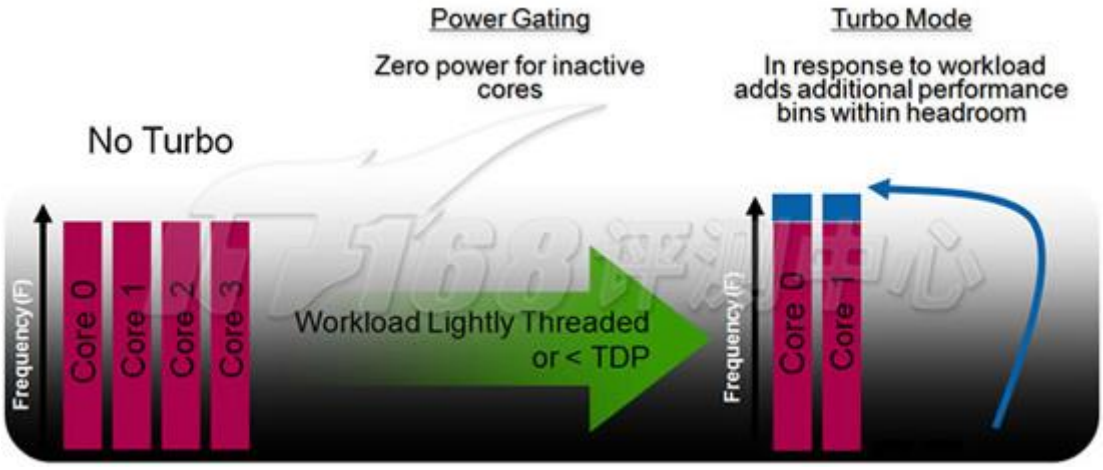
为了实现 PCU，Nehalem 使用了特别的工艺，在第 9 金属层上实现了非常低导通电阻和非常高关闭电阻及极低晶体管漏电的 Power Gate 电路。



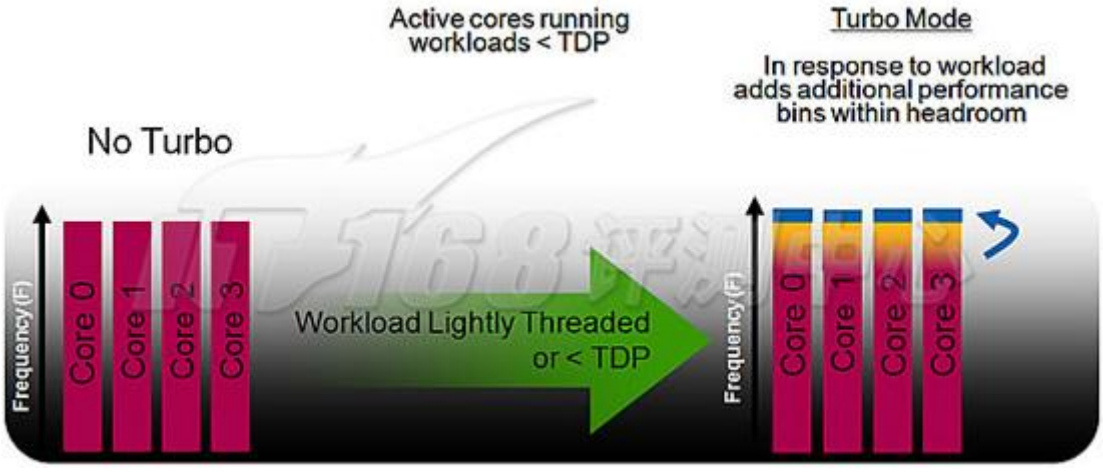
第 23 页：深入 Nehalem 微架构：能耗比控制

通过 Power Gate，在 Nehalem 上还实现了了一种新的能耗比控制技术：Turbo Mode，或者叫做 Turbo Boosting，这种技术在笔记本上曾经出现过。作用就是当一些核心处于空闲状态，被 Power Gate 关闭之后，剩余的核心可以动态提升频率以提升负载的响应能力。

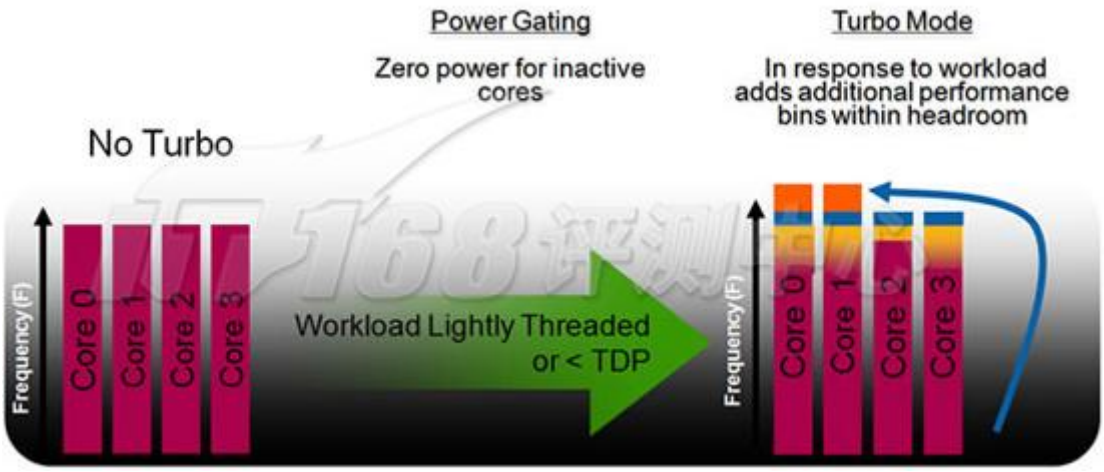
通常状态



两个核心被 Turbo Boost



TDP 允许的情况下，所有的核心都被 Turbo Boost



TDP 允许的情况下，部分核心允许更进一步地 Turbo Boost

并不是所有的处理器都具有 Turbo Mode 功能，在 Xeon 5500 系列处理器当中，只有最后一位数字为 0 的处理器具备超线程技术和 Turbo Mode，为其他数字的则没有。一开始，桌面版本的 Nehalem 也是没有超线程技术和 Turbo Mode 的，后来 Intel 改变了主意，这个举动应该是为了刺激市场，通过培养消费者来扩展它们的应用领域。

第 24 页：小结：Nehalem 架构的优势

从前面可以看出，Nehalem 架构比以往 Intel 处理器具有了较大的变迁，这个变迁带来了非常直接的性能提升，总结起来，Nehalem-EP/Gainestown 比 Penryn/Harperton 具备的主要优势有三点：

直联架构带来了 IMC 和 QPI

IMC：CISC 的 x86 架构对缓存/内存带宽极度渴求，集成内存控制器让处理器避开了访问内存需要通过 FSB 总线的限制，并将带宽提升到三通道 DDR3 1333（8 核心 Nehalem-EX

支持四通道 DDR3）每处理器，极大提升了 Nehalem 处理器的内存带宽，对服务器应用提升巨大。

QPI：新的点对点总线带宽更高，并且让处理器之间可以直接连接，避免了共享的 FSB 总线在处理器核心过多时的效率急剧下降，更适合扩展到大规模并行系统。同样处理器数量下，QPI 点对点形成的 ccNUMA 拓扑比共享 FSB 的星型总线具有更高的效率。

虽然 SMT 有不少处理器采用，不过，在 x86 处理器上只有 Intel 具有

HTT：超线程技术在打游戏的时候或许看不出有作用，不过在企业级别应用上效果明显。在主要竞争对手也有 IMC 和类似 QPI 的情况下，HTT 就成为了 Nehalem 的特别武器。这项据说耗资十亿开发费用的技术终于从 Nehalem 开始大放光芒。

第 25 页：Nehalem-EP：处理器规格对照表

要了解一款处理器，可以先看它的规格表。在 Nehalem-EP 新 Xeon 5500 处理器首度曝光中我们已经有了一个简单的表格介绍 Nehalem-EP/Gainestown 处理器的规格，不过这个规格表不是非常完善，而且只有 Nehalem-EP 部分的数据，因此我们整理了以下表格，包括了 Core i7/Bloomfield、Xeon Harptown 和 Nehalem-EP/Gainestown 的完整处理器资料：

Intel Core i7/Bloomfield 规格表				
名称	Core i7 920	Core i7 940	Core i7 Extreme 965	Core i7 Extreme 975
系列	Nehalem/Core i7		Nehalem/Core i7 Extreme	
多处理器数量	1			
频率	2.66GHz	2.93GHz	3.20GHz	3.33GHz
QPI 速率	4.80GT/s		6.40GT/s	
Turbo Boost	○			

HTT(SMT)	○
核心/线程	4/8
L2 缓存	4 x 256KB
L3 缓存	8MB
TDP	130W

普通的 Core i7 和 Core i7 Extreme 的区别就在于主频，以及 QPI 总线规格。

Intel Xeon/Harpertown规格表

名称	Xeon L5410	Xeon L5420	Xeon L5430	Xeon E5405	Xeon E5410	Xeon E5420	Xeon E5430	Xeon E5440	Xeon E5450	Xeon X5450	Xeon X5460	Xeon X5470	Xeon E5462	Xeon E5472	Xeon X5472	Xeon X5482	Xeon X5492
系列	Harpertown Low Voltage			Harpertown													
每系统数	2																
频率	2.33GHz	2.50GHz	2.66GHz	2.00GHz	2.33GHz	2.50GHz	2.66GHz	2.83GHz	3.00GHz	3.00GHz	3.16GHz	3.33GHz	2.80GHz	3.20GHz	3.20GHz	3.33GHz	3.40GHz
FSB速率	1333MHz									1600MHz							
核心/线程	4/4																
L2缓存	2x6MB																
TDP	50W			80W						120W			80W		120W	150W	

Penryn/Harpertown Xeon 规格对照表

45nm Harpertown Xeon 的型号众多，可以按照 FSB 分为 1333MHz(5400)和 1600MHz(5402)，或者分为低电压版和普通版。不同型号的差别只是在于主频、FSB 总线和 TDP。

Intel Nehalem-EP/Gainestown规格表

名称	Xeon W3520	Xeon W3540	Xeon W3570	Xeon L5506	Xeon L5520	Xeon E5502	Xeon E5504	Xeon E5506	Xeon E5520	Xeon E5530	Xeon E5540	Xeon X5550	Xeon X5560	Xeon X5570	Xeon W5580
系列	Nehalem-WS 1S/Bloomfield			Nehalem-EP/Gainestown											Nehalem-WS 2S/Gainestown
每系统数	1			2											
频率	2.66GHz	2.93GHz	3.20GHz	2.13GHz	2.26GHz	1.86GHz	2.00GHz	2.13GHz	2.26GHz	2.40GHz	2.53GHz	2.66GHz	2.80GHz	2.93GHz	3.20GHz
支持内存频率	1066MHz		1333MHz	800MHz	1066MHz	800MHz			1066MHz			1333MHz			
QPI速率	4.80GT/s		6.40GT/s	4.80GT/s	5.86GT/s	4.80GT/s			5.86GT/s			6.40GT/s			
Turbo Boost	○			×	○	×			○						
HTT (SMT)	○			×	○	×			○						
核心/线程	4/8			4/4	4/8	2/2	4/4		4/8						
L2缓存	4 x 256KB					2 x 256KB	4 x 256KB								
L3缓存	8MB			4MB	8MB	4MB			8MB						
TDP	130W			60W		80W						95W			130W
千颗售价	$284	$562	$999	$423	$530	$188	$224	$266	$373	$530	$744	$958	$1172	$1386	$1600

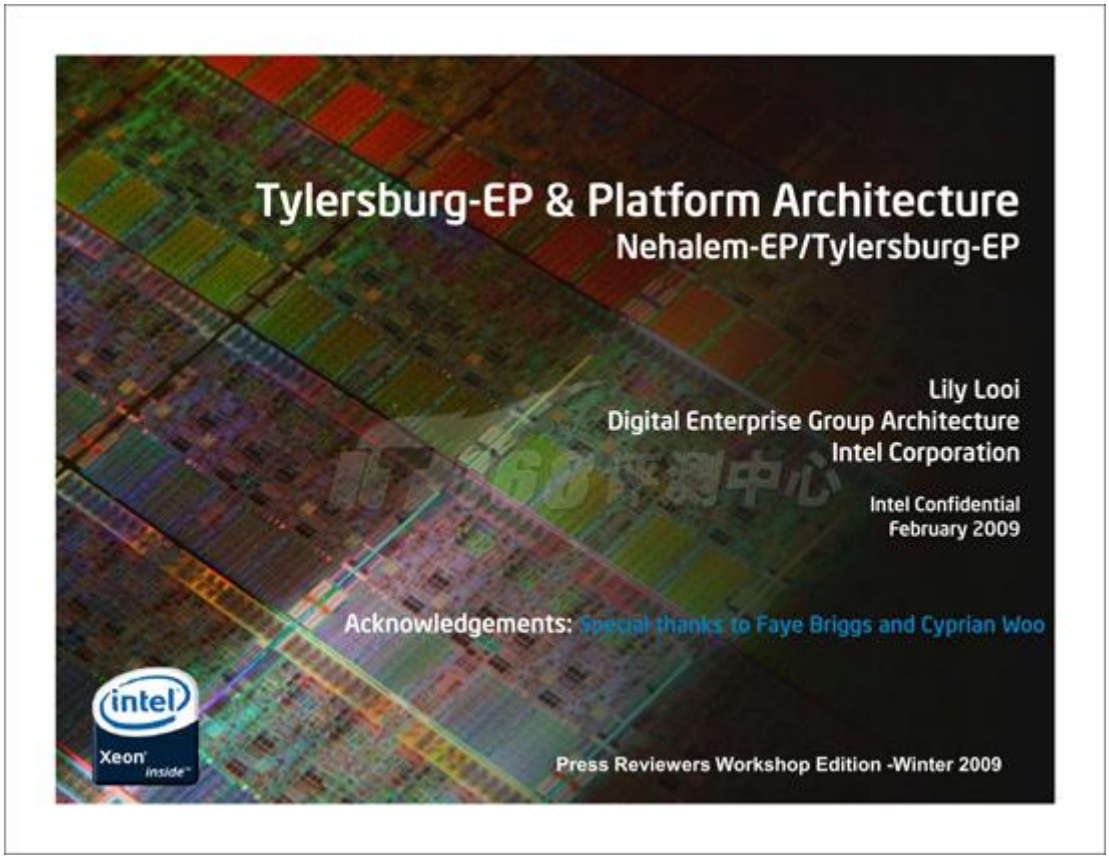
Nehalem-EP/Nehalem-WS Xeon 规格对照表

上表除包括了 Nehalem-EP/Gainestown 之外，还包括了 Nehalem-WS——这一系列 CPU 部分属于 Bloomfield（Nehalem-WS 1S 系列），部分属于 Gainestown（Nehalem-WS 2S 系列，只有一款型号：Xeon W5580）不过是面向 Workstation 市场，它们和 Nehalem-EP 的区别就是它们大部分只支持一路处理器系统（也就是 Nehalem-WS 1S 系列；支持二路系统的是

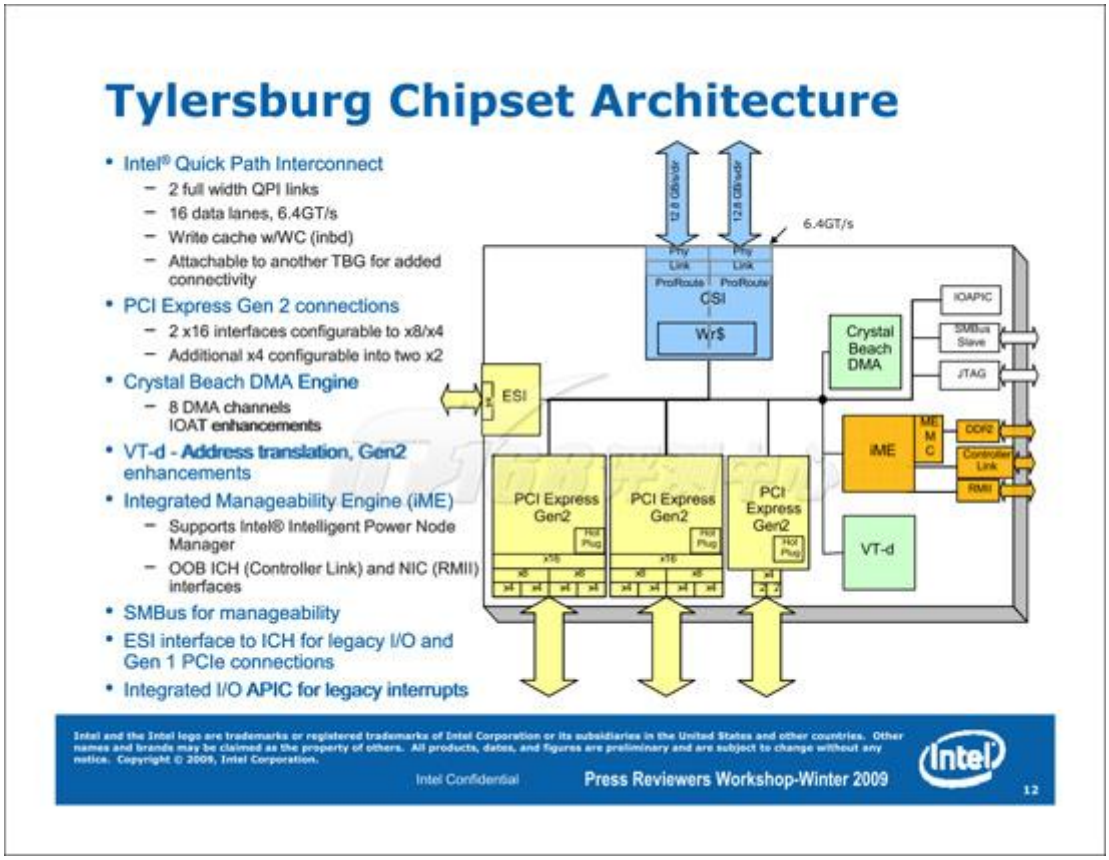
Nehalem-WS 2S 系列并只有一款处理器：W5580）。不同型号的差别在于主频、QPI 总线（有三种）、L3 容量（有两种）和 TDP（有四种）。Nehalem-EP 也提供了两款低电压版型号。Nehalem-EP 还提供了一款双核的型号，此外并不是所有的 Nehalem-EP 都搭载了 HTT 超线程技术（同时和 Turbo Boost 技术）。

第 26 页：Nehalem 座驾：Tylersburg 芯片组结构

为了搭载全新的Nehalem处理器,需要同样是全新的芯片组,这个芯片组需要QPI总线，并且不需要内存控制器，这个芯片组就是 Tylersburg。



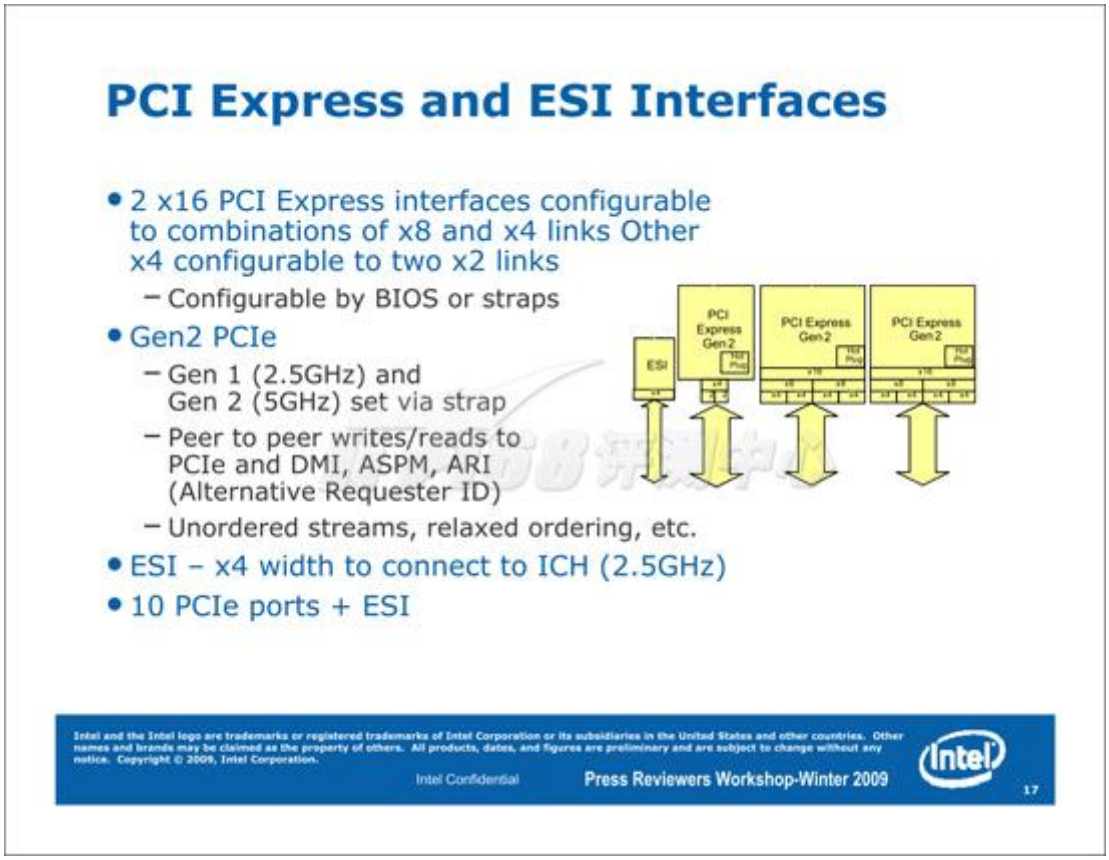
如同 Nehalem 处理器有普通版本和 Nehalem-EP 版本一样，Tylersburg 芯片组也有普通版本和 Tylersburg-EP 版本，普通版本的 Tylersburg 我们都已经很熟悉，就是搭配 Core i7 处理器用的 X58 芯片组。



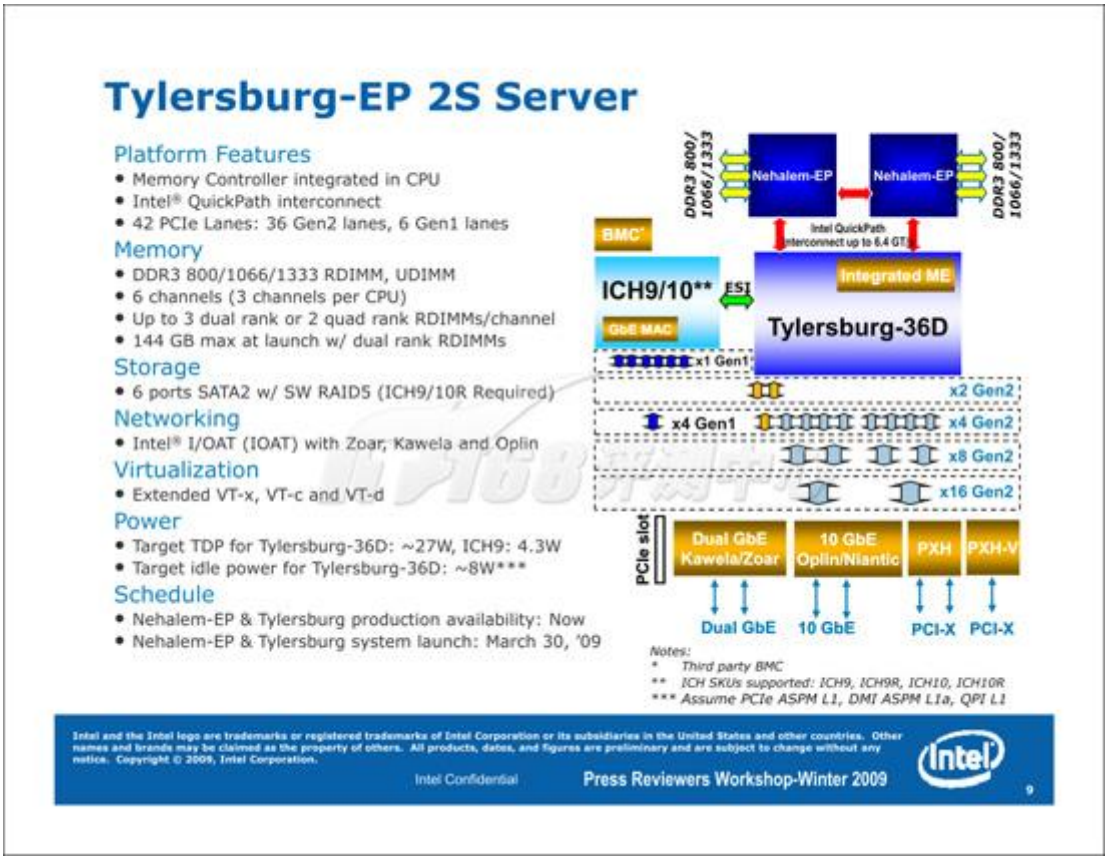
由于 Nehalem 处理器已经集成了内存控制器，因此主板芯片组上就没有必要再有，因此北桥芯片组的名称也就不能再叫 MCH（Memory Controller Hub），现在的 Tylersburg 叫做（IOH，I/O Hub）。

