CNCC 2020 特邀报告

计算机体系结构四个时代的结束和第五个时代的兴起

约翰·汉尼斯（John Hennessy）
斯坦福大学

关键词：计算机体系结构

非常高兴通过视频与大家在 CNCC 大会上见面，一起探讨计算机体系结构和最近我所做的一些工作。我很高兴能有这样一个机会与中国的同行进行交流。真切希望我能够有幸亲自到场，但在当前的情况下，这种线上的形式已经是我们能做到的最佳选择了。

自身经历

我于 1977 年加入斯坦福大学，成为计算机系统方向的一名年轻教授。这一方向在当时还处于发展初期：苹果公司还未成立，微处理器也还是一个相对新鲜的事物。我正是在这样的背景下开始了研究生涯。当时我们观察到 VLSI（超大规模电路集成，Very Large-Scale Integration）成为了一种实现处理器的新方式，这促使我们重新思考应该如何设计计算机。最终，我们在 80 年代初期形成了 RISC（精简指令集计算机，Reduced Instruction Set Computers）的概念，并完成了相关设计和论文发表。坦白地讲，因为我们的研究成果非常显著，我以为工业界会快速采纳这一思路，然而实际上并没有。虽然有几家大公司开展了一些实验性质的项目，但后来都被一一取消了。于是我决定自己创立一家公司来证明这一技术不仅是实际有效的，而且具有很强的吸引力。这就是 MIPS 这家公司的由来。后来它被硅图公司（Silicon Graphics）收购。MIPS 实际上经历了两次上市，书写了一段不同寻常的历史。这是我在硅谷创业的第一段经历。在那之后，我很快和学校的同事一起创立了一家叫 Atheros 的公司。这家公司是 Wi-Fi 领域的先锋之一。早在 Wi-Fi 成为标准之前，我们就率先研发出基于 CMOS 技术的高性能 Wi-Fi 芯片组产品，推动了 Wi-Fi 技术的发展。最后这家公司也成功地被高通（Qualcomm）收购。后来，我回到斯坦福大学，历任工程学院院长、学校教务长，并最终担任了 16 年斯坦福大学校长。

现在，我领导着 Knight-Hennessy 学者项目，这是一个全球范围内的奖学金计划，面向包括中国在内的全世界的学者，帮助他们在完成研究生学业的同时，在课外通过一系列课程和活动提升领导力。在担任校长期间，我发现在全世界的政府、工业界、非营利组织等各个领域中都很缺乏真正强有力的领导者。因此，我们希望通过这一项目的推广，在国际上培养出色的未来领导者，以帮助我们建立一个更美好的世界。

计算机体系结构的四个时代

言归正传，今天在这里我们主要探讨科技。当

时代	时期	工艺技术	主要特点	终结
I	1955—1964： 指令集架构出现以前	晶体管	计算机模型各不相同	统一的指令集架构
II	1964 年至 80 年代： IBM 360 时代	小规模与中规模集成（计算核心，DRAM 存储）	微编程	1. 场效应管（MOS）与集成电路的兴起 2. 指令集架构扩展（减小代码大小）但高级编程语言主导，使指令级并行变难 3. 与 CMOS 相比的成本太高（即使其性能更好）
II.a	1970—1990： 小型机	中规模集成（计算核心，DRAM 存储）	微编程	
II.b	1964—1984： 经典超级计算机	发射极耦合逻辑技术（SRAM 作为存储）	简单的指令集；矢量；无存储层次	
III	1977—2005： 复杂指令集微处理器	大规模与超大规模集成	宽发射，集成；微编程	指令级并行耗尽； 功耗和面积效率变低
III.a	1985—2010： 精简指令集微处理器	超大规模集成（计算核心）	精简指令集（单位功耗或面积下更高吞吐率）	第一个时期：多于五个指令集架构；无半导体代工场；商业化软件 第二个时期：功耗效率变低
IV	2005 年至今： 多核处理器	超大规模集成（多核）	多个核心；复杂与精简指令集市场分化	登纳德缩放终止； 摩尔定律减缓
V	2016 年开始： 领域专用处理器	超大规模集成（加速器）	领域专用架构	未知

