山东大学

调研报告

计算机系统结构与硬件技术方向

学　　院	山东大学软件学院
专　　业	软件工程
年　　级	2020 级
姓　　名	贾星宇 杨钰润 易格名 石嘉晖

2022 年 12 月 6 日

目　录

第一章　总体概述

1.1　计算机系统层次结构

计算机系统按照从高级层次到低级层次可以分成 6 层，分别为：应用语言机器级、高级语言机器级、汇编语言机器级、操作系统机器级、传统机器语言机器级、微程序机器级。其中，应用语言机器集指的是通过程序包翻译成高级语言程序；高级语言机器集指的是通过编译程序，将程序翻译成汇编语言程序；汇编语言机器集指的是通过汇编器，将程序翻译成机器语言程序；操作系统机器集指的是通过机器语言程序，来解释控制语句作业；传统语言机器级，主要为固件，指通过微指令程序解释指令；微程序机器级，主要为硬件，指通过微指令系统直接执行微指令。

对于层次结构，在任何级别，下一个机器级别都是无关紧要的。从封装的角度来看，每一层都提供一种服务。机器级划分可以简化系统逻辑，但也引入了不可控的问题。通过翻译和口译实现机器级交互。转换是指使用转换程序将较高机器级程序转换成下一个机器级的等价程序，并在下一个机器级继续执行。解释是在较低的机器层次上使用一系列语句或指令来模拟较高机器层次上的语句或指令的功能。总的来说，笔译技术在翻译后执行效率更高，而口译技术在解释时执行更灵活且效率较低。

我们研究的一个主要子课题是计算机系统结构，也称计算机体系结构，是系统结构的一部分，即“传统机器语言机器级”的结构。然后是操作系统机器级、汇编语言机器级、高级语言机器级、应用语言机器级。其下是固件和硬件，是汇编语言程序员和编译语言程序员看到的抽象概念，是软件和硬件之间的接口。

1.2　计算机硬件技术

计算机硬件是计算机技术运用中的关键内容。其主要由处理器、控制器、存储器、输入设备与输出设备等组成。从物理本质来说，计算机硬件即是组成计算机结构体系中的电子元件、内外部组成结构设备等。计算机硬件系统主要由主机和外部设备两大部分组成。主机，从专业角度来说由 CPU 和内存组成，而在一般情况下指主机箱内的所有部件，包括 CPU、主板、内存、硬盘、电源、风扇、软驱、光驱、显卡、声卡、网卡、MODEM 卡等各种系统功能扩展卡。外部设备，包含键盘、鼠标、显示器、外存储器、打印机、扫描仪、投影仪等。

我们所熟知的 CPU 出现于大规模集成电路时代，处理器架构设计的迭代更新以及集成电路工艺的不断提升促使其不断发展完善。从最初专用于数学计算

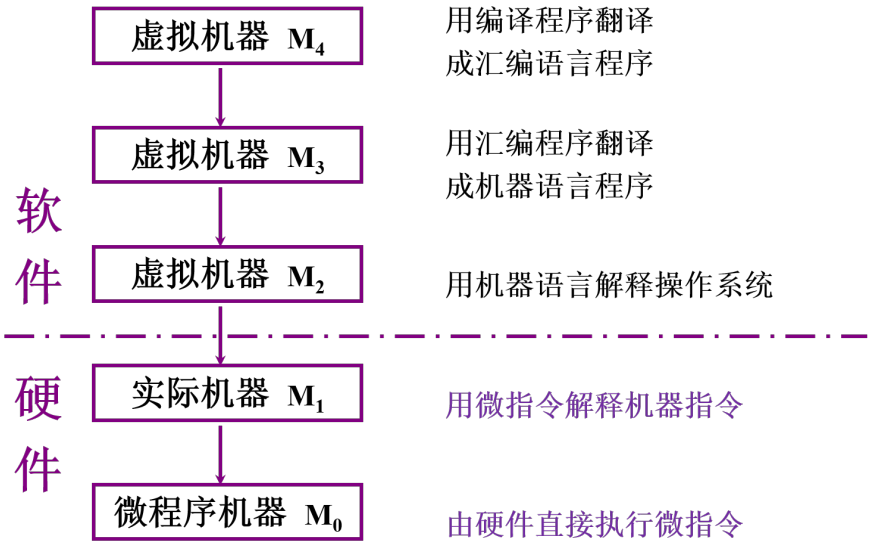

图 1-1　　计算机系统层次结构

到广泛应用于通用计算，从 4 位到 8 位、16 位、32 位处理器，最后到 64 位处理器，从各厂商互不兼容到不同指令集架构规范的出现，CPU 自诞生以来一直在飞速发展。CPU 包括运算器和控制器两个部分；其中，ALU 是运算器的核心器件，用来完成算数和逻辑运算；CU 是控制器的核心器件，用来解释存储器中的指令，并发出各种操作命令来执行指令。主存储器用来存放程序和数据，它可以直接与 CPU 交换信息；I/O 设备受 CPU 控制，用来完成相应的输入输出操作。

第一台电子计算机 ENIAC 于 1944-1946 年在美国宾夕法尼亚大学莫尔学院创建，重达 30 吨，造价 48 万美金，占地 170 平方米，内装 18000 个电子管，计算速度 5000 次/秒加法。但是，这台计算机存储容量小：仅有 20 个字长 10 位。而且非自动，采用线路连接来编程。Von Neumann 在此基础上提出了现代计算机的模型。随后，计算机之父冯 · 诺依曼提出了冯诺依曼计算机：计算机由五大部件组成；指令和数据以同等地位存于存储器，可按地址寻访；指令和数据用二进制表示；指令由操作码和地址码组成；能够存储程序；以运算器为中心。

其中，采用二进制形式表示数据和指令指的是数据和指令在代码的外形上并无区别。都是由 0 和 1 组成的代码序列，只是各自约定的含义不同而已。采用此方法的原因是：逻辑电路的技术实现简单；物理上最易实现存储；便于进行加、减运算和计数编码；便于逻辑判断（是或非）；用二进制表示数据具有抗干扰能力强，可靠性高。采用存储程序方式是诺依曼思想的核心内容。通过事先编制程序—存入主存储器—运行程序，让计算机能高速自动运行。计算机的工作体现为执行程序，计算机功能的扩展在很大程度上体现为所存储程序的扩展。

图 1-2　　计算机硬件

计算机技术的发展，得益于计算机硬件技术的逐代革新。计算机技术运用的一大内容组成：软件运行，其发展也是离不开硬件系统的。简言之，硬件是软件的前提条件，只有提升硬件技术的全面运用，才能促使软件技术得以创新发展。

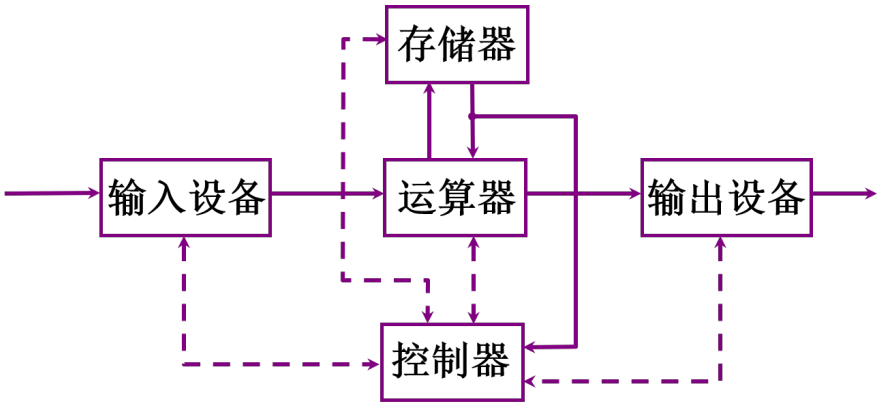

图 1-3　　计算机硬件框图

下面，我们将以计算机架构划分、计算机性能描述、软硬件的发展和系统结构的并行性四个方面说明计算机硬件技术当前的发展。

1.2.1　计算机架构划分

有很多方法来划分计算机结构，我们广泛使用的是佛林分类法，即 Flynn’s Taxonomy，它根据指令和数据进入 CPU 的方式分为了四类：单指令单数据

SISD，如传统串行计算机 386；单指令多数据 SIMD，特点为并行架构，比如向量机，所有核心指令唯一，但是数据不同，现在 CPU 基本都有这类的向量指令；多指令单数据 MISD，这种类型比较少见，类似于多个指令共同处理一个数据；多指令多数据 MIMD，特点为并行架构，多核心，多指令，异步处理多个数据流，从而实现空间上的并行，MIMD 多数情况下包含 SIMD，即 MIMD 拥有很多计算核，而计算核支持 SIMD。另外，根据内存的类型，我们还可以将计算机结构划分为分布式内存的多节点系统和共享内存的多处理器系统。

再参考市面上很多文档和说明对于计算机架构的分类，我们发现，吞吐量和计算速率逐渐成为近年来计算机结构的分类方向。这是因为人们对于高性能计算的要求提高。高性能计算拥有更快的计算速度、更大的负载能力和更高的可靠性。故人们希望，让高性能计算的每一个单一处理器的计算性都足够强劲，并在此基础上集成多个处理器。这些多个处理器构成了一个更高性能的计算机系统，并且实现了多处理器之间的协同分布式计算和并行计算。这种全新的思想被称为“高性能计算机群”，即 HPC。此外，由于支持在云端自动创建和删除集群，HPC 还能有效降低负载成本。

1.2.2 计算机性能描述

为了描述一个计算机的性能，我们定义了多种属性，并分析了这些属性对于性能的参考价值，结果如下：

一、主频：主频作为性能描述的指标是不准确的，主频高的 cpu 不一定比主频低的 cpu 性能好。因为可能只是主频高的 cpu 核心数更多。只有在核心数等条件相同的情况下，才可以用主频进行比较。

二、每秒百万条指令（MIPS）：参考该项指标时，单位时间内，执行指令越多，性能越好。

公式：MIPS = 执行条数 / 执行时间 $*10^6$

然而，这个标准也是不准确的。假设对于一个游戏的功能来说，用硬件实现只需要 100 条指令，用软件实现需要 10000 条指令。但是，硬件实现 100 条指令的性能不一定比软件 10000 条指令的差。

三、每秒百万条浮点指令（MFLOPS）：该项指标即为 MIPS 的精准版，所用是小数，而 MIPS 使用的是整数。超级计算机比赛使用的就是这个指标，因为执行的指令等条件都一样。

四、真实程序运行实现：这是唯一一项较为准确的指标。测试时多个计算机同时执行一项任务，谁先完成的早，谁性能最好。为避免计算机有各自擅长的领域而有失公允，通常会选择将多个领域都测试一遍，然后综合表现得出结论。

1.2.3 软硬件的发展对计算机系统结构的影响

一、软件的发展对计算机系统结构的影响

软件的发展使得计算机的可移植性提高。可移植性具体指的是：软件能长期使用，不会因为机器更新而需要重写；可以大大减少了编程软件的工作量；能迅速用上新的硬件技术，更新系统，让新系统立即发挥效能；软件设计者能够将更多精力投入到开发全新的软件中去。

下面介绍实现可移植性的几种方法：模拟和仿真、采用系列机、统一高级语言。

（一）模拟和仿真

模拟是指通过软件解释技术，在现有操作系统上模拟指令集，从而在一个系统上运行另一个系统。但是模拟的运行性能较低，而且只有当功能没有任何问题时我们才方便使用模拟。仿真是指通过现有机器的微程序，去实现另一个机器的指令集。这种方法性能高，但是它的难度也更高，更加接近底层，有时候人们难以实现。

模拟和仿真的区别主要有如下几个方面：首先，模拟使用机器语言程序解释实现，解释程序存储在主存中。而仿真通过微程序解释实现，解释程序存储在控制存储器中。其次，由于微程序机器级对传统机器语言机器级的依赖，两台计算机在系统结构差异很大时难以仿真。最后，仿真可以提高性能而模拟不能。

所以，不同软件系列移植一般会采用模拟和仿真并行。对于经常使用且系统结构差异不大的软件一般会使用仿真，而那种不经常使用，而且 I/O 等难以仿真的软件，一般会使用模拟。即使两台机器差别较大，也需要同时使用仿真和模拟来完成机器间的映像。

（二）采用系列机

通常系列机系统结构差别不大，易于移植，但也并非可以百分之百实现移植。系列机在发布更新版本时，一般只能增加功能，不能删除，也不能更改之前提供的更新内容。

（三）统一高级语言

统一高级语言软件移植技术应用于系统结构相同以至完全不同的的机器之间高级语言程序的软件移植，这是一种理想的移植方式，但是不容易实现。究其原因，主要是因为程序员之间开发习惯不同，熟悉的开发语言不同，对语法的需求也不同。并且对于语言的一些字段，不同的程序员之间也存在一些争议。更糟糕的是，不同厂家为了发展自己的特色，会在编译程序后嵌入汇编语言，这就导致即使是相同的高级语言，其对应的编译程序都不相同，很难适配。

二、应用的发展对系统结构的影响

应用是系统结构发展的第一动力，如多媒体、网络应用的不断需求促进了个人 PC 不断更新换代。近年来大型游戏的开发也对系统结构提出了更高的要求。

三、器件的发展对系统结构的影响

（一）非用户片、现场片和用户片的发展影响

非用户片又称通用片，是在厂家制作后发售的器件，用户只能使用，不能进行任何更改；现场片是厂家发售后，用户可根据需求进行更改的器件，使用更灵活，功能强。如 BIOS 就在 prom 可编程只读存储器中；用户片是用户向厂家定制的器件，销量低、成本高、设计费用高、设计周期长。如果是完全按照用户要求定做的用户片，我们称之为全用户片。

（二）器件的发展是推动计算机系统结构发展的动力

器件的发展有如下的规律：对于电路来说，为电子管-晶体管-小规模集成电路-大规模集成电路-超大规模集成电路。同时，也有着非用户片-现场片-用户片的发展规律。

（三）器件发展改变了设计逻辑

当下计算机系统结构的设计主要是看微汇编、微高级语言、计算机辅助设计软件的方法来设计。同时，对用户片计算机而言，机器设计和芯片设计也有着密不可分的关系，其中最主要的是对芯片的设计。有时，为了芯片的利用率，我们必须采用计算机辅助设计，通过计算机辅助系统来提高芯片的利用率。

1.2.4 系统结构的并行性

一、并行性

（一）并行性的概念

在同一时刻或同一时间间隔，同时做出多个操作。并行性有双重概念：其一是同时性，即同一时刻做出多个操作。其二是并发性，即同一时间间隔做出多个操作。并行性的目的是为了提高计算机处理性能。我们常说的并行和并发分别指代的就是同时性和并发性。

（二）并行性的等级划分

从计算机系统执行程序的角度，可以划分为指令内部和指令之间、任务和进程之间、作业和程序之间。其中，指令内部是指一条指令的多个微操作并行性。指令之间是指指令和微操作都可以并行性。任务或进程之间是指多个任务或程序段并行性。作业或程序之间是指多个计算机协同执行程序。

从计算机处理数据的角度，可以划分为：位串字串，即每次只处理一个字（word），一个位（bit）；位并字串，即每次处理一个字（word）的多个位（bit）；位片串字并，即每次处理多个字（word），每个字处理一个位（bit）；全并行，即字并行处理，字的位也并行处理。

从计算机信息加工的各个步骤和阶段的角度，可以划分为：存储器操作并行，即存储器用各种访问方式，对要操作的内容，进行并行的比较、检索、更新、变换等操作。典型例子是并行储存系统和以相联储存器微核心构成的相联处理机；处理器操作步骤并行，即将指令的取指、分析、执行等操作，按执行步骤在流水线上处理。典型例子是流水线处理机；处理器操作并行，即多个处理机在一个控制器下，对指令的多个微操作（步骤）进行并行性的处理；指令、任务、作业并行，即多个处理机，对多个指令进行并行性处理，同时，也对每个指令的步骤进行并行性处理。

