Custom Artificial Intelligence Streaming Accelerator

Corerain
鲲云科技

定制数据流架构技术

白皮书

Corerain CAISA® 3.0

CAISA® 3.0

Corerain CAISA® 3.0

www.corerain.com

免责声明

本文档仅提供阶段性信息，所含内容可根据产品的实际情况随时更新，我们将不另行通知。如因文档使用不当造成的直接、间接或其他损失，本公司不承担任何责任。

本文档可能包含第三方信息、产品、服务、组件、数据或内容（统称“第三方内容”）。本公司不控制且不对第三方内容承担任何责任，包括但不限于准确性、兼容性、可靠性、可用性、合法性、适当性、性能、不侵权、更新状态等，除非本文档另有明确说明。在本文档中提及或引用任何第三方内容不代表本公司对第三方内容的认可或保证。

用户若需要第三方许可，须通过合法途径获取第三方许可，除非本文档另有明确说明。

除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的保证。

商标声明

本文档中所涉及的所有注册商标及产品名称仅供识别所用，商标或产品名称均为各自拥有者的商标或产品名称。

深圳鲲云信息科技有限公司

地址：深圳福田保税区市花路长富金茂大厦 14 层
邮编：518000
电话：深圳 0755-88917937
网址：www.corerain.com
联系我们：contact@corerain.com

技术支持
可通过邮箱或问题反馈网站向我司提交产品使用的问题，并获取技术支持。

问题反馈邮箱：support@corerain.com
问题反馈网址：https://community.corerain.com

修订历史

下表为本文的修订历史。

日期	版本	修订内容
2020-06-20	1.0	基于内容发布

目录

1 导言

1.1 AI 计算平台的核心是芯片

算力、算法和数据是当前人工智能（Artificial Intelligence, AI）发展的三个核心要素，其中算力是承载和推动 AI 走向实际应用的基础平台和决定性力量。

算力在具体 AI 应用中是以计算平台的方式体现的，AI 计算平台可以简单定义为可运行各种人工智能算法的计算系统。它的物理形式可以是边缘侧的智能盒子，也可以是数据中心的大型服务器。在智慧城市、自动驾驶、智能零售、安全生产、智能制造等众多的应用场景中，各种算法都最终需要运行于一个 AI 计算平台上，以完成应用的部署。无论何种应用场景，都希望 AI 计算平台具有高通用性、易于部署、高性价比等特点，而决定 AI 计算平台特性的核心在于其配置的 AI 芯片。

1.2 下一代人工智能计算平台的指标突破

计算平台的架构在持续演进，X86、RISC 及 CUDA 等都是架构演进过程中的典型代表，由这些架构构建了不同的算力平台。不夸张的说，每次计算架构的重大创新都代表着一个新的算力时代，而在每一个新的算力时代里，我们会看到一个共同的现象：由创新架构构成的算力平台相对于上一代平台都会在某一核心指标上提供一个数量级级别的性能提升。比如：

（1） X86 计算平台相较于 IBM 大型机提供了更好的兼容性及可拓展性，

（2） RISC 计算平台相较于 X86 计算平台提供 10 倍以上能效比提升；

（3） CUDA 相比于 X86 计算平台提供 10 倍以上峰值算力提升；

为什么会出现这种现象呢？因为计算平台是一个系统工程，用户在应用某个计算平台时获得的实测性能是受诸多因素影响的，比如算力架构、工艺水平、接口带宽以及软件生态等。新一代的算力平台相对于上一代的平台可以在某些因素上实现巨大的突破，但是其他的影响因素很大的可能是不如上一代先进，这种综合的影响使新的算力平台必须相对于上一代算力平台在某个指标上高 10 倍以上，才能实现用户实测性能的提升。

图 1-1 计算平台性能的影响因素

这一现象最近的一个典型案例是英伟达 GPU 与 intel CPU 的性能对比[1]。在 2010 年左右，GPU 在高性能计算领域异军突起，业界及学术界宣称 GPU 的算力是 intel CPU 的 10 倍以上。以当时主流的

GPU GTX280 为例，其具有 240 个 CUDA 核心，主频 602MHz，相对于当时主流的 intel CPU Core i7 960（4 核，3.5GHz）确实是实现了 10.32 倍的峰值算力提升，但是 intel 具有 Multithreading，cache blocking，reorganizing SIMD memory access 等架构和编译技术，最终在吞吐率对比中，GTX280 仅比 i7 960 高 2.5 倍左右，而非峰值算力的 10.32 倍。但对于 GPU 而言，这已经是技术的巨大胜利，因为实测算力 GPU 确实是相对于 CPU 提升了 2.5 倍，虽然代价是需要 10 倍的峰值算力，这充分证明了增加峰值算力的技术途径是完全可行的，所以后续英伟达一直在其 GPU 芯片上不断增强峰值算力以巩固其优势。

回到人工智能芯片的领域，下一代 AI 芯片的什么指标可能实现 10 倍以上的突破呢？鲲云认为是芯片利用率，其定义如下：

$$\text{芯片利用率} = \frac{\text{芯片实测算力}}{\text{芯片峰值算力}}$$

现有人工智能芯片主流架构为基于 GPU 与 Tensor Core 的指令集架构，其中 Tensor Core 针对卷积及矩阵计算提供性能提升，GPU 提供其他算子所需的算力。在此架构中，核心问题在于芯片利用率不足，无法充分发挥芯片的理论峰值算力。我们以英伟达旗舰 AI 加速卡 T4 为例，其 benchmark 数据如下表所示。

表 1-1 T4 Benchmark 测试结果

Model Name （Batch Size=128）	GOPS/FPS	Measured FPS	Measured Performance TOPS	Peak Performance TOPS	Chip Utilization Ratio
ResNet50	7.7	5415	41.7	130	32.1%
YOLOV3	71.4	128.3	9.16	130	7.05%
SSD-ResNet50	46.1	212	9.77	130	7.52%
U-Net Industrial	131.1	118	15.47	130	11.9%

备注：1. ResNet50, SSD-ResNet50, U-Net Industrial 数据来源自英伟达官方数据：
https://developer.nvidia.com/deep-learning-performance-training-inference#resnet50-latency
2. YOLOv3 数据官方数据缺失，采用 YOLOv3 开源网络在 T4 进行测试，Batch=128

通过上表可以看出，T4 的芯片利用率在 7%-31.1%之间，这意味着在运行 AI 算法时，T4 上有 66.8%-93%的芯片算力处于闲置状态。

如果沿着英伟达的技术路线，在峰值算力方面实现 10 倍以上提升的机会很小，或者需要付出巨大的成本（很多的芯片面积，最先进的工艺等）。所以，本白皮书旨在介绍一种新的 AI 计算平台的架构实现方式，定制数据流架构（Customized AI streaming Accelerator，CAISA），该架构可以在芯片利用率指标上实现 10 倍的提升，从而在同样的峰值算力条件下，为用户提供更高的实测性能。

1.3 定制数据流架构的意义

传统计算平台，如 CPU、GPU 及面向 AI 的 TPU，底层架构都是依托于冯诺依曼体系的指令集架构。其核心思路在于将计算分为处理单元、控制单元、存储指令的指令存储器、以及存储数据的数据存储器。控制单元取指令和数据，控制处理单元完成整个计算过程。