图 1　过去 65 年间计算机经历的四个时代

下正是一个重要的转折时刻。当摩尔定律（Moore's Law）、登纳德缩放定律（Dennard Scaling）等一系列半导体规律逐渐失效时，设计面向未来的计算机系统需要我们改变旧有的观念。65 年以来，计算机体系结构以惊人的发展速度历经了四个时代，最终来到了当前第五个时代的开端。这 65 年里的巨大进步使得计算机的性能得到了不可思议的提升。单单在过去的 40 年，微处理器的性能就提高了 10^6 之多，整整快了 100 万倍。这来源于一系列架构层面的创新。第一，系统架构由 8 比特、16 比特变宽到 64 比特；第二，指令级并行度提高了一个数量级；第三，多核开始出现，我们可以在一个芯片上实现多个处理器核，以获得更高性能。同时，在这 65 年中，电路的时钟频率也提高了 1000 余倍，这得益于底层半导体工艺技术的革新。摩尔定律成功预言了晶体管数量的增长，每 18 至 24 个月翻一番；登纳德缩放定律表明每个晶体管的功耗会随着晶体管数量的增长而降低，从而实现在算力提升的同时，单次计算的能耗反而可以下降。不幸的是，这两个重要的规律目前都开始渐渐失效了。

让我们回顾一下计算机系统发展史上的各个时代（见图 1）。最早的时候，每个计算机模型甚至每台计算机都不相同（第一代）。IBM 360 的出现改变了这一情况，它开始采用中小规模的集成电路，使我们能够建造大型机、小型机以及后来的超级计算机（第二代）。但是这些系统的实现成本高昂，导致它们逐步退出历史舞台，被微处理器所取代。得益于摩尔定律带来的晶体管规模增长，微处理器逐渐具备了更强大的功能，能更好地挖掘出指令级并行，性能与效率得以提升。RISC 和 CISC（复杂指令集计算机，Complex Instruction Set Computers）架构也是在这一时期被提出并得到发展，使微处理器的性能越来越强（第三代）。然而有一天，指令级并行的效果逼近上限，我们开始关注多核微处理器——在同一芯片上实现多个独立计算核心以持续提高性能（第四代）。当前，由于登纳德缩放定律的结束和摩尔定律的减缓，第四个时代也即将终结。我们现在正处于第五个时代的开端，从通用计算转向领域专用。