（三）沿三种并行性开发途径的多机系列类型和特点

常见的三种并行性开发途径为时间重叠、资源重叠、资源共享。其特点如下：时间重叠是在同一个硬件条件下，通过流水线的方式处理，以提升处理效率；资源重复是通过推硬件来提高性能；资源共享是用软件实现，使多个任务按一定顺序轮流使用同一套硬件设备。多道程序分时系统就是典型的资源共享案列。

（四）并行性的耦合度

耦合度可以视为物理硬件的紧密程度，即交叉作用能力的强弱。并行性的耦合度可以分为：最低耦合、松散耦合、紧密耦合。若各机器之间除了存储硬件之外没有任何物理连接，也没有任何共享的联机硬件，就可称为最低耦合。各脱机处理机之间就是最低耦合；若多台计算机通过通道或网络通信通道实现互联并能共享外围设备，可以从最低频带在文件或数据集间相互作用，则称之为松散耦合系统或间接耦合系统。若多台计算机通过总线或高速开关互联，有较高的信息传播率，可实现数据集、任务级、作业级并行，则称之为紧密耦合系统或直接耦合系统。

1.3 文章架构

本文架构如下：

第一章，讲述计算机系统结构和计算机硬件技术的相关介绍，并介绍文章架构；第二章，分析计算机系统结构和硬件技术的子主题划分；第三章，讲述各个子主题的研究热点；第四章，讲述为了实现各个子主题而创建的关键技术；第五章，讲述各个子主题的主要研究机构；第六章，讲述与计算机系统结构和硬件技术相关的重点企业；第七章，设想计算机系统结构和硬件技术的各个子主题的发展趋势；最后的第八章，是关于文章的总结以及报告的分工情况说明。

第二章　　子主题划分

计算机系统结构又称计算机层次结构，是传统机器级的系统结构。它是软硬件的交界面它研究的内容是传统机器级上界面的定义，软硬件功能的分配，为各机器级提供应有应遵循的规范。根据要求和我们收集到的资料，将计算机系统结构与硬件技术分为三个子方向：计算机体系结构、高性能计算与算力影响、计算机硬件与人工智能的结合。

计算机体系结构的主要问题在功耗和散热、CPU 的设计和运行、新兴的设备技术、基于 GPU 的图形处理、新兴计算模型，如类脑和量子计等问题。在现代处理器中，功耗和散热问题已经愈发重要；其次，关于 CPU 设计运行中也遇到了众多现实性问题；同时，计算机图形处理也发生着高速的发展。

在高性能计算及算力影响上，近年来计算机结构的分类方向上有着逐渐趋近于影响吞吐量和计算速率因素的趋势。这是因为人们对于高性能计算的要求提高，同时高性能计算拥有更快的计算速度、更大的负载能力和更高的可靠性。故人们希望高性能计算的每一个单一处理器的计算性都足够强劲，在此基础上，我们集成多个处理器，完成更高速的计算。

同时，人工智能的火热发展推动着人们倾向于将神经网络部署到各种低功耗、低成本的硬件上，如卷积神经网络在区域可编程门阵列 FPGA 上部署并实现目标检测。比如，图像理解正在成为越来越多的应用程序的重要功能，包括自动驾驶汽车的医疗诊断。许多应用程序需要嵌入式集成到具有严格实时性和功率限制的现有系统中的解决方案。卷积神经网络（CNN）目前在所有图像理解基准，但具有非常高的计算复杂性。因此，嵌入式网络需要小型、高效、功能强大的计算平台。CNN 的这种卓越性能是以巨大的计算量为代价的复杂性 CNN 对实时视频流图像分类的实时评估每秒可能需要数十亿或万亿次操作。图像分割的努力并且场景标记甚至显著更高。虽然可以达到此性能级别使用最新的图形处理单元（GPU），同时希望将此类解决方案嵌入其他系统，例如汽车、无人机，甚至可穿戴设备，其在物理尺寸和能量消耗方面表现出严格的限制。因此，嵌入式 CNNs 需要小型高效、功能强大的计算平台。已经考虑了不同的平台来实现。其中最有前途的是 CMOS、CMOS、CMOS 和现场可编程门阵列（FPGA）。这些通用集成电路提供数十万种可编程逻辑块和可配置的互连，这使得能够构建定制的硬件加速器架构。它们有可能提供计算嵌入式网络中心所需的功率在其规定的大小和功率范围内各自的应用程序。

第三章　研究热点

3.1　计算机体系结构

计算机体系结构的研究热点聚焦于现有问题的解决和新兴的设备技术。现有问题主要集中于功耗和散热问题，以及 CPU 设计运行时遇到的现实问题。新兴设备技术则如基于 GPU 的图形处理、新兴计算模型（如类脑和量子计算）等。

功耗和散热问题在现代处理器中已经越来越重要，因此，重要的问题是让芯片架构师和编译器编写者更清楚地了解应该如何权衡功率和性能。大多数现有的功耗分析工具仅在布局或布局规划完成后，它才会对功耗估算值进行计算，从而帮助技术人员实现期望的高精度。但是，除了在设计过程的后期可以使用之外，此类工具并不具有在其他时段进行评测的功能。这无疑大大增加了技术人员在设计时还想要提高性能的难度。并且，由于时钟速率的提高和管芯尺寸的增大，电力问题已经成为了一些非常高端的微处理器的主要设计限制。据预测，功耗和散热问题将很快成为单片微处理器性能的关键限制因素。目前，传统空气冷却技术已经不足以支撑高端微处理器进行散热。当芯片上的电流过大时，尽管电压调节和专门的电路技术是低功耗设计的主要策略，但电池的寿命也会受到影响。因此，工厂难以交付数额巨大的高端微处理器合格品。因而至今计算机系统领域的一大研究热点，仍然是对于降低功耗与电量消耗的方式的研究。

其次，关于 CPU 设计运行中所遇到的现实性问题，我们做出总结。芯片设计师通常会面临权衡和取舍，而其中最大的一个问题就是要低的功耗，还是要好的性能，这两者不能兼得。设计师在设计每一代芯片时都要提升芯片的性能，通常设计师会选择提升 IPC 或者最大频率等方面，在过去几十年间，因为摩尔定律，性能的提升较为容易。但放到现在，由于摩尔定律效用放缓，性能提升变得不再那么容易。在这种情况下，设计师就需要在微架构上有更多的创新，但在这个过程中，伴随运行速度的增加，功耗往往也不断增加。

一方面，输电资源技术（power delivery sources）的发展非常缓慢。输电线上的电阻很大，导致不能提供足够的功率。另一方面，封装技术有限，封装电感会导致无法提供所需的快速变化的电流或功率。功率和电流通常成正比，因此很难得到一个快速变化的电流。如果一瞬间电流突然增大，则功率也会瞬间增大，而这种情况是我们不希望看到的。结合两方面因素，促使设计师不仅要在设计时对功耗有更多的了解，而且在运行中也要对功率变化进行管控，避免出现停机等糟糕的情况。

如下图所示：

Service	Cost of One Hour of Downtime
Brokerage Operations	$6,450,000
Credit Card Authorization	$2,600,000
eBay	$225,000
Amazon.com	$180,000
Package Shipping Services	$150,000
Home Shopping Channel	$113,000
Catalog Sales Center	$90,000

图 3-1　　系统停机一小时的估计成本

计算机图形处理器（Graphics Processing Unit,GPU）的高速发展, 不但促进了图像处理、虚拟现实、计算机仿真等应用领域的快速发展, 同时也为人们利用 GPU 进行图形处理以外的通用计算提供了良好的运行平台。正因如此，基于 GPU 的图形处理及其通用计算成为图形学及高性能计算领域的热点研究课题之一。

早期 GPU 是一种图形处理设备，但是如今它已经作为一种通用设备服务于多个领域：在超算领域，它赋能当今最快的超级计算机；在人工智能领域它作为深度学习的主要平台，为自动驾驶、机器人以及智能摄像头等诸多设备提供智能。与此同时，它还能够以实时的帧率生成逼真的高质量图片。GPU 的演进则主要是通过增加新特性支持新的应用场景。

3.2　高性能计算（HPC）

1972 年费林根据资讯流将计算机分成指令和资料两种，而今又可细化成如下四种计算机类型：单一指令流单一资料流计算机、单一指令流多资料流计算机、多指令流单一资料流计算机、多指令流多资料流计算机。按照费林的分类，分布式的高性能计算属于多指令流多资料流计算机的范畴。

HPC 系统由计算、存储、网络、集群软件四部分组成。其类型繁多，范围从标准计算机的大型集群，到高度专用的硬件不等。大多数基于集群的 HPC 系统使用高性能网络互连，而基于网络拓扑和组织的 HPC 系统则可以使用一个简单的总线拓扑。HPC 系统的主流处理器是 X86 处理器，操作系统是 linux 系统 (包括 Intel、AMD、NEC、Power、PowerPC、Sparc 等)、构建方式采用刀片系统，互联网络使用 IB 和 10GE。

HPC 集群中的计算节点一般分 3 种: MPI 节点、胖节点、GPU 加速节点。MPI 节点又称为瘦节点、双路节点，双路以上称为胖节点；胖节点需要配置大容量内存；集群中胖节点的数量要根据实际应用需求而定。

HPC 的算法 (或者算法中的部分满足，其他部分通过 CPU 协调) 必须满足

一下几点要求：每个数据（数据包）都需要经过相同的流程来处理；数据之间并没有相关性，即数据的计算不依赖另外一些数据的计算结果；数据量庞大。

为了解决这些问题，研究者提出了一种功率感知算法，可以自动调整电压和频率设置从而实现降低功耗，同时对性能的影响最小化。具体来说，我们利用名为“动态电压和频率缩放（DVFS）”的技术让 HPC 系统在运行时实现功率感知算法，从而实现 DVFS。举例来看，如果笔记本电脑用户在使用电池供电的情况下长时间阅读文档，笔记本电脑就会自动降低 CPU 的频率和供电电压以降低功耗，因为功耗与 CPU 频率和 CPU 电源电压的平方成正比。在这个例子中，为 CPU 实现上述频率和电压自动调节的技术就是 DVFS。相比于普通计算机，功率感知（或低功率）的概念对于 HPC 系统来说是一个全新的突破点，实现起来也相对更加困难。

大型 HPC 系统的可靠性问题也尤为重要。例如，表 1 显示了当前前沿超级计算机的可靠性 [21]。随着功率密度每 18-24 个月翻一番（图 1），并且大型 HPC 系统的尺寸不断增加，产生的热量（以及因此产生的温度）继续上升。根据经验，应用于微电子学的 Arrenhius 方程指出，温度每升高 10°C (18°F)，系统的故障率就会翻倍。

System	CPUs	Reliability
ASCI Q	8,192	**MTBI: 6.5 hrs.** HW outage sources: storage, CPU , memory.
ASCI White	8,192	**MTBF: 5 hrs ('01) and 40 hrs ('03).** HW outage sources: storage, CPU, 3rd-party HW.
PSC Lemieux	3,016	**MTBI: 9.7 hours.**
Google	15,000	**20 reboots/day; 2-3% machines replaced/year.** HW outage sources: storage, memory.

MTBF/I: mean time between failures/interrupts

图 3-2　前沿超级计算机的可靠性

目前，存在一些关于使用 DVFS 降低高性能计算 (HPC) 节点的热功率包络的可行性的有见地的案例研究，从而提高可靠性，同时最大限度地减少对性能的影响 [2] , [6]。知道 DVFS 确实可以在 HPC 中发挥作用后，下一步就是开发利用 DVFS 的各种方法。此类方法包括手动 DVFS 调优 [5]、[7]、带概要分析的编译器分析 [12]、基于 MPI 库的扩展或自适应运行时系统 [11]。

手动 DVFS 调整通常涉及在所有可能的频率电压设置下分析程序（或其结构）的执行行为。它可以简单到记录程序在每个可用 CPU 频率下的执行时间，然后使用配置文件选择满足性能约束的最低频率来执行程序。[2]、[6] 的可行性研究属于这一类。

当前用于功率感知的自适应运行时系统的方法主要基于 CPU 利用率，例如笔记本电脑上的 cpus-peed。对于 cpuspeed，当 CPU 利用率低于某个阈值时，CPU 电压和频率会降低以节省能源；当 CPU 利用率超过某个阈值时，会提高 CPU 电压和频率以提高性能。尽管这种简单的方法既独立于应用程序又独立于输入，但它仅对交互式使用有效，例如，Microsoft Office 的笔记本电脑使用，并且主要取决于阈值 [10] 的选择。对于科学应用，它的效率很低，因为这样的应用没有大量的 CPU 空闲时间 [11]。

其他基于 CPU 利用率的更复杂的方法，例如 [25] 中的方法，仅提供对 DVFS 引起的性能下降的松散控制，例如，37% 的速度下降，SPEC 仅节省 12% 的系统能源去基准测试，因为 CPU 利用率本身并不能提供足够的计时信息。因此，我们得出结论，需要一种具有严格的性能减速控制并可以节省大量能源的节能运行时系统。

3.3 计算机硬件与人工智能

随着硬件描述语言以及高层次硬件语言的发展，我们甚至可以利用小规模或者功耗极低的硬件，如区域可编程逻辑门阵列（FPGA）或者 ARM 芯片进行部署并计算。

3.3.1 神经网络

近年来，我们以实现类人工智能的机器学习技术为目标，开发了一种模仿人脑的新型神经网络，即神经网络。在人脑中，神经网络是高度复杂和僵化的组织。据统计，一个成年人的大脑可能包含多达 1000 亿个神经元。因此，我构建了一个包含输入/输出和计算函数的神经元模型。一般来说，一个神经元有多个树突，主要用来接收传入的信息，还有轴突，在轴突的尾部有无数的神经元，可以发送其他神经元，还有轴突末端，它们像神经冲动一样传递信息。轴突末端连接到其他神经元的树突，因此它们可以传输信号。生物学家称这些连接为“突触”。

数据的输入，我们可以将其类比为神经元的树突，同理的，我们将输出类比为神经元的轴突，将细胞核的功能类比为计算。在神经元中，连接是最重要的一个操作。我们在每一个连接上都赋予一个权重，神经网络训练算即为将权重的值调整到最优状态，这样能够让整个网络的预测效果达到最好。在其他的绘制模型里，有向的箭头有可能表示的是值的不变化的传递，而在我们神经元模型里，有向箭头表示的是数据值的加权传递。

3.3.2 卷积神经网络

视觉是人类获取信息的主要来源，在通过视觉获取到的信息中，图像是人们生活中最重要的数据信息，所以图像处理一直是近年来研究的热点。目标检测将

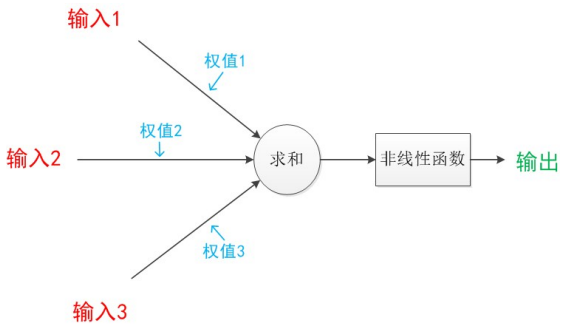

图 3-3　神经元模型

图像中目标的分割和识别合二为一，即需要对图像中的所有目标完成位置确认和类别识别，是计算机视觉领域最基本也是最重要的任务之一。因而，随着计算机视觉技术的不断发展，目标检测技术也被广泛运用于各个领域。

当研究人员训练有 30 万个突触权重的神经网络层时，很容易出现过拟合。过拟合会把图像中对于分类不重要的信息当成重要的信息，抓不住问题的本质，导致模型虽然在训练数据上表现好，但在更广泛的测试数据上表现差，即模型的泛化能力差。而实际应用中神经网络的权重远多于 30 万个。在深度学习的神经网络中，通常有很多个隐层，每一层的神经元的数量可能远不止 100 个。如果采用简单粗暴的全连接方式，当输入是 224×224 大小的 RGB 图像（第一个隐层有 1000 个神经元）时，仅输入和隐层之间的权重就有 1.5 亿。有如此多参数的神经网络是很难训练的，即使训练出来也往往会过拟合。