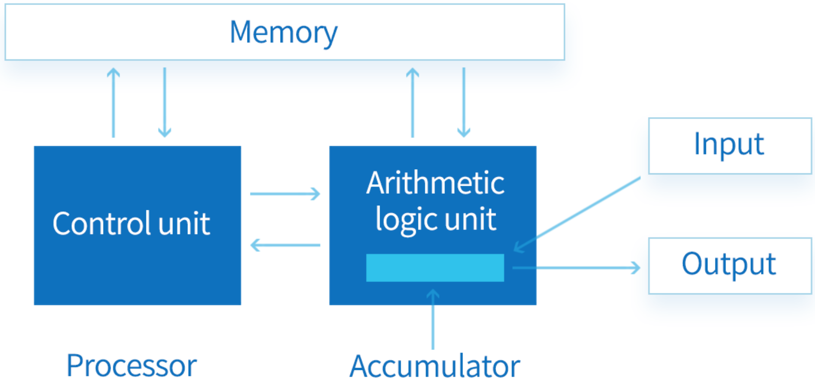

图 1-2 基于冯诺依曼的典型指令集架构示意图

作为一个计算平台，其核心任务为完成计算任务。在理想情况下，我们希望一个计算架构中每个计算单元在每一个时钟周期都能够完成一次有效计算（这也是芯片峰值算力的计算方法：芯片计算单元时钟频率×芯片中计算单元数量）。然而在实际架构中，指令集架构所发挥性能远低于芯片峰值算力：芯片中的计算单元在大部分时间内，都保持闲置状态等待计算所需数据到来，亦即冯诺依曼瓶颈。以上文中的 T4 芯片利用率为例，在每个时钟周期，平均 66.8%-92%的计算单元处于闲置状态。这些闲置的计算单元，主要处于以下两个状态之一：

（1） 等待指令：等待指令读取及译码；

（2） 等待数据：等待计算所需数据处于 ready 状态。

现代指令集架构引入流水线处理、数据缓存、数据预取等多种架构创新，以不断降低由于数据和指令等待带来的计算资源闲置。然而，导致指令集架构计算空闲的核心问题并没有解决，亦即指令协同运行的时钟不精确性：指令集架构无法准确预知下一时钟的数据、计算及指令运行状态。当数据状态无法确认时，计算单元的计算流水线必然被打破以等待特定指令完成执行。而指令集架构不断提升的峰值算力进一步加剧了指令集架构的闲置时间。一方面，为不断提高指令集架构性能，指令集架构引入多线程、多核并行处理等技术，进一步加剧了指令协同的不确定性；另一方面，峰值算力提升带来了计算单元的增加，计算单元之间的协同处理加剧了等待闲置时间。以英伟达 P4 加速卡与 T4 加速卡为例，虽然峰值算力提升了 5.9 倍（自 22TOPS 至 130TOPS），ResNet50、YOLOV3 等人工智能算法的实测性能只提升了 2-3 倍，芯片利用率下降了 50%-70%，处于不断降低状态。

定制数据流架构是指依托于数据流动次序，而非指令执行次序，来完成目标计算的计算架构。不同于基于冯诺依曼的指令集架构，数据流架构依托数据流的流动次序控制计算执行次序。如下图 1-3 所示，上半部分流水线计算一定按照加法、乘法、移位、累加次序执行，而下半部分流水线一定按照移位、对比、累加次序执行。从而完成算法执行，在这种计算方式下，解决指令集架构的几大核心问题：

（1） 数据流架构中数据流动次序即计算执行次序，没有指令概念，移除了由于指令带来的控制冗余以及等待指令读取译码带来的计算单元闲置；

（2） 数据流架构支持时钟级的精确计算，每个数据流动及计算在每个时钟都可精确预计，从而支持将数据流动与计算深度重叠，大幅降低计算单元闲置；

（3） 数据流架构中一个数据流流水线中可深度整合大量计算单元，从而打破指令集架构中峰值算力提升与芯片利用率的冲突问题：通过不断加深数据流流水线，可以在提升峰值算力同时，不降低流水线中计算单元闲置时间。

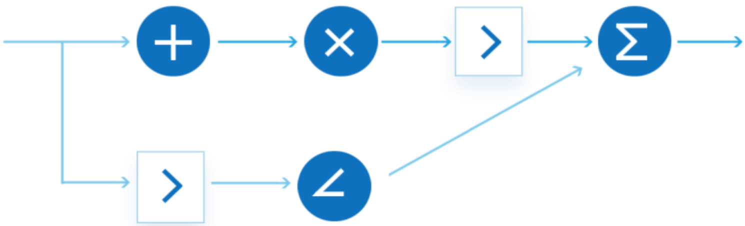

图 1-3 定制数据流计算示意图

2 数据流人工智能计算平台面临的挑战

无论何种架构的人工智能计算平台，都需要解决作为算力平台的几个核心挑战：

（1） 算力性价比：用户单位价格能够买到的 AI 算力，越高越好；

（2） 架构通用性：架构对于不同 AI 算法的支持，支持的算法越多越好；

（3） 软件易用性：AI 算法迁移至新平台的支持程度，越易用，迁移成本越低越好。

数据流技术的核心特征在于依托数据流动方式支持计算，而不依托指令集执行次序，从而突破指令集架构的芯片利用率限制。数据流架构带来技术突破的同时，作为一个全新的计算架构，其也面临着算力性价比、架构通用性以及软件易用性的挑战：

（1） 算力性价比：对于数据流架构，算力性价比意味着不断提升芯片利用率直至逼近 100%，如何在保证计算正确性的前提下，实现芯片利用率的持续提升，是一个巨大的挑战；

（2） 架构通用性：在 AI 领域，不同算子的不同组合带来种类繁多的 AI 算法需要支持，而定制数据流架构中深度定制的特性与对不同算法支持所需的通用性存着直接的矛盾；

（3） 软件易用性：成熟的 AI 计算平台无需用户了解底层硬件架构实现的细节，而定制数据流架构的高性能来源于底层硬件的时钟级精确计算，这种精细的底层控制与简单易用的高层软件开发之间存着矛盾。

2.1 算力性价比：时钟级精确的计算控制与计算正确性

100%的芯片利用率是芯片设计和使用的最理想状态，即：在每个时钟，芯片内的所有计算单元都处于有效运行状态。100%芯片利用率意味着芯片的所有算力成本都转换为实际算力提供到用户，从而达到算力性价比最优。指令集架构，如下图 2-1 所示，数据流动与数据计算是交替进行的，虽然数据传输过程会导致计算单元的闲置，但这种严格的指令执行次序保护了程序中的数据依赖，可以保证计算的正确性。反观数据流架构，深度定制的流水线支持在整个 AI 算法的执行过程中，计算与数据流动完全重叠，从而能够减少计算单元的等待时间，逼近 100%的芯片利用率。然而，不断重叠的数据流动与计算增加了数据正确性的压力：由于数据流完全依托数据流动控制计算次序，在单个 AI 算法的数百万时钟周期执行过程中，数据流动与计算间即使只有 1 个时钟周期的错位就会导致整体的计算错误。