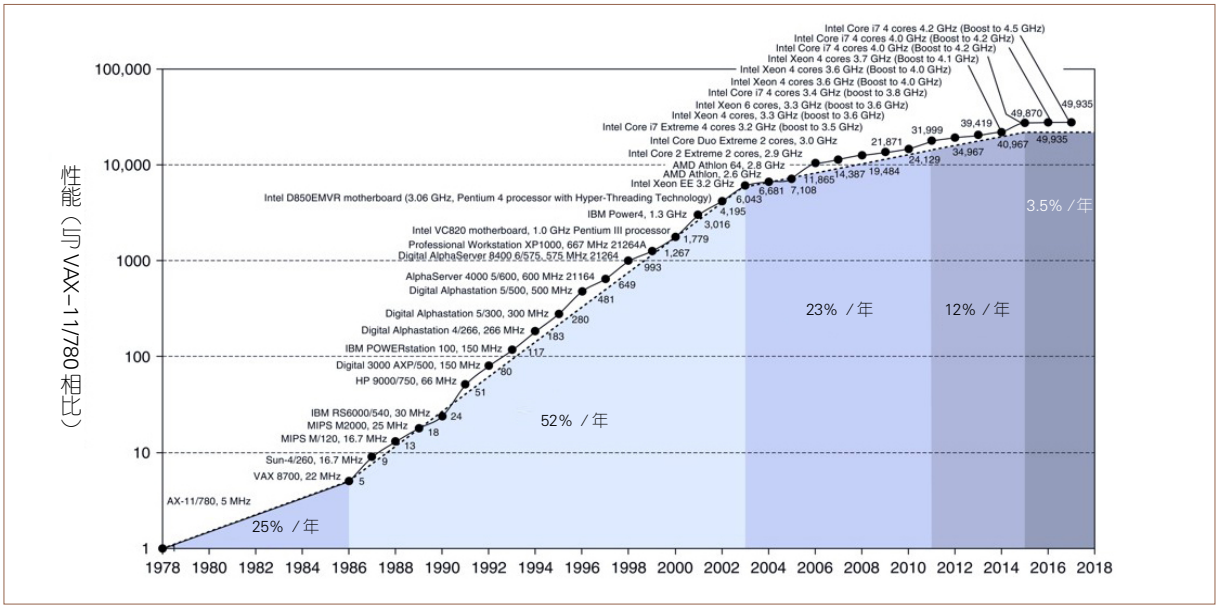

图 2　单核处理器性能趋势

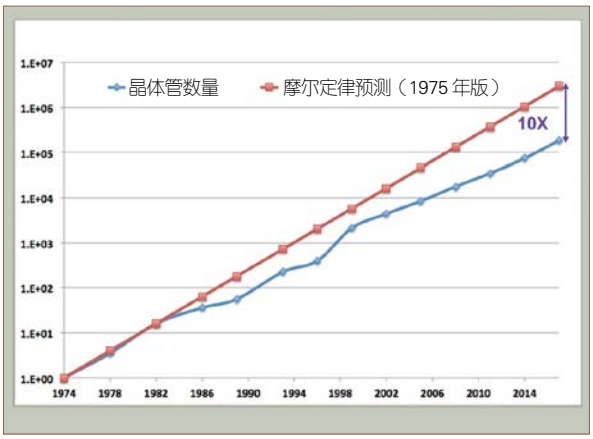

图 3　晶体管数量变化：预测与实际情况对比

图 4　半导体工艺、能耗与登纳德缩放

半导体工艺的趋势和挑战

图 2 描绘了当前单核单线程处理器性能增长逐渐减缓的趋势。从 20 世纪 80 年代中期至 21 世纪初，处理器性能以每年 52% 的速度增长。与之相对的是，在过去的五六年中仅仅取得了每年 3.5% 的缓慢增速。摩尔定律的减缓只能解释其中一部分原因。如果我们根据戈登·摩尔 1975 年的预测做出外推，所应达到的晶体管数量与实际情况之间相差了一个数量级（见图 3）。但是，晶体管的总体数量仍然实现了 100 万倍的增长。在 100 万倍的基础上仅仅相差一个数量级，其实并不是一个巨大的差距，但这一差距正在慢慢变大。同时，由于晶体管的制造成本也在增加，单个晶体管的平均成本的降低速度正在变缓。因此，这仍然是一个现实存在的挑战。

另一个更大的挑战在于登纳德缩放定律。登纳德缩放让我们能够在维持同等能耗的情况下持续提高性能。但从图 4 可以看到，自 2007 年开始，这一缩放效应开始变缓，出现了几次阶跃。直至 2012 年，每平方纳米的功耗开始快速升高，导致能耗增大，效率下降。

能效是一个非常重要的指标。过去 20 年，能

效对于各个层面应用的重要性与日俱增。不论在移动端和物联网应用上，还是在大规模的云计算数据中心中，能效都成为了一个至关重要的因素。例如，在大规模数据中心中，除了计算机系统本身，制冷设施也会消耗能量，使功耗成为了数据中心里第二大成本因素。另一方面，由于处理器的散热目前已经达到了极限，我们现在不得不采用一些在以前看来不可想象的技术：芯片会自动关闭一些功能来避免过热。即使有更好的封装技术，散热能力和电池容量仍会成为能效的瓶颈。为了提高能效，我们需要降低功耗，或者在不增加功耗的条件下提升性能。

事实上，真正终结了之前几个时代的原因正是我们达到了能效的极限。2005 年左右，单核处理器“榨干”了指令级并行，达到了其效率上限。指令级并行的终结开启了第四个时代，在多核处理器上由程序员显式指定并行机会。当然，即使是在多核处理器上，由于阿姆达尔定律的制约，我们永远也无法达到完全理想的高并行效率。这对于提升处理器能效是一个非常严峻的限制。于是，我们看到多核通用处理器的发展也开始碰壁。