第 27 页：Nehalem 座驾：Tylersburg 芯片组 PCI Express

作为一个 IOH，IO 自然是其目的，Tylersburg 的 IO 主要针对三个方面：CPU、PCIE 设备和 ICH 南桥，这三种设备的连接分别由 QPI、PCI Express、ESI 来完成。其中 PCI Express 支持是 Tylersburg 最重要的部分。

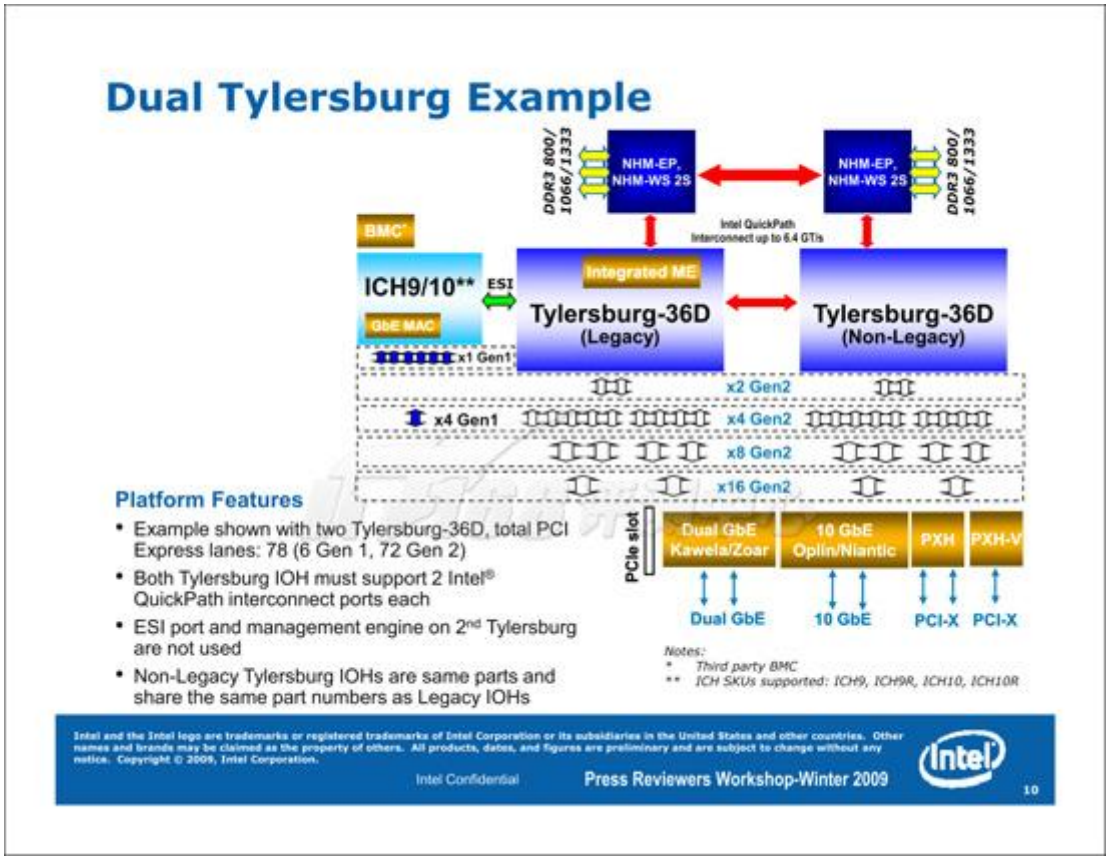


Tylersburg-EP IOH最多可以提供2个x16规格的PCI Express总线(Tylersburg-36D)，并且可以分割为多个细小的连接，如分解为4个x8，或者最多分解为8个x4。除了两个这两个可以用来连接显卡的x16界面之外，Tylersburg-EP还可以额外提供一个x4界面用来在连接两块x16显卡之后连接其它如阵列卡这样的设备，这个额外的端口可以分割为两个x2界面。因此，Tylersburg-EP最多具有10个PCI Express端口，并且这些端口都属于第二代（PCI Express Gen 2，或者2.0），每信道带宽达到了500MB/s，是其上一代的两倍。



包括 ICH10R 在内，Tylersburg 最多可以提供 42 个 PCIe Lanes：36 个 Gen2，6 个 Gen1

Tylersburg-EP 最多提供两个 QPI 总线，可以最多支持两路 Nehalem-EP 处理器（我们尝试了将 Core i7 放上去，结果无法启动……）。Tylersburg-EP 使用的南桥是 ICH10R，而不是以往的 ESB63x1 系列，这一点和桌面版本的 Tylersburg/X58 一样。

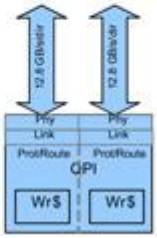


Tylersburg-EP 还支持特别的双芯片组设置:两个 Tylersburg 通过 QPI 总线互相连接，并分别连接一个 Nehalem-EP 处理器，这样，整个系统就可以提供非常多的 PCI Express 信道，例如，连接 4 块全速的 PCI Exress 2.0 x16 显卡等。

第 28 页：Nehalem 座驾：Tylersburg 芯片组 QPI

TBG Specific QPI Features

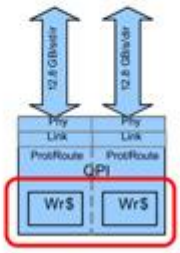
- Physical Layer
  - Up to 6.4GT/s Transfer rates
  - Low power state support
    - > L0, L0s, L1, L2
  - Polarity and Lane reversal
  - Dual Simplex Differential Signaling
  - Forwarded clock
- Link layer
  - 3-bit NodeID
  - 40-bit addressing
  - No Port Bifurcation
- Routing Layer
  - Routing Table picks correct QPI link given a target NodeID
    - > DP only, disabled for UP
    - > 3 NodeID bits means 8 entry routing table
- Protocol Layer
  - QPI caching agent (write cache)
  - Non-coherent home agent for I/O
  - Supports interleaving across ports to prevent hot spots



Tylersburg-EP 最多支持两个 QPI 总线，最高 3.2GHz，6.4GT/s

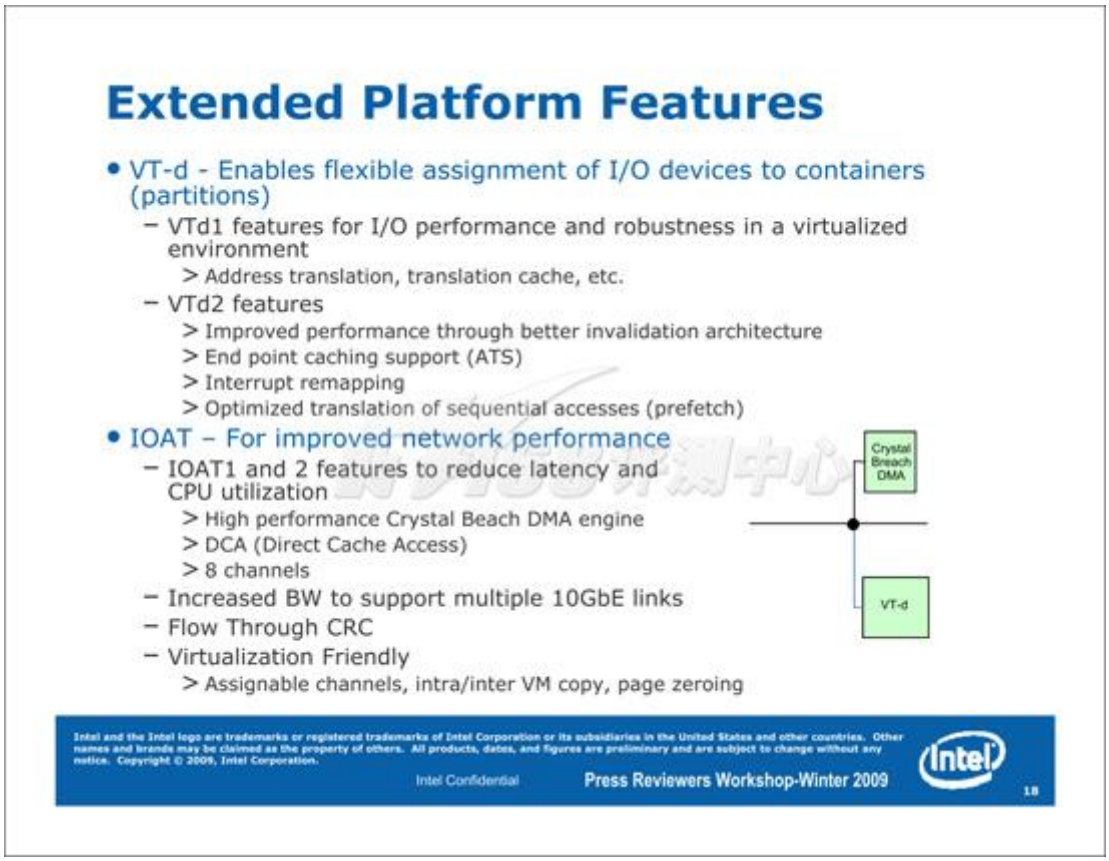
Write Cache

- Used to prefetch ownership for inbound writes
  - Performance requirement for full coherent write bandwidth
- Eviction policy - Immediate eviction for full line writes
- Opportunistic inbound write combining
- Support with prefetch hint for network optimization
- Supports M/E/I states



Tylersburg-EP 还拥有写入缓存，用于缓冲 QPI 发来的写入请求以提高性能

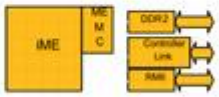
第 29 页：Nehalem 座驾：Tylersburg 芯片组其它特性



除了在 Nehalem 处理器集成的部分 VT-d 虚拟化增强技术之外，Tylersburg-EP 也集成了这个技术的一部分（依赖于 PCI Express 的那部分），Tylersburg-EP 支持 IOAT2，支持 ATS 和中断重映射特性，并提供更强的性能

Integrated Manageability Engine

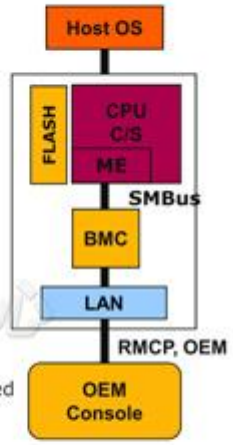
- Supports Intel® Intelligent Power Node Manager Firmware
  - Power & thermal monitoring
  - Power control capability
  - Exposes interfaces which allow management controllers to query platform and set policies
- ARC4 RISC microcontroller
  - Internal caches and general purpose RAM for improved performance
- Watch dog timers
- Private external DDRII interface
- Host interfaces (HECI, IDE)
- DMA engine for data transfer to/from host
- Out of Band Controller Link for communication with ICH
  - Private low pin count, low power communication interface
- RMII (Reduced Media Intelligent Interface)
  - 10/100 MAC interface to another communication device (NIC)



Tylersburg-EP 还支持 Intel Intelligent Power Node Manager（Intel 智能电源节点管理器），进一步提升 Intel 平台的远程管理能力

Intel® Intelligent Power Node Manager

POR DPNM Features on Tylersburg Platforms	
Monitoring & Querying (Enable data center power modeling & usage planning)	• Platform & CPU subsystem power consumption • Temp monitoring • Inlet air temp
Policy-based control (Enable data center policy-based power mgt)	• Power budget policy • Maintain platform power budget • OOB P and T State Control (synchronized with OSPM) • Observe policy activation periods
Actions, Alerts & Notifications (Enable data center automation)	• Report interesting states (e.g. "Consumption approaching budget threshold", etc.) • Initiate shutdown



- A power & thermal monitoring and power control capability embedded in the platform
- Exposes interfaces which allow management controllers to
  - Query platform about its power capability & consumption
  - Specify policy directive (e.g. set a platform power budget)
- Controls subsystem knobs to achieve policy directives
- See Additional Resources section on use case and policy based node power management

第 30 页：Nehalem 座驾：四种 Tylersburg 规格对照表

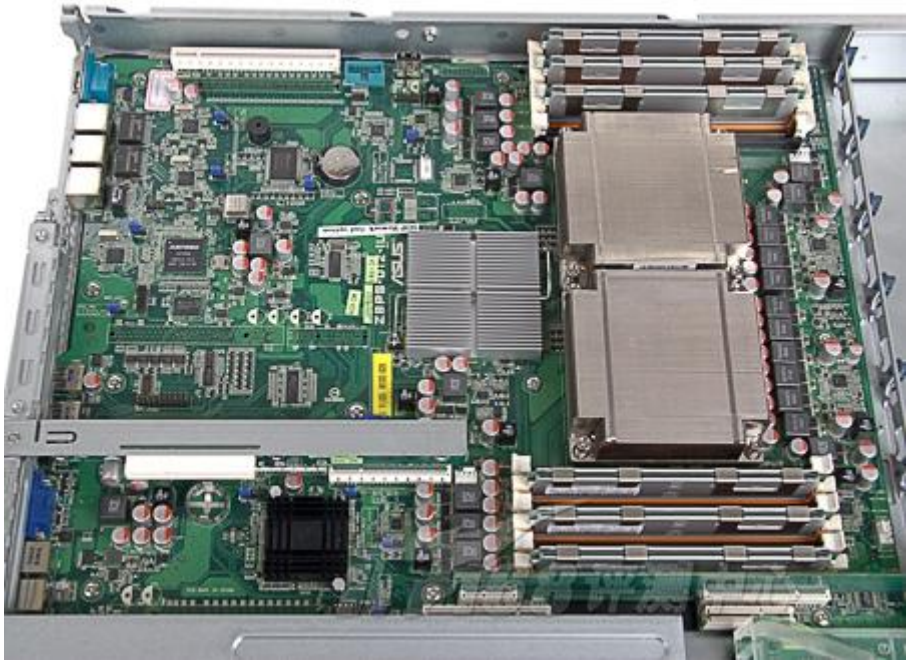
Tylersburg 芯片其实分为四个型号，按照单处理器/双处理器分两种，按照 PCI Express 信道的数量又分为两种，二二得四，最终的型号就有：Tylersburg-24S、Tylersburg-36S、Tylersburg-24D 和 Tylersburg-36D，这些型号非常好记：数字表明了 PCI Express 信道的数量，字母表示单处理器还是双处理器，S 就是 Single 单处理器，D 就是 Dual 双处理器。

Intel Tylersburg 芯片组规格表				
名称	Tylersburg-24S	Tylersburg-36S	Tylersburg-24D	Tylersburg-36D
系列	Tylersburg-DT		Tylersburg-EP	
对应处理器	Nehalem/Bloomfield		Nehalem-EP/Gainestown	
QPI 数/处理器数	1		2	
PCIE Lanes	24(16+8)	36(16+16+4)	24(16+8)	36(16+16+4)
QPI 速率	6.4OGT/s			
VT-d Gen 2	○			
IOAT2	○			
南桥总线	DMI/ESI			
南桥	ICH10R			

单处理器版本是因为只具有一个 QPI 总线，双处理器则是因为具有两个。我们目前接触到的 Tylersburg-EP 都是 Tylersburg-36D。桌面的 Core i7 处理器搭配的 X58 芯片组实际上就是 Tylersburg-36S。

Intel Tylersburg-36D 实物

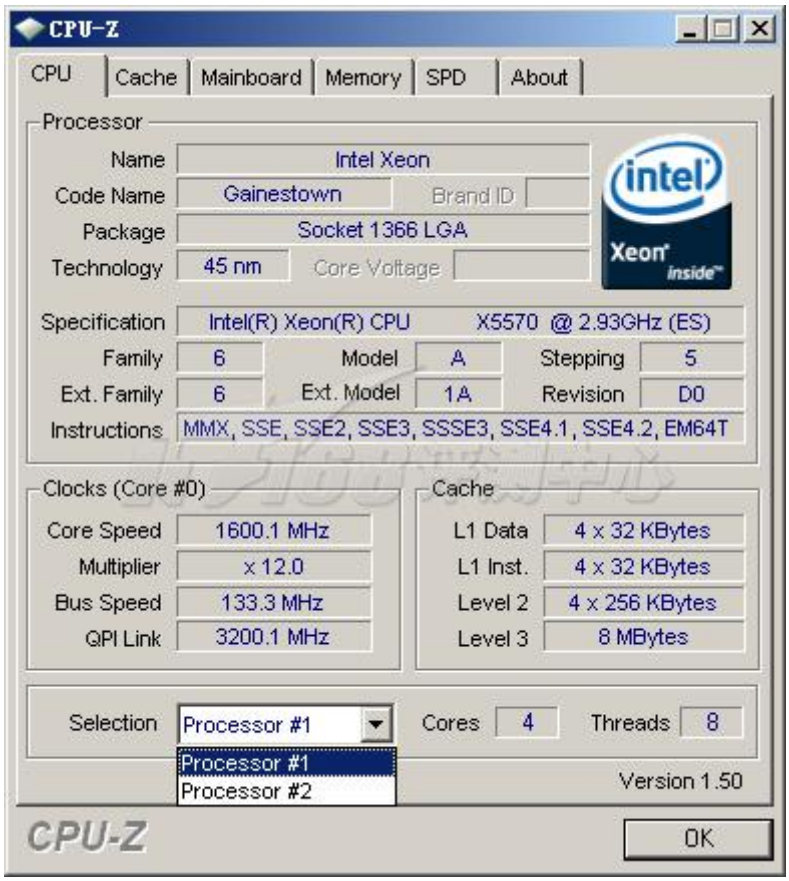
第 31 页：实物图：Nehalem-EP 最高型号 Xeon X5570



两个大方形铝散热器下方就是 Nehalem-EP 处理器

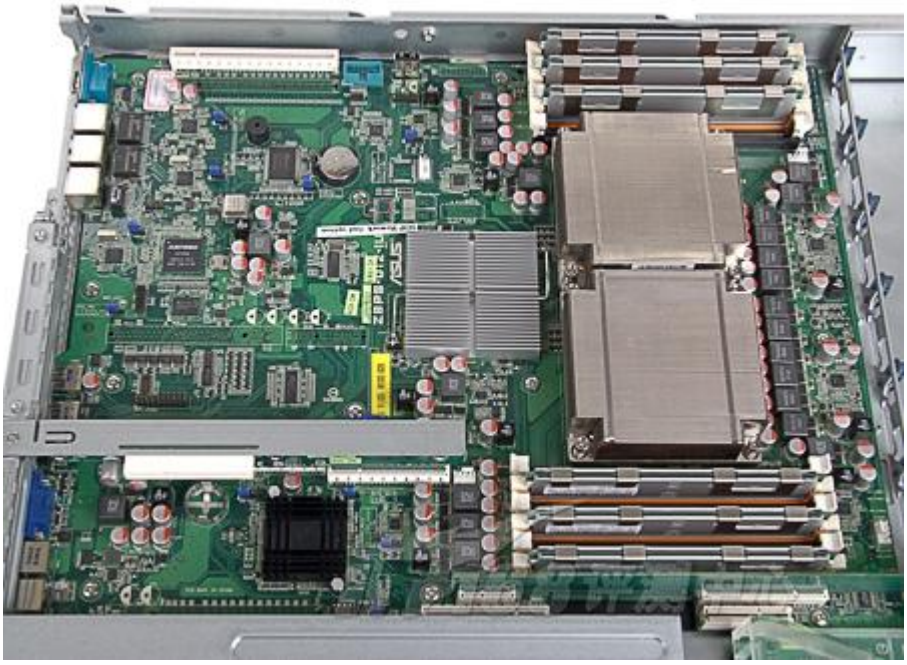
Nehalem-EP Xeon X5570

型号为 Xeon X5570，是目前 Nehalem-EP 种规格最高的处理器型号，频率达到了 2.93GHz，比它高的 W5580（3.20GHz）属于 Nehalem-WS 2S 系列，不属于 Nehalem-EP 系列。