卷积神经网络（Convolutional Neural Network，CNN）在某种意义上能够很好地解决上述问题，其核心思想包括两点：

其一，局部连接。视觉具有很强的局部性，相邻的多个点很可能构成一个完整的物体，距离越远的两个点之间的联系可能越弱，所以用神经网络做图像处理时，一般不需要做全连接，应该充分考虑邻域信息，对局部做稠密连接即可。

其二，权重共享。卷积神经网络使用卷积核（也称为滤波器或卷积模板）做卷积处理，一张图片中不同的位置可以用同样的卷积系数（即突触权重）。例如一张图片的左上角和右下角，神经网络的突触权重可以是同一组值。其原理是，每一组权重抽取图像的一种特征。例如，抽取形状特征时，在图像的不同位置都可以用同一组权重。基于局部连接和权重共享两种技术，卷积神经网络可以大幅减少处理图像时所需的权重的数量，从而避免过拟合。

卷积神经网络中最重要的层是卷积层（convolution layer），每个卷积层有一组卷积核来抽取特定的特征。卷积层输入、输出的特征图（feature map）尺寸一般变化不大。为了缩小特征图尺寸，一个卷积层之后，一般会有一个池化层

(pooling layer)。例如图 3.2 中，输人图像是一张 224×224 大小的 RGB 图片。通过第一组卷积层后变成 64 个 224×224 大小的特征图，这 64 个特征图代表 64 个不同的卷积核抽取出的图像中的 64 种不同的特征，同时特征图尺寸保持不变；卷积层后面是池化层，通过池化把特征图的尺寸从 224×224 变为 112×112；然后交替地出现卷积层和池化层，特征图的尺寸随之不断变小，从 112×112 变为 56×56，最后变成 7×7。与此同时，抽取出来的特征数量在不断增多，从第一组卷积层抽取出 64 种特征，第二组卷积层抽取出 128 种特征，到最后一组卷积层抽取出 512 种特征。然后用全连接层，将输入的神经元和输出的神经元全部一一连接起来。当然，在每一个卷积层和每一个全连接层内部，除了向量内积，还需要有激活函数。最后，还会用到 softmax 函数进行分类概率的凸显、抑制以及归一化。

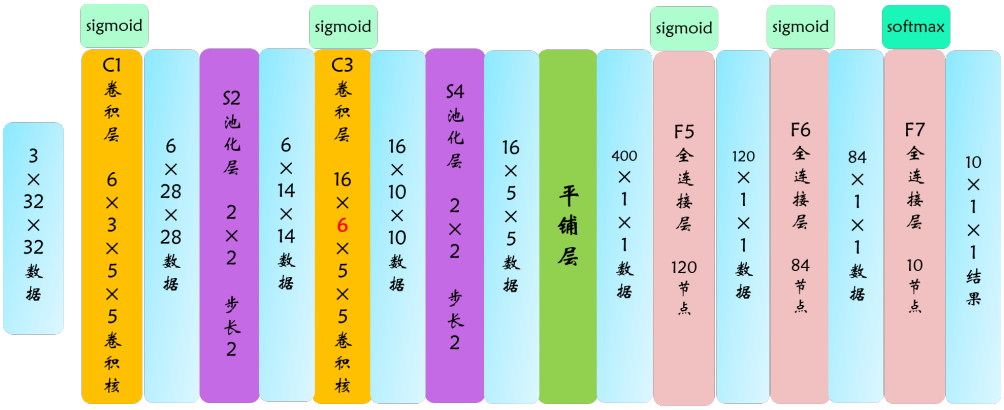

图 3-4　　lenet 卷积神经网络

一、卷积层

卷积层通过卷积可以抽取出图像中一些比较复杂的特征。由于卷积层的局部连接和权重共享的特点，对一张图片做卷积时，不相邻的区域不会放在一起计算。浅层神经网络采用全连接方式，计算一个输出需要用到所有输入。而卷积神经网络计算卷积层的一个输出只需要用到 K×K 个输入，其中 K×K 是卷积核的大小，K 可以是 1、3、5、7、9、11 等。此外，浅层神经网络中所有神经元之间的连接都采用不同的权重，因此一个 N 输人、N 输出的全连接的权重为 N×N 个。而卷积神经网络中卷积层的一对输入特征图和输出特征图共用同一组权重，权重仅为 K×K 个，大幅减少了权重的数量。

卷积神经网络的卷积运算是对输人子矩阵和卷积核做矩阵内积。假设图 3.4 是一张图片或输人的特征矩阵 X，矩阵大小为 6×6；卷积核 K 是 3×3 的矩阵；卷积步长为 1。为了计算输出矩阵 Y 的第一个值 $Y_{0.0}$，将卷积核的中心放在矩阵

X 的（1，1）位置，将对应位置的矩阵 X 的系数和卷积核的系数一一相乘后加和。随后，将卷积核的中心在矩阵 X 上右移一格，再将对应位置的矩阵 的系数和卷积核的系数一一相乘后加和，得到输出矩阵的第二个值 $Y_{0,1} = 40$；将卷积核在矩阵 X 上每次移动一格，再做乘加计算可以得到输出矩阵 Y 的第一行的所有值。然后，将卷积核的中心移动到矩阵 X 的（2，1）位置，再将对应位置的矩阵 X 的系数和卷积核的系数一一相乘后加和，得到输出矩阵的第二行的第一个值 $Y_{1,0}$=5。移动卷积核在矩阵 X 上的位置，可以得到所有其他输出值。以上就是卷积运算的过程。该运算过程中，只用了一个卷积核，因此只能提取出一种特征。

二、池化层

池化层（pooling layer）可以主动减小图片的尺寸，从而减少参数的数量和计算量，抑制过拟合。例如，输入图片或特征图的大小为 100×100，经过池化，可能变成 50×50。池化层一般没有参数，训练时很简单。池化的方法有很多种，例如最大池化（max poo- ling)、平均池化（avg pooling)、L2 池化等。最大池化法是一种常用的池化方法，在池化窗口 K,×K. 内找最大值作为输出。以图 3.9 为例，假设池化窗口为 2×2，步长为 2，不做边界扩充，从输人特征图的左上角的 2×2 子矩阵找到最大值 7 作为第 1 个输出，池化窗口在输人特征图上右移 2 格后找到最大值 5 作为第 2 个输出，池化窗口继续右移 2 格找到最大值 3 作为第 3 个输出，继续向下滑动池化窗口可以得到所有的输出值。最大池化法仅保留池化窗口内特征的最大值，可以提高特征的鲁棒性。

平均池化法也是一种常见的池化方法。该方法在池化窗口内对所有的数取平均值，会把图像的一些特征平均化，也就是模糊化。L^2 池化法是在池化窗口内对所有的数计算平方并累加和后再开平方。对于硬件设计而言，最大池化法只需要找几个数中的最大值，很容易实现。而 L^2 池化法需要计算开平方，硬件实现复杂度高。以前还有用几何平均做池化，复杂度更高。如果几何池化窗口为 2×2，则需要开 4 次方；如果几何池化窗口为 3×3，则需要开 9 次方。几何池化计算时间很长，可能会带来一点精度提升，但实际使用时会有很多麻烦。因此最大池化法是最常用的。

三、全连接层

卷积层和池化层构成特征提取器，而全连接层（fully-connected layer）是分类器。全连接层将特征提取得到的高维特征图映射成一维特征向量，该特征向量包含所有特征信息，可以转化为最终分类为各个类别的概率。例如，一个 224×224 大小的输入图片经过多层卷积和池化，可能变成 4096 个 1×1 大小的特征图，根据这 4096 个特征可以做一个全连接层，来判定最后是猪狗猫牛羊中的哪一个。

3.3.3 SoftMax 层

有的卷积神经网络的最终输出是由全连接层决定，但也有的卷积神经网络是用 softmax 层做最终输出。softmax 对输出进行归一化，输出分类概率。其计算过程为

$$f(z_i) = \frac{e^{z_j}}{\sum_{i=0}^{n} e_{z_i}}$$

其中，z 是 softmax 层的第 j 个输入。从 softmax 的计算过程可以看出，输入和输出的数据规模是相同的；通过归一化计算，可以凸显较大的值并抑制较小的值，从而显著地抑制次要特征，决定分类概率。

3.3.4 FPGA

FPGA 是用途较为广泛的通用可编程的加速设备。在深度学习加速中，FPGA 也常常用来实现各种不同的架构设计，也常常用于架构设计的前期验证。我们同样通过计算、存储和控制三个方面来简单介绍 FGPA。在计算单元上，FPGA 采用了大量基本的可配置逻辑单元模块（Configurable Logic Block, CLB），这些模块通过查找表（Loop-Up Table，LUT）的方式实现各种功能。简单来说，LUT 的基本思想就是把所有输入组合对应的输出情况保存在查找表中，然后根据输入的值通过查表输出相应的值。例如，对于实现一个异或 XOR 基本逻辑，LUT 需要把整个真值表存储下来（四行，输入组合为 00，01，11，10 的情况）。在使用的时候，由于单个 CLB 的计算能力极其有限，FPGA 需要通过编程决定 CLB 的自身功能和相互之间的互联方式，从而通过联合多个 CLB（包括多个互联模块和存储模块）来完成复杂的功能。在存储方面，出于灵活性的考虑，通常 FPGA 在片内提供了很多存储资源，可以配置成不同的形式来使用。例如 Xilinx 公司的 FPGA 在内部集成了很多块存储（Block RAM），专门实现数据暂存功能，且每个时钟区域都布置了若干个 BlockRAM。通过将这些 Block RAM 组织、配置成所需要的存储模块，FPGA 可以提供例如单端口 ROM、RAM 或者双端口 ROM、RAM 等功能。在控制上，FPGA 则需要设计者通过配置 CLB 的方式来控制和使用片内的资源。在实际使用中，FPGA 开发工具会帮助用户把他们所编程的电路切分成小的模块，然后把这些模块映射到 FPGA 的 CLB 上。通过对 FPGA 内部可编程连线资源进行配置，这些 CLB 就可以有机地组织联合成完整的电路，包括时序电路。

FPGA 是一种可编程、可定制的芯片，与 GPU 类似，具有并行处理的能力。传统的 FPGA 只具有可编程逻辑部分，芯片的灵活性较差，但近年来，随着 SOC 技术的不断发展，推出了大量高性能的异构 FPGA 产品，最典型的就是 Xilinx 公司的用于高性能需求的异构平台 ZYNQ 系列芯片。

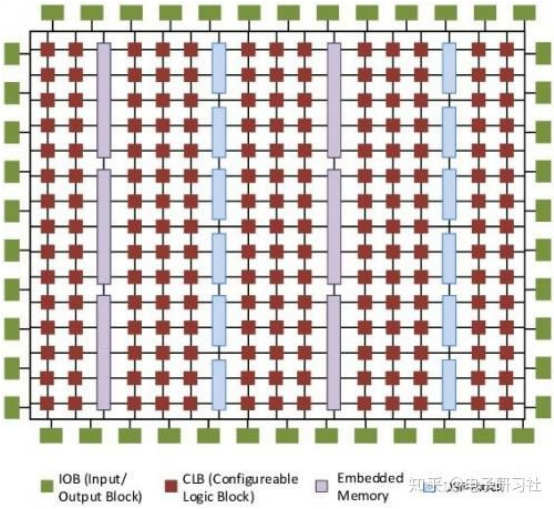

图 3-5　FPGA 结构

3.3.5　硬件描述语言

为了能够实现以描述硬件的方式来构造卷积神经网络，我们首先需要对不同级别的硬件描述语言有一个大致的了解。

逻辑门级描述：通过直接绘制各种逻辑门组成电路的形式。

RTL（Register Transfer Level，寄存器传输级）指：不关注寄存器和组合逻辑的细节（如使用了多少逻辑门，逻辑门之间的连接拓扑结构等），通过描述寄存器到寄存器之间的逻辑功能描述电路的 HDL 层次。RTL 级是比门级更高的抽象层次，使用 RTL 级语言描述硬件电路一般比门级描述简单高效得多。综合是指将 HDL 语言、原理图等设计输入翻译成由与、或、非门等基本逻辑单元组成的门级连接（网表），并根据设计目标与要求，也就是约束条件优化所生成的逻辑连接，输出门级网表文件。通俗来讲，RTL 代码不是在“写代码”，是在画电路结构。RTL 代码需要“画”出输入输出端口，各级寄存器，寄存器之间的组合逻辑和前三者之间的连接。对于组合逻辑，只需要软件级描述，将其功能包装在“黑匣子”中即可，无需考虑其门级结构。

HLS(High-Level Synthesis) 高层综合：就是将 C/C++ 的功能用 RTL 来实现，将 FPGA 的组件在一个软件环境中来开发，这个模块的功能验证在软件环境中来实现，无缝的将硬件仿真环境集合在一起，使用软件为中心的工具、报告以及优化设计，很容易的在 FPGA 传统的设计工具中生成 IP。传统的 FPGA 开发，首先写 HDL 代码，然后做行为仿真，最后做综合、时序分析等，最后生成可执行文件下载到 FPGA 使用，开发周期比较漫长。使用 HLS，用高级语言开发可以提高效率。因为在软件中调试比硬件快很多，在软件中可以很容易的实现

指定的功能，而且做 RTL 仿真比软件需要的时间多上千倍。

Vitis AI 开发环境可在赛灵思硬件平台上加速 AI 推断，包括边缘器件和 Alveo™ 加速器卡。此环境由经过最优化的 IP 核、工具、库、模型和设计示例组成。其设计以高效和易用为核心，旨在通过赛灵思 FPGA 和自适应计算加速平台 (ACAP) 来充分发掘 AI 加速的全部潜能。Vitis AI 开发环境将底层 FPGA 和 ACAP 的繁复细节加以抽象化，从而帮助不具备 FPGA 知识的用户轻松开发深度学习推断应用。

Vitis AI 具有如下功能特性：

- 支持主流框架和最新模型，能够完成多样化的深度学习任务。
- 提供一整套经过预优化的模型，这些模型可直接部署在赛灵思器件上。
- 提供强大的量化器，支持模型量化、校准和微调。对于高级用户，赛灵思还提供了可选 AI 优化器，它能以可接受的精度损失对模型执行高达 90% 的剪枝。
- 提供逐层分析，以帮助识别瓶颈。
- 提供统一的高层次 C++ 和 Python API，从而最大程度提升从边缘到云端的可移植性。
- 自定义高效且可缩放的 IP 核，满足诸多应用的不同需求，如吞吐量、时延和功耗等方面的需求。

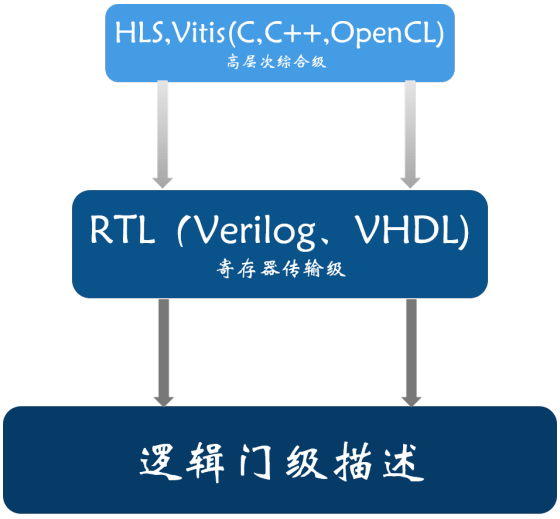

图 3-6　硬件描述语言结构层次

第四章　关键技术

4.1　计算机体系结构

一、CPU 生态及国产 CPU 发展态势

CPU 生态包括硬件和软件。在硬件方面，从指令集和 IP 核心到接口、板和机器制造商，所有硬件链路都相互控制，形成硬件协作生态系统；对于运行的软件、操作系统、编译器、Java NET 等核心软件、会议软件等顶级软件，存在相互适应，形成了密集的软件生态。一旦处理器生态系统形成，它就非常稳定。生态比技术更重要。历史上曾有一些处理器产品具有出色的性能，但它们的 Windows 生态系统较低。当处理器的生态形成时，它是一种寡头格局，强者恒强。第三方生产商将无法单独通过商业手段打破竞争格局，而是将成为一个新的世界级巨头，完全依靠行政权力和足够大的内部市场。目前，处理器行业由两个生态系统主导：一个是基于 X86 控制系统和 Windows 操作系统的 Windows。第二个是基于 ARM 指令系统和 Android 操作系统的 ARM 系统。