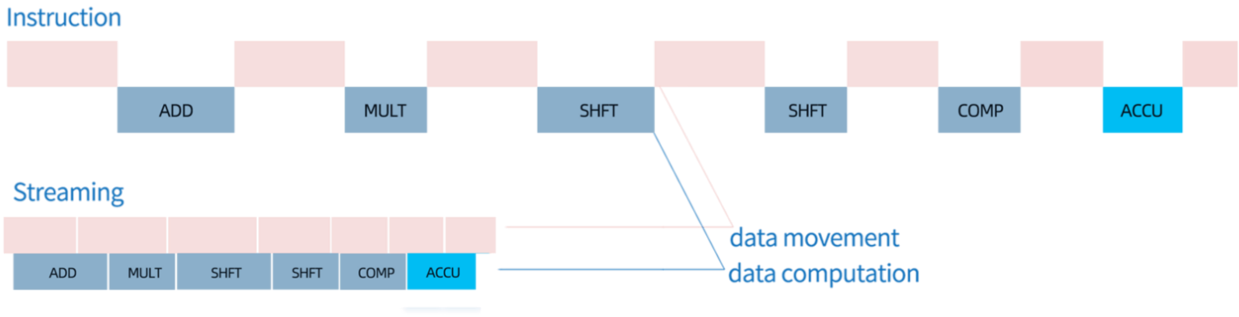

图 2-1 指令集架构与数据流架构在数据流动和计算顺序上的区别

2.2 架构通用性：针对所有 AI 的算法高性能支持

作为 AI 计算平台，如下图所示，数据流架构需要支持各种不同的 AI 算法，而数据流架构的技术特点是通过流水线的深度定制，并挤压流水线的每个闲置时钟周期来提升性能和芯片利用率，这种高度优化的定制设计，与种类繁多且需求不一的 AI 算法的通用支持之间存在着巨大的矛盾，需要通过细致的设计来解决。

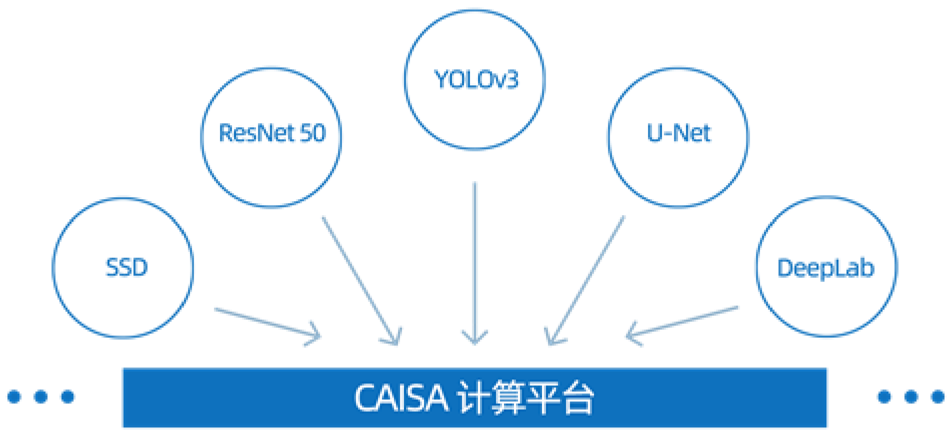

图 2-2 数据流平台支持算法示意图

2.3 软件易用性：时钟级精细计算控制与硬件无关的软件设计

当前 AI 算法主要依托于 Tensorflow、Caffe、ONNX、PyTorch 等开发框架进行开发，软件工程师通过将 AI 算法集成到整体应用系统中来发挥算法的价值。从行业生态角度出发，AI 应用开发者希望能够以最小开发成本发挥最大的芯片性能，这要求数据流架构配套的软件开发工具必须达到以下两点，以降低开发者在新计算平台上的算法部署或迁移成本：

（1） 底层架构对开发者透明，即开发者无需关心架构细节即可实现硬件最优性能；

（2） 工具的端到端支持，从开发框架端输入，直到 API 接口调用端输出，无需开发者过多干预。

另一方面，数据流架构性能优势的核心之一是对计算过程时钟级准确的控制，并且架构中需要控制的计算模块、存储模块以及互联网络的数量非常巨大，这种在硬件层面的精确控制必须依赖软件来完成，否则数据流架构是难以实用的。

所以为保证架构配套的软件的易用性，必须将开发者接口和硬件控制接口进行合理解耦，硬件控制应该对开发者透明，从而保证开发者实现简单快捷的算法部署和系统集成。

3 定制数据流架构概述

定制数据流架构（Custom Streaming Artificial Intelligence Accelerator，CAISA）为鲲云科技自主研发数据流架构，依托于 CAISA3.0 架构的 CAISA 芯片已完成量产，为全球首颗数据流人工智能芯片。作为下一代人工智能计算平台，CAISA3.0 架构具有以下技术优势：

（1） 高算力性价比：CAISA 芯片实测芯片利用率高达 95.4%，相比英伟达旗舰芯片提供最高 10 倍以上芯片利用率提升；

（2） 高架构通用性：支持各种深度学习算法，包括目标检测、分割、分类等领域应用的主流算法；

（3） 高软件易用性：提供 RainBuilder 开发工具链，自顶层深度学习开源框架中算法至底层 CAISA 架构时钟精确映射，端到端自动化工具兼容现有人工智能开发框架、生态及软件。