设想一个类比的情况，缓存是计算机系统中最为重要的技术之一。然而，如果我们不断增大缓存的容量，最终我们的性能收益会逐步递减，但功耗却会持续不断地增加。

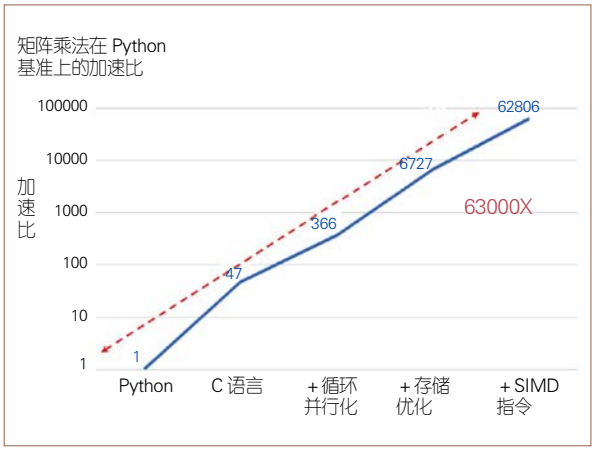

图 5　矩阵相乘性能优化（相对于基准 Python 版本，在 18 核英特尔处理器上）

软件和硬件中的机会

那么，我们还有什么其他机会吗？一方面，在软件层面，一个方案是试图提高程序本身的效率。当前最为流行的编程方法是运用现代高级语言和脚本语言，如 Java、Python 等。它们采用解释型执行，支持动态类型，并具有很高的灵活性来支持复用。这类编程语言对程序员非常友好，可以用于快速开发大型工程，但它们的执行效率不高。另一方面，在硬件层面，我们也可以通过改进架构来提升性能。但这个方向上的唯一出路是转变为领域专用架构，而非继续保持通用性。我们可以为一套特征相近的任务专门定制一种系统架构的设计，以达到最佳的性能。事实上，我们也可以结合软件和硬件，协同设计一门领域专用的编程语言，支持其特别优化的系统架构。

让我们来看看这两个方向具体的机会。首先是软件层面。图 5 引用的是来自麻省理工学院一个团队的论文“There’s plenty of room at the Top”。他们观察到，当前我们常用的软件编程环境的效率尚有很大的提升空间。他们选择了一个小规模但具有足够代表性的应用——矩阵相乘。比较的基准是一个 Python 的实现版本。首先，他们用 C 语言实现了同样的算法，得到了 47 倍的加速；然后在 C 语言版本的基础上把循环并行化，得到了额外的 8 倍提速；接着，引入缓存和内存访问等相关优化，使性能再次提高了 20 倍；最后，通过采用 x86 架构上的 SIMD 指令，又获得了 10 倍加速。结合以上的所有软件优化技术，最终的实现版本比原始的 Python 基准快了约 63000 倍。不得不说这是一个极大的性能提升。虽然这个程序只是一个比较易于优化的例子，但我认为它揭示了许多通过改进软件实现系统效率提升的机会。

那么，领域专用的硬件架构又拥有什么样的机会呢？这类架构通过针对特定应用领域的定制化设计来达到更高的效率，支持整个领域的应用，而不只是单一的一个应用。它们是可编程的，这是与专用的 ASIC（专用应用集成电路，Application-Specific Integrated Circuits）芯片的不同之处。后者常见于手机等设备之上，通常用来执行唯一的固定任务，例如调