硬件结构的发展是计算机体系结构发展的重要一环，关注各国生产 CPU 的发展态势有助于本国的计算机体系生态构建。中国的信创产业发展是中国乃至全球信息产业一次格局重构，其过程就是中国基础软硬件厂商的崛起，这已经不仅仅是对 Windows 架构安全与否的质疑，更是要发展中国 IT 产业完整的产业链和核心竞争力。CPU 是信息产业的基础硬件底座，是整个信创产品中最重要的环节，整个软件生态架构都建立在底层 CPU 架构之上。信创加快推进，CPU 国产化先行。目前，国内共有海光信息、兆芯、龙芯中科、华为鲲鹏、飞腾信息和申威科技六家国产 CPU 厂商。海光和兆芯凭借 X86 架构拥有最强生态，能够兼容目前存在的数百万款基于 X86 指令集的系统软件和应用软件；鲲鹏和龙芯基于 ARM 架构在移动端一家独大，逐步向桌面和服务器延伸，生态建设较为繁荣。龙芯和申威从 MIPS 和 Alpha 转向自主指令集，逐步构建自主生态。国产 CPU 性能参差不齐，但总体上逐步向国际主流水平逼近。不同指令集架构下部分产品参数如核心数、超线程对产品性能影响程度不同，海光 7285 可对标 Intel 在 2020 年发布的铂金级旗舰芯片，鲲鹏 920 可对标 Inte1 在 2017 年发布的 Xeon Platium8180，兆芯的开胜 KH-30000 可对标 Inte1 在 2017 年发布的酷睿 i5，飞腾 S2500 可对标 Inte1 在 2016 年发布的至强 E5，龙芯 3C5000 性能基本与典型 64 核 ARM 相当。

从政策角度来看，近年来欧美频繁对我国关键科技领域发动科技封锁及贸易战，影响了我国创新领域的正常发展，因此在核心关键领域的国产替代为大势

所趋、迫在眉睫：从行业角度看，目前在 CU 等核心领域，国产化率较低，国产替代仍存在较大空间，而目前一些国产产品已由可用完善至好用，在满足国产替代在党政军及重点行业的替换之后，伴随生态及产品不断迭代完善，有望迎来更广阔的自主发展空间：从市场角度来看，国产替代产品近几年捷报频传，不断有代表性产品发布上市，并且部分厂商业绩迎来拐点初步验证国产替代逻辑，后续有望持续爆发。

国产 CPU	产品型号	发布时间	用途	内核	主频	工艺
海光	7285	2020Q1	服务器	32	3.0GHz	14nm
兆芯	KX-6000	2019Q2	桌面	8	3.0GHz	16nm
鲲鹏	920-7260	2019Q1	服务器	64	2.6GHz	7nm
飞腾	S2500	2020Q3	服务器	64	2.1GHz	16nm
	D2000	2020Q4	桌面	8	2.3GHz	14nm
龙芯	3C5000	2022Q2	服务器	16	2.2GHz	12nm
申威	1621	2018	桌面/服务器	16	2.0GHz	28nm
Intel	酷睿 i9-12900KS	2022Q1	桌面	16	5.5GHz	Intel 7nm
	至强铂金 8368Q	2021Q1	服务器	38	3.70GHz	10nm

图 4-1　　国产 CPU.

CPU 系科技行业算力基础，而技术及生态是行业发展的关键要素。CPU 是计算机的运算和控制核心，作为算力基础，是整个科技行业的底座，是多个创新方向的基础支撑。纵览 CPU 发展历史，每个 CPU 巨头的崛起都是技术与生态的“共振”从而引领时代发展，如 Intel 凭借性能及“WINDOWS”联盟称霸服务器及桌面端，ARM 架构凭借能耗及“ARM 体系”在移动时代异军突起。目前来看，CPU 市场仍主要为海外巨头占据，国内 CPU 厂商起步较晚仍处于劣势，但十四五以来国产 CPU 自主研发受到高度重视，目前已走出三条独立自主化道路，国家政策层面始终高度重视“卡脖子”领域自主创新度，国产 CPU 产品也捷报频传，技术产品层面不断突破，政策支持叠加行业发展加快生态建设，国产 CPU 或迎来重大发展机遇。

二、关键技术及框架介绍

（一）Watch

这是一种用于在架构级别分析和优化微处理器功耗的框架。Watch 比现有布局级电动工具快 1000 倍或更多，并且在其估计值的 10% 以内保持准确度，这些准确性已在前沿设计中使用行业工具进行了验证。

（二）ARM 架构

ARM 架构是一个 32 位精简指令集处理器架构，其广泛地使用在许多嵌入式系统设计。由于节能的特点，ARM 处理器非常适用于移动通讯领域，符合其主要设计目标为低耗电的特性。如今，ARM 家族占了所有 32 位嵌入式处理器

75% 的比例，使它成为占全世界最多数的 32 位架构之一。ARM 处理器可以在很多消费性电子产品上看到，从可携式装置到电脑外设甚至在导弹的弹载计算机等军用设施中都有它的存在。

（三）RISC-V 架构

RISC-V 架构是基于精简指令集计算（RISC）原理建立的开放指令集架构（ISA），RISC-V 是在指令集不断发展和成熟的基础上建立的全新指令。RISC-V 指令集完全开源，设计简单，易于移植 Unix 系统，模块化设计，完整工具链，同时有大量的开源实现和流片案例，得到很多芯片公司的认可。

RISC-V 架构的起步相对较晚，但发展很快。它可以根据具体场景选择适合指令集的指令集架构。基于 RISC-V 指令集架构可以设计服务器 CPU，家用电器 CPU，工控 CPU 和用在比指头小的传感器中的 CPU。

（四）MIPS 架构

MIPS 架构是一种采取精简指令集（RISC）的处理器架构，1981 年出现，由 MIPS 科技公司开发并授权，它是基于一种固定长度的定期编码指令集，并采用导入/存储（Load/Store）数据模型。经改进，这种架构可支持高级语言的优化执行。其算术和逻辑运算采用三个操作数的形式，允许编译器优化复杂的表达式。如今基于该架构的芯片广泛被使用在许多电子产品、网络设备、个人娱乐装置与商业装置上。最早的 MIPS 架构是 32 位，最新的版本已经变成 64 位。

三、计算机设计的主要方法

计算机设计的主要方法分别是从上往下、从下往上、从中间向两边。目前最佳的设计方式是第三者。

（一）“从顶向低”设计

先考虑满足应用要求，定好面向应用的那个虚拟机器级的特性和环境，在解决优化问题上，逐级向下设计，每设计下一级，考虑对下一级的优化。在设计效率的问题上，因为从顶向下，所以效率极低。通用机的应用对象和环境不停改变，一旦改变，软硬件很快就会不适应，系统效率极具下降。其传统机器级和微程序机器级，都在已有机器中选型。通常不专门设计。

（二）“由底向顶”设计

先不管应用要求，按照已有机器的特点来做器件，将传统机器机器级和微程序机器级研发。然后适配不同领域的操作系统和编译系统软件，采用合适的系统软件和算法来满足应用需求。硬件不能改变，仅靠设计软件来被动适配硬件。有时候仅仅需要稍微更改或增加硬件的功能就能大大简化软件设计，都做不到。软硬件脱节，软件得不到硬件的支持而变得繁杂，部分机器指标不真实。串行设计，会延长设计周期。同样已经很少使用。

（三）“由中间向两边”设计

由中间向两边的方法较为通用，克服前两种软硬件设计分离和脱节的问题，从中间开始设计。往往会先选择一个软硬件界面，然后从这个界面，分别向上下进行软硬件设计。

4.2 高性能计算

一、提高处理器性能的关键技术

提高通用应用程序性能的新方法之一使将硬件可编程资源整合到处理器微体系结构中。通过对编译时例程和硬件综合工具耦合性的分析，我们自动配置一组给定的硬件可编程功能单元，也就是 PFU，用来扩充基本的指令集架构，用来使其更好地满足每个应用程序的指令集需求。我们将这种新型通用计算机称为可编程指令集计算机，也就是 PRISC。尽管在概念上他们比较类似，但 PRISC 方法与动态可编程微代码是有不相同的，因为在 PRISC 中我们定义了崭新的比较初始的数据通路操作。PRISC 的微型体系结构设计，也就是具有单个 PFU 的 RISC 微处理器。研究表明，处理器性能在 SPECint92 的基准测试中我们提高了 22%。几十年来，高性能计算的社区一直专注于性能，其中性能被定义为速度。为了在每个计算节点实现更好的性能，微处理器供应商不仅每 18 到 24 个月内将晶体管的数量和速度翻一番，而且还将功率的密度增加一倍。因此，保持大型的 HPC 系统正常运行需要在大型机房中持续进行冷却，这也导致了大量的运营成本。此外，功率密度的增加有可能部分地导致系统可靠性下降，从而引发了生产力下降。

二、图形算法及性能提高

在过去的二十年里，并行处理器逐渐成为一种商品——图形处理器单元，也就是我们常说的 GPU，这种形式的通用处理器，以及张量处理器单元，即 TPU，和图形处理器等特定领域的处理器在 DARPA HIVE 计划下开发的处理器。对开发并行硬件的研究以及 DIMACS 和 HPEC 图形挑战等举措已成功加速图形算法。然而，图形性能的提高是以更具挑战性的编程模型为代价的。结果是用户和图形算法设计者更喜欢使用的高级语言（例如 Python）与并行硬件的编程语言（例如 C++、CUDA、OpenMP 或 MPI）之间不匹配。GraphBLAS 的计算模型是为了解决这个问题而发展的产物。无论是以顶点为中心、以边为中心还是基于线性代数，图形框架的 GPU 实现在实现高性能方面都面临着几个常见的挑战。

（一）细粒度负载不平衡

我们在处理图问题时，图问题中最直接的并行形式是跨顶点并行。然而，在许多的图中，尤其是无标度图中，每个顶点的出站边数可能会有很大差异。因此，每个顶点的工作量以相同的方式变化。因此，将顶点分配给相邻线程的 GPU 实现会导致这些相邻线程之间出现显着的细粒度负载不平衡。这种不平衡与稀疏

矩阵运算中的不平衡相同，每行具有可变数量的非零元素，选择为每个矩阵行分配一个线程。原生图框架通常通过各种技术解决这个问题，包括根据工作负载的大小动态地对顶点进行装箱，以及使用适当大小的计算粒度处理类似大小的箱，或者使用前缀将顶点并行性转换为边并行性-sum-like 方法。挑战在于选择正确的负载平衡方法并平衡负载平衡的成本与其性能优势。

（二）最小化开销

在每个输入元素需要大量工作的大型负载平衡数据集上运行的 GPU 内核可达到其峰值吞吐量。然而，在图形计算过程中，GPU 框架可能经常面临运行时间不是由处理时间支配而是由开销支配的情况。当 GPU 没有足够的工作来保持整个 GPU 忙碌时，就会出现一种形式的开销；在这种情况下，内核的运行时间主要由内核启动的成本而不是工作成本决定。批量同步操作的一种相关形式的开销是在每个内核末尾进行全局同步的要求；对于第一顺序，整个 GPU 必须等到内核处理的最后一个元素完成后才能启动下一个内核。

另一种形式的开销是当使用多个内核来执行可以组合成单个内核（“内核融合”）的计算时，这种计算可能会利用内核中的生产者-消费者局部性。内核启动的开销大约为几微秒。因此，对于需要多次迭代且每次迭代有许多内核启动的图形计算，内核启动的总成本非常高。因此，最小化内核启动是任何高性能图形框架的重要目标。

4.3 计算机硬件与人工智能

在此小节中我们主要介绍如何将卷积神经网络 LeNet-5 部署到 FPGA 中。

4.3.1 卷积层的硬件实现

卷积操作主要涉及如下几个属性：Width：输入数据的行数；Length：输入数据的列数；K ：卷积核的行列数；img[][] ：输入数据；Kernel[][]：卷积核数据；通过滑窗并且乘加的方式我们可以实现卷积操作，我们以二维数据、单卷积核、步长为 1 的卷积操作为例，为了实现内存的连续读取，我们可以利用如下方法将卷积操作转化为矩阵乘法操作：

如图，由于每次卷积操作便是卷积核在图像上对应区域每一像素与卷积核对应位置权重的乘积并加和，因此我们可以提取出每次要做乘法的图像框并将其拉长为一维行向量，并在卷积核每一次滑动后所对应的图像框皆拉成行向量并存储为一个二维矩阵，随后将卷积核拉长为一维列向量，这样，两个矩阵相乘得到的结果便为我们普通卷积操作所得到的结果。其中，拉成矩阵的图像行数为 $(length-k+1)*(width-k+1)$，列数为 $k*k$。

同理，我们也可以根据此方法对于多通道图像、多个卷积核的卷积操作转化

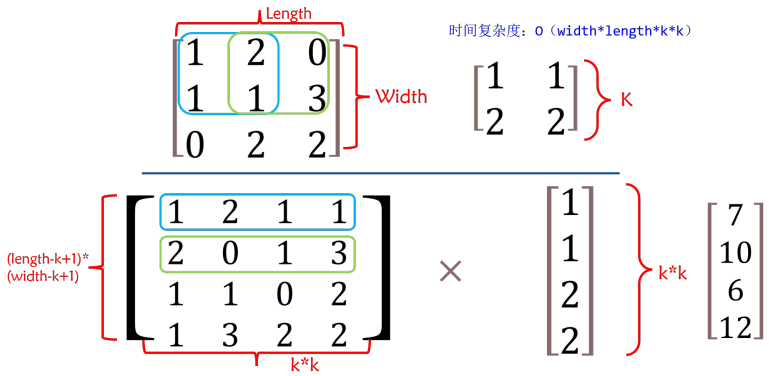

图 4-2　卷积操作转为矩阵乘法

为矩阵乘法：

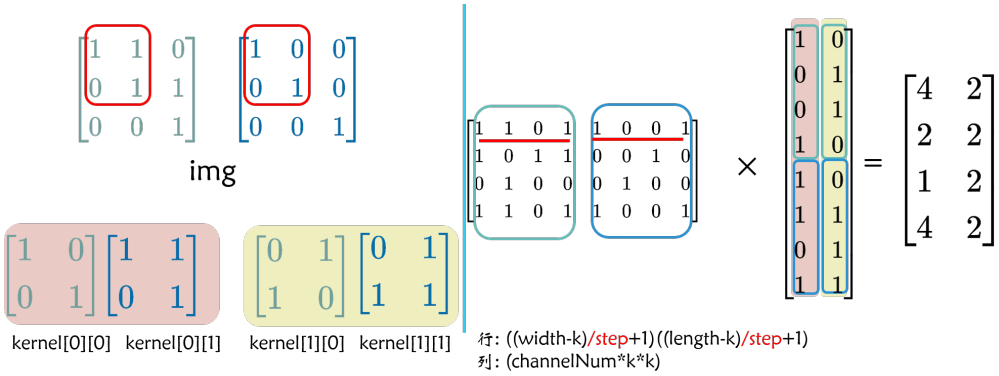

图 4-3　三维多卷积核卷积操作

此时，图像矩阵列数为：$channelNum*k*k$，行数为：

$$\left(\frac{width-k}{step}+1\right)*\left(\frac{length-k}{step}+1\right) \tag{4-1}$$