CAISA 架构图如图 3-1 所示

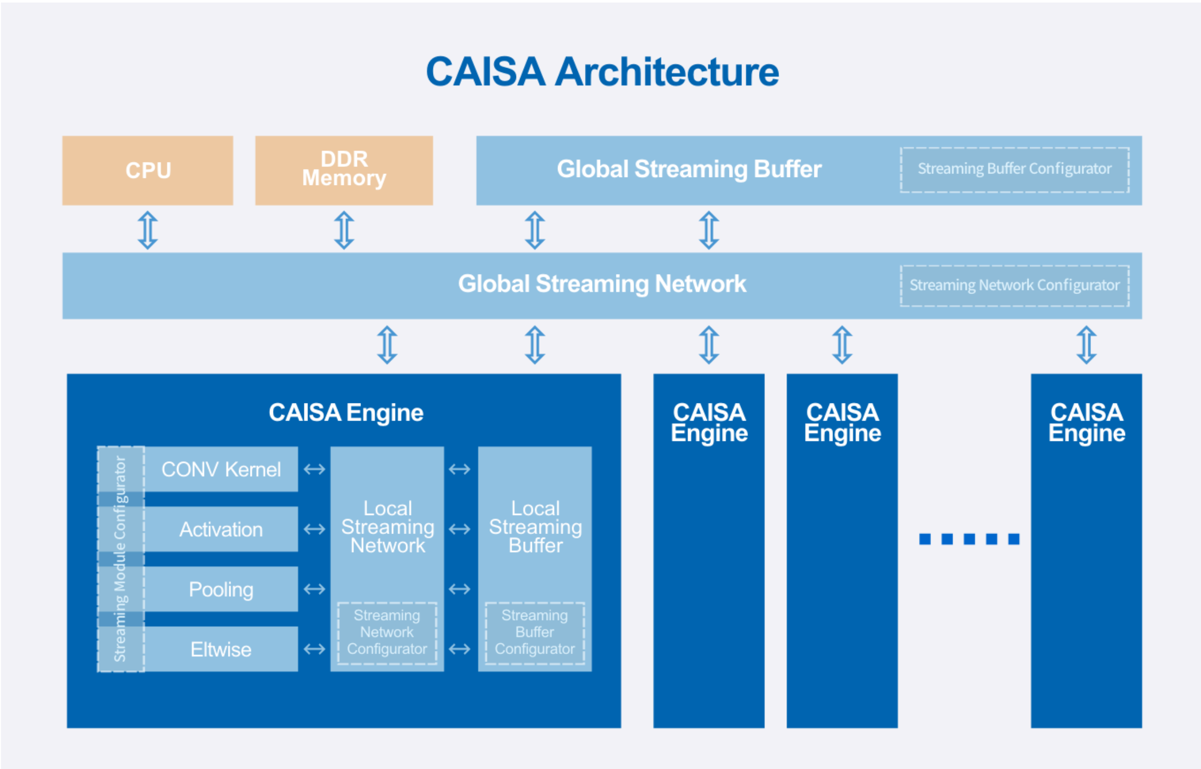

图 3-1 CAISA3.0 架构示意图

CAISA 架构包含三大部分：

（1） **CAISA 引擎（CAISA Engine）：** CAISA 引擎可以被分时配置为针对特定 AI 算法的定制流水线，其包含四个主要组成部分，可以支持不同的 AI 算法的计算加速。

- 数据流计算模块：包含卷积、池化、激活等不同 AI 算子计算模块，是实现计算功能的具体硬件单元；

- 局部数据流网络：实现数据流计算模块、局部数据流缓存与全局数据流网络三者之间的动态连接；
- 局部数据流缓存：支持引擎内数据流的存储、共享、及汇聚；支持数据流在不同数据流计算模块间流动，以组成定制的流水线。
- 数据流重组配置器：其根据配套软件工具生成的 CAISA 引擎配置，定义数据流连接关系及状态、以及数据流计算模块的计算模式，从而完成不同 AI 算法在 CAISA3.0 架构的映射。主要包含三种：第一种为数据流网络配置器，其配置数据流网络的连接关系及运行状态；第二种为数据流计算模块配置器，其配置数据流计算模块的计算模式；第三种为数据流缓存配置器，其配置局部数据流缓存的连接关系和运行状态。

（2） **全局数据流网络：**实现不同 CAISA 引擎、全局数据流缓存与外部存储器之间的动态连接；

（3） **全局数据流缓存：**支持数据流的存储、共享、及汇聚，支持数据流在不同 CAISA 引擎间的流动。

CAISA 架构提供三项核心技术，解决了数据流架构的三大挑战性难题：

（1） 时钟精确计算：提供最优算力性价比；

（2） 流水线动态重组：支持针对不同算法形成定制化流水线；

（3） 算法端到端自动化部署：支持算法从开发框架到芯片的端到端自动化部署，提升软件易用性。

图 3-2 CAISA 架构的三大技术突破

对三种技术突破的详细描述，如下文四、五、六章所述。

4 时钟级精确的计算：提供高算力性价比

CAISA 架构突破指令集限制，整体架构基于数据流方式设计，底层硬件设计中每个时钟的数据传输和计算操作都可精确预知，从而可实现时钟级精确的计算。在时钟级精确的硬件系统中，CAISA 架构通过数据计算与数据传输重叠、算力动态均衡、数据流带宽复用等技术，进一步提升算力利用率。

4.1 数据计算与数据流动重叠：压缩每个闲置计算时钟

CAISA 架构通过数据流流动方式及数据顺序控制计算次序，通过这种方式，CAISA 去除所有指令执行带来的冗余，并将数据传输时间隐藏于计算时间之内，从而充分压缩计算单元的空闲时间，发挥其最大的利用率。

以简化的深度学习算法为例，如下图 4-1 所示。其中 A、B、C、D 四种计算模块依次连接构成异构计算的流水线。在第一个时钟周期，数据流 1 抵达模块 A，并在第二个时钟周期输出数据流 2、3 至模块 B1 和 B2。数据流在模块中一个个向后传递，并在第四个时钟周期抵达模块 D。由于每个模块和每个数据流的时钟级精确，在第五个时钟周期后，每个时钟，所有 8 个数据流以及 7 个数据流计算模块中都在并行进行数据流动以及数据计算，利用率为 100%，从而能够突破受限于指令执行以及数据等待带来的芯片利用率瓶颈。

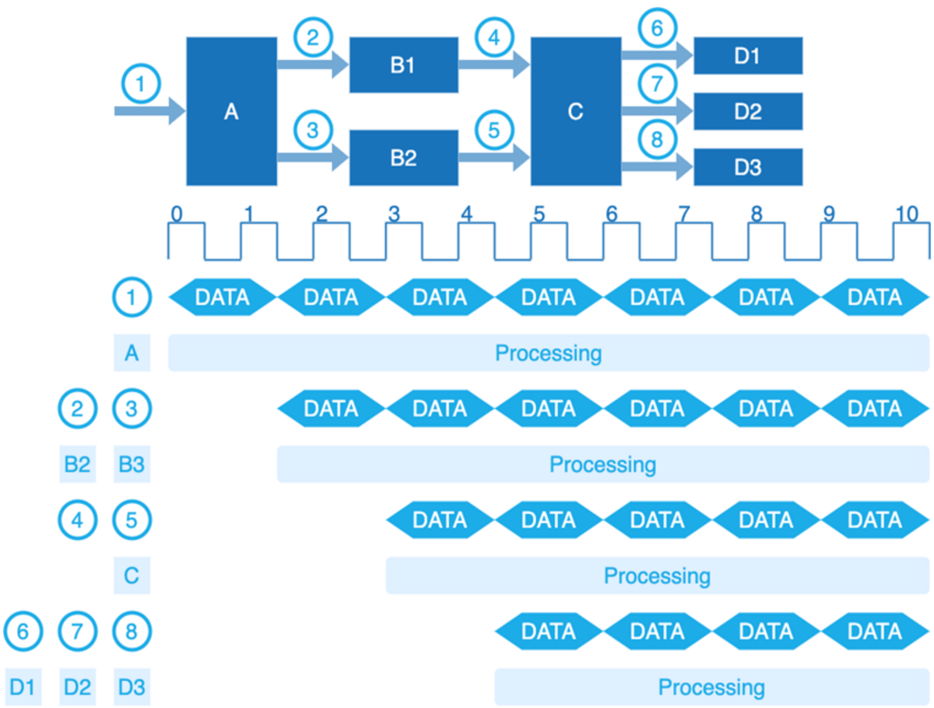

图 4-1 数据流动与数据计算重叠示意图

4.2 算力动态均衡：消除端到端流水线性能瓶颈

深度学习算法中包含卷积、池化及激活函数等多种不同的算子，对应的，CAISA 架构中每个计算引擎包含数个大小不一的计算模块，相互之间通过数据流网络连接组成深度定制的 CAISA 流水线。在每个 CAISA 流水线中，不同数据流模块所提供的计算吞吐率不同。为进一步提升芯片利用率，CAISA 架构

支持不同数据流模块间以串行和并行的方式互联，以平衡不同模块的吞吐率差异，实现流水线性能的最大化。

在下图 4-2 所示案例中：ABCD 四个计算模块串行连接，其中 A 模块每个时钟周期输入 1 个数据，输出 2 个结算结果，C 模块每个时钟周期输入两个数据，输出 3 个计算结果。BD 模块每个时钟周期输入 1 个数据，输出 1 个计算结果。