Nehalem-EP/Gainestown Xeon X5570 处理器，主频 2.93GHz。QPI 总线频率 3.2GHz，传输速率是 6.4GT/s

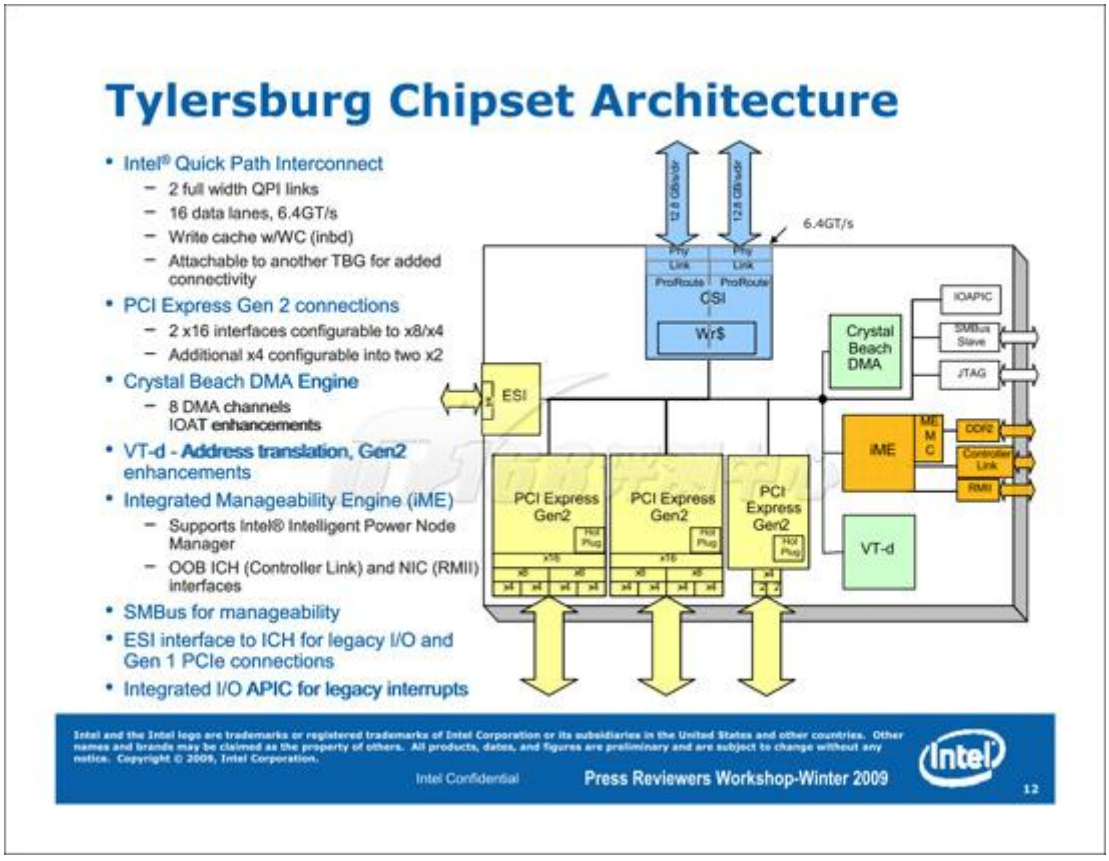
第 32 页：实物图：Tylersburg-EP 芯片组



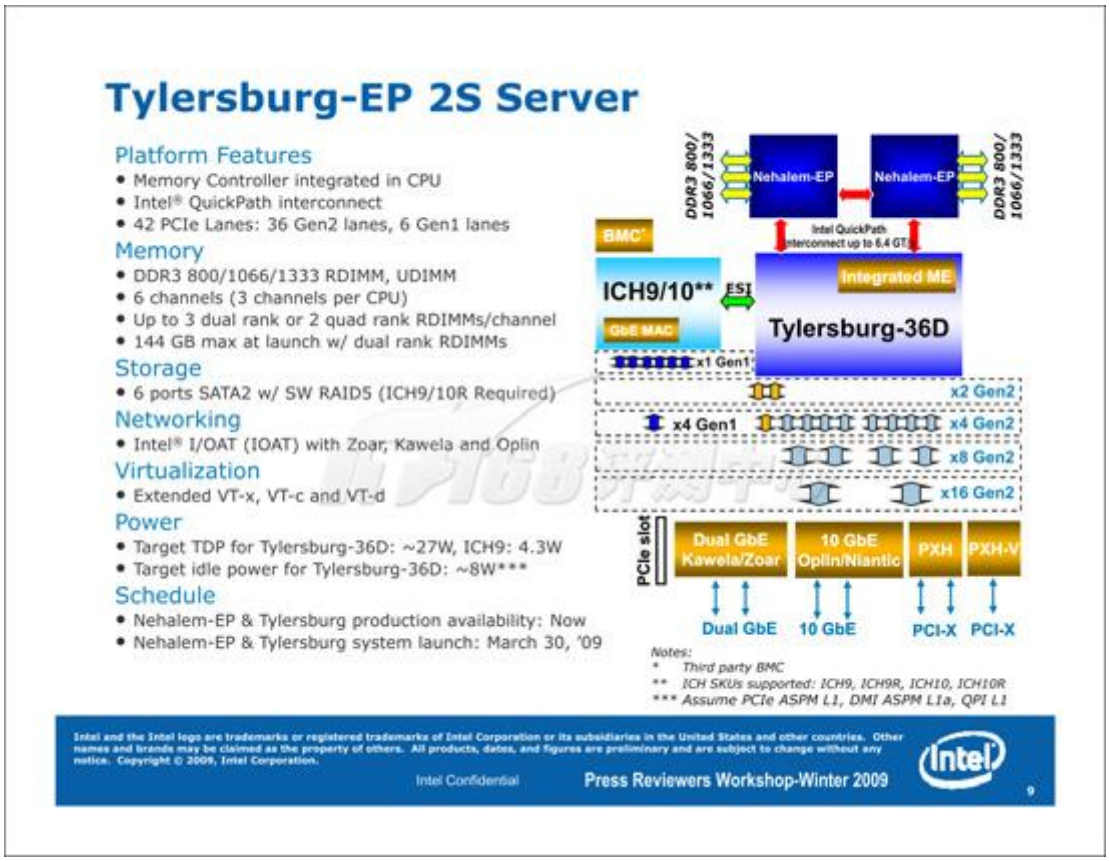
主板正中央的扁平铝散热器下方就是 Tylersburg-EP 芯片

Intel Tylersburg-EP 芯片组实物

这个 Intel TYlersburg-EP 芯片型号为 Intel 5520，属于 Tylersburg-36D 系列，提供了 36 条 PCI Express 信道，其结构如下：



Intel 5520/Tylersburg-36D 结构图



Intel 5520/Tylersburg-36D 结构图

第 33 页：实物图：Intel Nehalem-EP 测试样机

Intel 提供的测试样机实际型号上由华硕生产，型号为 RS700-E4

这是一台 1U 高度的机架式 Nehalem-EP 服务器，个头虽小，动力却是非常澎湃

4 个 3.5 英寸热插拔 SAS 硬盘槽，一个超薄 DVDRW Multi 光驱

两个前置 USB 端口。这是一个很具华硕风格的前面板……

Intel Nehalem-EP 官方评测样机，配置了双路 Xeon X5570 处理器和 24GB DDR3 内存

第 34 页：实物图：Intel Nehalem-EP 测试样机

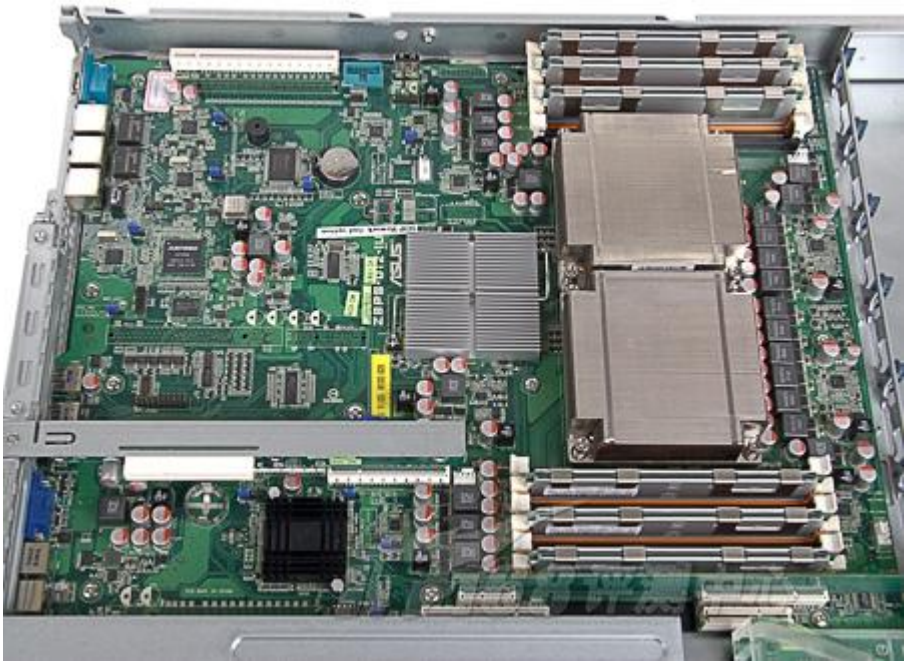
后面板比较特别的地方是具有三个 RJ-45 端口——除了两个千兆以太网端口之外，另外一个是 100Mbps 的远程 IPMI 管理端口

这台测试样机支持冗余电源配置，从这个电源的个头和重量来看，功率不低

著名厂商台达出品，最大总功率 770W，+12V 输出电流 62.5A，输出功率 737.5W（主要输出功率都在+12V 这里了）

数一数，共 7 个热插拔冗余风扇，个头很小——声音很吵

第 35 页：实物图：Intel Nehalem-EP 测试样机

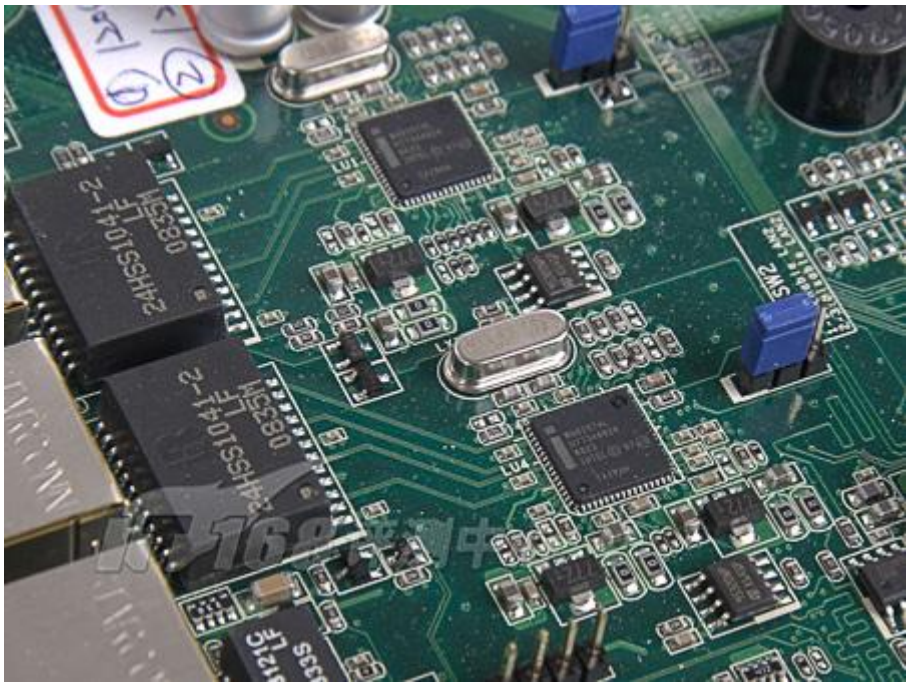


主板是华硕的 Z8PS-D12-1U，两个大方形铝散热器下方就是 Nehalem-EP 处理器

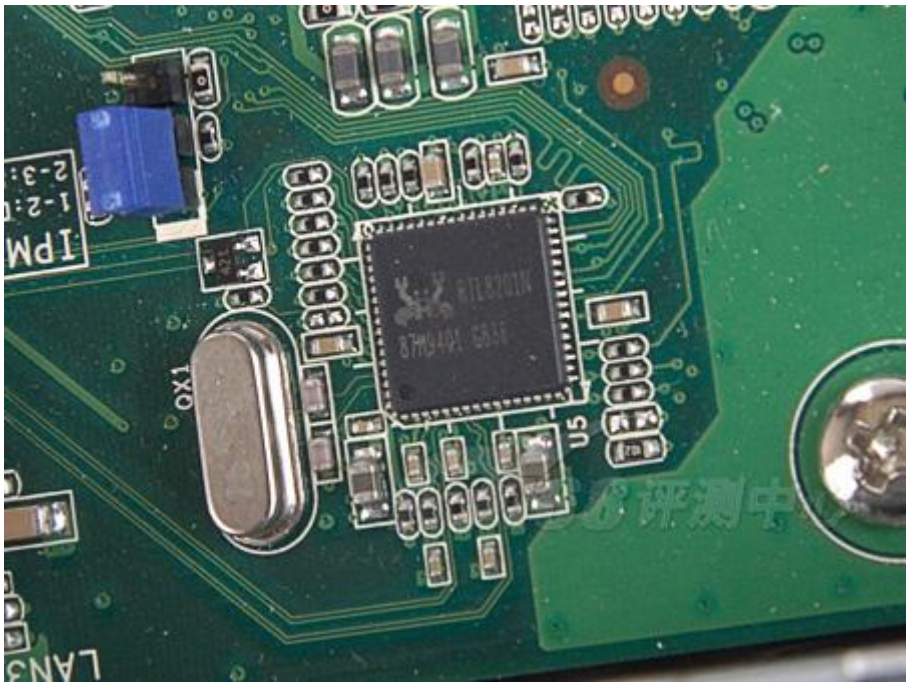
Nehalem-EP：Xeon X5570，主频 2.93GHz，QPI 频率 3.2GHz

Nehalem-EP 集成了内存控制器，因此内存插槽分布在处理器两旁。Nehalem-EP Xeon X5570 处理器支持三通道 DDR3-1333

自然，官方评测样机搭配的也就是 DDR3-1333（小把戏：写成 PC3-10600 的 10600 代表的是传输带宽），共 6 条，合六个内存通道。规格为 R-ECC。Nehalem-EP 也能支持普通的不带 R 也不带 ECC 的内存，这样的胃口适应度就比其上一代好多了



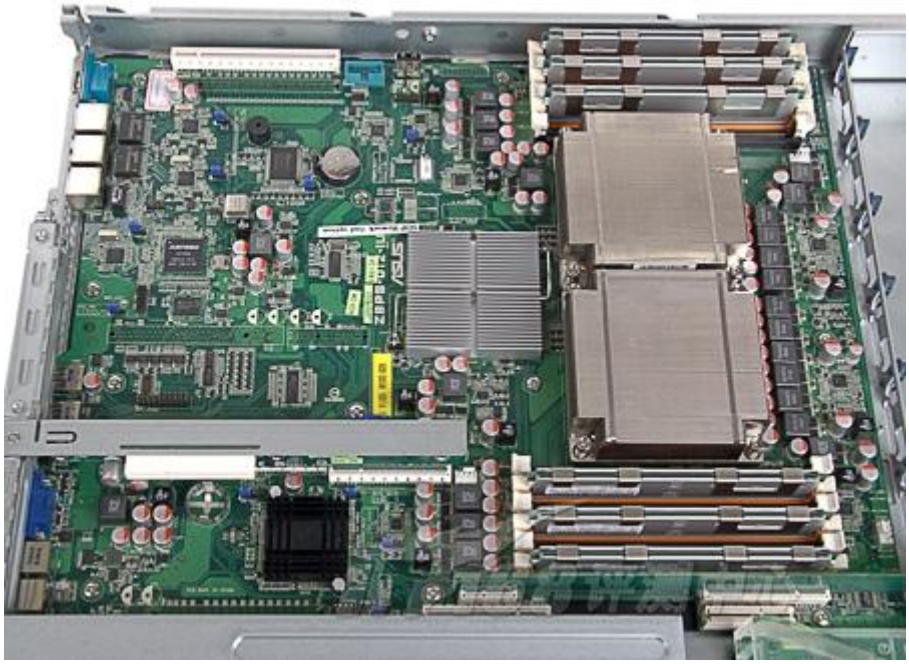
官方评测样机板载了两个 Intel G82574L 千兆网络芯片，这是一种成熟的千兆网络解决方案



这个 Realtek RTL8201N 提供的百兆网络实际上是用来提供远程管理界面的

ASPEED AST2050 是一块 2D 显卡，同时又是一个支持 IPMI 2.0 的 iKVM 模块芯片，集成 200MHz 的 ARM926EJ 处理器和 32KB L1 缓存（16K+16K），其下方的 8MB Qimonda 芯片是其运行内存

第 36 页：实物图：Intel Nehalem-EP 测试样机

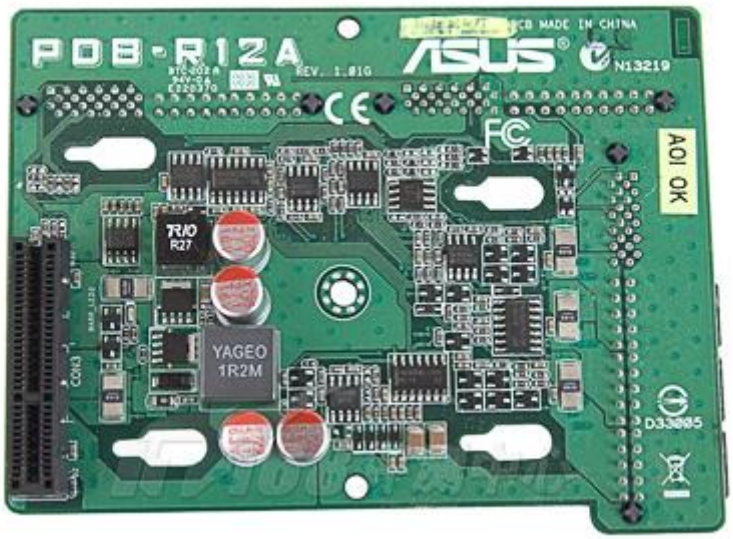


由于是 1U 高度，因此半高的 Low Profile 外形的插卡都无法使用——需要用到 RISER 组件才行

这就是那个 RISER 组件，评测样机提供并且最多支持两个全高度的 PCI Express x16 插槽——不过仅支持单插槽宽度，或许想用 1U 服务器用作工作站的用户可以插两块单槽的显卡来 CrossFire 或者 SLI……

主板上还有几个特别的插槽，看起来很像 PCI Express 插槽

他们使用类似这样的桥



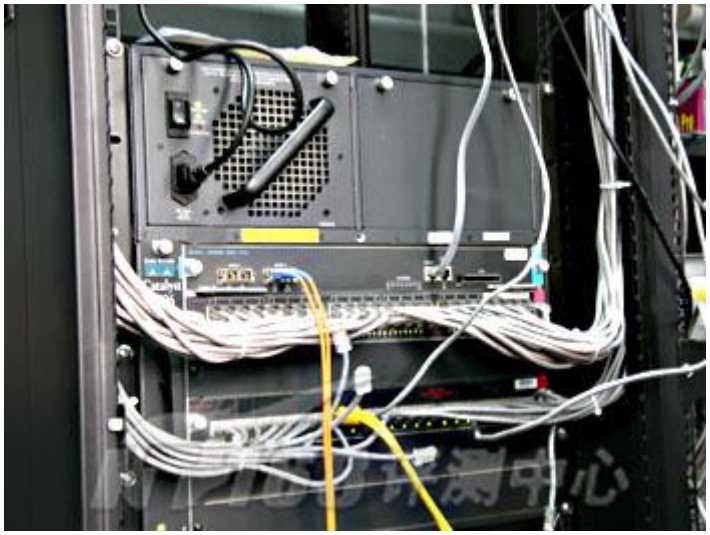
连接的是这样的一块卡，这块卡连接的是磁盘背板，不过这块卡上面没什么线路，因此磁盘控制器实际上是在主板上面

使用了两个 Seagate 的 3.5 英寸硬盘

认真一看，原来是 Barracuda 7200.11 320GB SATA 硬盘。对于服务器应用来说，性能一般

第 37 页：测试环境与测试方法

在 2005 年度服务器横评之后，我们认为当时的网络实验室无法满足今后继续发展的服务器测试的需要。所以，2006 年我们 IT168 评测中心又斥资几十万对于 IT168 网络实验室的服务器测试平台进行了大幅度的升级，为思科 Catalyst4500 千兆交换机（WS-X4013+ Supervisor Engine II-Plus 和 WS-X4548-GB-RJ45）增加了一个思科全千兆 24 口模块 WS-X4424-GB-RJ45，可同时连接 72 个千兆铜缆设备和 2 个光缆设备。另外，我们还购置了 29 台 Dell PowerEdge SC430 塔式服务器和原来的 32 台主流配置 PC 一起为服务器测试平台的提供负载。2007 年，我们又采购性能更强的部分客户端，来确保为新一代的服务器提供足够的测试负载。2009 年初，我们又对所有客户端的内存子系统进行了全面的升级。



Catalyst4500 千兆交换机

部分 Dell PowerEdge SC430 服务器

在新的测试环境下，我们进一步完善了服务器性能测试方案：

- SPEC CPU 2006 v1.0.1

SPEC 是标准性能评估公司（Standard Performance Evaluation Corporation）的简称。SPEC 是由计算机厂商、系统集成商、大学、研究机构、咨询等多家公司组成的非营利性组织，这个组织的目标是建立、维护一套用于评估计算机系统的标准。

SPEC CPU 2006 是 SPEC 组织推出的 CPU 子系统评估软件最新版，我们之前使用的是 SPEC CPU 2000。和上一个版本一样，SPEC CPU 2006 包括了 CINT2006 和 CFP2006 两个子项目，前者用于测量和对比整数性能，而后者则用于测量和对比浮点性能，SPEC CPU 2006 中对 SPEC CPU 2000 中的一些测试进行了升级，并抛弃/加入了一些测试，因此两个版本测试得分并没有可比较性。

SPEC CPU 测试中，测试系统的处理器、内存子系统和使用到的编译器（SPEC CPU 提供的是源代码，并且允许测试用户进行一定的编译优化）都会影响最终的测试性能，而 I/O（磁盘）、网络、操作系统和图形子系统对于 SPEC CPU2006 的影响非常的小。

SPECfp 测试过程中同时执行多个实例（instance），测量系统执行计算密集型浮点操作的能力，比如 CAD/CAM、DCC 以及科学计算等方面应用可以参考这个结果。SPECint 测试

过程中同时执行多个实例（instances），然后测试系统同时执行多个计算密集型整数操作的能力，可以很好的反映诸如数据库服务器、电子邮件服务器和 Web 服务器等基于整数应用的多处理器系统的性能。

我们在被测服务器中安装了当前最新版本的 Intel C++ 10.1.025 Compiler、Intel Fortran 10.1.025 Compiler 这两款 SPEC CPU 2006 必需的编译器，通过最新出现的 QxS 编译参数，Intel Compiler 10 版本开始支持对 Intel SSE4 指令集进行优化（假如只支持 SSE3，则使用 QxT 编译参数）。我们另外安装了 Microsoft Visual Studio 2003 SP1 提供必要的库文件。按照 SPEC 的要求我们根据自己的情况编辑了新的 Config 文件，使用了较多的编译选项。我们根据被测系统选择实际可同时处理的线程数量，最后得到 SPEC rate base 测试结果（基于 base 标准编译，SPEC base rate 测试代表系统同时处理多个任务的能力）。

和其它测试部件不同，SPEC CPU 2006 需要大量的系统物理内存，我们的 SPEC 测试在 64bit Windows Server 2008 Enterprise 下完成，对于每个运算核心，配置 1.5GB 内存。

- Iometer 2006.7.27

  Iometer 是一款功能非常强大的 IO 测试软件，它除了可以在本机运行测试本机的 IO（磁盘）性能之外，还提供了模拟网络应用的能力。在这次的测试中，我们仅仅让它在本机运行测试服务器的磁盘性能。为了全面测试被测服务器的 IO 性能，我们分别选择了不同的测试脚本。

  - Max_throughput（read）：文件尺寸为 64KB，100%读取操作，随机率为 0%，用于检测磁盘系统的最大读取吞吐量
  - Max_IO（read）：文件尺寸为 512B，100%读取操作，随机率为 0%，用于检测磁盘系统的最大读取操作 IO 处理能力
  - Max_throughput（write）：文件尺寸为 64KB，0%读取操作，随机率为 0%，用于检测磁盘系统的最大写入吞吐量
  - Max_IO（write）：文件尺寸为 512B，0%读取操作，随机率为 0%，用于检测磁盘系统的最大写入操作 IO 处理能力

- SiSoftware Sandra v2009

  SiSoftware Sandra 是一款可运行在 32bit 和 64bit Windows 操作系统上的分析软件，这款软件可以对于系统进行方便、快捷的基准测试，还可以用于查看系统的软件、硬件等信息。从 2007 开始，Sandra 的 Arithmetic benchmarks 增加了对 SSE3 & SSE4 SSE4 的支持，在 Multi-Media benchmark 中增加了对于 SSE4 的支持，另外还升级了 File System benchmark 和 Removable Storage benchmark 两个子项目。对于新的硬件的支持当然也是该软件每次升级的重要内容之一。SiSoftware Sandra 所有的基准测试都针对 SMP 和 SMT 进行了优化，最高可支持 32/64 路平台，这也是我们选择这款软件的原因之一。

- NetBench v7.03

  NetBench 是针对文件服务器的性能测试软件，影响 NetBench 性能的主要是服务器的磁盘子系统，服务器磁盘控制器、条带大小、读写缓存、硬盘类型、组建磁盘阵列模式、内存容量、网络拓朴结构等都会对测试结果有明显的影响。我们在被测服务器上设立了文件服务器，NetBench 通过网络实验室中 60 个客户端来模拟网络中的 PC 向文件服务器所发出的文件传输请求，文件服务器则将存储在磁盘上的文件数据发送给相应的客户端。在测试过程中，

客户端会以每四台一组的步进依次增加并且向服务器发送文件传输请求，测试结束后控制台收集数据并绘制出服务器的数据传输变化曲线。

- Benchmark Factory 4.6

  大部分的服务器应用都同数据库有着密切的联系，因此我们今年开始着手在在服务器测试中加入对于数据库性能的测试。我们选择了 Benchmark Factory 4.6 软件和 Microsoft SQL2005 SP3 来测试不同的硬件平台在数据库应用中的表现。

  我们选择了 Benchmark Factory 内置的标准测试脚本 AS3AP，这项测试可用于对于 ANSI 结构化查询语言（SQL）关系型数据库进行测试，它可用于测试 DBMS（单用户微机数据库管理系统），也可用于测试高性能并行或者分布式数据库。