卷积核矩阵的行数为：$k*k$ 列数为：$kernelNum$ 。

4.3.2　池化层、全连接层的硬件实现

在池化层，我们只需要通过依次滑动窗口并计算相应的最大值或者平均值来得到池化后的结果，而在全连接层我们只需要将相应的权重与输入图像做乘加操作，并输出即可得到想要的结果。

4.3.3　卷积操作关键代码

```
//图像平铺为矩阵
for (int c = 0; c < channelNum; c++)
```

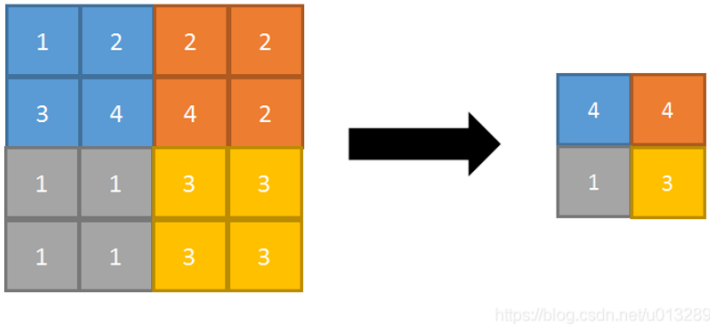

图 4-4　　最大池化操作

```
    {
        for (int i = 0; i < width - k + 1; i += step)
        {
            for (int j = 0; j < length - k + 1; j += step)
            {
                int imgMatrixline = (i / step) * ((length - k) / step + 1) + j / step;
                for (int m = 0; m < k; m++)
                {
                    for (int n = 0; n < k; n++)
                    {
                        if (ifPadding == 0)
                        {
                            imgMatrix[imgMatrixline][(m * k + n) + c * k * k] =
    inputImg[c][i + m][j + n];
                        }
                        else if (ifPadding == 1)
                        {
                            imgMatrix[imgMatrixline][(m * k + n) + c * k * k] =
    paddingInputImg[c][i + m][j + n];
                        }
                    }
                }
            }
        }
    }
//卷积核平铺为矩阵
for (int n = 0; n < kernelNum; n++)
    {
        for (int c = 0; c < channelNum; c++)
        {
```

```
            for (int i = 0; i < k; i++)
            {
                for (int j = 0; j < k; j++)
                {
                    kernelMatrix[i * k + j + c * k * k][n] = kernel[n][c][i][j];
                }
            }
        }
    }
//矩阵乘法
for (int i = 0; i < imgMatrixWidth; i++)
    {
        for (int j = 0; j < kernelMatrixLength; j++)
        {
            int tempRes = 0;
            for (int m = 0; m < imgMatrixLength; m++)
            {
                tempRes += imgMatrix[i][m] * kernelMatrix[m][j];
            }
            convRes[i][j] = tempRes;
        }
    }
```

第五章　主要研究机构

5.1　计算机体系结构

5.1.1　微处理器的发展

自 1960 年代以来，许多人员就研究致力于计算机架构复杂性降低——其中最著名的是 IBM 801 项目——由 Hennessy 和 Patterson 分别在斯坦福和伯克利领导的研究项目，其确立了该架构的可行性 RISC 方法并普及了相关概念，同时将其介绍给学术界和工业界。RISC 方法与当时流行的复杂指令集计算机（CISC）计算机的不同之处在于，它需要一小组简单通用的指令（计算机必须执行的功能）。与复杂指令集相比，RISC 方法所需晶体管更少，且减轻了计算机的执行工作。

创建 RISC 术语的是 Berkeley Patterson 团队，他们在 1982 年建造而且演示了新型处理器，称为 RISC-1 处理器。RISC-1 原型拥有 44000 个晶体管，比使用 100000 个晶体管的传统 CISC 设计更大。轩尼诗于 1984 年由 MIPS 计算机系统公司等创立，以促进斯坦福团队的工作。后来，Sun microsystems 促进了伯克利团队在 SPARC 微体系结构中的工作。尽管许多专业领域的架构师最初对 RISC 架构有着猜测的想法，但是 SPARC 和 MIPS 的成功创业、RISC 设计的较低生产成本以及更多的研究进展导致 RISC 得到更广泛的接受。到 20 世纪 90 年代中期，RISC 微处理器在整个领域占据主导地位。

随着人工智能算法和高清传感器的需求不断增长，当代微处理器设计在内存带宽（即冯诺依曼瓶颈）、处理速度和功耗方面面临巨大挑战。内存计算（IMC）利用器件技术和设计技术的进步，在内存阵列中嵌入模拟深度学习运算，实现高存储密度的大规模并行计算。另一方面，其性能仍然受到器件非理想性、电路精度、片上互连和算法特性的限制。

在最新的 RSS2 会议之中，亚利桑那州立大学的计算机研究所将关注重点放在了基于稳健 RRAM-b 的 IMC 设计。基于全集成 65nm CMOS/RRAM 测试芯片的统计数据，我们将说明当前 IMC 系统的瓶颈，包括 RRAM 变化、机器学习模型的稳定性等。它们相互作用，限制了片上推理的准确性。我们将演示两种恢复精度损失的方法：在映射到硬件之前训练模型稳定性，以及用于映射后恢复的混合 SRAM/RRAM 架构。这些方法应用于各种数据集以及 65 纳米 SRAM/RRAM 测试芯片，有助于阐明未来 IMC 的研究重点。

5.1.2 体系结构发展与安全性

在 2022 年的研究中，虽然计算机系统架构领域的研究人员们关注到了 DNN 的效率，但 Software 2.0 暴露了健壮性、安全性和弹性方面等一系列问题，这些问题也是 Software 2.0 成为普遍计算范例之前的主要障碍。例如，对输入的小扰动很容易“愚弄”DNN 以产生不正确的结果，从而引发所谓的对抗性攻击。同样，虽然 DNN 通常对硬件故障具有弹性，但很少有人研究过 DNN 对硬件故障的最坏情况弹性，这通常决定了很关键的系统安全性。

提高 Software 2.0 的健壮性、安全性和弹性必然是一项跨层任务，就像算法、编程语言、体系结构和电路设计在 Software 1.0 时代聚集在一起一样。不要过度优化单个系统组件也很重要；相反，我们必须采用全系统方法来理解端到端系统的要求和约束，这些系统通常是多芯片的，并且跨越客户端、边缘和云。

在最新的 RSS2 会议之中，特定于应用程序的加速在云计算和数据中心盛行。但当前的基础架构设计很少或根本不支持外部加速器的安全性。现有的可信计算解决方案（如 Intel SGX 或 ARM TrustZone）仅针对纯 CPU 环境，而使外部加速器和外围设备得不到保护。这项工作提出了 AccGuard，这是一种扩展远程 FPGA 加速器信任计算的新方案。AccGuard 由一个安全管理器 (SM) 组成，该安全管理器具有硬件信任根和通过标准加密原语进行的远程证明，以形成 FPGA 加速器的飞地框架。与最先进的基于 CPU 的 enclave 框架 Intel SGX 相比，它最大限度地减少了性能开销（由于安全功能）

5.2 高性能计算

参考高性能计算顶级会议–超算高性能计算、网络、存储和分析国际会议。

高效的 GPU 资源调度对于最大化资源利用率和节省培训成本至关重要，以应对共享 GPU 集群中不断增加的深度学习工作负载。现有的 GPU 调度器在很大程度上依赖于静态策略来利用深度学习作业的性能特征。然而，由于缺乏弹性，它们很难达到最佳效率。为了解决这个问题，新加坡国立大学研究学者 Zhengda Bian 的研究团队提出了 ONES，一种用于弹性批量编排的在线进化调度程序。ONES 根据训练 batch size 自动管理每个作业的弹性，从而最大化 GPU 利用率，提高调度效率。它通过可以持续优化调度决策的在线进化搜索来确定每个作业的批量大小。他们用 64 个 GPU 评估 ONES 在 TACC 上的有效性 s Longhorn 超级计算机。结果表明，ONES 可以以更短的平均作业完成时间胜过之前的深度学习调度程序。

在现代数据中心，大量并发图形处理作业正在大型图形上进行处理。然而，现有的硬件/软件解决方案存在不规则的图形遍历和激烈的资源争用。在本文中，华中科技大学高性能研究团队提出了 LCCG，这是一种以局部为中心的可编程

加速器，可增强众核处理器以实现更高的并发图处理作业吞吐量。具体来说，华中科技大学高性能研究团队在加速器设计中开发了一种新颖的拓扑感知执行方法，以根据图拓扑动态调整多个作业的图遍历，从而能够完全整合来自并发作业的图数据访问。通过在更多作业之间重用相同的图形数据并合并这些作业的顶点状态的访问，LCCG 可以提高核心利用率。华中科技大学高性能研究团队在模拟的 64 核处理器上进行了大量实验。结果表明，LCCG 仅以 0.5% 的额外面积成本将前沿软件系统的吞吐量提高了 11.3 23.9 倍。此外，LCCG 比最先进的硬件图形处理加速器（分别为 HATS、Minnow 和 PHI）获得了 4.7 10.3、5.5 13.2 和 3.8 8.4 倍的加速。

5.2.1 多处理器系统

随着在新兴的实时应用程序中实现越来越复杂的功能，需要多处理器系统来实现高性能，并且有向无环图 (DAG) 用于对功能依赖性进行建模。在这项工作中，我们研究了在同类多处理器平台上运行的单个周期性非抢占式 DAG，这是许多领域的常见设置，例如汽车、机器人和工业自动化。为了减少 DAG 的 makespan 并提供紧密而安全的界限，约克大学的研究团队的贡献涉及节点级并行性和节点间依赖性的利用，这是 DAG 拓扑的两个关键因素。首先，他们引入了一个并发的提供者和消费者 (CPC) 模型，它精确地捕获了上述两个因素，并且可以在解析 DAG 时递归应用。

其次，提出了新的响应时间分析，它为非关键节点的任何执行顺序提供了通用界限，并为固定的此类顺序提供了特定（更严格）的界限。综合评估表明，我们的调度方法和分析优于最先进的方法。它为非关键节点的任何执行顺序提供了通用界限，并为固定的此类顺序提供了特定（更严格）的界限。综合评估表明，我们的调度方法和分析优于最先进的方法。它为非关键节点的任何执行顺序提供了通用界限，并为固定的此类顺序提供了特定（更严格）的界限。综合评估表明，我们的调度方法和分析优于最先进的方法。

5.2.2 嵌入式计算机近期研究

通过阅读第 28 届 IEEE 实时和嵌入式技术与应用研讨会中的相关文章，我们收集分析到了最新的嵌入式计算机研究方向和内容。

安全关键系统的设计和实施受到综合标准的监管，作为预防危险事件的一种手段。例如，错过时序截止日期是不可接受的事件，解决方案是使用足够的时序界限来表征这些系统的时序行为。最坏情况时序分析能够计算出安全（和精确）的时序界限并提供必要的保证。

最先进的静态时序分析器，如 aiT 、Otawa 、Heptane 或 Chronos 考虑架构模型（缓存或管道）并使用静态分析来表征它们的时序。这些架构模型通常是

手工开发的，有时会针对硬件模拟器进行一致性验证（例如 aiT). 对其管道模型进行更仔细的代码检查会暴露出它们的共同属性。首先，流水线阶段用单个变量表示，将流水线阶段简化为标识属性。其次，这些模型关注的是指令进程而不是它的实际计算（即程序执行的正确性是在静态时序分析的上下文中假设的）。第三，这些模型执行基本块（即直线代码），因此，像转发这样的流水线优化没有明确编码在硬件流水线模型中，而是通过任意时序惩罚在代码级处理。

在硬件的不断发展之下，许多处理器设计变得可用 1，由高级 HDL（如 Chisel）和相关的可扩展编译链（如 FIRRTL ）支持. 这些编译链带有可配置的优化和传递基础结构，有助于硬件设计的分析（更难从 Verilog/VHDL 解决）。因此，在本文中，我们提出了对基于 Chisel/FIRRTL 的处理器设计的分析，以确定用于静态时序分析的流水线模型。我们的分析解决了 (1) 如何确定流水线深度（即级数）和 (2) 如何连接这些级，以形成数据路径，以后可以很容易地将其抽象出来以匹配 WCET 分析器的数据路径。此分析侧重于流水线寄存器以捕获周期准确的行为，如 WCET 分析所要求的，同时改进 WCET 分析器的流水线模型。更准确地说，我们的分析可以暴露寄存器转发，在体系结构和程序模型之间创建关注点分离，并使该模型适用于更广泛的时序属性（例如时序异常、安全相关等）。他们将分析应用于多个处理器并报告初步结果。我们的工作解决了（硬件设计的）代码级分析以提取用于时序分析的流水线模型，和 [14] 中的工作解决了类似的目标，即分析处理器（Verilog/VHDL）设计代码。我们的方法考虑使用更具表现力的语言（Chisel/FIRRTL）开发的设计。与我们设计解决方案然后旨在验证它的方法相反，[13] 中的工作使用抽象解释框架来构建声音处理器抽象模型。这项特殊工作应用于 [14] 使用半自动程序确定体系结构模型，而我们的分析在构建管道模型时是完全自动化的。抽象管道模型（包括转发）在中得到解决，来自处理器图（即组合组合逻辑和顺序逻辑的结构，可以从 Verilog/VHDL 代码生成）。实际构造也依赖于寄存器放置，作为我们的方法，但需要输入流水线深度。此外，的目标是正式验证流水线优化的功能正确性，这与我们的保持时序行为的功能是互补的。

复杂的嵌入式系统现在支持多个操作系统的共存，以管理曾经分配给单独的嵌入式微控制器的服务。例如，汽车系统现在使用多个操作系统来整合中央嵌入式计算平台上的电子控制单元 (ECU) 功能。此类平台具有工业嵌入式 PC 的复杂性，具有多核和硬件虚拟化功能。这使分区管理程序能够在空间和时间上与单独的来宾操作系统共享物理机，这些操作系统管理不同关键级别的服务。然而，PC 级硬件在引导操作系统和相关应用程序级服务时会产生较大的延迟。固件 BIOS 执行开机自检，然后将操作系统映像从可启动存储设备加载到内存中。这种延迟在时间关键的嵌入式系统中是不可接受的，其中重要的服务必须在系

统启动后的几毫秒内运行。

在本文中，我们介绍了 Jumpstart，这是一种 PC 级电源管理方法，可最大限度地减少用于嵌入式系统的分区管理程序的唤醒延迟。我们展示了 Jumpstart 如何在大约 600 毫秒内从低功耗挂起状态恢复关键操作系统服务和任务，并将整个系统启动延迟减少 23 倍。此外，与需要从先前通电的系统启动系统的方法相比，Jumpstart 消耗的电量最少-关闭状态。相比之下，另一种固件优化的引导加载程序，称为 Slim，将引导延迟减少了 1.8 倍。重要服务必须在启动系统后的几毫秒内运行。

5.3 计算机硬件与人工智能

5.3.1 Vitis

Vitis 软件平台为人工智能的嵌入式硬件开发者提供了一个很好的开发方式。Vitis 统一软件平台包括：全面的内核开发套件，可无缝构建加速应用；不断发展的硬件加速合作伙伴库和预建应用生态系统；Vitis Model Composer 是一款基于模型的设计工具，支持在 MathWorks MATALB^R 和 Simulink^R 环境中进行快速设计探索和验证，并加速 Xilinx 器件的量产；Vitis Networking P4 允许创建软定义网络。VitisNetP4 数据平面构建器生成的系统可以针对从简单的数据包分类到复杂的数据包编辑的各种数据包处理功能进行编程。