（1） 如果四个模块直接串行连接，受限于 B 和 D 模块的处理带宽，最终本流水线每个时钟只能产生 1 个计算结果，AC 模块中大部分算力处于闲置状态；

（2） 在 CAISA 架构中，通过算力动态均衡，2 个 B 模块以及 3 个 D 模块接入本 CAISA 流水线中，从而保证本 CAISA 流水线每个时钟能够产生 3 个结果，所有 ABCD 模块每个时钟均处于满载计算状态；

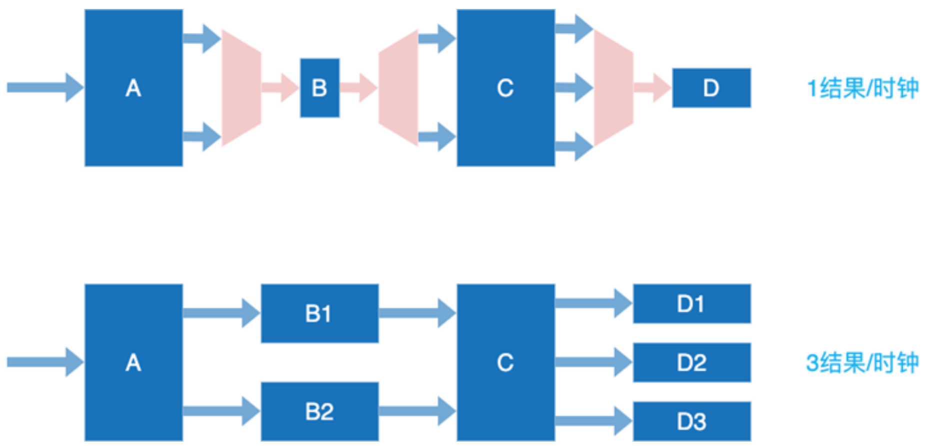

图 4-2 算力动态平衡示意图

4.3 数据流时空映射：最大化复用内部数据流带宽

AI 计算芯片的另一个核心资源为数据传输带宽，尤其是与外部存储器的数据交互带宽。在 CAISA 架构内部通过不同的映射方式支持不同的数据流流动方式，从而将数据流在芯片内部最大化复用，降低对于片外存储带宽的需求。

CAISA 架构中缓存空间在不同情况下连接数据流，提供在物理空间上的数据传输带宽以及在时间上的数据缓存。下图中列举了两个数据流映射的情况：

（1） 在空间映射（a）中，A、B、C 三个计算模块都需要外部存储带宽以完成各自的计算任务。

（2） 在时间映射（b）中，A、B 模块所需数据流在时间上具有差异：模块 A 在 T1 时间写出数据流，而模块 B 在 T2 时间读回数据流。

为最大化复用外部存储的带宽资源，CAISA 架构可将数据流连接映射至芯片内部数据流网络及数据流缓存，从而降低对于带宽资源的需求：

（1） 空间映射：通过将 ABC 模块的连接关系映射至 CAISA 内部数据流网络，同一计算过程对于 DDR 带宽的需求从 6 次读写降低至 2 次读写，带宽需求降低 3 倍；

（2） 时间映射：通过将 AB 模块间嵌入数据流缓存模块，在保证数据流计算正确性的同时，同一计算过程对于 DDR 带宽的需求从 4 次读写降低至 2 次读写，带宽需求降低 2 倍。

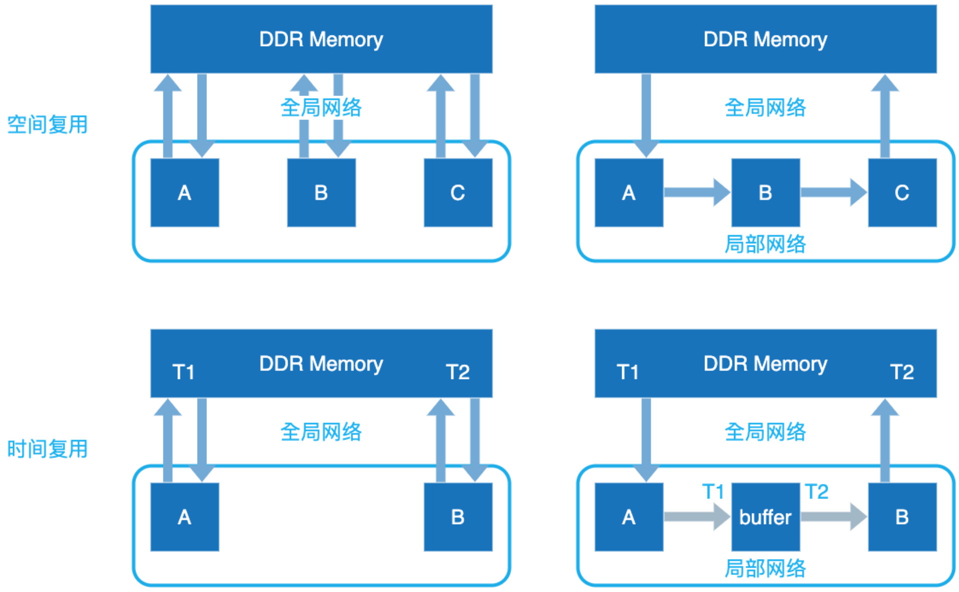

图 4-3 数据流时空映射示意图

5 流水线动态重组：实现高架构通用性

CAISA 架构可通过数据流网络对流水线进行动态重组，从而可以支持不同算法的计算需求，满足计算平台的通用性要求。如下图 5-1 所示，CAISA 架构中资源的不同配置可对应支持不同的人工智能算法，并且可以在运行过程中对流水线进行动态重组，从而支持架构在不同算法之间实现动态切换。