- 系统功耗监测

  我们使用 UNI-T UT71E 智能数字万用表对于被测服务器系统的整体功耗进行了监测，利用随机附带的接口程序，我们可以记录被测服务器任意时间段内的功率变化。

- CineBench R10

  CineBench 是基于 Cinem4D 工业三维设计软件引擎的测试软件，用来测试对象在进行三维设计时的性能，它可以同时测试处理器子系统、内存子系统以及显示子系统，在服务器测试平台中显示子系统不重要，因此就只有前两个的成绩具有意义。和大多数工业设计软件一样，CineBench 可以完善地支持多核/多处理器，它的显示子系统测试基于 OpenGL。

- ScienceMark 2.0

  ScienceMark 2.0 可以用来评估测试对象在执行科学计算时的运算效能，这部分效能主要和处理器子系统和内存子系统相关。我们主要用来评估测试对象的内存子系统的性能。

第 38 页：Nehalem-EP 服务器对比测试平台

本次 Nehalem-EP 评测基于一台曙光的服务器，配置的是双路 Nehalem-EP Xeon E5540 处理器，测试结果并会与我们 IT168 评测中心的 DELL PowerEdge 2900 III 服务器进行对比，测试对比平台的详细参数如下：

测试平台、测试环境			
测试分组			
类别	Intel Nehalem-EP 官方送测样机 华硕 RS700-E4 服务器 双路 Intel Gainestown Xeon X5570	Dawning A650 服务器双路 AMD Shanghai Opteron 2378	双路 Xeon E5430 基准平台 DELL PE2900 III 服务器
处理器子系统			
处理器	双路 Intel Xeon X5570	双路 AMD Opteron 2378	双路 Intel Xeon E5430
处理器架构	Intel 45nm Nehalem	AMD 45nm Shanghai	Intel 45nm Penryn
处理器	Gainestown	Shanghai	Harpertown

代号			
处理器封装	Socket 1366 LGA	Socket F 1207	Socke 771 LGA
处理器规格	四核	四核	四核
处理器指令集	MMX, SSE, SSE2, SSE3, SSSE3, SSE4.1, SSE4.2, EM64T, VT	MMX, 3DNow!, SSE, SSE2, SSE3, SSE4A, AMD-64, AMD-V	MMX, SSE, SSE2, SSE3, SSSE3, SSE4.1, EM64T, VT
主频	2.93GHz	2.40GHz	2.66GHz
处理器外部总线	2x QPI 3200MHz 6.40GT/s 单向 12.8GB/s（每 QPI） 双向 25.6GB/s（每 QPI）	2x HT 1000MHz 2.00GT/s 单向 4.0GB/s（每 HT） 双向 8.0GB/s（每 HT）	FSB 333MHz 1333MT/s 10.6GB/s
L1 D-Cache	4x 32KB 8 路集合关联	4x 64KB 2 路集合关联	4x 32KB 8 路集合关联
L1 I-Cache	4x 32KB 4 路集合关联	4x 64KB 2 路集合关联	4x 32KB 8 路集合关联
L2 Cache	4x 256KB 8 路集合关联	4x 512KB 16 路集合关联	2x 6144KB 16 路集合关联
L3 Cache	8MB 16 路集合关联	2MB 32 路集合关联	
主板			
主板型号	ASUS Z8PS-D12-1U	Tyan S2932-E	DELL PE2900 III
芯片组	Intel Tylersburg-EP IOH:Intel 5520(Tylersburg-36D) ICH:Intel 82801JR(ICH10R)	NVIDIA nForce PRO 3600	MCH:Intel 5000X ICH:Intel ESB6321
芯片特性	2xQPI VT-d Gen 2	1x HT	2xFSB1333 12MB Snoop Filter VT-d Gen 1
内存控	每 CPU 集成三通道 R-ECC	每 CPU 集成双通道 R-ECC DDR2 800	北桥集成四通道 FBD DDR2

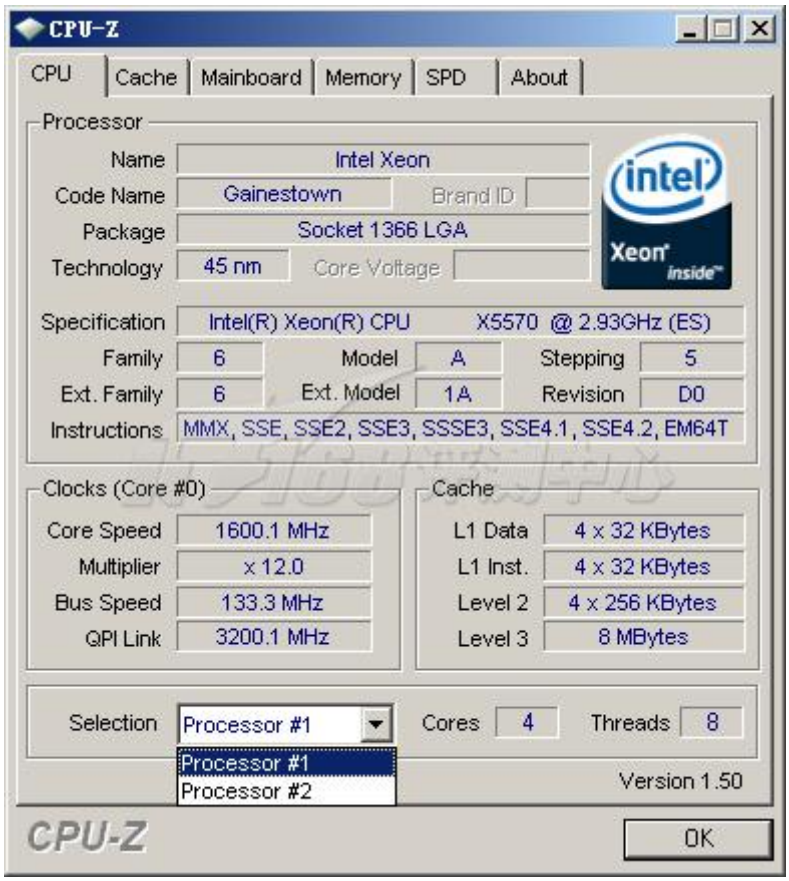
制器	DDR3 1333		667
内存	4GB R-ECC DDR3 1333 SDRAM x6	2GB R-ECC DDR2 667 SDRAM x4	2GB FBD DDR2 667 SDRAM x4
系统磁盘子系统			
磁盘控制器	LSI Embedded MegaRAID SAS RAID Controller	LSI MegaRAID SAS RAID Controller	DELL Perc 5/i RAID Controller
磁盘控制器规格	8x SAS 3Gbps	8x SAS 3Gbps	8x SAS 3Gbps
磁盘控制器设置	RAID 0	RAID 5	RAID 5
磁盘控制器驱动	LSI MegaSR 13.06.0212.2009	LSI SAS 3.8.0.64	LSI SAS 3.8.0.64
磁盘	Fujitsu MBA3300RC x2	Fujitsu MBA3147RC x3	Seagate Cheetah 15K.5 ST314655SS x3
磁盘规格	15000RPM 300GB SAS 3Gbps 16MB Cache	15000RPM 147GB SAS 3Gbps 16MB Cache	15000RPM 146GB SAS 3Gbps 16MB Cache
磁盘设置	SAS 3Gbps 50GB 系统分区	SAS 3Gbps 30GB 系统分区	SAS 3Gbps 20GB 系统分区
网络子系统			
网卡	Intel 82574 Gigabit Network Controller x2	NVIDIA nForce Pro 3600 integrated MAC with Marvell 88E1121 PHY GbE Controller x2	Broadcom BCM5708C PCI-E 千兆网卡 x2
网卡设置	PCI Express x1 @ ICH10R I/OAT Intel Teaming Load Balancing	Forceware Teaming Load Balancing	PCI Express x1 @ ESB6321 Broadcom NIC Teaming Load Balancing
网卡驱动	Intel PRO Set 13.5	NVIDIA NIC/LAN v67.76.1	Broadcom NetXtreme 2 11.04.01
软件环境			

操作系统	Microsoft Windows Server 2008 Enterprise Edition SP1 x64	Microsoft Windows Server 2003 R2 Enterprise Edition SP2 x64	Microsoft Windows Server 2008 Enterprise Edition SP1 x64

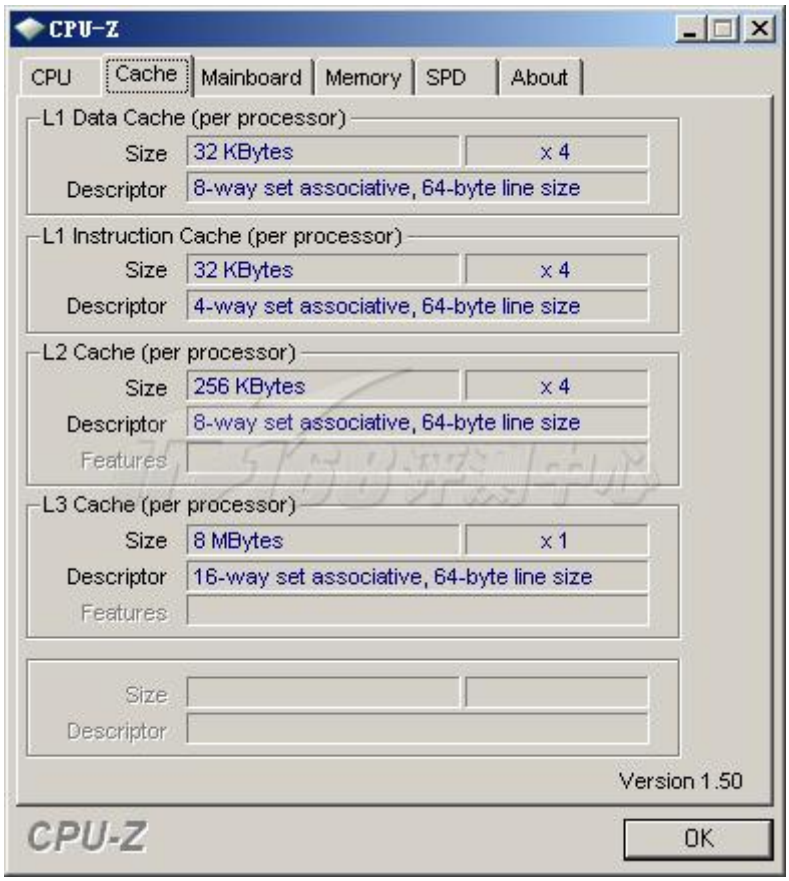
用来对比的 45nm Shanghai Opteron 2378（左）

第 39 页：软件测试信息、系统部件简介

Nehalem-EP：Xeon X5570，主频 2.93GHz，QPI 频率 3.2GHz



Nehalem-EP/Gainestown Xeon X5570 处理器，主频 2.93GHz。QPI 总线频率 3.2GHz，传输速率是 6.4GT/s

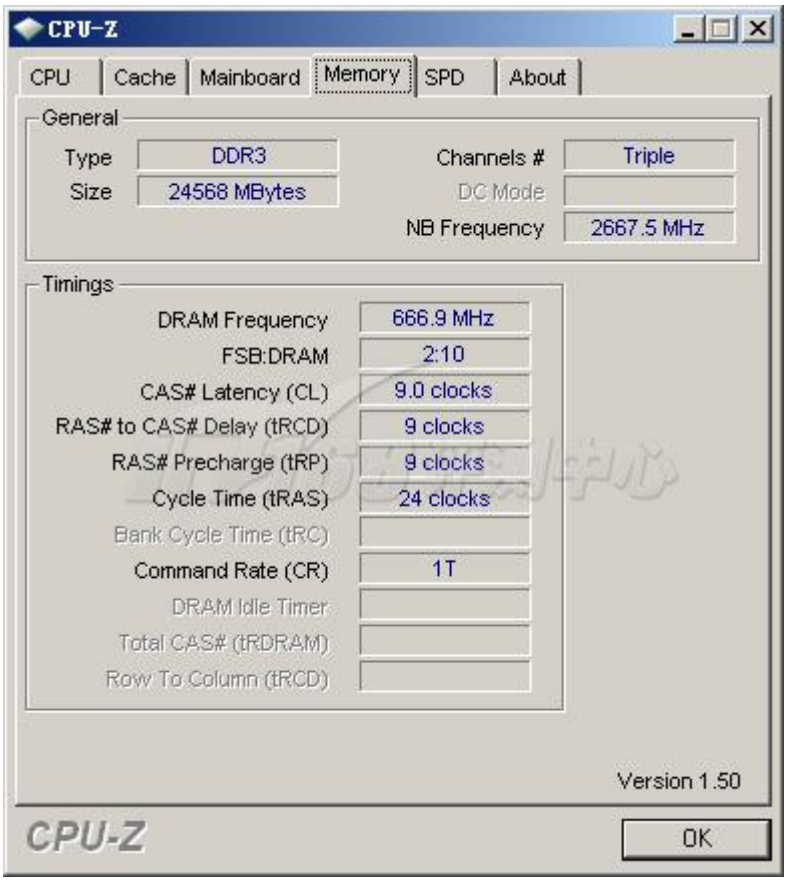


64KB L1，256KBL2，8MB 共享 L3

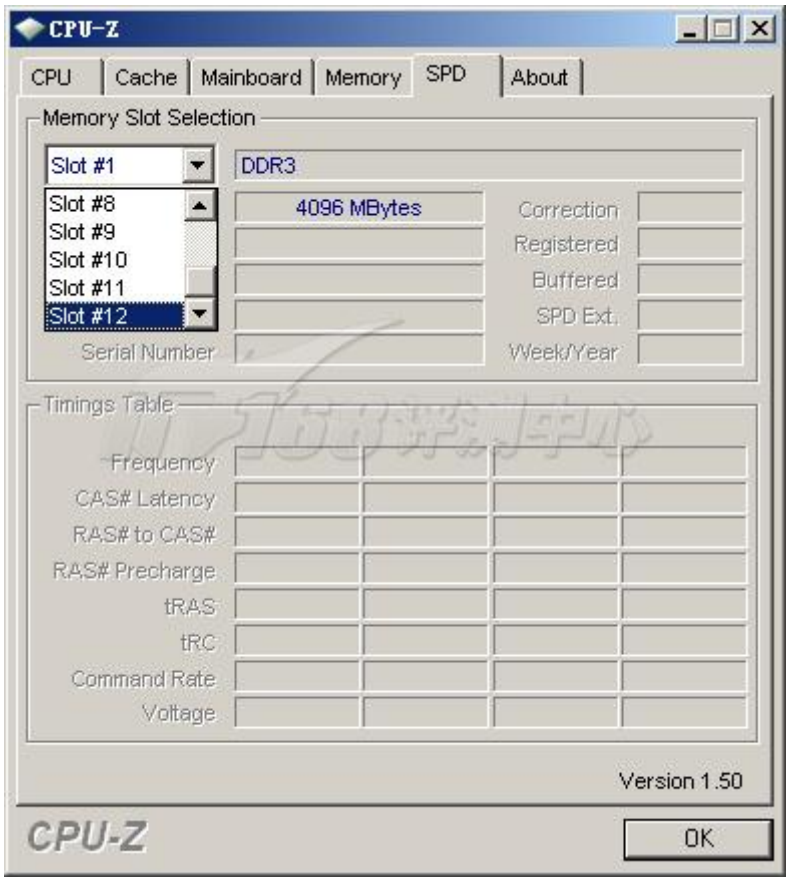
ASUS Z8PS-D12-1U 主板，采用 Intel 5520 + ICH10R 芯片组，也就是 Intel Tylersburg 芯

片组

D12 就是具备 12 个 DIMM 的意思，1U 就是专门为 1U 机架服务器设计



24GB R-ECC DDR3 1333 内存，NB Frequency 是 Nehalem-EP 处理器 Uncore 部分的频率（实际上就是 L3 的频率），而不是 Tylersburg 芯片组的频率：这个 Xeon X5570 的 Uncore 频率（也就是 L3 频率）是 2.67GHz



每条内存 4GB，总共 6 条 DDR3-1333 内存

第 40 页：SiSoftware Sandra 2009 处理器性能测试

SiSoftware Sandra 是一款可运行在 32bit 和 64bit Windows 操作系统上的分析软件，它可以对于系统进行方便、快捷的基准测试，还可以用于查看系统的软件、硬件等信息。SiSoftware Sandra 所有的基准测试都针对 SMP 和 SMT 进行了优化，最高可支持 32/64 路平台。我们利用了其中多个性能测试模块对于被测系统的性能进行了快速的测试。

有一点需要说明的是，Sandra 的处理器架构性能测试是根据处理器所能支持的所有指令集中选择进行的，不同的处理器支持的指令集不同，测试使用到的指令集也就不同。例如，Nehalem 在这个测试当中就可以使用 SSE4.2，而 Penryn 就只能使用 SSE4.1，而 Opteron 可能就只能使用 SSE3 了。一般而言，由于可以使用 SSE4，Intel 的处理器理论性能会比较好。

SiSoftware Sandra Pro Business 2009			
测试对象	Intel Nehalem-EP 双路 Intel Gainestown Xeon X5570 2.93GHz	Dawning A650 双路 AMD Shanghai Operton 2378 2.40GHz	DELL PE2900 III 双路 Intel Harptown Xeon E5430 2.66GHz
Processor Arithmetic Benchmark 处理器架构测试			

Dhrystone ALU	142977MIPS	63082MIPS	91006MIPS
Dhrystone ALU vs SPEED	48.75MIPS/MHz	26.28MIPS/MHz	34.21MIPS/MHz
Whetstone iSSE3	124035MFLOPS	62993MFLOPS	78385MFLOPS
Dhrystone iSSE3 vs SPEED	42.29MFLOPS/MHz	26.25MFLOPS/MHz	29.47MFLOPS/MHz
Processor Multi-Media Benchmark 处理器多媒体测试			
Multi-Media Int x16 iSSE4.1	296.85MPixel/s		
Multi-Media Int x8 aSSE2		187.70MPixel/s	
Multi-Media Int x8 iSSE4.1			199.33MPixel/s
Multi-Media Int x16 iSSE4.1 vs SPEED	101.21MPixel/s/MHz		
Multi-Media Int x8 aSSE2 vs SPEED		78.21MPixel/s/MHz	
Multi-Media Int x8 iSSE4.1 vs SPEED			74.94MPixel/s/MHz
Multi-Media Float x8 iSSE2	228.24MPixel/s		
Multi-Media Float x4 iSSE2		81.53MPixel/s	108.69MPixel/s
Multi-Media Float x8 iSSE2 vs SPEED	77.82kPixels/s/MHz		
Multi-Media Float x4 iSSE2 vs SPEED		33.97kPixels/s/MHz	40.86kPixels/s/MHz
Multi-Media Double x4 iSSE2	125.88MPixel/s		
Multi-Media Double x2 iSSE2		44.51MPixel/s	55.75MPixel/s

Multi-Media Double x4 iSSE2 vs SPEED	42.92kPixels/s/MHz		
Multi-Media Double x2 iSSE2 vs SPEED		18.55kPixels/s/MHz	20.96kPixels/s/MHz
Multi-Core Efficiency Benchmark			
Inter-Core Bandwidth	75.61GB/s	6.54GB/s	20.54GB/s
Inter-Core Bandwidth vs SPEED	26.40MB/s/MHz	2.79MB/s/MHz	7.91MB/s/MHz
Inter-Core Latency (越小越好)	16ns	128ns	90ns
Inter-Core Latency vs SPEED (越小越好)	0.01ns/MHz	0.05ns/MHz	0.03ns/MHz
.NET Arithmetic Benchmark .NET 架构测试			
Dhrystone .NET	32904MIPS	12736MIPS	10562MIPS
Dhrystone .NET vs SPEED	11.22MIPS/MHz	5.31MIPS/MHz	3.97MIPS/MHz
Whetstone .NET	78286MFLOPS	38737MFLOPS	45399MFLOPS
Whetstone .NET vs SPEED	26.69MFLOPS/MHz	15.62MFLOPS/MHz	17.07MFLOPS/MHz
.NET Multi-Media Benchmark .NET 多媒体测试			
Multi-Media Int x1 .NET	62.28MPixel/s	24.48MPixel/s	31.28MPixel/s
Multi-Media Int x1 .NET vs SPEED	21.23kPixels/s/MHz	10.20kPixels/s/MHz	11.76kPixels/s/MHz
Multi-Media Float x1 .NET	26.19MPixel/s	5.29MPixel/s	8.68MPixel/s
Multi-Media Float x1 .NET vs	8.93kPixels/s/MHz	2.20kPixels/s/MHz	3.26kPixels/s/MHz

SPEED			
Multi-Media Double x1 .NET	51.45MPixel/s	21.31MPixel/s	24.75MPixel/s
Multi-Media Double x1 .NET vs SPEED	17.54kPixels/s/MHz	8.88kPixels/s/MHz	9.30kPixels/s/MHz

SiSoftware Sandra 对比，用蓝色标出了性能特出的项目

处理器架构性能测试分为整数和浮点两个部分，在频率更低的情况下，Nehalem-EP 处理器的测试成绩全面强于对比的处理器，领先幅度在 50%~100%左右。

第 41 页：SiSoftware Sandra 2009 缓存内存性能测试

SiSoftware Sandra 缓存内存测试主要包括内存带宽、内存延迟等性能的测试。

SiSoftware Sandra Pro Business 2009			
测试对象	Intel Nehalem-EP 双路 Intel Gainestown Xeon X5570 2.93GHz	Dawning A650 双路 AMD Shanghai Operton 2378 2.40GHz	DELL PE2900 III 双路 Intel Harptown Xeon E5430 2.66GHz
Memory Bandwidth Benchmark 内存带宽测试			
Int Buff'd iSSE2 Memory Bandwidth	16.93GB/s	16.59GB/s	6.13GB/s
Int Buff'd iSSE2 Memory Bandwidth vs SPEED		25.52MB/s/MHz	9.43MB/s/MHz
Float Buff'd iSSE2 Memory Bandwidth	16.90GB/s	16.58GB/s	6.13GB/s
Float Buff'd iSSE2 Memory Bandwidth vs SPEED		25.50MB/s/MHz	9.43MB/s/MHz
Memory Latency Benchmark 内存延迟测试			
Memory(Random Access) Latency (越小越好)	81ns	106ns	108ns
Memory(Random Access) Latency vs SPEED (越小越好)		0.16ns/MHz	0.16ns/MHz