制解调等，并不能通过编程来完成其他功能，而领域专用架构可以支持一系列相似的应用。但是与通用处理器相比，设计此类架构需要更多的领域专用知识，需要深入理解应用的特征才能得到好的性能。这里也存在着一个权衡：更灵活、更通用意味着更低的性能；更高性能则需要更加专注于特定领域。一个典型的可能也是目前最大的例子，就是深度学习类的应用：现代神经网络处理器可以用于模型训练和任务推理。又如GPU，革命性地改变了图形学应用的性能。好消息是，虽然这类领域专用处理器不够通用，但在这些特定领域中，需要持续追求更高的性能。实际上，我经常听到大家讨论，机器学习也许是目前最重要、甚至唯一的新型编程方式——对数据进行“编程”来产生有潜力的应用，而不是按传统的方式去写许多代码。深度学习正在引发一场机器学习的革命。现在，这一领域每周发表的论文数量已经达到了一百篇左右。机器学习方向发表的论文数量，正如同摩尔定律一般飞速增长。人们对这一领域充满着兴趣和热情，这就是一场新的革命。

领域专用架构

我们如何判断领域专用架构有多大机会？以典型的英特尔多核处理器中的一个核心为例，我们来分析其上每条指令执行时所产生的能量消耗，例如一条从一级缓存中读取数据至寄存器的指令，或者是一条完成一次浮点数乘法的指令。我们注意到，在这两种情况下，多于一半的能量是被控制逻辑所消耗的。另外，从缓存中访问指令也耗费了许多能量。所以，有高达60%~80%的能耗都是来源于获取指令和解析其相应的控制行为。如果可以改善这几部分，那将带来极大的能效提升。

领域专用架构又是怎样超越通用处理器的呢？如果你熟悉戴维·帕特森（David Patterson）与我近些年来编写的教材中的内容，你就会知道应该用量化的分析方法来理解系统的行为。领域专用架构的优势来自其针对特定领域的定制化设计，这里我们给出几条思路。

1. 领域专用架构可以采用更简单的并行方式来处理一个特定应用领域的问题，以减少所需的硬件控制逻辑。我们刚刚谈到，通用处理器中60%以上的能量都消耗在控制上。通过利用更简单的并行方式，我们可以减少控制逻辑，从而减少处理指令的额外开销。这包括两种方法：一是从传统多核处理器的MIMD架构（多指令多数据，即每个核心独立获取其指令流）转向所谓的SIMD架构（单指令多数据，即将一条指令广播至多个运算单元以并行处理多个数据）。由于取指令的数量大大减少，效率将大幅提升。二是采用所谓的VLIW（超长指令字）架构，用更宽的指令来配置各个单元，以取代现代通用多核处理器中的猜测执行和乱序执行方式。后者被广泛用于英特尔处理器和高端的ARM处理器中，但其需要大量的控制逻辑。因此我们再一次在控制方面取得了优势，降低了能耗。

2. 领域专用架构可以更加有效地利用片上和片外的存储带宽，以用户显式控制的存储来取代传统缓存。当我们无法预知程序代码的行为时，缓存对于通用处理器是非常灵活和高效的，但它们也有非常高的额外开销。对比之下，当应用程序具有非常明确的数据流动模式时，我们则可以采用让用户显式控制的存储系统来达到更高的效率。所以我们可以将传统的缓存系统替换成处理器结合显式存储的结构，并在需要时采用高效的片外数据预取。总之，正是由于我们可以明确知道应用程序的特征，使得我们可以完成上述的优化设计。

3. 降低不必要的精度。IEEE的浮点数标准支持很高的精度，但也有很大的代价，可以将其替换成低精度的浮点数格式。类似地，也可以将32比特和64比特的整数降为8比特或16比特。例如，对于机器学习应用，我们可以用小浮点数做模型训练，用小整数做任务推理。

4. 领域专用的编程模型，无论是TensorFlow还是CUDA等，都可以帮助我们将上层应用与下层架构相匹配。

结合以上方法，领域专用架构完全可以在晶体管的使用效率和功耗开销上取得显著的优势。

一个领域专用架构的例子：谷歌 TPU-1

图 6 展示了谷歌的第一代张量处理单元 TPU-1 的芯片布局。注意其中最大的两个模块：一个是矩阵乘法单元（Matrix Multiply Unit），它是一个计算密集型单元，占据了 24% 的芯片面积，含有 256×256 个 8 比特的乘加器，可以在每个时钟周期内完成高达 6 万余次的低精度乘加操作；另一个是大容量的联合存储单元（Unified Buffer），用于本地存储计算的中间结果，可容纳 24 MB 的数据。这是一个用户显式控制的存储，不是传统缓存。