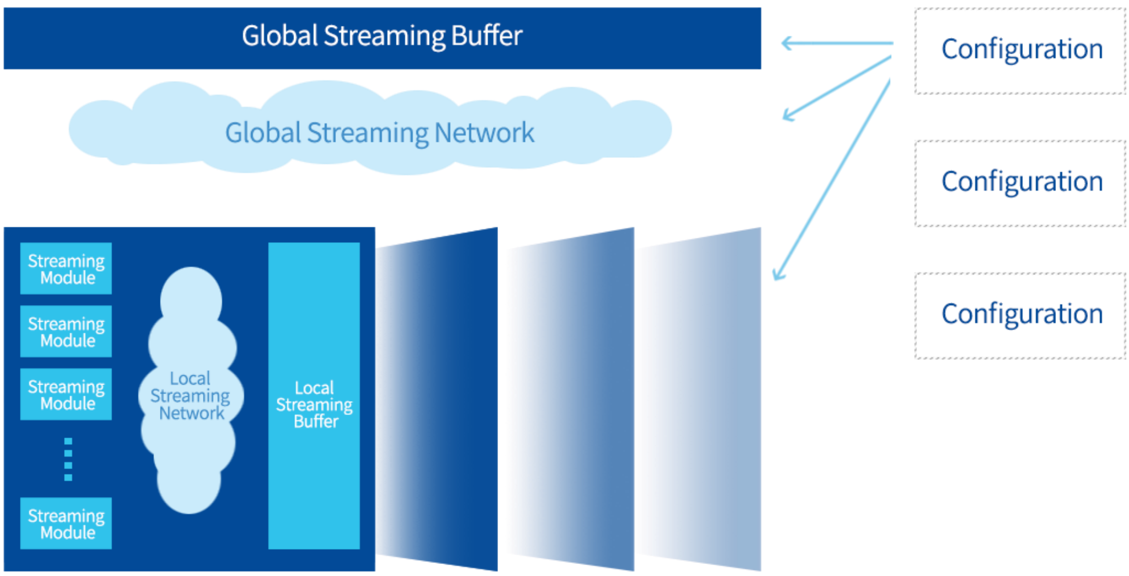

图 5-1 CAISA 架构中资源配置示意图

5.1 数据流连接关系映射：重组流水线计算次序

在 CAISA 中，采用数据流图来表达不同人工智能算法的计算过程。在数据流图中，AI 算法中的算子以图的方式相互连接，进而组成整个计算网络。

算法的数据流图中包含了本算法执行所需的所有算子，以及算子间的连接关系。不同 AI 算法对应不同算子的组合关系，为有效支持不同人工智能算法，CAISA 架构支持将 AI 算法内部的算子间连接关系映射成为不同的数据流流动方式。如下图 12 所示，针对两种不同的 AI 算法，局部和全局数据流网络可将不同计算模块按照不同先后顺序连接至数据流流水线内部，从而支持形成不同的 CAISA 流水线。而每条流水线都是针对算法定制的，可以充分保证计算的高性能。

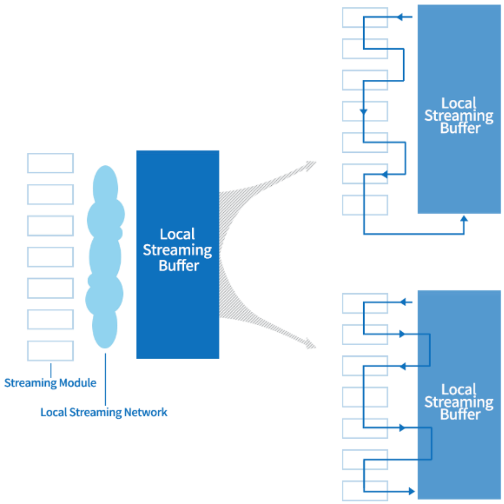

图 5-2 流水线重组示意图

5.2 数据流运行动态映射：计算模块时钟级协同

除了算子的连接关系外，算法的数据流图中包含每个数据流图节点的运行状态，在 CAISA 架构图中，数据流图的节点可被映射为：

（1） 数据流计算模块：完成目标算法中的计算操作；

（2） 数据流缓存模块：完成 CAISA 架构中数据缓存、共享与不同流水线间的连接；

数据流图的连接关系及数据流图节点可以映射至 CAISA 架构中对应的硬件模块，为保证 CAISA 流水线的正确运行，每个模块时钟级的运行状态需要在映射时做出定义。以下图 5-3 为例，流水线中包含 7 个数据流节点以及 11 个数据流，其中 7 个数据流节点映射至数据流计算模块及数据流缓存模块，而 11 个数据流映射至全局及局部互联网络。

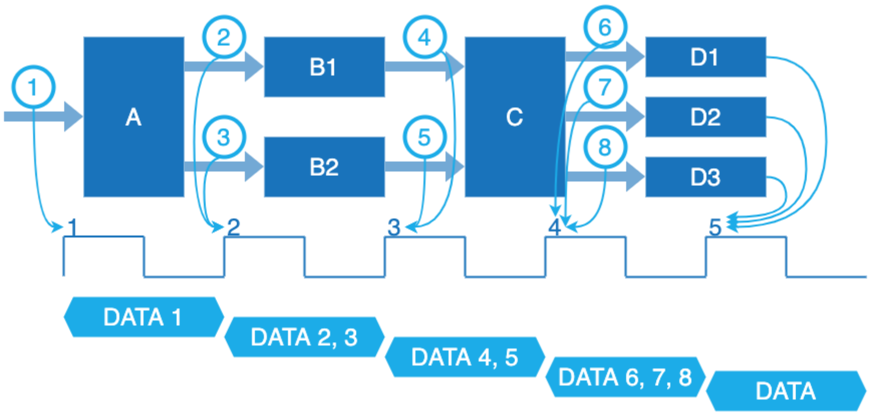

图 5-3 计算模块时钟级协同示意图

完成映射后，7 个数据流节点状态配置如下：

（1） 在时钟 1，数据流 1 及模块 A 启动；

（2） 在时钟 2，数据流 2、3 及模块 B1、B2 启动；

（3） 在时钟 4，数据流 4、5 及模块 C 启动；

（4） 在时钟 5，数据流 6、7、8 及模块 D1、D2、D3 启动

（5） 在时钟 8，数据流 9、10、11 启动。

当配置为以上状态时，数据流架构可以保证每个时钟周期所有计算单元进行有效计算，并产生正确结果。在 CAISA 架构中，数据流图节点运行状态包含启动、暂停、清空等多种运行状态，在每个状态进行时钟级的精确定义后，CAISA 流水线完成整体映射，可输入原始数据启动算法计算。

5.3 数据流配置器：流水线配置自动化

CAISA 架构的映射需要芯片内部的数据流配置器来完成：当所有数据流动态重组所需状态已明确，数据流配置器读取对应配置，并将对应连接关系、模块运行状态、运行所需参数配置到 CAISA 架构的各个模块。CAISA 架构中，设计了数据流缓存配置器、数据流网络配置器以及数据流计算模块配置器，这些配置器可以自动化地将不同功能单元面向不同算法形成定制的流水线，并可以在面向不同算法的不同流水线间无缝切换。流水线动态切换的顺序如下：