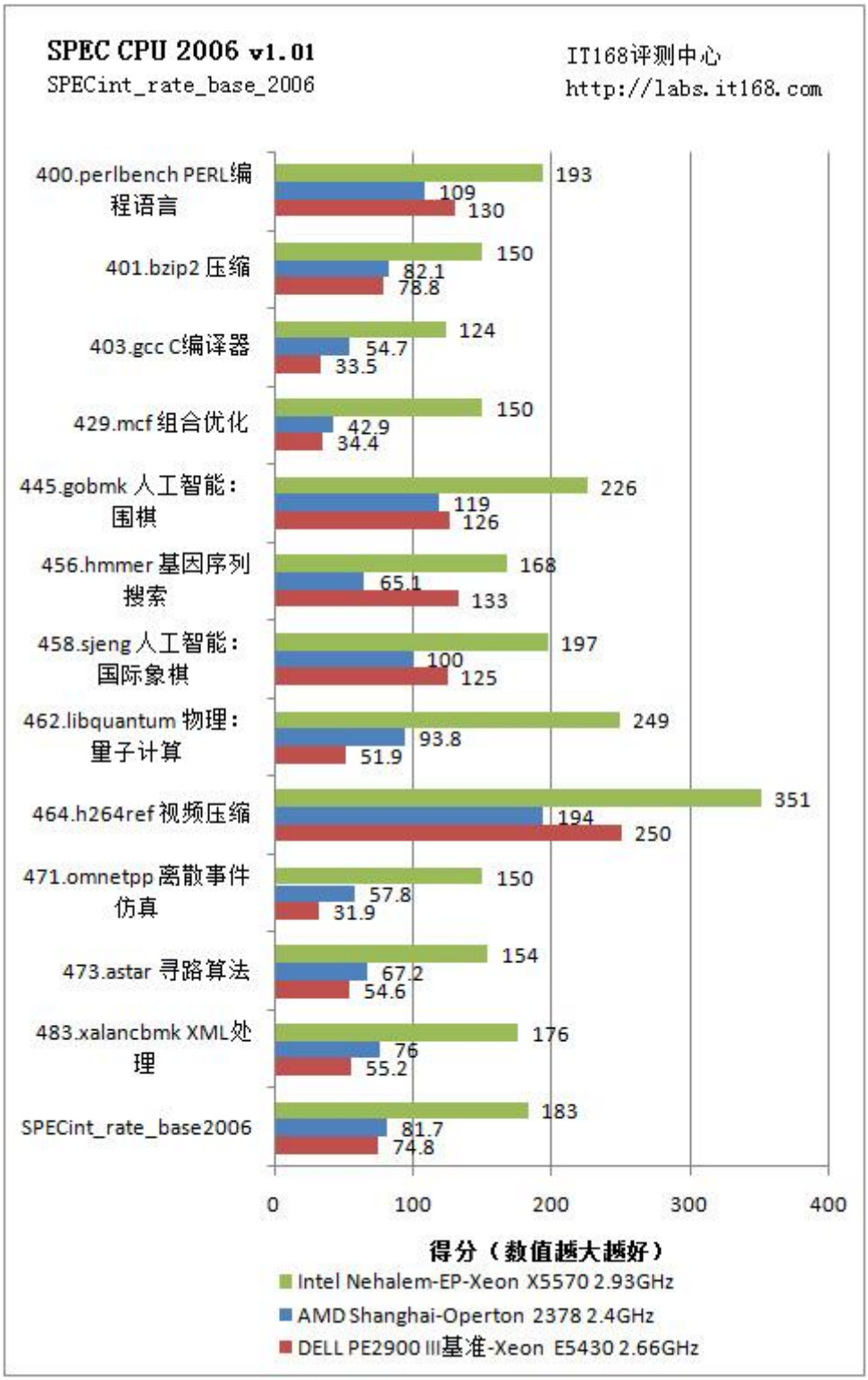
Speed Factor (越小越好)	61.40	83.80	95.20
Internal Data Cache	4clocks	3clocks	3clocks
L2 On-board Cache	10clocks	16clocks	18clocks
L3 On-board Cache	48clocks	58clocks	
Cache and Memory Benchmark 缓存及内存测试			
Cache/Memory Bandwidth	143.24GB/s	77.08GB/s	68.88GB/s
Cache/Memory Bandwidth vs SPEED	50.01MB/s/MHz	32.89MB/s/MHz	26.52MB/s/MHz
Speed Factor (越小越好)	20.90	36.00	111.90
Internal Data Cache	448.46GB/s	299.00GB/s	421.23GB/s
L2 On-board Cache	421.42GB/s	162.91GB/s	122.68GB/s

SiSoftware Sandra 对比，用蓝色标出了性能特出的项目

在频率更低的情况下，Nehalem-EP 处理器的测试成绩全面强于对比的处理器。

第 42 页：SPEC CPU 2006 整数性能测试

SPEC CPU 2006 整数运算主要包含编译、压缩、人工智能、视频压缩转换、XML 处理等，此外，各种日常操作也主要是基于整数操作。SPEC CPU 2006 的整数运算包含了 400.perlbench PERL 编程语言、401.bzip2 压缩、403.gcc C 编译器、429.mcf 组合优化、445.gobmk 人工智能：围棋、456.hmmer 基因序列搜索、458.sjeng 人工智能：国际象棋、462.libquantum 物理：量子计算、464.h264ref 视频压缩、471.omnetpp 离散事件仿真、473.astar 寻路算法、483.xalancbmk XML 处理共 12 项。



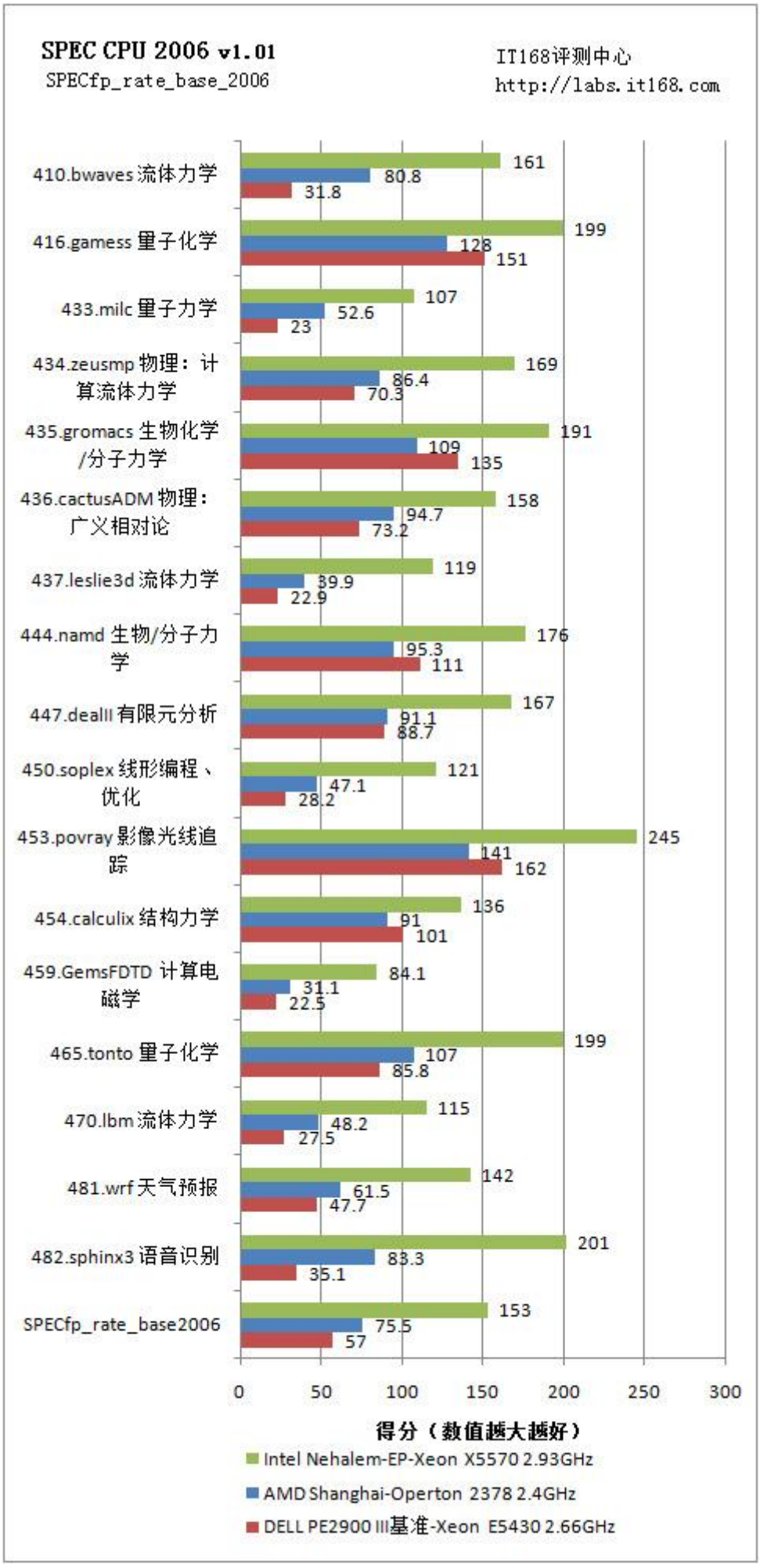
Intel Nehalem-EP/Gainestown Xeon X5570 SPEC CPU 2006 整数运算性能

对比频率更高的 Harpertown，Nehalem-EP/Gainestown 的性能可谓让人大吃一惊：提升超过了 100%，Xeon X5570 的得分为 183，比 Xeon E5430 的 74.8 分高 144.7%，成绩斐然——当然 CPU 的主频也高了 10.2%，同频率下的提升也达到了 122.1%。在测试当中，403.gcc C 编译器（270.1%）、429.mcf 组合优化（336.0%）、462.libquantum 物理：量子计算（379.8%）、471.omnetpp 离散事件仿真（370.2%）、473.astar 寻路算法（182.1%）、483.xalancbmk XML

处理（218.8%）这 6 项的提升都很明显，这些项目都能因直联架构而获益。所有的项目都能从超线程当中获得提升。

第 43 页：SPEC CPU 2006 浮点性能测试

SPEC CPU 2006 的浮点运算测试包括的全部都是科学运算，科学运算需要用到大量的高精度浮点数据，如 410.bwaves 流体力学、416.gamess 量子化学、433.milc 量子力学、434.zeusmp 物理：计算流体力学、435.gromacs 生物化学/分子力学、436.cactusADM 物理：广义相对论、437.leslie3d 流体力学、444.namd 生物/分子、447.dealII 有限元分析、450.soplex 线形编程、优化、453.povray 影像光线追踪、454.calculix 结构力学、459.GemsFDTD 计算电磁学、465.tonto 量子化学、470.lbm 流体力学、481.wrf 天气预报、482.sphinx3 语音识别共 17 项测试。


SPEC CPU 2006 v1.01
SPECfp_rate_base_2006
IT168评测中心
http://labs.it168.com
410.bwaves 流体力学
161
80.8
31.8
416.gamess 量子化学
199
128
151
433.milc 量子力学
107
52.6
23
434.zeusmp 物理：计算流体力学
169
86.4
70.3
435.gromacs 生物化学/分子力学
191
109
135
436.cactusADM 物理：广义相对论
158
94.7
73.2
437.leslie3d 流体力学
119
39.9
22.9
444.namd 生物/分子力学
176
95.3
111
447.dealII 有限元分析
167
91.1
88.7
450.soplex 线形编程、优化
121
47.1
28.2
453.povray 影像光线追踪
245
141
162
454.calculix 结构力学
136
91
101
459.GemsFDTD 计算电磁学
84.1
31.1
22.5
465.tonto 量子化学
199
107
85.8
470.lbm 流体力学
115
48.2
27.5
481.wrf 天气预报
142
61.5
47.7
482.sphinx3 语音识别
201
83.3
35.1
SPECfp_rate_base2006
153
75.5
57
0
50
100
150
200
250
300
得分（数值越大越好）
Intel Nehalem-EP-Xeon X5570 2.93GHz
AMD Shanghai-Operton 2378 2.4GHz
DELL PE2900 III基准-Xeon E5430 2.66GHz

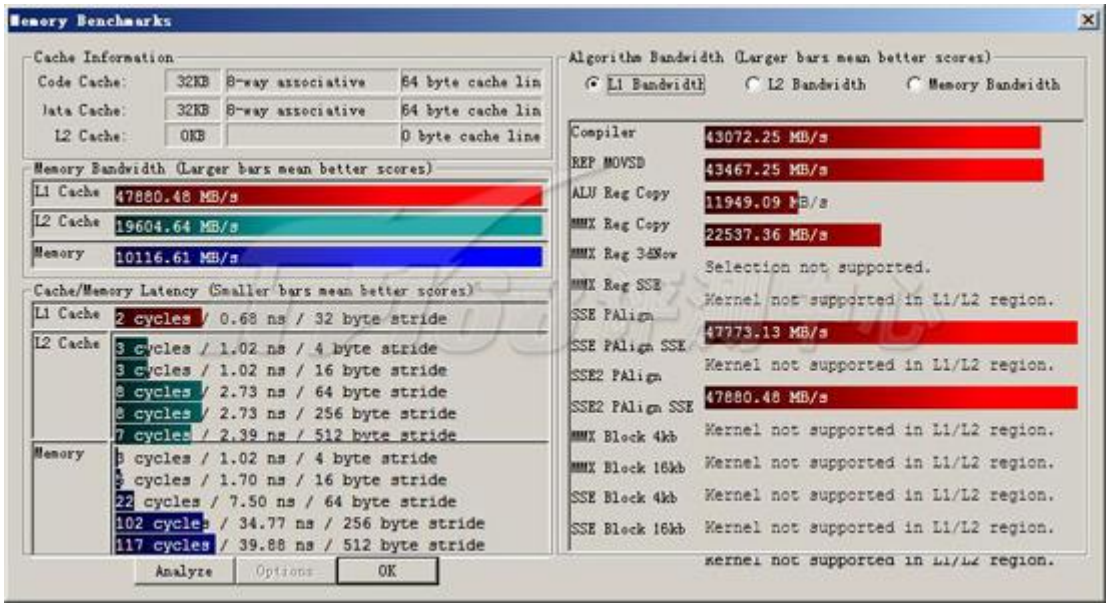
Intel Nehalem-EP/Gainestown Xeon X5570 SPEC CPU 2006 浮点运算性能

浮点运算上的提升比整数上更大，Nehalem-EP/Gainestown 的得分为 153，比 Harpertown 的 57 分高 168.4%，单位频率的提升达到了 **143.6%**，这是 IMC、QPI、HTT 的集合成果，表明了 Nehalem 架构的强大优势（Nehalem-EP 测试上仍然是整数性能表现强于浮点性能表现）。在测试当中，410.bwaves 流体力学（406.3%）、433.milc 量子力学（365.2%）、434.zeusmp 物理：计算流体力学（140.4%）、436.cactusADM 物理：广义相对论（115.8%）、437.leslie3d 流体力学（410.7%）、450.soplex 线形编程、优化（329.1%）、459.GemsFDTD 计算电磁学（273.8%）、465.tonto 量子化学（131.9%）、470.lbm 流体力学（318.2%）、481.wrf 天气预报（197.7%）、482.sphinx3 语音识别（472.6%）这 11 个项目的提升都很大，提升幅度都是几倍几倍的，最高的是 482.sphinx3 语音识别（472.6%），Xeon X5570 的性能是 Xeon E5430 的 5.7 倍以上。

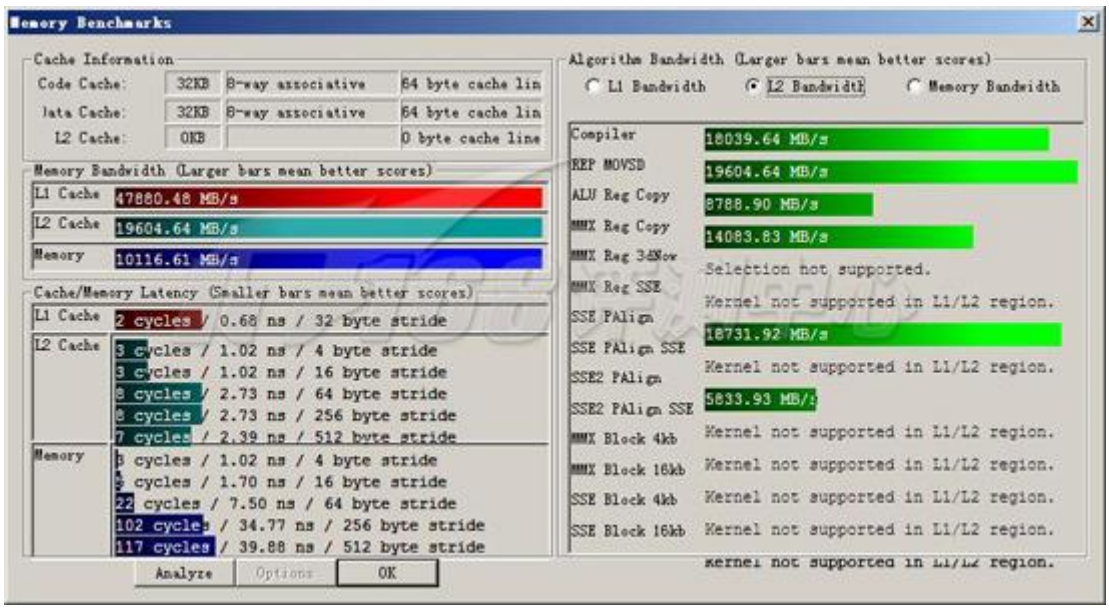
第 44 页：ScienceMark 缓存内存子系统性能测试

ScienceMark v2.0 Membench

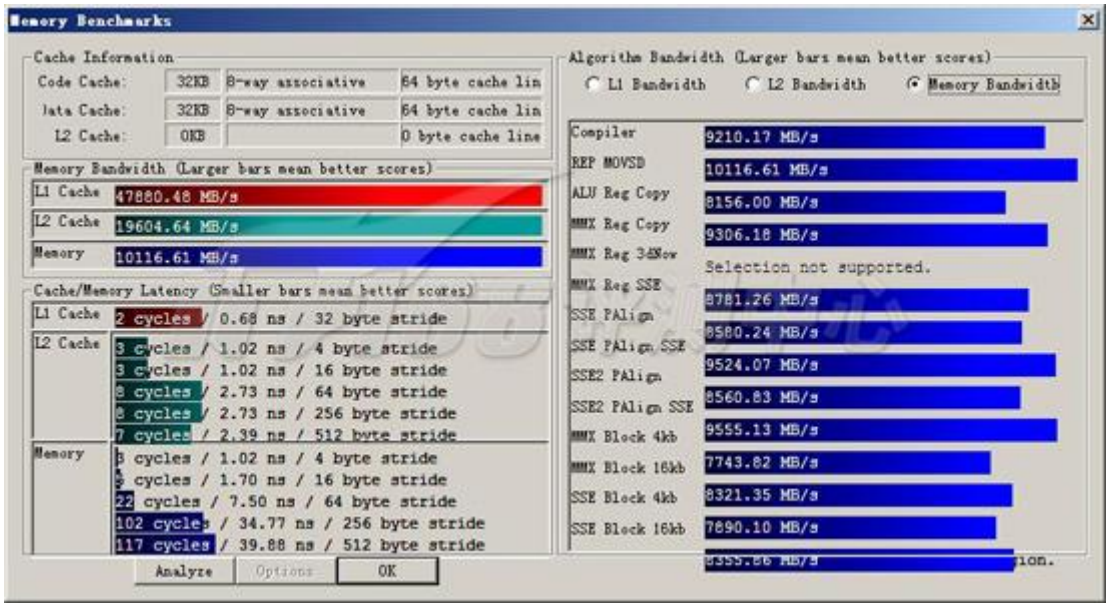
ScienceMark v2.0 是一款用于测试系统特别是处理器在科学计算应用中的性能的软件，MemBenchmark 是其中针对处理器缓存、系统内存而设计的功能模块，它可以测试系统内存带宽、L1 Cache 延迟、L2 Cache 延迟和系统内存延迟，另外还可以测试不同指令集的性能差异。



ScienceMark v2.0 Membench L1 测试成绩



ScienceMark v2.0 Membench L2 测试成绩



ScienceMark v2.0 Membench 内存测试成绩

首先我们进行的是 ScienceMark 的测试，主要考察系统的缓存和内存子系统情况。L1/L2 Cache 的成绩主要是跟处理器频率相关，因为目前的处理器当中 L1 Cache 都是和处理器核心同频率的，而 L2 Cache 基本上也是——当前的处理器 L2 都是全速的（放置在处理器内但不在同一个芯片上的 Pentium II 为半速 L2，而 Pentium 之前的处理器 L2 则和处理器分离，速度更低）。越快的频率，L1/L2 性能就越好。而内存带宽主要由两部分相关：比较大的部分是内存架构，小部分是内存操作指令（集），例如使用最新的 SSE 指令集比通常的 ALU 指令集会得到更大的吞吐量，而不同的 SSE 版本性能也有不同。

ScienceMark Membench			
厂商	Intel	Intel	Intel
产品型号	Nehalem-EP Intel Gainestown	A650 AMD Shanghai	PowerEdge 2900 III Intel Harpertown

	Xeon X5570 2.93GHz	Operton 2378 2.40GHz	Xeon E5430 2.66GHz
内存技术参数	4GB R-ECC DDR3-1333 SDRAM x6	4GB R-ECC DDR3-1333 SDRAM x6	4GB R-ECC DDR3-1333 SDRAM x6
L1 带宽(MB/s)	47880.48	48167.88	55376.16
L2 带宽(MB/s)	19604.64	14314.34	16757.55
内存带宽(MB/s)	10116.61	6672.76	4485.09
L1 Cache Latency(ns)			
32 Bytes Stride	2 cycles 0.68 ns	1.25 ns	1.13 ns
L1 Algorithm Bandwidth(MB/s)			
Compiler	43072.25	34042.63	25201.96
REP MOVSD	43467.25	34864.10	25467.15
ALU Reg Copy	11949.09	12166.94	13093.65
MMX Reg Copy	22537.36	25698.47	25242.19
SSE PAlign	47773.13	48167.40	52826.21
SSE2 PAlign	47880.48	48167.88	55376.16
L2 Cache Latency(ns)			
4 Bytes Stride	3 cycles 1.02 ns	1.25 ns	1.13 ns
16 Bytes Stride	3 cycles 1.02 ns	1.25 ns	1.50 ns
64 Bytes Stride	8 cycles 2.73 ns	3.75 ns	4.51 ns
256 Bytes Stride	8 cycles 2.73 ns	6.25 ns	4.51 ns
512 Bytes Stride	7 cycles 2.39 ns	6.25 ns	4.89 ns
L2 Algorithm Bandwidth(MB/s)			
Compiler	18039.64	11609.57	11880.48
REP MOVSD	19604.64	12140.00	12536.88
ALU Reg Copy	8788.90	9273.71	8577.86
MMX Reg Copy	14083.83	12042.45	13408.31
SSE PAlign	18731.92	14314.34	16719.97
SSE2 PAlign	5833.93	14289.88	16757.55
Memory Latency(ns)			
4 Bytes Stride	3 cycles 1.02 ns	1.67 ns	1.13 ns

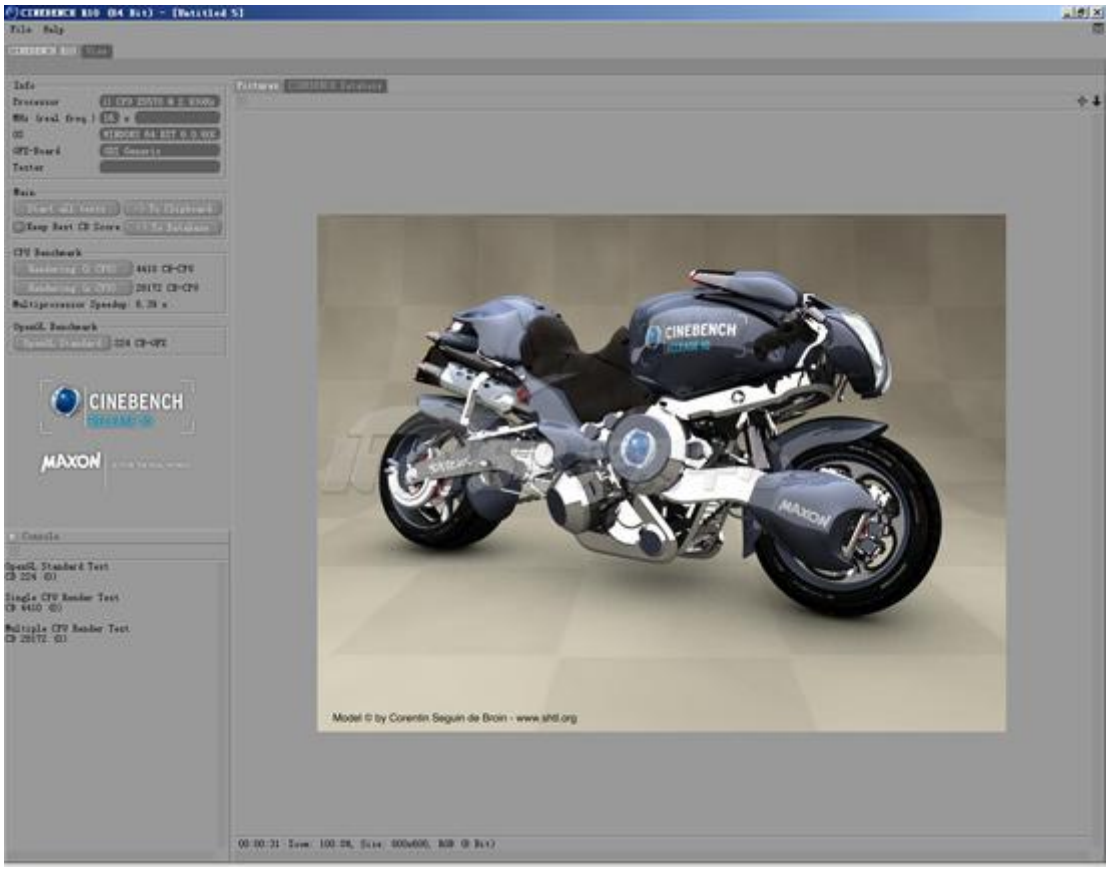
16 Bytes Stride	5 cycles 1.70 ns	5.00 ns	4.89 ns
64 Bytes Stride	22 cycles 7.50 ns	20.00 ns	19.17 ns
256 Bytes Stride	102 cycles 34.77 ns	34.58 ns	59.77 ns
512 Bytes Stride	117 cycles 39.88 ns	81.24 ns	68.04 ns
Memory Algorithm Bandwidth(MB/s)			
Compiler	9210.17	2872.77	3178.45
REP MOVSD	10116.61	2887.02	3220.23
ALU Reg Copy	8156.00	2654.29	2789.34
MMX Reg Copy	9306.18	2943.85	2972.91
MMX Reg 3dNow	-	6631.75	-
MMX Reg SSE	8781.26	6672.76	3978.53
SSE PAlign	8580.24	5765.46	4128.59
SSE PAlign SSE	9524.07	6611.10	4390.48
SSE2 PAlign	8560.83	5766.87	4326.42
SSE2 PAlign SSE	9555.13	6612.42	4441.71
MMX Block 4kb	7743.82	4450.46	4063.30
MMX Block 16kb	8321.35	4677.49	4479.88
SSE Block 4kb	7890.10	4441.71	4074.79
SSE Block 16kb	8355.86	4681.34	4485.09