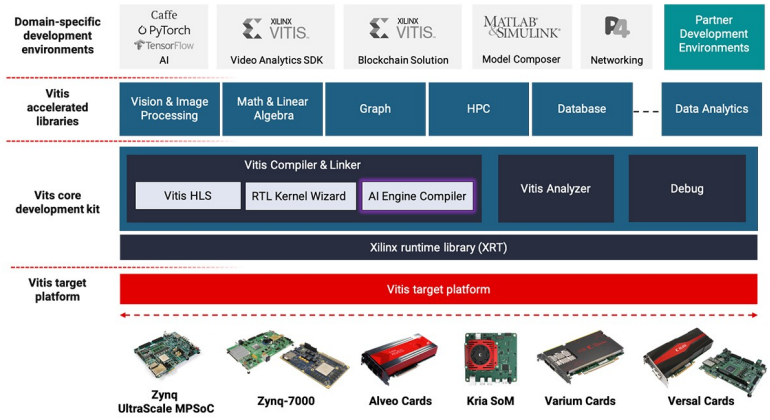

图 5-1　　Vitis 软件平台

从 Vivado 2019.2 开始，对应的软件开发环境从 SDK 变成了 Vitis。2019.1 及之前版本，SDK 是 Vivado 的一个附属组件，可以从 Vivado 启动 SDK。从 2019.2 开始，vitis 不能从 vivado 启动，必须单独启动。地位和 vivado 齐平了，不在是一个附件。

第六章　重点企业

6.1　NVIDIA 英伟达篇

一、著名产品

NVIDIA –GeForce 256 , 一款用于实时图形处理的专用处理器。随着实时图形处理技术的进步，GPU 拥有了可编程性。图形实时处理应用需要高内存带宽和大量的浮点计算能力用于 vertex 和 fragment 的着色计算。可编程性与浮点计算能力的结合，使得 GPU 对于科学计算拥有了巨大的吸引力。

1999 年推出的 GeForce 256 是第一款将顶点变换和光照以及片元计算集成到一块芯片上的芯片。早期，科学家通过把科学计算映射为顶点和片元着色器 (shader) 来使用 GPU 的计算能力。这一阶段，GPU 通过不断的降低硬件编程的复杂度，同时提供双精度浮点计算等通用能力来满足科学计算领域的需求。

随着 PC 游戏越来越复杂，为了实现更加复杂的游戏效果，对修改顶点和像素的计算需求也与日俱增。从 20 世纪 80 年代开始，可编程的“着色器 (shaders)”已经被应用在离线图形软件中。GeForce 3(NV20) GPU 在 2001 年发布了一种可编程的顶点着色器 (vertes shader) 特性，2002 年在 GeForce FX 上发布了可编程的片元着色器 (fragment shaders) 特性。

GeForce，是由英伟达公司开发的个人电脑的图形处理器品牌。第一款 GeForce 产品是为高边际利润（高阶）游戏社群和计算机用户的市场所设计开发，但是后来的产品发布扩展了产品线，涵盖图形市场的所有细分市场，从低阶、中阶到高阶。到 2020 年为止，GeForce 的设计已经包含 16 个世代。它现在的竞争对手是 AMD 的 Radeon 系列图形处理器。NVIDIA 亦拥有定位于专业图形处理领域的 Quadro 系列，多数产品使用与 GeForce 相同的核心，GeForce 的早期产品甚至可以借由改刷固件的方式软改为 Quadro 系列。

二、发展趋势

在 2019 中国 GTC 大会选择中的主题，GPU 的未来趋势无外乎 3 个：大规模扩展计算能力的高性能计算（GPGPU）、人工智能计算（AIGPU）、更加逼真的图形展现（光线追踪 Ray Tracing GPU)。虽然 GPU 的最基本功能-显示技术在大会主题中没有“显式”的提及，但是众多应用方向均与之密切相关，譬如：智慧医疗和生命科学、游戏、虚拟现实/增强现实、工业设计与工程、自动驾驶与交通等，因此支持更加清晰和动感的高清显示是无需强调的未来趋势。此外，由于 GPU 越来越广泛地应用到手机、终端、边缘计算节点等嵌入式设备，所以高效能也是一个永恒的追求。

正如我们在十年前的文章中大体预测的那样，GPU FP64 性能从 2010 年的 Fermi 到 2020 年的 Ampere，十年间增长了 20 倍，年增长率为 35%（见图 5）。我们预计这一趋势将会在未来的 GPU 迭代中持续，但是会放缓。虽然 2005 年 Dennard 微缩技术的终结停止了传统的电压微缩技术，但半导体特征尺寸仍在变得更小。随着 GPU 的技术曲线从 Ampere 的 7 nm，到 5 nm，3 nm 以及更高的工艺，我们将会在芯片上集成更多的器件和电容，因此一个特定功能所需的能量将会减少，单位面积或者单位功耗将产生更高的性能。深度学习推理的性能已经以很高的速率（见图 3）在增加，每年增速超过 100%。我们预计这一趋势将会持续几代，但是随着架构的成熟，主要增长来自技术的驱动，最终将会趋于每年 35% 。

GPU 性能持续提升的主要挑战来自于内存带宽。虽然像 Linpack 这样的基准测试可以很好的使用 cache 以保证 A100 的计算单元可以被填满，但是大多数 HPC 和光追图形应用依然受到带宽限制。一个经典的 HPC 应用，每个 FP64 算术操作需要从内存获取 1-4 bytes 的数据。例如，HPCG 基准测试是使用共轭梯度求解器代码的代表，它的访存计算比 (Byte/Flop) 达到 4 : 1。内存带宽是从 Pascal 一代开始出现重大飞跃，从此开始向高带宽内存（high-bandwidth memory, HBM）发展。Ampere 的 FP64 Tensor Core 和 HBM2E 的 DRAM 使得访存计算比达到 0.1。为了继续扩展内存带宽和 HPC 应用程序性能，我们预计还需要在封装和内存架构方面进行更多的创新。

GPU 是融入新的专用需求的理想硬件加速平台。它提供了可编程平台和高速的片上带宽，以及片外存储系统和高速的片外互联技术。专用需求的硬件可以以新的指令添加到 SM 中（就像为深度学习添加的 Tensor Core 指令），或者作为独立的内存事务，从 SM 启动相关操作（例如 RT cores）。我们预测，在未来几年，新的指令和基于内存的加速器将被集成到 GPU 中，以支持数据库、稀疏线性代数和生物信息学等应用领域。

另外，除了纵向扩展单个 GPU 的性能，我们预测 GPU 将被横向扩展到更大的集群，以赋能深度学习训练和 HPC 应用。如今，通过 NVLink 连接的 Ampere 系统能够实现每组超过 8 个 GPU 之间以 600 GB/s 的带宽共享彼此的内存，就像一块 GPU 一样。在未来几代中，我们预计芯片外通信带宽将继续扩大，NVLink 连接节点和 Infiniband 连接集群的规模将继续增长。我们也期望 GPU、CPU 和 DPUs(智能网卡) 之间更紧密的集成，以更少的开销实现更高带宽的通信和 I/O。

CUDA 和相关库的使用已经简化了 gpu 编程，超过 600 个 HPC 应用程序都是 gpu 加速的。由于有更好的工具和对编程的抽象，我们预计未来的 GPU 编程将更加简单。例如，Legate 编程系统允许 Numpy 程序在任意数量的 GPU 上运

行，从单个 GPU 到 4,480 个 GPU 集群 (如 Selene)。Legate 是建立在 Legion 之上的一个运行时系统，它能够管理多 GPU 应用的任务图及其数据模型。Legion 负责任务的调度和管理数据通信、复制和迁移，大大简化了多 gpu 系统的编程任务。

未来 GPU 编程的一个挑战是，从不同的 SMs 访问不同的内存位置时非统一内存访问 (Nonuniform memory access，NUMA) 所带来的固有开销。曾经 GPU 提供过一个统一内存访问（Uniform memory access，UMA）模型，从任意 SM 访问内存的任意位置开销都是均等的。随着 GPU 尺寸和多 GPU 系统规模的逐渐增大，要求所有的内存出现在同样距离的位置上变得不可能，也就导致 UMA 模型变得不可维护。我们预计向非统一内存访问模型的转变能够促使 SMs 接近内存的某些部分，以便它们能够利用局部性来提高性能和效率。这一转变，将要求 GPU 程序具有更好的局部性表达，以便 GPU threads 能够和他们所操作的数据具有更好的一致局部性。虽然程序员已经在多 GPU 系统中采用了这样的策略，但是我们希望 NUMA 模型在单 GPU 中也能够更有效。

自 GeForce 256 面世的 20 多年里，GPU 已经取得了长足的发展。它们在性能方面增加了 390 倍，同时变得高度可编程，并增加了专用领域的硬件来支持 HPC、深度学习和光线追踪。我们期待在未来 20 年里，在性能和功能上也会有类似的进步。

三、整理总结

除了观看采访外，我们还了解整理了英伟达本年度的发展方向和报告：

在 GTC2020 大会上，英伟达推出了安培架构的首款超算 GPU——A100。A100 引入了有着里程碑式意义的 Tensor Cores 双精度计算技术，这使得 A100 的算力比前一代 V100 提高了 175%。NVIDIA A100 Tensor Core GPU 针对 AI、数据分析和 HPC（high performance computing，高性能计算）等应用上，实现了更强的加速，针对极其严峻的计算挑战上有了更大作为。

作为 A100 GPU 系列中的最新力作，在架构特性上有如下特点：采用第三代 Tensor Core。使用新的 TF32，上一代 Volta 架构的 AI 吞吐量提高了 20 倍。使用 FP64，HPC 性能提高了 2.5 倍。通过 INT8，AI 推理性能提高了 20 倍，并支持 BF16 数据格式。使用更大更快的 HBM2e GPU 内存。因此，内存容量增加了一倍，超过 2TB/s 的内存带宽是业界第一。使用 MIG（多实例 GPU，多实例 GPU）技术，单个独立实例的内存增加了一倍，最多可提供七个 MIG，每个实例具有 10GB 内存。采用结构化稀疏技术，将稀疏模型的推理速度提高了两倍。

如今，分布式键值存储 (KVS) 正在成为最重要的存储范例之一，以补偿大规模集群上的传统文件系统。例如，云供应商广泛采用分布式 KVS，如 Cassandra、Dynamo 和 Memcached。高性能计算遵循类似的趋势，利用分布式 KVS 来提高

性能或提供前所未有的服务，其中一些已在我们之前的工作. 为分布式 KVS 实现高可靠性和可用性的最先进方法是复制，然而，它本质上具有以下缺点，显着的空间开销，额外的磁盘 I/O，以及更多的网络带宽消耗。

英伟达公司提出了两种技术来减轻这种 I/O 开销并提高键值存储性能：GPU 编码和位置感知编码。我们不是通过网络迁移完整大小的副本，而是将原始文件分成更小的块，并在将它们分散到远程节点之前使用 GPU 使用一些额外的奇偶校验码对它们进行编码。奇偶校验码通常比原始文件小得多，这节省了高可用性所需的额外空间并减少了 I/O 开销。同时，大量的 GPU 内核极大地加速了计算密集型编码过程。然而，将原始文件拆分为存储在多个节点上的更小块会破坏数据局部性从应用程序的角度来看。为此，我们提出了一种位置感知编码机制，允许在所需块所在的节点上将作业作为更细粒度的任务进行调度。因此，数据局部性被保存在子作业（即任务）级别的更细粒度上。我们对所提出的方法进行了深入分析，并实施了一个名为 Gest 的系统原型。Gest 已在各种测试平台上部署和评估，证明可以同时实现高数据可用性、高空间效率和高 I/O 性能。第三代 NVLink 和 NVSwitch，相较于上一代互连技术，可使 GPU 之间的带宽增加至原来的两倍，将数据密集型工作负载的 GPU 数据传输速度提高至 600 GB/s。

6.2 Intel 英特尔篇

一、公司介绍

英特尔成立于 1968 年，是全球最大的半导体芯片制造商，也是 36 年创新历史中的产品领导者。全球范围内，致力于为服务器、客户、互联网解决方案、在线通信互联网服务的互联网经济全球增长奠定基础。英特尔是全球领先的信息行业之一，在全球 500 家公司中，目前英特尔处于领先地位。2003 年，净收入和年收入达到 56 亿美元和 301 亿美元。由于中国信息科技市场的进一步发展，英特尔在上海科学园区设立了一个国外研发中心。公司凭借其市场优势和人才优势，英特尔（上海）研发中心有限公司研发中心目前在研发中心投资 1500 万美元。

研发中心是利用先进的技术、设备和科学的管理方法，研发先进的信息和通信技术产品，从而获得经济效益。理想主义有助于中国的经济发展。业务研发中心将具体包括：高科技领域（包括电子商务技术），以及英特尔产品和相关技术的研发转让和许可研发成果；测试英特尔产品和相关技术的生产；提供技术服务，包括测试、维护、咨询和解决方案。一系列英特尔产品，包括母公司产品及其子公司。为英特尔客户提供技术支持，以帮助和加强信息产业客户设备制造商和供应商生产基于体系结构的英特尔产品，并开发下一代新产品，以改进和增强英特尔产品的技术内容和功能。Intel SSG（软件解决方案部）、知名人士小组

(项目部)、MPG (移动部)、著名人士小组 (桌面部) 和 ICG 电信产品部 (CEG 消费电子服务部) 以及其他相关部门将进入科学园开展这些业务活动，包括开发下一代固件、编译器以及中国电子教室的数字家居、高性能计算、商业解决方案和电子学习开发技术创新活动。

二、发展趋势

英特尔的使命是创造技术，改善地球上每个人的生活，改变世界。该公司致力于使世界更安全，促进创新，提高生产力，建设健康和充满活力的社区，并利用其全球影响力改变世界。社会、商业和土地。改善并使行业同行更负责任、更具包容性和可持续性。我们感到迫切需要与其他国家合作，以应对世界上任何人都无法独自应对的挑战。英特尔为企业责任制定了新标准，并推动了我们全球网络的重大变革。从推动气候解决方案到多样性和包容性，英特尔将我们的合作伙伴和全球技术组合客户聚集在一起，以实现伟大的目标。

英特尔有一个强大的产品路线图，强调集成设计和制造的力量，以创造真正差异化的产品。凭借我们多样化的 xPU 产品组合和深入的客户合作，我们有能力巩固我们在 CPU 方面的领导地位。但要超越竞争对手，成为既定的领导者，在相邻细分市场建立影响力，使用数据和分析来推动决策制定，更快地学习和利用新机会。为此，我们需要更快地采取行动。我们投资于未来的制造领导地位。它还利用代工厂和供应商生态系统作为敏捷 IDM 2.0 模型的一部分。这种模式使客户能够信任以敏捷和规模交付的行业最佳创新。英特尔通过世界领先的工程设计、尖端工艺技术和对下一代计算架构的研究不断创新。领先的新芯片设计和技术，可加速客户在云、边缘和客户端中的高价值工作负载。随着世界和人类的各个方面都联机以及数字技术转型机会的增加，将继续追求激进创新领域。

6.3 Apple 苹果篇

一、公司介绍

苹果公司，前身为苹果电脑公司，总部位于美国加利福尼亚州库比蒂诺。其核心业务是电子技术产品，目前全球计算机市场份额为 7.96%。20 世纪 80 年代，麦金塔电脑的问世彻底改变了个人电脑。通过其创造性的硬件、软件和互联网技术和设备，苹果公司致力于为全世界的学生、教育工作者、创意专家和普通消费者带来最佳的计算机体验。1993 年苹果电脑公司北京办事处的成立，标志着世界上最大的电脑公司之一苹果公司正式进入中国市场，而 2008 年开始的苹果大学是一所培训苹果公司中层员工和管理人员的培训机构。该机构是由史蒂夫乔布斯和其他高级管理人员建立的。为了保持苹果文化的活力，乔布斯特意聘请了前耶鲁商学院院长乔尔波多尔尼 (Joel Podolny) 担任苹果大学首任校长。