（1） 启动数据流动态重组，停止数据流输入，清空流水线；

（2） 各配置器完成所需配置，实现流水线重组；

（3） 启动数据流，输入新算法所需数据。

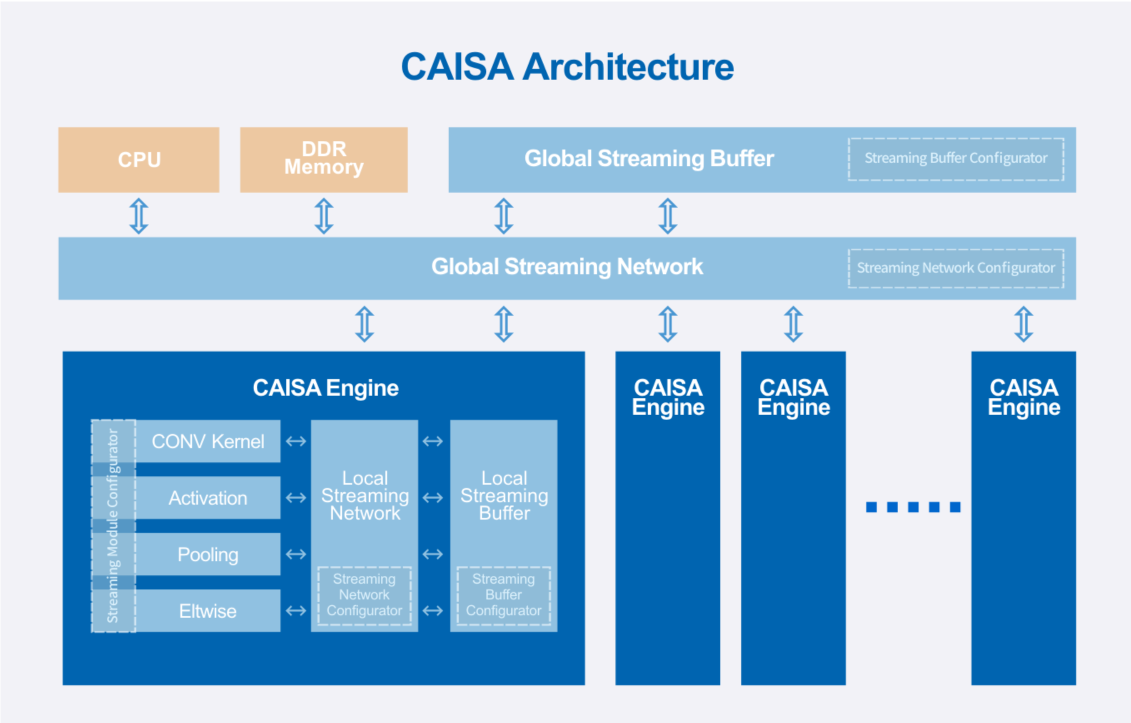

图 5-4 CAISA 架构中各配置器示意图

6 算法端到端自动部署：保证高软件易用性

作为计算平台，软件易用性是核心性能之一。我们以基于 CAISA3.0 架构的单颗 CAISA 芯片为例，其中包含接近 2 万个数据流计算模块，这 2 万个数据流计算模块通过嵌入其中的数据流网络、数据流缓存组成 4 个 CAISA 数据流引擎，其中每个引擎可被动态重组为一个 CAISA 流水线。换句话说，在 CAISA 芯片内部高效支持一个人工智能算法运算，需要准确配置 2 万个数据流模块、所有数据流网络以及数据流缓存的时钟级精确运行状态。显然，如果没有有效的软件支持，这种配置是难以实现的，CAISA 架构将无法实际使用。

CAISA 芯片配备的 RainBuilder 工具支持端到端自动化映射人工智能算法，其包含两大模块：

（1） 编译器 RbCompiler：自动化将深度学习开发框架开发的算法转换为针对 CAISA 底层架构的数据流图；

（2） 运行时系统 RbRuntime：提供用户级软件接口，与高级语言编程环境兼容，支持用户最大化发挥 CAISA 架构的性能。

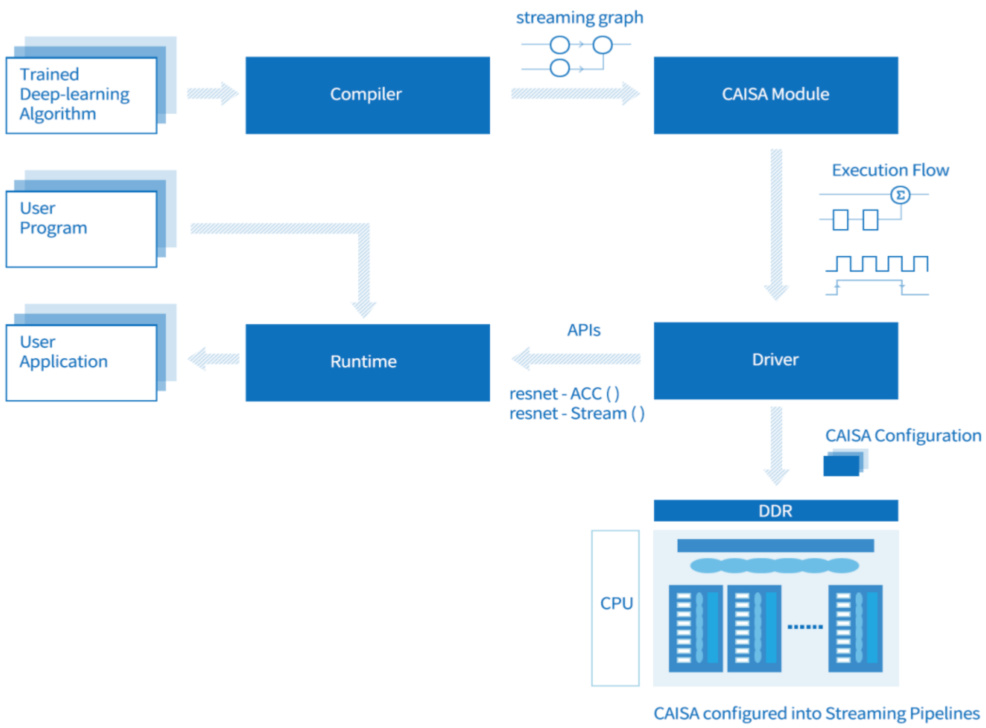

图 6-1 RainBuilder 工具工作流程示意图

6.1 RbCompiler：算法信息自动提取、优化及转换

编译器的输入为通过 AI 算法开发框架训练好的网络，输出为面向 CAISA 架构的数据流图。该数据流图定义了目标深度学习模型到 CAISA 平台上时钟级精确运算流的映射。RainBuilder 的编译器支持 Tensorflow、Caffe、ONNX、PyTorch 等业界主流人工智能开发框架，主要包含三个功能模块：

（1） 算法解析模块：读取不同 AI 开发框架下训练模型文件，提取对应模型中的模型结构和超参数信息，以计算图的形式存储；

（2） 图优化模块：根据 CAISA 底层硬件设计对原始计算图进行优化，包括冗余结构折叠、算子融合、定制化拓扑排序、计算资源分配等。

（3） 数据流图生产模块：针对优化后算法信息，对应转换为数据流中间表达，并生成数据流计算图。数据流计算图中涵盖了目标深度学习网络在 CAISA 架构上运行的所有信息（CAISA 模块及其连接关系），以.sg 的文件格式保存为模型文件。

图 6-2 RainBuilder Compiler 示意图

6.2 RbRuntime：与人工智能应用集成，完成算法自动部署

当完成深度学习算法模型到 CAISA 架构的映射后，运行时系统提供用户高层软件开发接口。为了方便用户实现基于 CAISA 架构的算法部署，RbRuntime 中整合以下特性：

（1） 硬件架构透明：RbRuntime 通过调用数据流驱动 RbDriver 完成 CAISA 模块的调用，而驱动对域用户是不可见的。RbDriver 提供了软件与底层芯片交互的接口，通过各配置器控制 CAISA 流水线的行为，并以此完成计算图在芯片上的流转。

图 6-3 RainBuilder RbDriver 示意图

（2） 软硬件高效协同：RbRuntime 中包含了两种线程池，分别对应管理调度线程和计算线程。其中调度线程负责在时间上软硬件的并行，计算线程负责在计算资源上的并行。两种线程池的联合保证了用户应用的端到端性能。

（3） 多算法上下文更新：RbRuntime 支持多算法的在 CAISA 平台上统一的计算资源和数据依赖管理，实现不同算法之间的无缝切换。 当多种 AI 算法出现在同一应用中时，RbRuntime 可以自动按需调用对应算法的 runner 示例，最小化不同算法间的调度开销。