基本上，与处理器结合最紧密的 L1，或 L2（在有 L3 的情况下）的延迟总是跟处理器频率密集相关的，从总体测试结果来看，Nehalem-EP Xeon X5570 全面强于基准平台，不过有两项数值很奇怪：SSE2 PAlign 的 L1 测试和 L2 测试，这个数值明显不正常。

第 45 页：CineBench R10 性能测试

CineBench R10

CineBench 是基于 Cinem4D 工业三维设计软件引擎的测试软件，用来测试对象在进行三维设计时的性能，它可以同时测试处理器子系统、内存子系统以及显示子系统，我们的平台偏向于服务器多一些，因此就只有前两个的成绩具有意义。和大多数工业设计软件一样，CineBench 可以完善地支持多核/多处理器，它的显示子系统测试基于 OpenGL。



Nehalem-EP/Gainestown Xeon X5570 测试成绩

CineBench R10			
处理器	双路 Intel Gainestown Xeon X5570	双路 AMD Shanghai Operton 2378	双路 Intel Harpertown Xeon E5430
显卡	-	-	-
CPU Benchmark			
Rendering (1 CPU)	4410 CB-CPU	1797 CB-CPU	2931 CB-CPU
Rendering (x CPU)	28172 CB-CPU	10734 CB-CPU	16806 CB-CPU
Multiprocessor Speedup	6.39x	5.97x	5.73x
OpenGL Benchmark			
OpenGL Standard	224 CB-GFX	98 CB-GFX	176 CB-GFX

Intel Nehalem-EP/Gainestown Xeon X5570 测试成绩对比

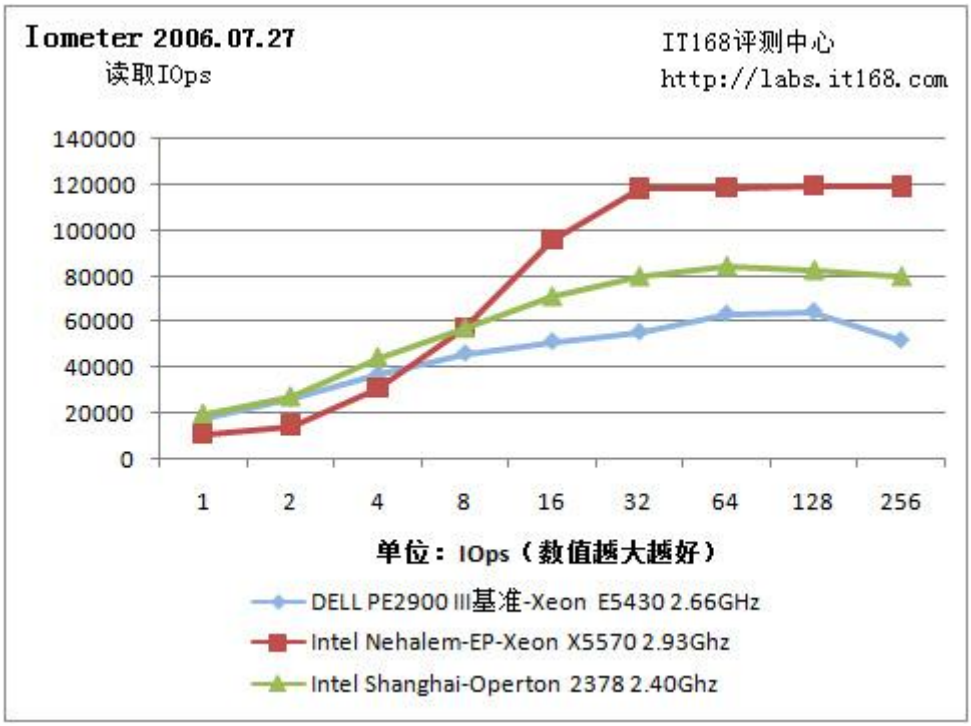
单处理器的渲染性能，Xeon X5570 要比 Xeon E5430 要高 50.5%，频率上要高 10.2%，架构提升很明显。

在多处理器的渲染测试中，X5570 性能要高 67.6%，多处理器加速比为 6.39x。

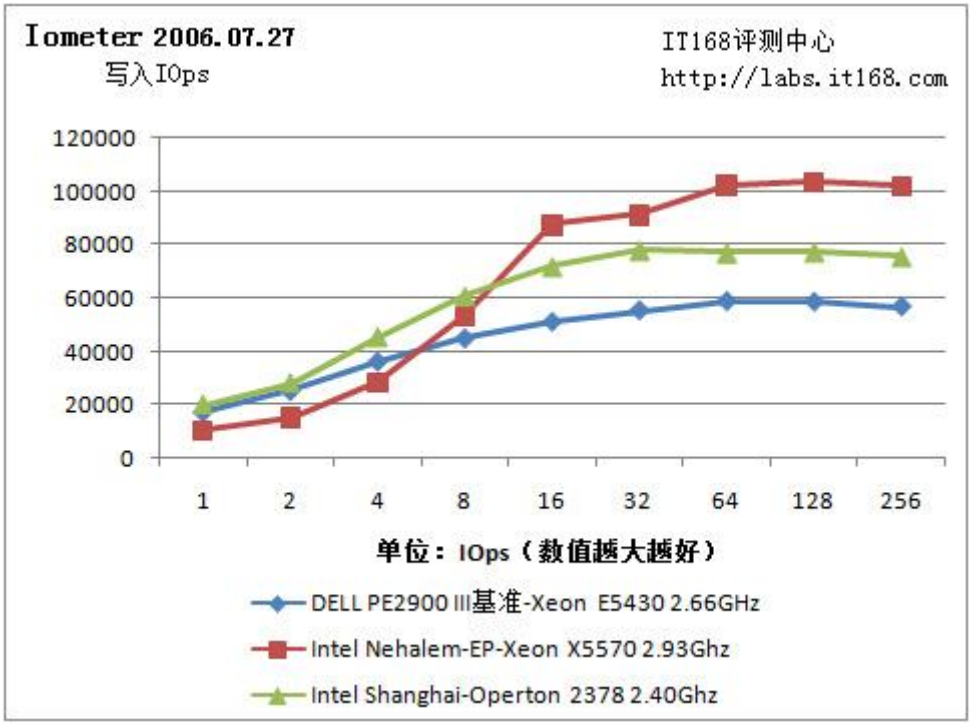
第 46 页：Iometer 磁盘子系统性能测试

Iometer 2006.07.27

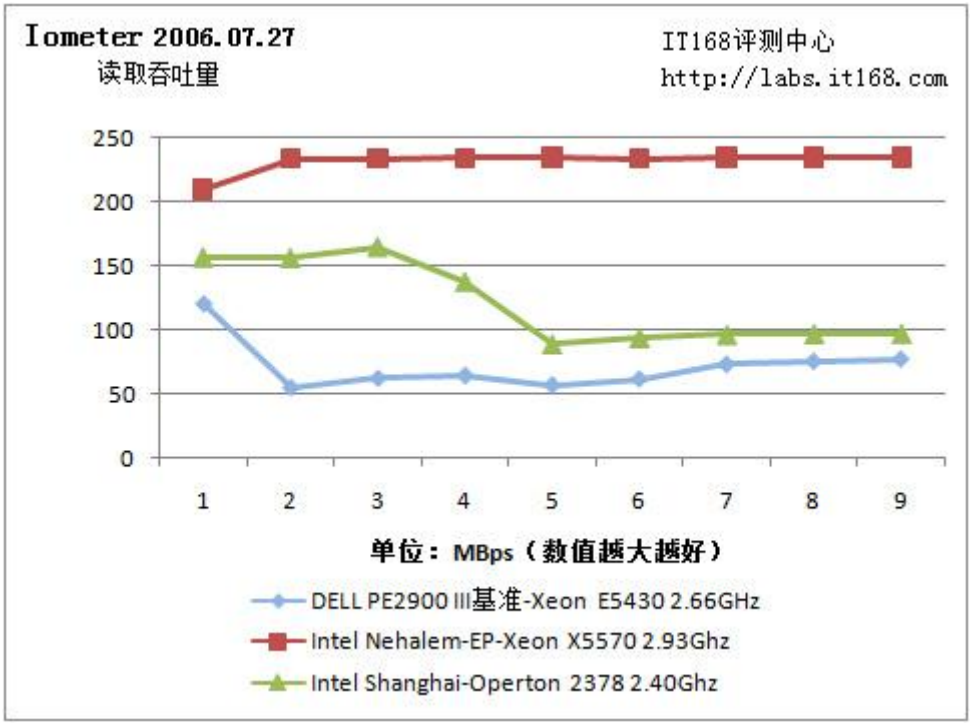
我们的基准服务器采用了三块 15000RPM 的 Seagate Cheetah 15K.5 硬盘。Nehalem-EP 测试样机则是用两块 7200RPM Seagate Barracuda 7200.11。基准平台使用了 LSI MegaRAID SAS 8408E 硬件阵列卡组建了 RAID 5 阵列，而测试样机使用了集成的 LSI Embedded MegaRAID SAS 阵列卡。显而易见，Nehalem-EP 测试样机的磁盘子系统比较糟糕。



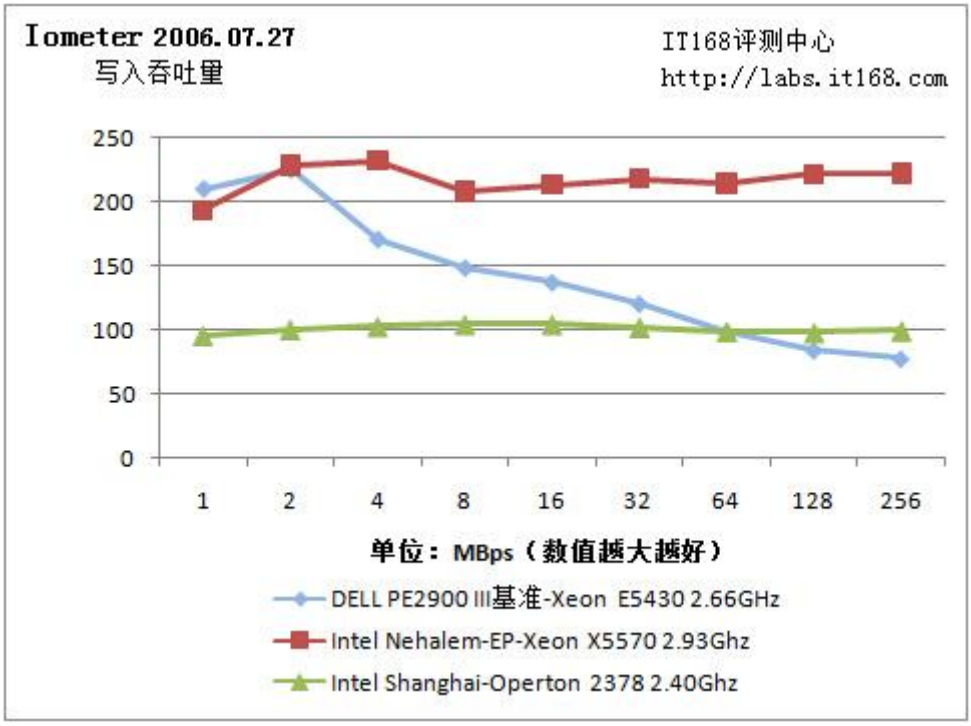
IO 读


Iometer 2006.07.27
写入IOps
IT168评测中心
http://labs.it168.com
120000
100000
80000
60000
40000
20000
0
1
2
4
8
16
32
64
128
256
单位：IOps（数值越大越好）
DELL PE2900 III基准-Xeon E5430 2.66GHz
Intel Nehalem-EP-Xeon X5570 2.93Ghz
Intel Shanghai-Operton 2378 2.40Ghz

IO 写


Iometer 2006.07.27
读取吞吐量
IT168评测中心
http://labs.it168.com
250
200
150
100
50
0
1
2
3
4
5
6
7
8
9
单位：MBps（数值越大越好）
DELL PE2900 III基准-Xeon E5430 2.66GHz
Intel Nehalem-EP-Xeon X5570 2.93Ghz
Intel Shanghai-Operton 2378 2.40Ghz

读吞吐量



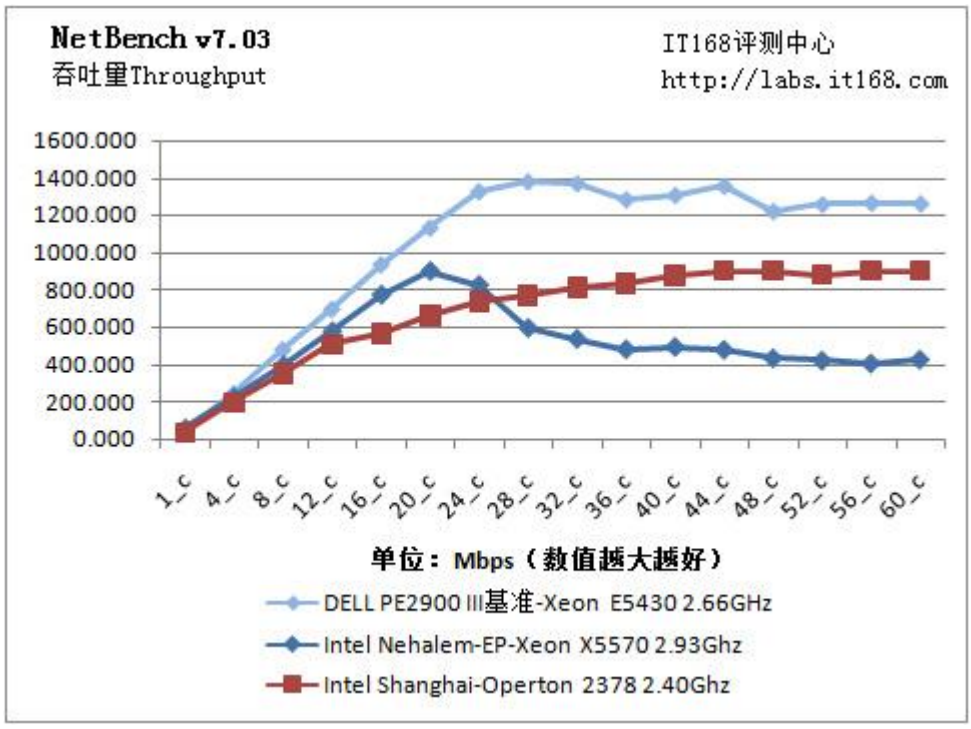
写吞吐量

由于是软阵列，阵列缓存由驱动在主内存中维护，因此 512B 连续读取 IOps 和连续吞吐量都很不错，当然……实际应用是另一回事。

第 47 页：NetBench 文件服务器性能测试

NetBench v7.03

NetBench 7.03 Ent_dm.tst 测试脚本模拟的是企业级文件服务器应用，它不但要求被测服务器的磁盘子系统可以提供足够的吞吐量，还需要其具有较高的 IO 处理能力，并且需要较为平衡的读取能力和写入能力。



NetBench 性能测试

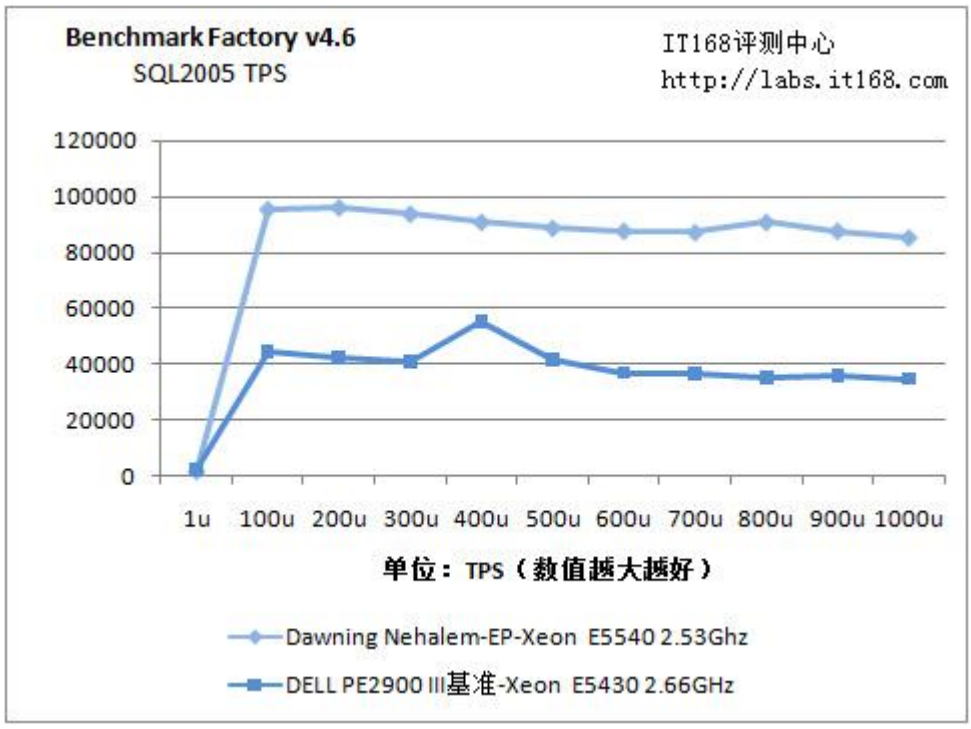
由于是 SATA 软阵列——它们的曲线都表现出类似于正态分布一样：在某处具有一个波峰，两侧则逐渐下滑。Nehalem-EP 测试样机的峰值吞吐量在 20 台测试客户机时达到，为 850Mbps，此后随着客户端的增加，滑落到 400Mbps 附近。基准平台属于硬件阵列，Shanghai 平台属于 SAS Host-RAID 半软半硬阵列。关于 NetBench 性能与处理器、内存、磁盘的关系可以看这里《评测机密：文件服务器性能提升 N 大要义》。

第 48 页：Benchmark Factory 数据库性能测试

Benchmark Factory 4.6

我们在被测服务器上安装了 Microsoft SQL 2005 SP1，按照测试要求建立了数据库。BF 在测试之前会在数据库中生成 9 个表，其中包括 4 个 500 万行的表格，每行包括 100 字节的数据，因此每个表格容量大约是 476MB，整个数据库容量为 1.86GB。我们用 60 个客户端模拟 1000 个用户，在这个数据库中进行查询、添加、删除、修改等操作。

由于时间紧迫，在测试 X5570 的同时，我们也对另一台 E5540 Nehalem-EP 进行了数据库测试。



SQL2005 数据库性能测试

数据库测试是一个综合性的测试，在较少客户端的时候，其性能依赖于处理器以及内存系统，在较多客户端的时候，则开始依赖于磁盘子系统。在这个测试里面，Nehalem-EP 的三个优势都得以完全发挥，最终成绩非常惊人：在频率更低的情况下，平均 TPS（每秒交易数）要高 114%（90557.2 对 40397.217），提升超过了一倍以上。峰值 TPS 是 96264.5。Nehalem 真是理想的数据库平台。

第 49 页：超线程能力对比测试：SiSoftware Sandra

为了体现出超线程对系统性能的影响，我们特地在另一台 Nehalem-EP 平台上作了打开/关闭超线程的测试。

SiSoftware Sandra Pro Business 2009		
测试对象	Intel Nehalem-EP 双路 Intel Gainestown Xeon X5570 2.93GHz	Intel Nehalem-EP 双路 Intel Gainestown Xeon X5570 2.93GHz 无超线程
Processor Arithmetic Benchmark 处理器架构测试		
Dhrystone ALU	142977MIPS	147034MIPS
Dhrystone ALU vs SPEED	48.75MIPS/MHz	50.13MIPS/MHz
Whetstone iSSE3	124035MFLOPS	80990MFLOPS

Dhrystone iSSE3 vs SPEED	42.29MFLOPS/MHz	27.61MFLOPS/MHz
Processor Multi-Media Benchmark 处理器多媒体测试		
Multi-Media Int x16 iSSE4.1	296.85MPixel/s	260.47MPixel/s
Multi-Media Int x16 iSSE4.1 vs SPEED	101.21MPixel/s/MHz	88.81MPixel/s/MHz
Multi-Media Float x8 iSSE2	228.24MPixel/s	196.13MPixel/s
Multi-Media Float x8 iSSE2 vs SPEED	77.82kPixels/s/MHz	66.87kPixels/s/MHz
Multi-Media Double x4 iSSE2	125.88MPixel/s	102.33MPixel/s
Multi-Media Double x4 iSSE2 vs SPEED	42.92kPixels/s/MHz	34.89kPixels/s/MHz
Multi-Core Efficiency Benchmark		
Inter-Core Bandwidth	75.61GB/s	32.66GB/s
Inter-Core Bandwidth vs SPEED	26.40MB/s/MHz	11.40MB/s/MHz
Inter-Core Latency (越小越好)	16ns	48ns
Inter-Core Latency vs SPEED (越小越好)	0.01ns/MHz	0.02ns/MHz
Memory Bandwidth Benchmark 内存带宽测试		
Int Buff'd iSSE2 Memory Bandwidth	16.93GB/s	38.71GB/s
Float Buff'd iSSE2 Memory Bandwidth	16.90GB/s	38.52GB/s
Memory Latency Benchmark 内存延迟测试		
Memory(Random Access) Latency (越小越好)	81ns	78ns
Speed Factor (越小越好)	61.40	61.60
Internal Data Cache	4clocks	4clocks
L2 On-board Cache	10clocks	9clocks
L3 On-board Cache	48clocks	46clocks
Cache and Memory Benchmark 缓存及内存测试		