乔布斯主张软硬件结合，从头到尾绝对控制。

创始人的风格成就了苹果的品牌核心价值观：诗意与工程紧密相连，艺术、创意与科技完美结合，设计风格既醒目又简洁。在 2007 年 1 月 9 日 Macworld Expo 的主题演讲中，乔布斯宣布苹果电脑公司从现在开始将被称为“苹果公司”，因为该公司已经将重心从电脑转移到消费电子产品上。这次活动也见证了 iPhone 和 Apple TV 的发布。该公司在前 30 个小时的销售中售出了 27 万部 iPhone，这款设备被称为“行业的游戏规则改变者”。

在 2007 年 2 月 6 日苹果网站上发表的一篇文章中，乔布斯写道，苹果愿意在没有数字版权管理 (DRM) 的情况下在 iTunes Store 上销售音乐，从而允许曲目在第三方播放器上播放，前提是唱片公司同意放弃这项技术。2007 年 4 月 2 日，苹果和百代联合宣布将 DRM 技术从 iTunes Store 的百代目录中移除，自 2007 年 5 月起生效。其他唱片公司最终跟进，苹果在 2009 年 1 月发布新闻稿，宣布 iTunes Store 上的所有歌曲都可以在没有 FairPlay DRM 的情况下使用。

2008 年 7 月，苹果推出了 App Store，销售 iPhone 和 iPod Touch 的第三方应用。一个月内，该商店销售了 6000 万个应用程序，平均每天收入 100 万美元。乔布斯在 2008 年 8 月推测，App Store 可能会成为苹果的一项价值 10 亿美元的业务。到 2008 年 10 月，由于 iPhone 的普及，苹果成为全球第三大手机供应商。2020 年 8 月 19 日，苹果股价一度突破 467.77 美元，成为第一家市值 2 万亿美元的美国公司。

在 2020 年 6 月 22 日的年度 WWDC 主题演讲中，苹果宣布将淘汰英特尔处理器，Mac 将过渡到内部开发的处理器。这一宣布是业内分析师所预期的，人们已经注意到，与目前基于英特尔的机型相比，采用苹果处理器的 MAC 电脑将大大提高性能。2020 年 11 月 10 日，MacBook Air、MacBook Pro 和 Mac Mini 成为首批采用苹果设计的处理器苹果 M1 的 Mac 设备。

6.4 IBM 蓝色巨人篇

一、公司介绍

100 多年来，IBM 一直坚守政企市场，主要客户是支付能力/意愿较强的大中型客户。在这样的背景下，IBM 只要做好两点就可以很舒服：一是保证自己的产品 (主要是硬件)/服务质量过硬；第二，服务好你的大客户。

IBM 的硬件产品性能，从已经销售给联想的 ThinkPad 品牌电脑，到小型机、大型机等网络和存储设备，确实是市场上其他竞争对手难以超越的，这使得 IBM 能够以远高于竞争对手的价格获得大量合同。然而，硬件技术的差异最终会缩小，除了金融机构和一些对硬件性能极其敏感的非金融商业机构，这个市场的绝大多数买家倾向于更注重性价比，而 IBM 产品/服务的高昂价格是无法接受的。更重要的是，通过合理的技术架构设计，很多企业发现其实可以用更低的

整体成本获得与购买高性能主机相同甚至更好的性能 (这其实也是“去 IOE”阵营的厂商一直在强调的)。所以，如果 IBM 还想单靠硬件性能赚取大量营收，那肯定不太可能。

至于大客户，IBM 会对客户的类型进行分类。一旦他们被列为顶级客户 (综合账户)，就会得到 IBM 的极大关注和资源投入，因此这些客户往往能感受到 IBM 的高价值服务。但是对于数量更大、短期贡献更低的一般客户，IBM 的整体资源投入实际上并不是很乐观。值得注意的是，往往是这些多元化的中小客户，而不是那些大客户，定义了 IT 行业未来的需求。如果 IBM 未来不能更好地理解和满足这些中小客户的业务需求，它将很难继续作为一个伟大的企业屹立于信息产业之林。另一个值得注意的趋势是所谓的 IT 消费化，这可能会改变整个企业 IT 市场的产品/交付模式。作为一家长期远离消费级 IT 市场的企业，IBM 如何适应这一趋势也颇具挑战性。随着技术和组织的发展，技术不再是企业 CTO 的专属领地，越来越多的前端人员更倾向于使用苹果的硬件、微软的软件和谷歌的服务。这将迫使企业的 IT 采购越来越多地转向在消费市场取得巨大成功的新 IT 产品/服务供应商。假以时日，他们也将有实力进入机构市场 (亚马逊就是一个很好的例子)。如果说企业/政府目前选择 IBM 是因为其强大的硬件技术，那么随着云计算时代计算能力的均等化，他们的选择域将大大扩展。到那时，IBM 是否还能屹立不倒，向客户收取高额费用，还不得而知。

二、发展趋势

对于公司未来的战略，IBM 选择了 SMAC(社交 + 移动 + 大数据 + 云)，比 Smart Planet 更有效：通过数据和分析为客户创造独特的商业洞察 (尤其是 MarketingSales 等前端部门)；通过云平台/产品/服务改变其商业模式，从卖硬件转向搭建统一的接入资源平台 (公有云 + 私有云 + 混合云)；通过社交和移动技术/产品，帮助企业客户的市场/客户趋势，增加商业价值。虽然 IBM 看起来总是一个行动上的跟随者，但不得不承认，这家公司对于整个行业的趋势和方向有着极佳的预见力。

IBM 在大数据和分析领域积累了广泛而深厚的技术和专业知识：超过 30 起相关并购；5000 名咨询师 +400 名数学家；2/3+ 研究工作与数据/分析和认知计算相关；分析领域的 4000 项专利；6000 个行业合作伙伴 +1000 个高校合作伙伴。到可以提供的实际产品/服务，包括：决策管理)/内容分析)/规划和预测)/发现和探索)/商业智能)/预测分析)/数据和内容管理)/流计算)/数据仓库)/信息集成和治理。

整个 IT 行业已经进入云时代，越来越多的 IT 基础设施和商业应用将基于云架构实现。预计到 2016 年，全球超过 1/4 的应用将拥有云版本，85% 的新软件也将拥有云版本。从市场价值来看，云计算市场预计到 2015 年将达到 2500 亿

美元的巨大规模。

目前，IBM 是企业云市场的领导者，这包括通过 15 次并购获得的能力，其中 2013 年收购的 SoftLayer 尤其值得注意。作为 IBM 云解决方案的基础支撑，极大提升了 IBM 在私有/公有云领域的能力。IBM 在自身积累和外部并购的基础上，构建了覆盖 IaaS/PaaS/SaaS/BPAAS(业务流程即服务) 的全方位云服务能力。目前，IBM 在云相关领域拥有 1500 多项专利，80% 的世界 500 强企业已经使用了 IBM 的云服务。目前，IBM 的公共云平台每天帮助客户处理超过 550 万笔交易请求。IBM 的 SaaS 服务已经被全球财富 500 强前 25 家公司中的 24 家使用。目前，IBM 在全球拥有 25 个大型数据中心，另有 15 个新的数据中心正在筹备中。

各种新技术已经加速了个人的赋权，人们对于企业提出了越来越多的要求—社交化、移动化等。越来越多的企业开始考虑将更多的 IT 预算投入到前台系统中，以加强对于顾客、雇员、合作伙伴、投资人以及民众的联系。IBM 在 2013 年发布了 IBM Mobile First，并完成了七项移动领域内的收购。目前，IBM 有超过 3000 名的移动领域专家，并且已经在移动和无线技术领域获得数百项专利。此外，IBM 在社交领域并购了 Kenexa；在安全领域并购了十几家相关企业，目前已经拥有超过 6000 名安全专家、以及 3000 项相关专利。同时，IBM 也在公司内部大力推进社交工具与移动平台的应用—IBM 内部的社交网站上已经有 300000 用户，以及 200000 个小组。此外，IBM 还建立了内部在线学习系统—Think Academy，帮助自己的员工加深对于各项技术/商业趋势的理解。

各种新技术加速了对个人的赋能，人们对企业提出了越来越多的要求——社会化、移动化等。越来越多的企业开始考虑将更多的 IT 预算投入前台系统，以加强与客户、员工、合作伙伴、投资者和公众的联系。

IBM 在 2013 年发布了 IBM Mobile First，并在移动领域完成了 7 次收购。目前，IBM 拥有超过 3000 名移动专家，并在移动和无线技术领域获得了数百项专利。此外，IBM 收购了 Kenexa 在社会领域；在安全领域，它收购了十几家相关公司。目前拥有 6000 多名安全专家，3000 多项相关专利。与此同时，IBM 也在公司内部大力推广社交工具和移动平台的应用——在 IBM 内部的社交网站上已经有 30 万用户和 20 万个群组。此外，IBM 还建立了内部在线学习系统——Think Academy，帮助员工加深对各种技术/业务趋势的理解。

6.5 Lenovo 联想篇

一、公司介绍

联想是中国著名的科技公司，也是全球最大的个人电脑制造商。其电脑销量连续多年位居世界第一，产品涵盖服务器、电脑、智能电视、手机、主板、打印

机等。2014 年，它还完成了对摩托罗拉的收购。

作为 ICT 领域的世界领导者，联想坚持“智能是万能的”的理念，为用户和行业提供集成应用、服务和最佳体验的智能终端，以及强大的云基础设施和工业智能解决方案。作为智能设备的全球领导者，联想每年为用户提供数以亿计的智能设备，包括 PC、平板电脑和智能手机。2018 年，联想排名全球第一。作为全球领先的企业数字化和智能解决方案提供商，联想积极推动整个行业“设备 + 云”和“基础设施 + 云”的发展以及智能解决方案的实施。面对智能转型的新产业创新机遇，联想提出了一项智能转型战略，重点关注物联网（Smart IoT）、智能基础设施和智能三个方向，以成为智能转型的行业领导者和推动者。2002 年 8 月 27 日，联想自主研发的沈唐 1800 电脑测得的峰值速度为每秒 1.027 万亿次，通过了由六位学者组成的专家组的评估。2002 年 11 月，权威的全球高性能计算机 500 强榜单重新发布。联想沈唐的 1800 万亿台服务器位居世界第 43 位，是第一家正式跻身前 100 名的中国公司。联想万亿计算机的成功开发对中国高性能计算机的产业化具有重要意义。

二、企业采访与信息收集

在国内，服务器与设备的发展很受关注。联想作为国内发展的龙头企业，进行业务结构调整。注重机构研究，而过去长达数十年的高研发投入，构建了联想集团科技护城河。

Q：如何解决人们对于计算机中固态性能的问题?

A：为了提高基于闪存的固态磁盘 (SSD) 的性能，联想集团研究，可以通过跨多个闪存芯片绑定相邻的闪存块来组装大型逻辑链接块。但是，闪存不允许就地覆盖，因此在这些块上合并写入的操作会明显降低性能。此外，当小的随机写入分布在磁盘地址空间时，性能往往会显着下降。因此，我们提出了一种有效管理随机写入的技术，以实现稳定的 SSD 性能。该结构主要由写入缓存和闪存转换层组成，闪存转换层按访问模式 (S-FTL) 分隔写入组。分别管理这两种类型的写入模式可实现更大的并行性并降低大块管理的成本，从而提高所提出的 SSD 的性能。仿真实验表明，与应用于现有并行 SSD 结构的基本 FTL 相比，所提出的模式自适应结构可以平均减少 39% 的额外闪存块擦除开销，写入性能可以提高约 60%。

在处理 NAND 闪存不能原地覆盖的限制上，不同的 FTL 方案采用了不同的方法。在日志块方案中，单个更新块被分配给每个逻辑块来处理写操作。但是，当在整个磁盘地址空间中以高频率发生短而随机的写入时，可能会发生块抖动。也就是说，更新块的数量不足会导致使用页面副本进行不必要的块擦除，这可能会降低性能。为了解决块抖动问题，全关联方案区分了随机写入和顺序写入。本文采用对数分块方案和全关联方案分别表示为 BAST 和 FAST。对于顺序写

操作，一个日志块被分配给每个逻辑块地址，就像在日志块方案中一样。另一方面，当发生随机写入时，数据被收集并以完全关联的方式存储在单独的随机写入日志块中。因此，任意随机写入不会影响日志块空间的使用，一定程度上防止块抖动。

总而言之，应根据随机/顺序或热/冷等访问特性来管理块。否则，在单个逻辑块中混合具有不同访问模式的数据会导致不必要的 SSD 垃圾收集和合并操作。此外，频繁的闪存块擦除会缩短 SSD 的使用寿命。在 SSD 等通用闪存应用中，逻辑块的大小应该很大，以便利用由更高程度的交错产生的更大并行性。然而，更大的块增加了未更新页面混合在逻辑块中的可能性，需要更多的努力来利用磁盘访问模式。

Q: 人们对与计算机传感器平台有了越来越高的要求，联想是如何看待呢?

A: 传感器网络必须就地处理大量间歇性可用的数据。这激发了对在需要时实现高性能但在闲置时实现超低功耗的方法的研究。应对这一挑战的一种方法是使用嵌入式多处理器系统，从而在并行性、性能、能效和成本之间进行权衡。为了评估这些权衡并获得对未来系统设计的洞察力，本文介绍了用于嵌入式传感器系统的微型、能量可扩展的 24 处理器模块 L24 的设计、实现和评估。

企业经过实验与测试，提供了激励此类嵌入式多处理器的分析结果和经验证据，并提出了并行定点快速傅里叶变换实现。此应用程序用作所提供硬件平台的具有挑战性但现实的评估器。通过结合硬件测量、指令级微架构仿真和分析建模，证明该平台提供的闲置功耗比采用具有同等性能的单片处理器的系统低一个数量级，而动态功耗仍然具有竞争力。

考虑到应用程序计算和处理器间通信需求，表明可能存在一个最佳工作电压，可以最大限度地减少求解时间、能量使用或能量延迟乘积。这个最佳工作点是通过分析制定的，通过系统测量进行校准，并针对所提供的硬件平台和应用程序进行评估。

我们考虑在集群架构上对本地代码块进行指令调度以提高性能。众所周知，空间和时间的安排是一个难题。以前的工作提出了基于列表调度的贪婪方法来同时执行空间和时间调度，以及基于首先划分代码块进行空间分配然后执行时间调度的分阶段方法。贪婪方法有犯错误的风险，而这些错误的恢复成本很高，而分区方法则存在众所周知的相序问题。

现在研究领域内出现了一种用于在集群架构上调度指令的约束编程方法。我们的研究人员正在对于这个方向进行开发。采用问题分解技术，以综合方式解决空间和时间调度问题。我们分析了不同硬件参数（例如集群数量、问题宽度和集群间通信成本）对应用程序性能的影响。

Q：云计算热日渐兴起，联想也开发了自己的云平台，在平台上高性能计算

开发有什么瓶颈吗?