更重要的一点是，如果把上述 TPU-1 和一个典型的高端通用处理器（例如英特尔 Skylake 处理器核）进行对比，在芯片面积的利用方式上我们可以发现明显的区别。TPU 在存储上使用了更多的面积，并且其使用方式更加高效。其存储单元消除了传统缓存所需的标签等额外代价，而是更专注于数据本身的存储。TPU 的计算资源也更多，将近 50% 的芯片面积贡献给了计算密集型模块，包括矩阵乘法单元和其他的累加器等。而 Skylake 核心在计算上用的面积更少。TPU 比 Skylake 核心增加的存储和计算资源是从何而来呢？看一看它们在控制逻辑上的区别就清楚了。Skylake 在控制上使用了 30% 的面积，

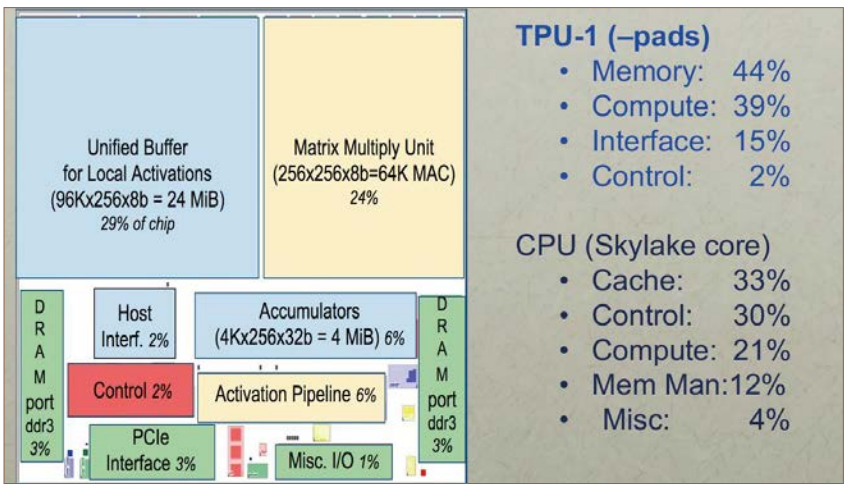

图 6　领域专用架构（谷歌 TPU-1）和传统处理器（英特尔 Skylake 处理器核心）的对比

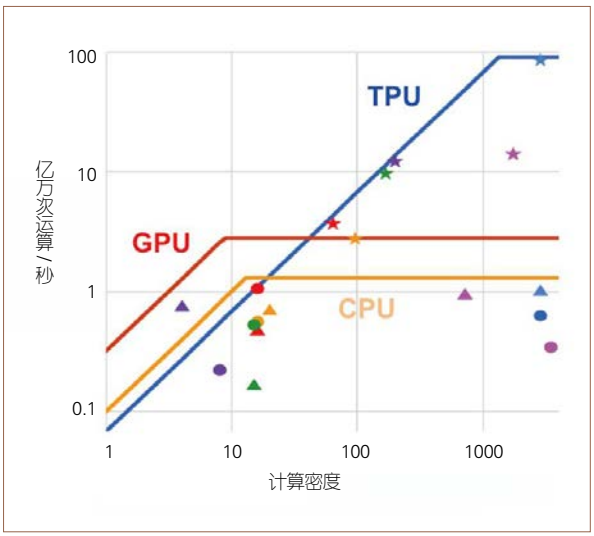

图 7　CPU、GPU、TPU-1 的 Roofline 曲线（对数坐标）

而 TPU 仅仅用了 2%，这就是一种权衡。更强的计算能力和更多的存储空间使计算更加快速，同时利用更少的控制逻辑。

以上这些设计如何转化成性能优势？图 7 作出了各个架构的 Roofline 曲线以进行分析。这些 Roofline 曲线展示了在增大计算密度（computational/operational intensity）的情况下峰值性能的变化。计算密度（即计算访存比）表征了应用程序在从内存读取到处理器中的每单位大小的数据（如一个字节或一个字）上所执行的代数逻辑运算量。我们看到，TPU 的 Roofline 可以升到很高的位置，如果可以让应用程序运行在这一区域，就可以获得比 CPU 和 GPU 高得多的性能。GPU 在低计算密度的区域表现得非常好，因为它有很高的内存带宽来访问数据。相比之下，TPU 的优势体现在当不需要访问太多数据时能高效利用其巨大的矩阵乘法单元。CPU 则显示出一个更平坦的 Roofline 曲线，主要是因为它要在内存带宽、计算能力和通用灵活性之间取得一个平衡。