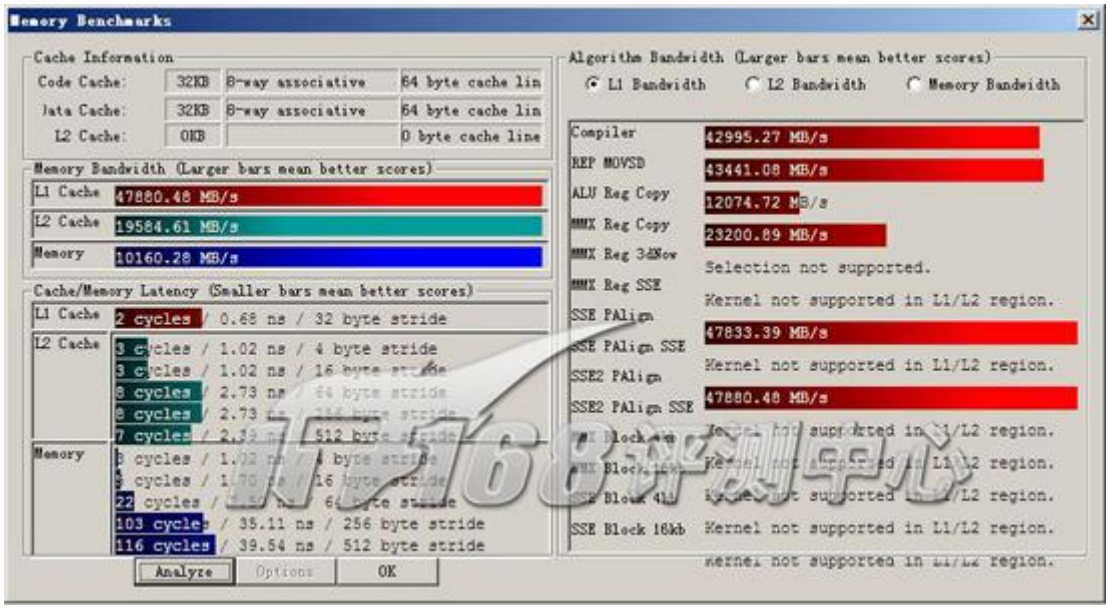
Cache/Memory Bandwidth	143.24GB/s	141.40GB/s
Cache/Memory Bandwidth vs SPEED	50.01MB/s/MHz	49.37MB/s/MHz
Speed Factor (越小越好)	20.90	21.90
Internal Data Cache	448.46GB/s	450.77GB/s
L2 On-board Cache	421.42GB/s	425.31GB/s
.NET Arithmetic Benchmark .NET 架构测试		
Dhrystone .NET	32904MIPS	31208MIPS
Dhrystone .NET vs SPEED	11.22MIPS/MHz	10.64MIPS/MHz
Whetstone .NET	78286MFLOPS	55638MFLOPS
Whetstone .NET vs SPEED	26.69MFLOPS/MHz	18.97MFLOPS/MHz
.NET Multi-Media Benchmark .NET 多媒体测试		
Multi-Media Int x1 .NET	62.28MPixel/s	55.60MPixel/s
Multi-Media Int x1 .NET vs SPEED	21.23kPixels/s/MHz	18.96kPixels/s/MHz
Multi-Media Float x1 .NET	26.19MPixel/s	15.95MPixel/s
Multi-Media Float x1 .NET vs SPEED	8.93kPixels/s/MHz	5.44kPixels/s/MHz
Multi-Media Double x1 .NET	51.45MPixel/s	29.85MPixel/s
Multi-Media Double x1 .NET vs SPEED	17.54kPixels/s/MHz	10.18kPixels/s/MHz

SiSoftware Sandra 对比，用蓝色标出了性能特出的项目

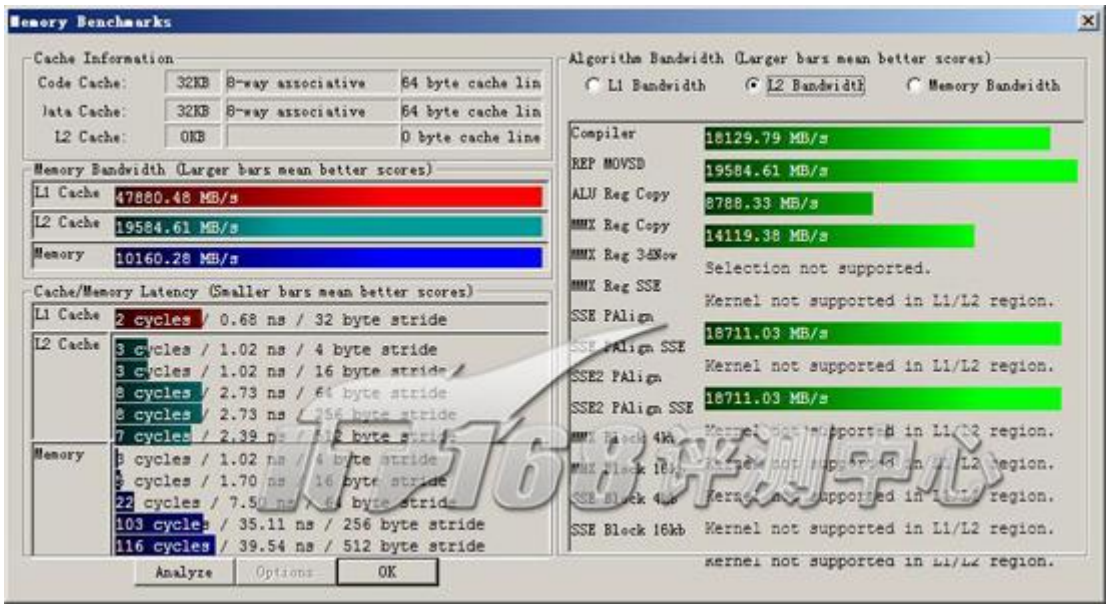
关闭超线程之后，计算能力普遍下降，对 L1/L2 缓存的压力降低（因而 L1/L2 缓存带宽也就略为上升了），然而内存吞吐量也下降了，这表明每 CPU 三通道 DDR3-1333 对 8 个逻辑处理器是足够的。

只有极少数的项目中，关闭超线程获得了更好的测试成绩。Nehalem-EP 的超线程比起 Pentium 4 时代有了不少的改进，你不应该将其关闭。

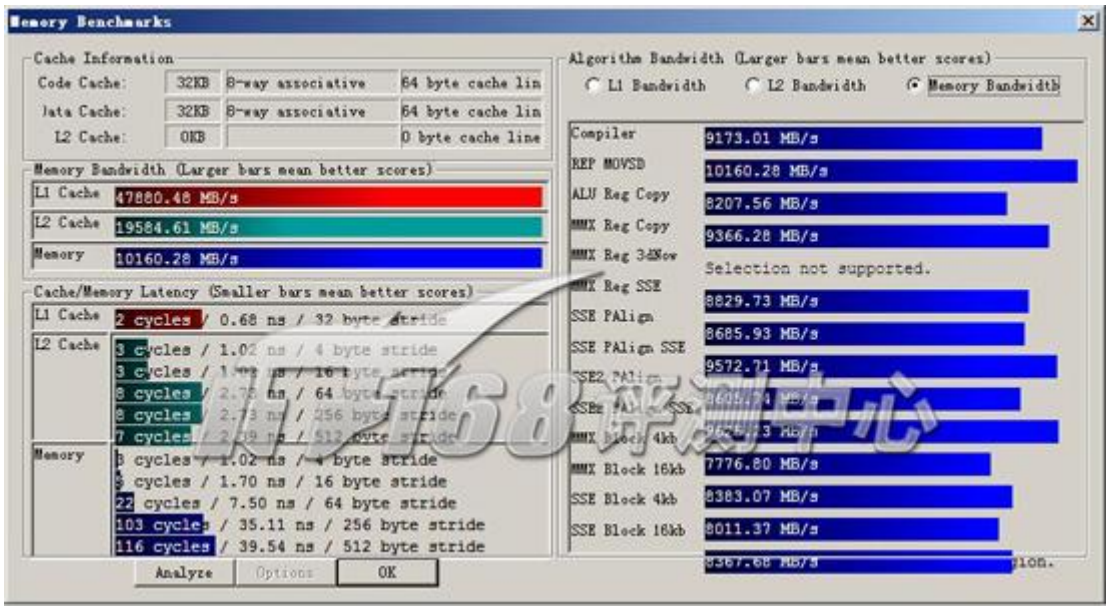
第 50 页：超线程能力对比测试：ScienceMark



Intel Nehalem-EP Xeon X5570 2.93GHz without SMT



Intel Nehalem-EP Xeon X5570 2.93GHz without SMT



Intel Nehalem-EP Xeon X5570 2.93GHz without SMT

ScienceMark Membench		
厂商	Intel	Intel
产品型号	Nehalem-EP Intel Gainestown Xeon X5570 2.93GHz	Nehalem-EP Intel Gainestown Xeon X5570 2.93GHz 无超线程
内存技术参数	4GB R-ECC DDR3-1333 SDRAM x6	4GB R-ECC DDR3-1333 SDRAM x6
L1 带宽(MB/s)	47880.48	47880.48
L2 带宽(MB/s)	19604.64	19584.61
内存带宽(MB/s)	10116.61	10160.28
L1 Cache Latency(ns)		
32 Bytes Stride	2 cycles 0.68 ns	2 cycles 0.68 ns
L1 Algorithm Bandwidth(MB/s)		
Compiler	43072.25	42995.27

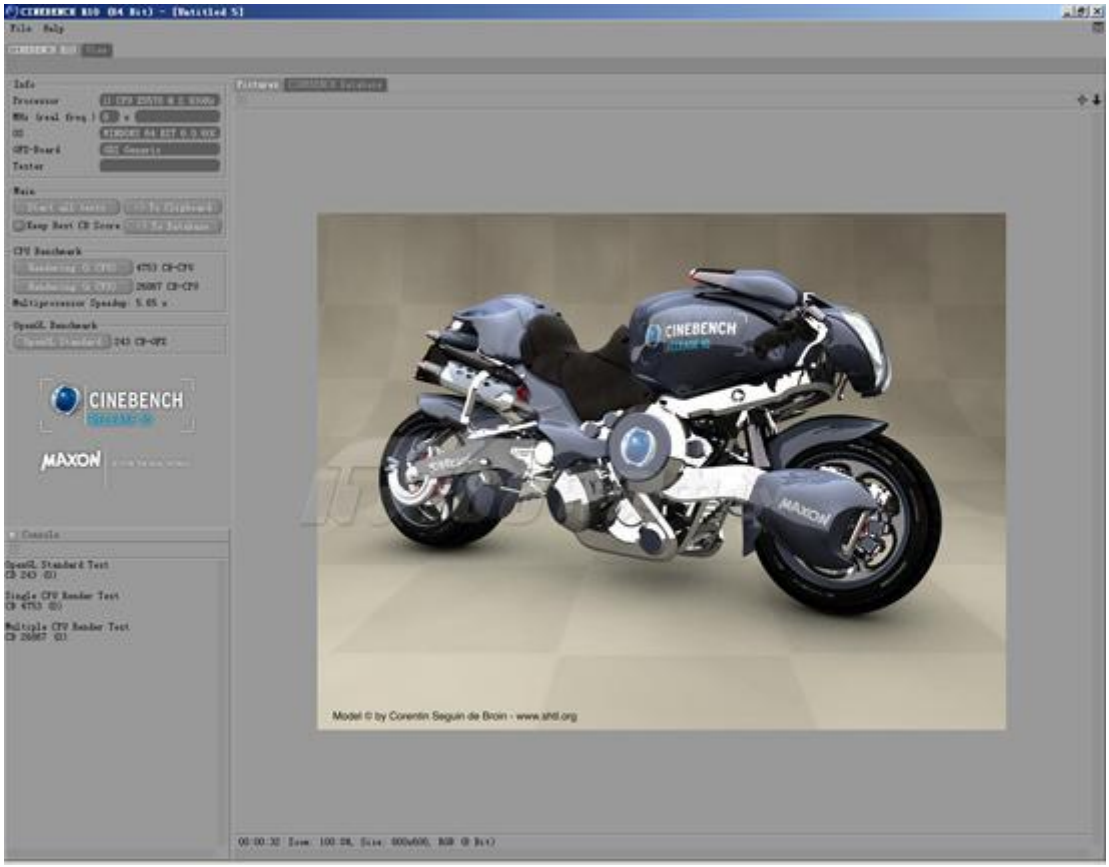
REP MOVSD	43467.25	43441.08
ALU Reg Copy	11949.09	12074.72
MMX Reg Copy	22537.36	23200.89
SSE PAlign	47773.13	47833.39
SSE2 PAlign	47880.48	47880.48
L2 Cache Latency(ns)		
4 Bytes Stride	3 cycles 1.02ns	3 cycles 1.02 ns
16 Bytes Stride	3 cycles 1.02ns	3 cycles 1.02 ns
64 Bytes Stride	8 cycles 2.73ns	8 cycles 2.73 ns
256 Bytes Stride	8 cycles 2.73ns	8 cycles 2.73s
512 Bytes Stride	7 cycles 2.39 ns	7 cycles 2.39 ns
L2 Algorithm Bandwidth(MB/s)		
Compiler	18039.64	18129.79
REP MOVSD	19604.64	19584.61
ALU Reg Copy	8788.90	8788.33
MMX Reg Copy	14083.83	14119.38

SSE PAlign	18731.92	18711.03
SSE2 PAlign	5833.93	18711.03
Memory Latency(ns)		
4 Bytes Stride	3 cycles 1.02	3 cycles 1.02
16 Bytes Stride	5 cycles 1.70	5 cycles 1.70
64 Bytes Stride	22 cycles 7.50	22 cycles 7.50
256 Bytes Stride	102 cycles 34.77	103 cycles 35.11
512 Bytes Stride	117 cycles 39.88	116 cycles 39.54
Memory Algorithm Bandwidth(MB/s)		
Compiler	9210.17	9173.01
REP MOVSD	10116.61	10160.28
ALU Reg Copy	8156.00	8207.56
MMX Reg Copy	9306.18	9366.28
MMX Reg 3dNow	-	-
MMX Reg SSE	8781.26	8829.73
SSE PAlign	8580.24	8685.93

SSE PAlign SSE	9524.07	9572.71
SSE2 PAlign	8560.83	8605.74
SSE2 PAlign SSE	9555.13	9625.23
MMX Block 4kb	7743.82	7776.80
MMX Block 16kb	8321.35	8383.07
SSE Block 4kb	7890.10	8011.37
SSE Block 16kb	8355.86	8367.68

关闭超线程之后，缓存和内存性能着微弱的提升，你不需要关闭超线程。

第 51 页：超线程能力对比测试：CineBench



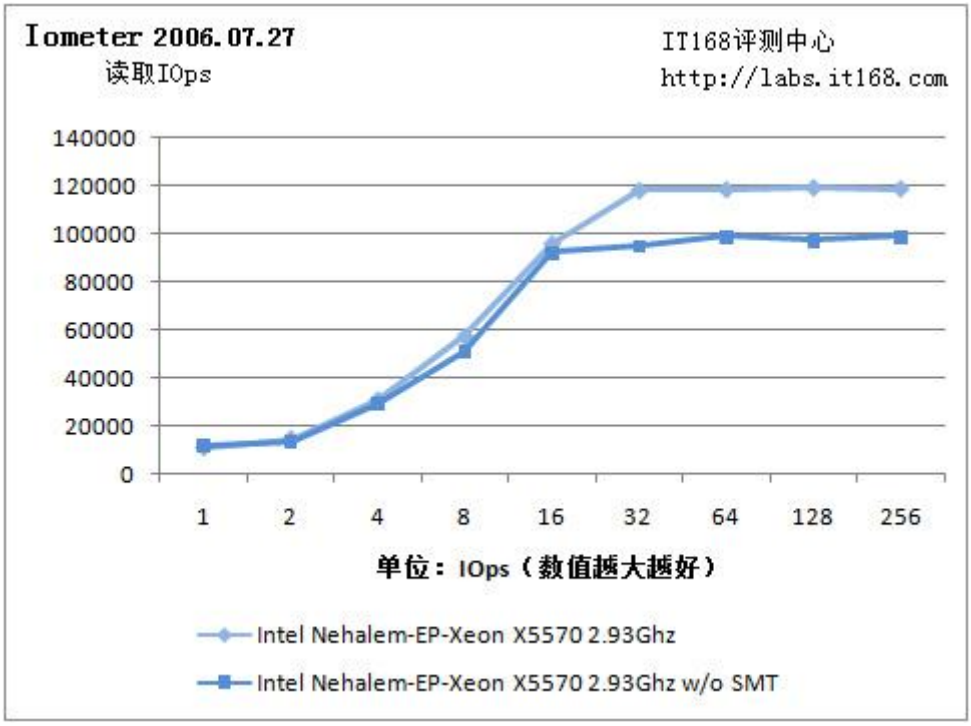
CineBench R10		
处理器	双路 Intel Gainestown	双路 Intel Gainestown

	Xeon X5570	Xeon X5570 无超线程
显卡	-	-
CPU Benchmark		
Rendering (1 CPU)	4410 CB-CPU	4753 CB-CPU
Rendering (x CPU)	28172 CB-CPU	26867 CB-CPU
Multiprocessor Speedup	6.39x	5.65x
OpenGL Benchmark		
OpenGL Standard	224 CB-GFX	243 CB-GFX

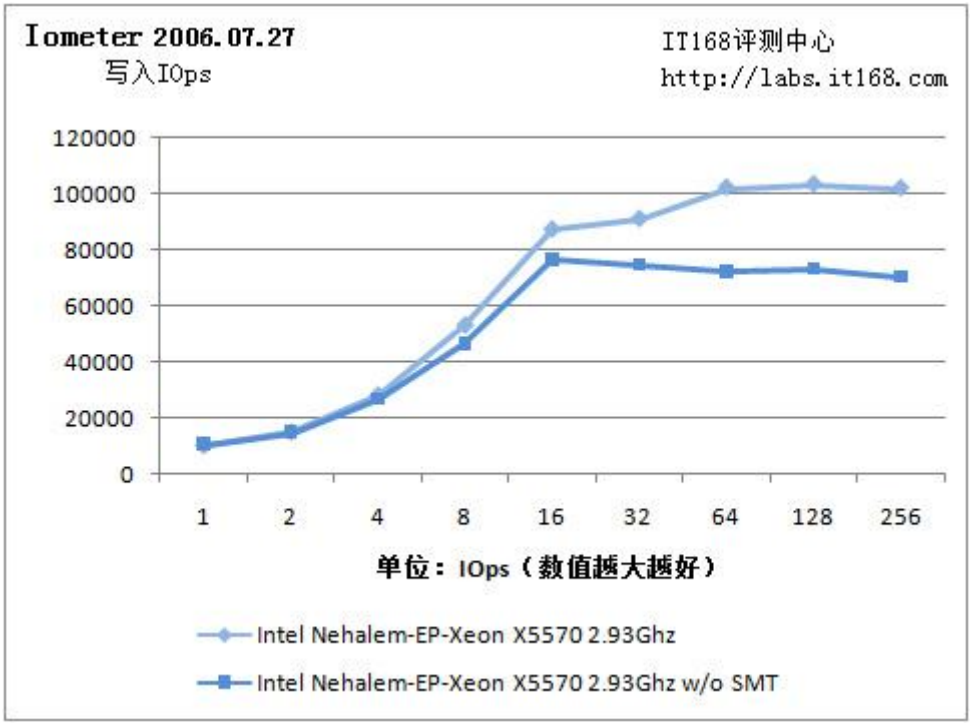
Intel Nehalem-EP/Gainestown Xeon X5570 超线程能力对比测试

没有超线程，单处理器渲染性能上升了 7.78%，不过，多处理器渲染性能下降了 4.63%。在一般情况下，你仍然没有必要关闭超线程。

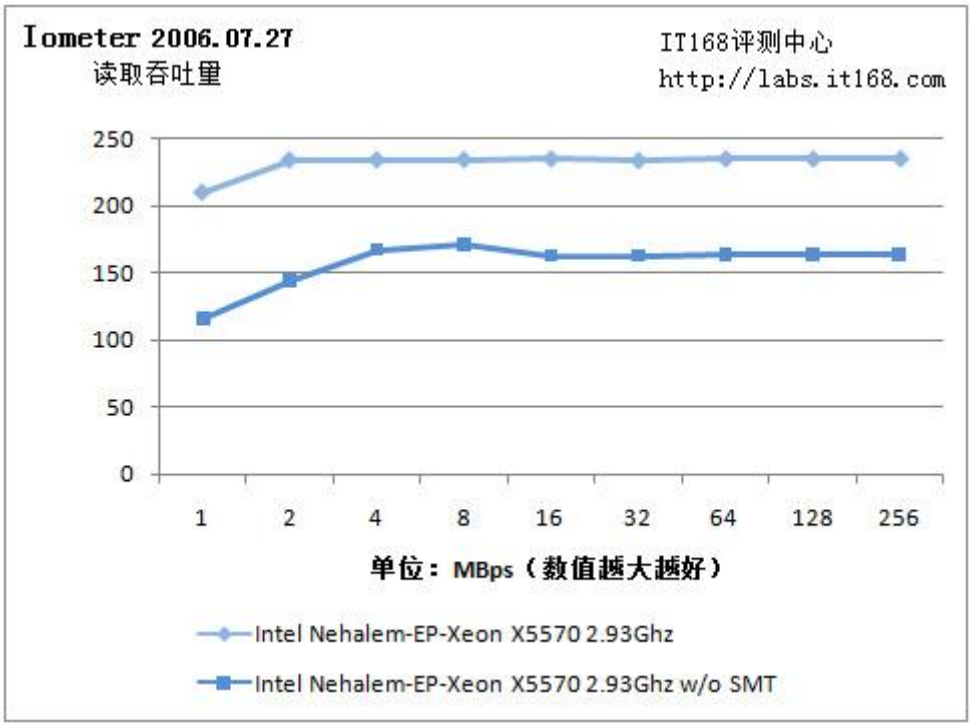
第 52 页：超线程能力对比测试：Iometer



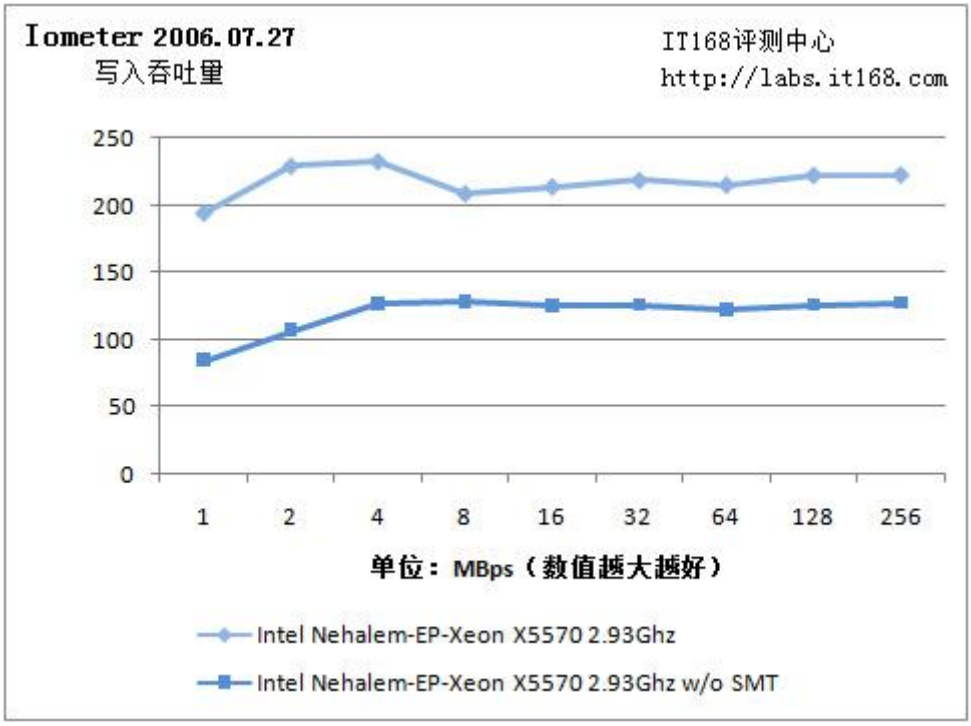
IO 读


Iometer 2006.07.27
写入IOps
IT168评测中心
http://labs.it168.com
120000
100000
80000
60000
40000
20000
0
1
2
4
8
16
32
64
128
256
单位：IOps（数值越大越好）
Intel Nehalem-EP-Xeon X5570 2.93Ghz
Intel Nehalem-EP-Xeon X5570 2.93Ghz w/o SMT

IO 写


Iometer 2006.07.27
读取吞吐量
IT168评测中心
http://labs.it168.com
250
200
150
100
50
0
1
2
4
8
16
32
64
128
256
单位：MBps（数值越大越好）
Intel Nehalem-EP-Xeon X5570 2.93Ghz
Intel Nehalem-EP-Xeon X5570 2.93Ghz w/o SMT