A：鉴于可用数据的爆炸式增长以及数据系统之间不断增加的连接性，可扩展数据分析的基础架构与以往一样重要。这个领域的一个基本操作是 join，它促进了基于公共连接键的记录组合。在通信和计算方面，这种数据密集型操作会产生巨大的成本。提高此操作的效率会对主要用于分析工作负载的应用程序的性能产生重大影响。我们在开发过程中也是遇到了诸多问题。尽管内部连接算法已在并行和分布式系统中得到广泛研究。但关于外部连接的主题所做的工作相对较少。事实上，外连接在复杂查询中很常见，并广泛应用于各种应用程序中。一旦发现，内部连接实现可以丢弃其键与连接另一端的键不匹配的记录。无论如何，DER 算法是专门为关系数据库管理系统设计的（RDBMS），其实现严重依赖于输入数据的模式（如后所述），这可能需要对其他环境进行多次修改，我们现在工作中研究的当前流行的云计算平台就是如此。

非阻塞数据结构是许多并行应用程序的重要组成部分，因为它们可以帮助提高容错性和性能。尽管有许多非阻塞数据结构，如堆栈、队列、双端队列 (deques)、列表，在实践中得到广泛应用，但它们中的大多数都设计为仅在共享内存机器上使用，而不能在分布式内存设置。最近的几项研究侧重于开发新的量身定制的非阻塞分布式数据结构，而忽略了为分布式内存情况调整大量现有非阻塞共享内存结构的潜力。因此，我们提出了一种通用方法来弥合这项工作中大多数现有非阻塞数据结构和分布式内存机器之间的差距。几个挑战，比如安全内存回收，解决 ABA 问题，都是必须攻克的。为了解决这些问题，我们提出了一个全局内存管理方案。该方案利用了广泛用于解决共享内存环境中问题的危险指针。

事实上，随着数据应用程序规模的扩大，云环境在应用程序横向扩展中发挥着关键作用，利用并行化来加速操作并扩展开发人员可用的可用内存量也是我们采用的一种重要技术。

第七章　发展趋势

7.1　计算机体系结构

7.1.1　完全串行访问的自动寻址架构

Racetrack 存储器是一种新兴的低功耗磁存储器，有望成为加速器中传统存储器的竞争性替代品。然而，内存中的随机访问是 CNN 加速器的时间和能量消耗，因为它存在大量无效移位。

以前解决这个问题的方法可以分为 move-ahead [8] 和 visit-nearby [9]。对于先行解决方案，端口对齐操作始终在数据访问之前执行，以便通过内存计算交错隐藏无效移位延迟。然而，该方法未能减少大量的无效移位，因此无法明显提高赛道存储器的能效。

西北工业大学计算机研究所提出了一种自动寻址架构，该架构构建了一种新颖的数据布局，以确保下一轮内存访问始终可以在当前轮的原位或严格相邻的单元中得到满足，从而产生完全序列化的访问足迹可以在赛道内存中没有任何无效移位的情况下驱动即时端口对齐。通过这种方式，原始的基于地址的访问退化为三个候选者之间重复的选择，即一个原位单元格和两个相邻单元格。

基于这种简化，轻量级访问管理可以根据 CNN 超参数定义的确定性访问行为生成三选一的序列。在浏览了论文的架构和结果之后，测试结果表示，当将五个流行的 CNN 应用程序部署到该架构时，赛道的物理位移比传统布局减少了 74.64%，分别实现了 54.2% 和 42.1% 的读取和写入能量减少。

7.1.2　计算的模块化和可重构方法的进步

现实生活中日益复杂的应用要求不断改进微处理器系统。最常采用的微处理器设计方案之一是冯 · 诺依曼架构。中央处理器 (CPU) 执行计算并在不断交换信息的过程中与内存通信。这两个组件之间不断的数据移动成为一个重要的性能瓶颈。在这种通信中浪费了大量的功率、能量和计算时间。借助超越冯诺依曼计算 (BvNC) 范例，计算在内存阵列内部或非常靠近内存阵列的地方执行。文献中提出了 BvNC 方法，主要基于对现有记忆的修改，从而实现简单的计算。其他人利用新兴技术来存储和计算数据，使用模拟操作。

采用了一种不同的方法，将计算单元放置在靠近内存单元的位置，从而提高了通用性和性能。他们提出了一种由交错结构中的内存和计算元素组成的混合 SIMD 架构。Hybrid-SIMD 既可以用作低密度存储器，也可以用作 SIMD 加速器。

7.2 高性能计算

7.2.1 上下文切换导致的时间延长问题

计算速度不断提高是行业发展的整体趋势，但是速度的提高离不开硬件技术的发展，但是硬件方向晶体管的研制在一定程度上，减慢了高性能计算的研发速度和科研成本。除此之外，还有一些操作系统的实现，例如图形处理单元使用大型寄存器文件来容纳所有活动线程并加速上下文切换。不幸的是，由于长访问延迟、高功耗和大硅面积供应，寄存器文件是未来 GPU 的可扩展性瓶颈。这个问题虽然不能从硬件方向着手切入在其他众多学者的不懈努力和研究之下提出了分层寄存器文件，其目的时通过将寄存器缓存在较小的寄存器文件缓存中来降低寄存器文件功耗。但是，即便前人付出了很大的努力由于寄存器文件高速缓存中的命中率较低，这种方法不会改善寄存器访问延迟。

但近期，有期刊上的文章提出的方法，已经淡化了这个问题：他们提出的延迟容忍寄存器文件 (LTRF) 架构，以在两级分层结构中实现低延迟，同时保持低功耗。我们观察到编译时间间隔分析使我们能够将 GPU 程序执行划分为多个间隔，并准确估计每个间隔内的 warp 聚合寄存器工作集。LTRF 的关键思想是在软件控制下，在每个间隔开始时，将估计的寄存器工作集从主寄存器文件预取到寄存器文件缓存，并将预取延迟与其他 warp 的执行重叠。我们观察到预取寄存器时寄存器组冲突会大大降低 LTRF 的有效性。所以，我们设计了一种编译时寄存器重新编号技术，以减少寄存器组冲突的可能性。我们的实验结果表明，LTRF 支持高容量但延迟时间长的主 GPU 寄存器文件，为各种优化铺平了道路。作为一个优化示例，我们使用新兴的高密度高延迟内存技术实现了主寄存器文件，使容量增加了 8 倍，并将整体 GPU 性能提高了 34%。

7.2.2 多处理器下资源共享问题

由于多处理器平台的发展，多处理器资源同步和锁定协议被提出并被广泛研究，例如分布式、多处理器、多处理器 SRP、灵活多处理器锁定协议、多处理器 PIP 锁定协议、多处理器带宽继承和多处理器资源共享协议。由于这些协议的性能高度依赖于任务划分，因此文献中开发了几种划分算法，例

几十年来，在考虑实时系统中的多处理器同步和锁定时，主要关注点一直是资源共享协议的设计和分析，其中协议决定新传入请求动态访问共享资源的顺序。相反，由 Chen 等人提出的依赖图方法 (DGA)。在 2018 年，预先计算允许任务访问资源的顺序，包括两个单独的步骤：构建依赖图以确定由一个二进制信号量或互斥锁保护的临界区的执行顺序。多处理器调度算法用于通过尊重构造的依赖图给出的约束来调度任务。

7.3 计算机硬件与人工智能

关于计算机硬件与人工智能，主要的发展趋势分为与目标检测算法的融合、卷积神经网络的加速等等。

7.3.1 与目标检测算法的融合

目标检测在民用、工业和军事领域都有巨大的应用需求，并且随着物联网技术的快速发展，在位于前端的嵌入式设备直接进行目标检测的需求也越来越大，目标检测算法直接部署到嵌入式设备上处理速度快、运行功耗低，适合物联网前端的应用环境，具有极大的现实意义。根据之前对于嵌入式加速器的分析，从功耗、开发成本、开发周期、灵活性等方面综合考虑，近年来推出的高性能、拥有异构处理器环境的 FPGA 嵌入式平台作为基于深度学习的目标检测算法部署硬件平台具有极大优势。

由于高性能 FPGA 产品的不断发布和 FPGA 的可重构、可定制和能源效率高等特性，越来越多的研究者开始着手于基于深度学习相关算法的 FPGA 高性能实现的研究，特别是对卷积神经网络的 FPGA 硬件加速研究做了大量的工作。国内有学者在 Xilinx 公司的 Virtex-5 系列 XC5VLX110T 器件上，针对完整的卷积神经网络推理过程采用 ISE 工具设计了一款加速器，该加速器能获得 Core i5 2500K 处理器的 4 倍的单帧图像处理速度，并且功耗仅为 CPU 的 2.68 但实现的网络深度只有 4 层。部分论文设计了“FPGA+CPU”的异构体系，完成卷积神经网络的硬件固化，系统准确率达到 92%，整体功耗控制在 2.93W 以内，但只是以 MNIST 手写数字识别为具体应用，所部署的网络模型参数量和深度都较小。部分论文利用 Xilinx SDAccel 工具在 OpenCL 框架下采用所提出的多种优化方法，设计了一个加速器，加速卷积层的性能提升了 14.4 倍，但该文献只加速了卷积操作，而池化、全连接等卷积神经网络中的常见操作并没有实现。文献通过采取不同的并行措施在 FPGA 上高性能地实现了完整 CNN 应用。部分论文主要利用了特征图内部卷积核的并行性，利用了输出内、输出间的并行性，但三者都并未使用片上缓冲区做数据重用，而是用很高带宽和动态重配置来提高性能还存在论文所设计的加速器最大化数据重用，通过尽可能地降低带宽需求从而有效地解决了外部存储带宽极有限下的加速问题，但并未考虑最大化计算性能，且加速器层间计算转换时需要采用 FPGA 重编程的方式，极大增加了网络的运行时间。

总体来说，将卷积神经网络部署到以 FPGA 为主的硬件平台上，通过采用可靠高效的硬件加速方案以获得高性能实现已成为研究热潮。目前基于 FPGA 的卷积神经网络加速器研究已经取得了不错的成果，但仍存在一些问题：(1) 有些研究成果只适用于小型卷积神经网络，然而以卷积神经网络为核心的目标检

测算法通常网络模型较复杂、参数量和计算量较大。(2) 一些加速器只实现了卷积层的加速，而 CNN 网络中往往不只有标准的卷积层，例如还有池化、归一化、全连接等操作。(3) 很多研究并没有协同考虑存储带宽和计算资源限制，有的未考虑数据重用，导致整个系统必须工作在外部储存带宽很大的前提之下，而有的过分侧重于降低带宽需求，导致忽略了系统的计算性能。(4) 对于一些实现了网络反向传播的加速器，反向传播过程的权值计算和更新等需要消耗大量计算资源和存储资源，并大大地增加系统的运行时间。所以，目前大多数研究都不适用于基于深度学习的目标检测算法在带宽限制和计算资源有限而对实时性和低功耗有着极大需求的移动端设备上实现。将目前最先进的基于 CNN 的目标检测算法部署到 FPGA 中以达到硬件加速的目的主要存在以两个难点：(1) 以卷积神经网络为基础的目标检测算法通常网络模型较复杂、参数量和计算量较大，而 FPGA 存储资源、计算资源和系统带宽都极其有限，这种矛盾造成了在实时性设计要求下的主要难点。(2) 大多数的基于 CNN 的目标检测算法模型具有不同的网络规模和结构，并且由于卷积神经网络的层次化结构，不同的网络层参数也可能是不同的 (例如不同的卷积层具有不同大小的卷积核和步长等)，这对设计的通用性和可扩展性提出了要求，以适应可变的参数。

7.3.2 卷积神经网络的加速

对于卷积神经网络的加速，主要分为如下几个部分：数据量化加速、并行度加速、数据存储加速等等，其中数据存储尤为重要。当一组数据以连续存储的方式存放到内存中时，我们可以消耗较小的开销和较少的内存完成数据的读取和运算。因此我们可以考虑多种数据存储和数据复用，完成加速。

图 7-1　　目标检测算法的 FPGA 实现

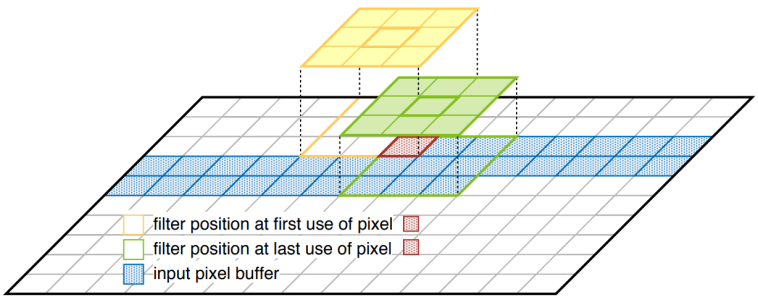

图 7-2　　数据复用加速

第八章　总结与分工说明

一、总结

通过本次对于计算机硬件的信息收集，资料理解和整理，我们对其方向的发展和行业的发展引领导向有了初步的了解。我们主要从计算机体系结构、高性能计算与算力影响、计算机硬件与人工智能的结合三个方面进行考察调研。

了解计算机体系结构首先要了解计算机系统层次结构的划分，明确各机器级之间的关系。在诸多层级中，我们重点关注了硬件技术和相关现实问题的解决。通过对领域研究热点的调研，我们分析了国产 CPU 的发展态势，关注了新兴设备技术，了解行业前沿知识。通过对现实中能耗问题的探讨，我们了解到 DVFS 算法，并进行扩展延伸，探索了其在 HPC 中的应用。我们还将计算机硬件知识与人工智能相结合，学习了解了计算机图形学的相关知识，温习了卷积层、FPGA 等相关基础知识。与此同时，我们还学习了计算机架构领域的基础知识，结合软件工程专业知识，探讨软硬件发展对计算机系统结构的影响。

此外，我们通过电子图书馆查找阅读了大量论文文献，了解了很多课题相关的研究机构，并对项目团队的报告和图表进行研究，学习高性能计算的相关知识。我们也明白，若想有所建树，一定要理论和实践相结合。所以我们关注行业重点企业的相关介绍和访谈内容，不仅仅从技术上学习，也从市场需求、商业竞争、国家政策等多方面看待计算机行业的发展态势。

文档上的文字并不是我们全部的学习成果，恰恰相反，它只是我们学习过程中的冰山一角。很多的段落都是经过反思和沉淀所凝练出来的文字，在整个摸索研习的过程中，我们也收获了很多。不仅仅是丰富的学科知识，更是具有启迪性的前辈们的思想方法和不同时代下都在熠熠生辉的科研精神。身为山大学子，我们也将秉持山东大学“学无止境”的校训继续在自己的专业领域不断探索、不断积累、不断创新。

二、报告分工

此报告由小组成员贾星宇、杨钰润、易格名、石嘉晖共同完成。由于本报告每一章节由计算机体系结构、高性能计算、计算机硬件与人工智能三个部分组成，因此本小组成员分工如下：

杨钰润同学负责报告中每一章节计算机体系结构、高性能计算部分的撰写。

贾星宇同学负责计算机硬件与人工智能部分的撰写以及计算机硬件与人工智能相关 PPT 的制作，具体 ppt 内容详见附件。

易格名同学负责资料搜寻以及重点企业的部分撰写。

石嘉晖同学负责资料的搜索。

参考文献

[1] 邹丹音. 基于深度学习的目标检测算法 FPGA 实现 [J], 2019.

[2] 王嘉晨. 基于深度学习的高清图像目标检测算法 FPGA 实现 [J], 2020.

[3] 吴瑞东, 刘冰, 付平, et al. 应用于极致边缘计算场景的卷积神经网络加速器架构设计 [J]. 电子与信息学报, 2022.

[4] Schmid E, Eberli F. ZynqNet:An FPGA-Accelerated Embedded Convolutional Neural Network [J], 2016.

[5] Wang E. PYNQ Classification - Python on Zynq FPGA for Neural Networks [M]. 2017.

[6] Redmon J, Farhadi A. YOLOv3: An Incremental Improvement [J], 2018.

[7] 袁春风, 张泽生, 杨若瑜, et al. “计算机组成与系统结构” 课程建设思路与教学实践 [J], 2012.

[8] 李大志, 张晓红. 基于新应用趋势的计算机系统结构研究的新方向 [J], 2006.

[9] 刘晓天. 浅析计算机系统结构的发展现状和发展方向 [J], 2007.

[10] Yang C. A High-Performance Linear Algebra-based Graph Framework on the GPU [J], 2002.

[11] Ghiasi N M. A High-Performance and Energy-Efficient In-Storage Computing System for Genome Sequence Analysis [J], 2022.

[12] Li Y. An Efficient CRT-Based Bit-Parallel Multiplier for Special Pentanomials [J], 2021.

[13]

[14] Di B. TLB-pilot: Mitigating TLB Contention Attack on GPUs with Microarchitecture-Aware Scheduling [J], 2022.