展望未来

面向未来，我们面临着许多的挑战和机遇。未来的计算机系统会是什么样的？我们应该如何更好地利用现有的优势？好消息是，这是一个新的时代，从基础研究到新兴公司，从硬件到软件，都孕育着新的机遇和新的方向。领域专用架构和领域专用编程语言的设计都还处于襁褓时期，仅仅发展了不到5年，还有巨大的空间来重新思考我们应该如何进行设计，如何向前发展。新兴的领域正在爆发式增长，还存在着诸多尚待解决的问题。例如，我们应该如何处理稀疏数据？在机器学习和深度学习应用中含有许多稀疏数据，在传统意义下难以高效处理，这是一个巨大的机会。又如，我们应该如何将硬件开发变得更加简单和容易？在领域专用架构的大背景下，我们将会需要设计许多不同的架构，这意味着我们需要搞清楚应该如何进行原型设计，复用设计模块，并将之抽象归纳，使得设计和验证工作能在现有方法上得到显著改进。这些硬件设计问题是否可以受益于机器学习算法呢？这都是非常好的研究课题。当然，如果你是一名器件工艺方向的研究者，也可以通过取得各种突破，来延续摩尔定律和登纳德缩放定律。所有像我们一样在思考如何设计更快的计算机系统的人都会非常期待器件工艺的突破。封装技术也同样重要。最后，我们是否还可以超越现有的以硅为基质的工艺呢？是碳纳米管还是量子计算机？这些技术都还处于一个早期的基础阶段，但都有潜力被用于延续半导体的发展路线图。

当畅想未来时，我保持乐观的态度。新的方法终将出现，但需要我们以不同的方式去思考和组织。戴夫·库克（Dave Kuck）是ILLIAC IV系统的软件架构师。ILLIAC IV是45年前非常有名的一个早期采用SIMD并行方式的高端计算机系统。他在口述回忆中谈到，想在应用程序和硬件系统中达到完美的匹配是一件非常困难的事情。当时他们未能实现这样一种满意的匹配，因为他们没有提前充分地设想好软件的部分。在领域专用架构的时代，那些仅仅设计硬件而不去同时好好构建和理解软件栈的人，终将失败。他们设计的机器不会被广泛应用，因为这些机器难以编程和使用。我们应该重新协同思考软件和硬件各个层面，思考跨越软硬件的界限会带来什么样的设计权衡，这样才能利用新的架构去运行新的应用，并得到好的性能。如果我们能做到这些，计算机体系结构将会有一个光明的第五时代，充满着性能提升的潜力和实现激动人心的新兴应用的机会。

最后，我非常高兴能有这次机会，通过该报告与中国同行交流沟通。我认为，不论是对于高校的科研，还是工业界的创新创业，当前都是一个令人无比兴奋的时刻，我们一同见证着领域专用架构的时代向前发展，这将带来无穷的新机会和新应用。处在这一领域中为计算科学的发展贡献力量，将再一次成为一个令人无比激动的使命！ ■

（本文根据CNCC 2020特邀报告整理而成）

作　者：

约翰·汉尼斯（John Hennessy）
斯坦福大学前校长，斯坦福大学荣誉教授。MIPS计算机系统公司和Atheros通信公司创始人。IEEE荣誉勋章和ACM图灵奖获得者。
hennessy@stanford.edu

整　理：

高鸣宇
清华大学交叉信息研究院助理教授。主要研究方向为计算机系统与体系结构、人工智能和大数据计算系统架构、硬件系统安全。
gaomy@tsinghua.edu.cn