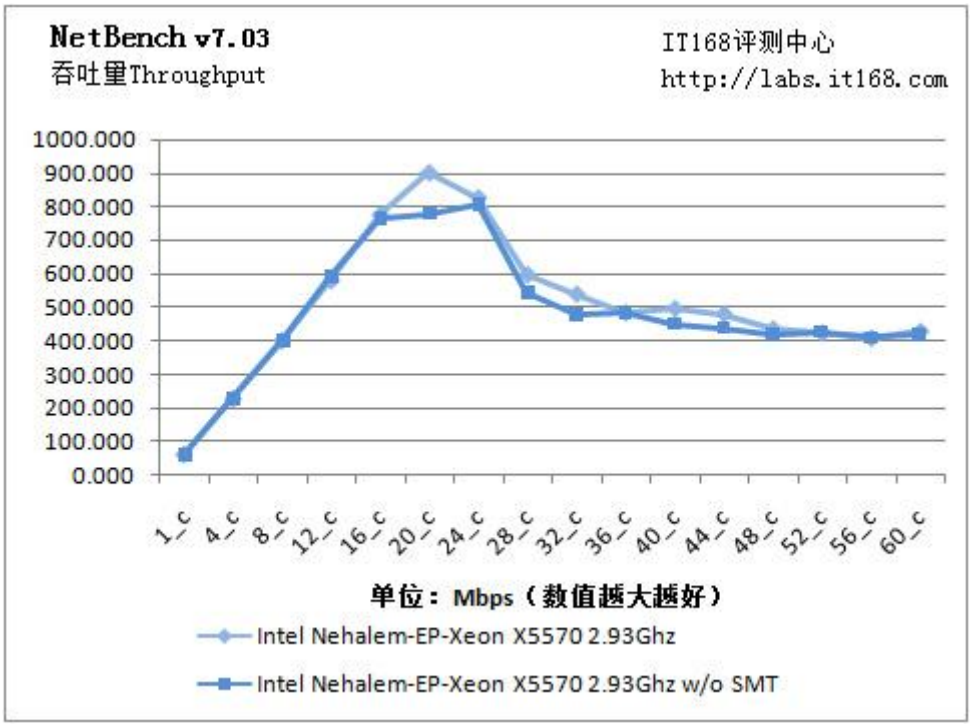
读吞吐量



写吞吐量

这台 Nehalem-EP 测试平台的磁盘子系统是一个软阵列，因此性能和处理器子系统和内存子系统相关，关闭超线程之后，软阵列的整体性能下降。因为沉重的磁盘操作负荷会占用较多的处理器资源，因此强烈建议打开超线程来解放处理器以进行其它计算。

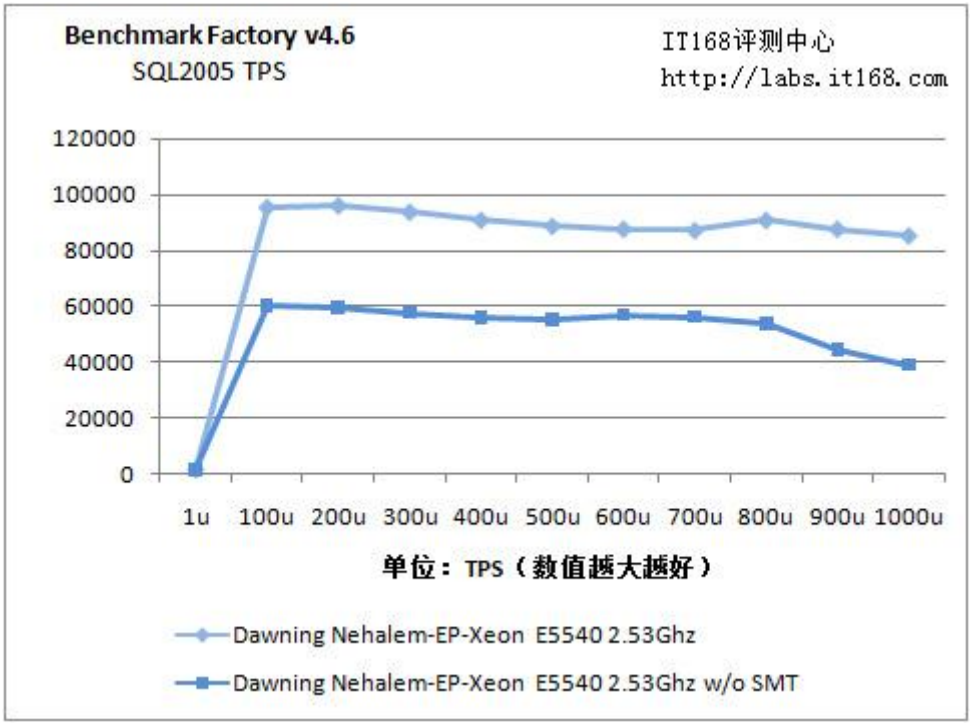
第 53 页：超线程能力对比测试：NetBench



with SMT vs withou SMT

关闭超线程成绩低一些，建议打开超线程。

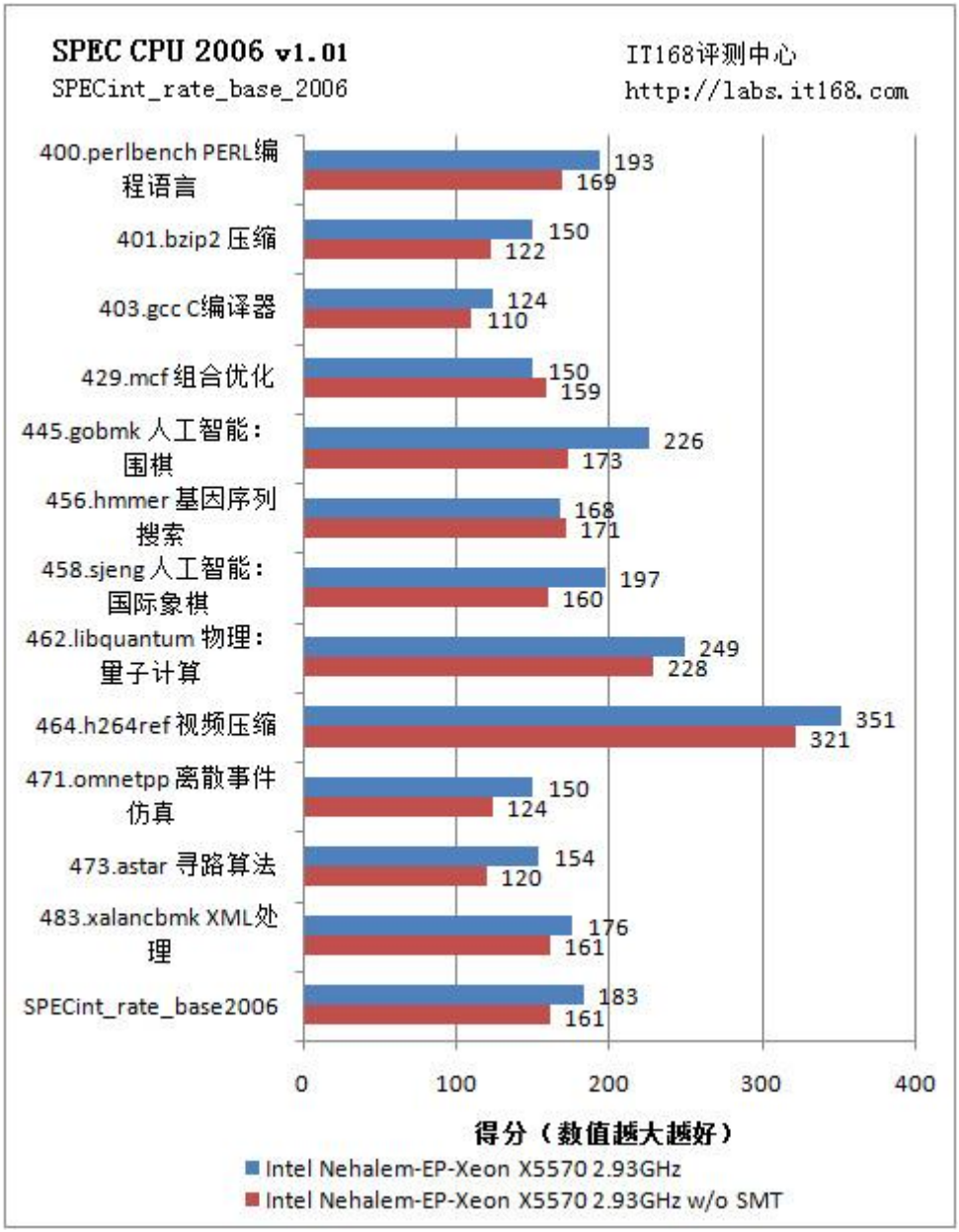
第 54 页：超线程能力对比测试：Benchmark Factory



with SMT vs without SMT

关闭 SMT 之后，数据库性能立降，降幅达 40.4%——你不应该关闭 SMT。

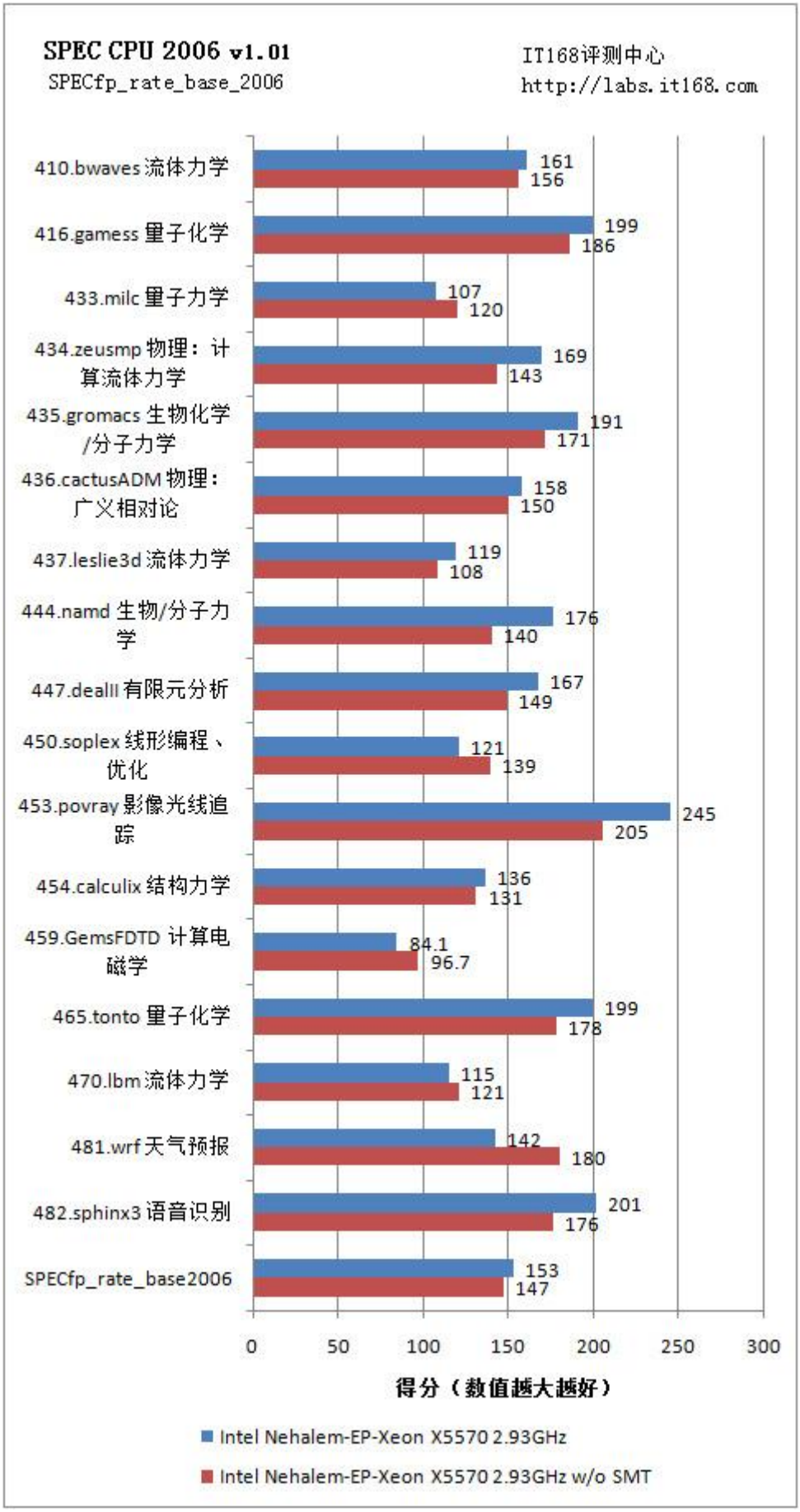
第 55 页：超线程能力对比测试：SPEC CPU 2006 整数



Intel Nehalem-EP/Gainestown Xeon X5570 SPEC CPU 2006 整数运算性能：with SMT vs without SMT

关闭超线程之后，Nehalem-EP 平台的测试成绩下降了 12.0%，非常明显。超线程对大部分整数测试项目都有着正面的提升作用，除了两个项目：429.mcf 组合优化（关闭后提升 1.06%）、456.hmmer 基因序列搜索（关闭后提升 4.05%），不算太明显，因此可以认为，在整数运算中，超线程可以很明显地提升处理器效能，你最好打开超线程技术。

第 56 页：超线程能力对比测试：SPEC CPU 2006 浮点


SPEC CPU 2006 v1.01
SPECfp_rate_base_2006
IT168评测中心
http://labs.it168.com
410.bwaves 流体力学
161
156
416.gamess 量子化学
199
186
433.milc 量子力学
107
120
434.zeusmp 物理：计算流体力学
169
143
435.gromacs 生物化学/分子力学
191
171
436.cactusADM 物理：广义相对论
158
150
437.leslie3d 流体力学
119
108
444.namd 生物/分子力学
176
140
447.dealII 有限元分析
167
149
450.soplex 线形编程、优化
121
139
453.povray 影像光线追踪
245
205
454.calculix 结构力学
136
131
459.GemsFDTD 计算电磁学
84.1
96.7
465.tonto 量子化学
199
178
470.lbm 流体力学
115
121
481.wrf 天气预报
142
180
482.sphinx3 语音识别
201
176
SPECfp_rate_base2006
153
147
0
50
100
150
200
250
300
得分（数值越大越好）
Intel Nehalem-EP-Xeon X5570 2.93GHz
Intel Nehalem-EP-Xeon X5570 2.93GHz w/o SMT

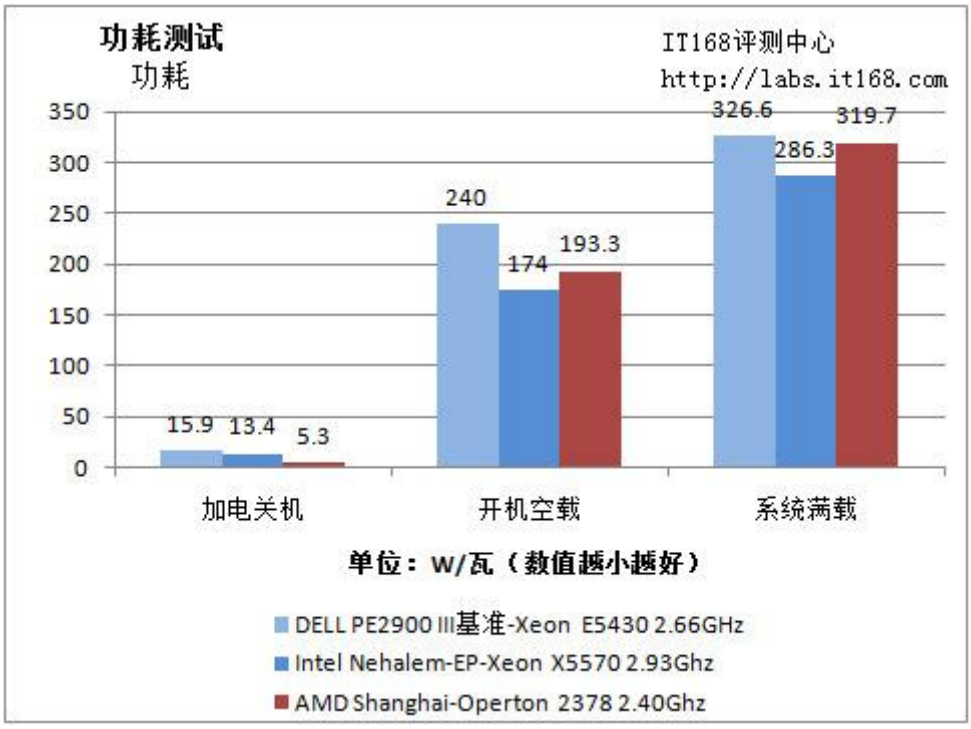
Intel Nehalem-EP/Gainestown Xeon X5570 SPEC CPU 2006 浮点运算性能：with SMT vs without SMT

关闭超线程之后，性能下降了 3.93%，并不算非常大——大部分测试成绩都下降了，少数项目在关闭超线程之后性能不降反升，这几个项目是：433.milc 量子力学（关闭后提升 12.1%）、450.soplex 线形编程、优化（关闭后提升 14.9%）、459.GemsFDTD 计算电磁学（关闭后提升 15.0%）、470.lbm 流体力学（关闭后提升 5.22%）、481.wrf 天气预报（关闭后提升 26.8%）共 5 项，其中 481.wrf 天气预报影响非常巨大。进行这些项目相关工作的用户在配置 Nehalem-EP 平台的时候最好先仔细测试一下。其他的浮点运算用户一般都不必关闭超线程。

第 57 页：Nehalem-EP 平台功耗测试

我们利用 UNI-T UT71E 智能数字万用表和相配套的软件对被测服务器在几种不同的状态下的功耗进行了监测，主要包括如下项目：

- P1：连接电源但不开机状态
- P2：系统启动完毕，5 分钟内无动作，但不休眠
- P3：系统启动完毕，处理器满载、磁盘以最大吞吐量工作



功耗：Intel Nehalem-EP 平台与 AMD Shanghai、DELL PE2900 III 平台

配置上，Nehalem-EP 官方评测样机具有 24GB 的内存，不过是 DDR3，Harpertown Xeon 则只有 16GB，不过是大发热量的 FBD DDR2。Harpertown Xeon 平台的硬盘要多一个，并且 Nehalem-EP 平台是 7200RPM 的桌面 SATA 硬盘。此外，Nehalem-EP 平台的机架式设计配置了 7 个暴力散热风扇，总体来看其功耗应该更高一些。上表仅作参考：Nehalem-EP 在闲置时功耗要比基准平台低不少，Nehalem 的长沟道晶体管、Power Control Unit、Power Gate 确实

发挥了作用。在满负荷情况下，Nehalem-EP 平台也仍然比基准平台更省电——同时性能更高。

参考的 AMD Shanghai 平台功耗要高一些。

第 58 页：IT168 评测中心观点

【IT168 评测中心】凭借着崭新的直联架构——集成内存控制器和双 QPI 总线，再配合超线程技术，Nehalem-EP 的性能比起其上一代有了一个大的飞跃，同频率下处理器密集型和内存密集型运算的性能提升达到了一倍以上。

Nehalem-EP：Xeon X5570，主频 2.93GHz，QPI 频率 3.2GHz

配合 Nehalem-EP 使用的 Intel Tylersburg-EP 芯片

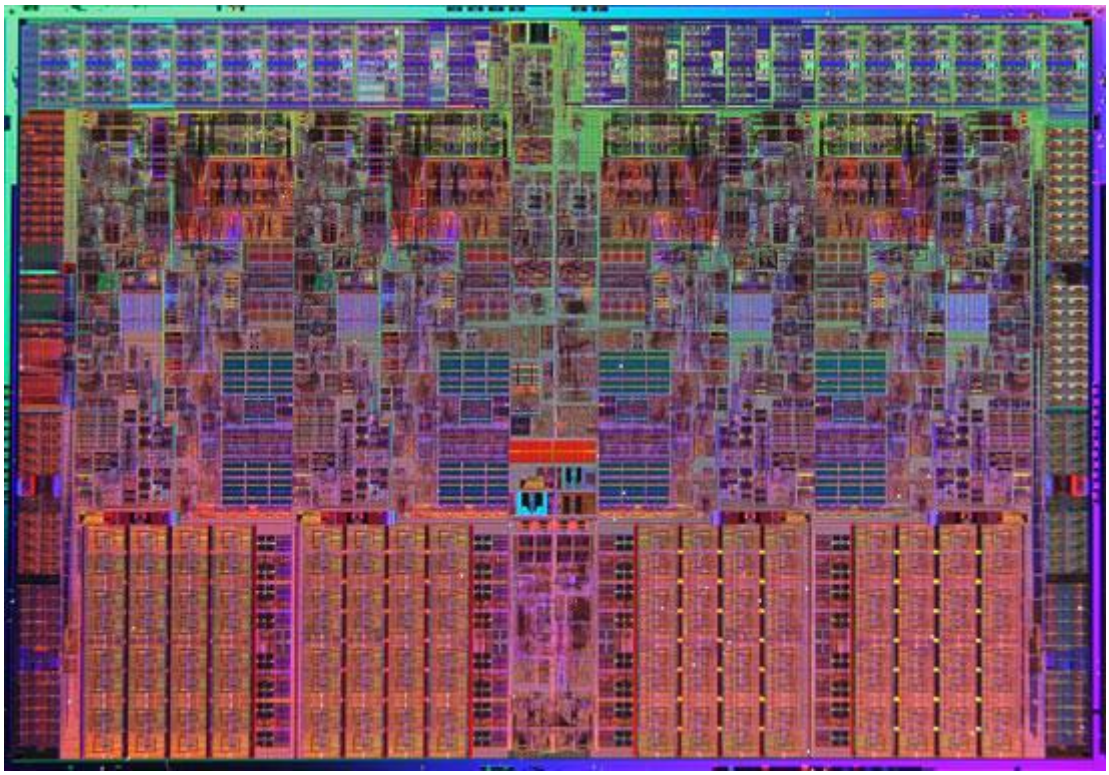
由于处理器指令集架构的缘故，x86 处理器非常依赖于缓存/内存性能，使用集成内存控制器之后，Nehalem-EP 消除了 FSB 总线引起的内存瓶颈，通过每处理器三通道 DDR3，提供了高带宽、低延迟的子系统，极大地提升了性能。

同样，高带宽的 QPI 总线也更有利于多处理器协同工作，虽然在双路系统中表现并不明显，不过，可以预见，在 4 路及 4 路以上市场以及非常多 PCI Express IO 设备的情况下，QPI 总线可以发挥巨大的作用。

超线程技术也是 Nehalem 处理器的要点之一，虽然不是所有的应用中都有正面效果，然而从测试来看，Xeon X5570 的超线程技术对 SPEC CPU 2006 的成绩提升为 13.7%（整数）和 4.08%（浮点），在应用测试，如 SQL 数据库性能测试中，超线程的存在让 Xeon E5540 的性能提升了 67.8%，这是一个巨大的数字。这表明如数据库应用这样的吞吐量计算可以将 Nehalem-EP 的超线程技术发挥到极致。

Intel Nehalem-EP 官方评测样机，配置了双路 Xeon X5570 处理器和 24GB DDR3 内存

比起同频率 Hartertown Xeon，Nehalem-EP 的性能提升在一倍以上，目前在双路 x86 服务器领域，Nehalem-EP 可以说是毫无敌手。



Nehalem-EP 处理器：独孤求败

[直联架构的威力 Nehalem-EP 处理器解析]

[Nehalem-EP 新 Xeon 5500 处理器首度曝光]

透视六核心至强 Dunnington 处理器解析

透视八核心至强 Nehalem-EX 处理器解析

2008 年度评测报告：深入 Nehalem 微架构

性能大幅提升 Core i7 服务器应用测试

再攀性能之巅 Intel 全新酷睿 i7 深度评测

机密揭露：Intel 超线程技术有多少种？

[IDF08]基辛格演讲:Nehalem 集群演示