6.3 基于 RainBuilder 的算法部署

通过 RbCompiler 和 RbRuntime 的协同，用户只需以下两步即可完成深度学习算法在 CAISA 架构上的运行：

（1） 使用 RbCompiler 将 TensorFlow、Caffe、ONNX、PyTorch 等框架下训练的模型文件转化为基于数据图的 sg 模型文件。

（2） 以 sg 文件输入，通过调用 RbRuntime 运行接口完成在鲲云加速卡上的推演过程。

图 6-4 使用 RainBuilder 进行算法部署与集成流程示意图

7 CAISA3：全球首颗数据流 AI 芯片

基于 CAISA3.0 架构，鲲云设计并实现了 AI 推理加速芯片 CAISA。该芯片采用 28nm 工艺制造，可达到 10.9 TOPS 的峰值性能，该芯片已完成全面验证并处于量产状态。

7.1 AI 加速器

CAISA 芯片被设计为辅助加速器，通过与主处理器配合实现 AI 计算的加速。其中主处理器用来运行应用程序并支持设备驱动。CAISA 芯片通过 PCIe Gen3 接口与主处理器通信，通过 PCIe 通道，处理器可以以 32Gbps 的吞吐量将数据传输到芯片中。该芯片还具有双 DDR4 通道，支持大容量设备侧本地存储器，以执行所需的深度学习网络计算。通过这种设计，用户可以将计算量大的深度学习负载卸载到 CAISA 芯片上，并以最少的数据传输获得计算果。

图 7-1 CAISA 芯片示意图

7.2 高性能设计

CAISA 芯片的内部结构如下图所示。单个芯片中放置了 4 个 CAISA 引擎，具有超过 1.6 万个 MAC（乘累加）单元以及所有辅助逻辑。为了支持较高的硬件资源利用率，鲲云设计了分布式数据流缓存，为每个 CAISA 引擎提供超过 340Gbps 的带宽。CAISA 引擎本身是基于对常用神经网络模型的计算量统计进行优化的。其不仅为常见的神经网络计算（如 Pooling，ReLU 等）实现了专用的硬件计算模块，而且它们与卷积计算的比例都经过平衡，从而可在常用 AI 算法中实现最佳性能。

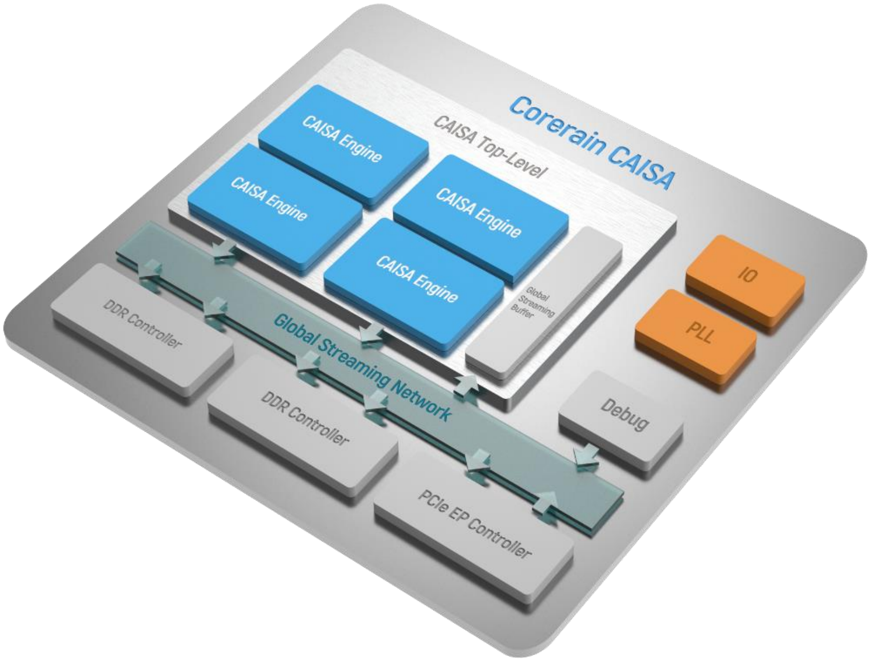

图 7-2 CAISA 芯片内部结构示意图

在 CAISA 引擎之外，还有一个全局数据流网络，该网络将输入数据和网络参数分发给引擎并获得结果。该网络以纳秒级精度设计和实现，以匹配引擎间的同步控制。在芯片级别，通过具有 AMBA 总线的 NoC(片上网络)，将 4 个 CAISA 引擎、 2 个独立的 DMA 控制器、PCIe 控制器和 2 个 DDR 控制器连接。为了进一步优化性能和效率，CAISA 中还设计了专用的 DMA 和 NoC。

7.3 软件支持

CAISA 芯片支持几乎所有的常用 AI 算子，通过对数据流网络中算子的不同组合，CAISA 芯片可以支持决大多数的 AI 算法，所有算子支持和算子组合都可以通过软件配置来实现，鲲云的 RainBuilder 3.0 工具链完全支持该芯片，该工具链为客户提供了端到端的推理模型部署解决方案。

7.4 芯片主要技术指标

- 峰值性能：10.9TOPS
- 外部接口：PCIe Gen3 4-lanes
- 存储接口：dual-channel DDR4-2400, 64 DQ per channel
- 工作温度：-40℃ ~ 125℃
- 芯片封装：FC-BGA 1152, 35mm^2

8 星空加速卡 X3

鲲云星空 X3 加速卡是全球首款搭载了 CAISA 芯片的数据流架构深度学习推断设备。X3 加速卡是一款工业级 HHHL（半高半长）单槽规格的 PCIe 板卡，其功耗小于 60W。得益于其轻量化的规格特点，这款高性能加速卡可以与不同类型的计算机设备进行适配，包括个人电脑、工业计算机、网络视频录像机、工作站、服务器等。有关 X3 加速卡详细的规格请见表 8-1。

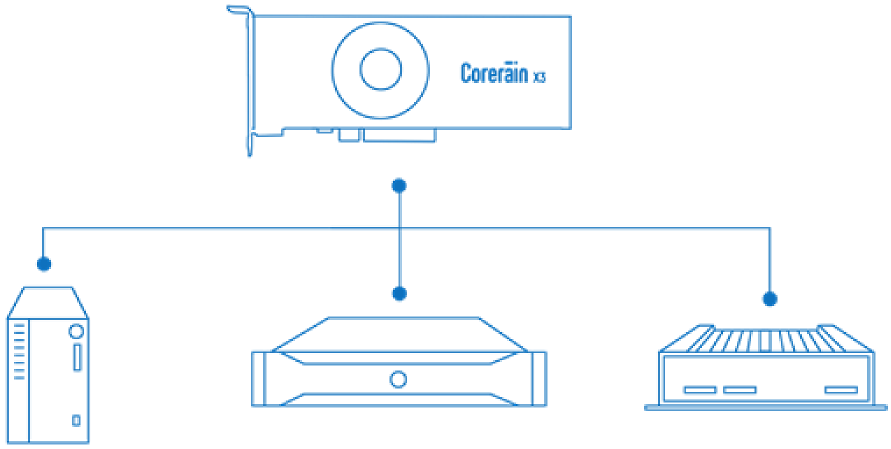

图 8-1 星空加速卡 X3 可适配不同类型计算设备

表 8-1 鲲云星空加速卡 X3 加速卡规格

主芯片	CAISA
峰值性能	10.9TOPS
芯片利用率	高达 95.4%
接口	PCIe 3.0 x8
内存	板载 8GB DDR4
电源	PCIe 供电
散热方式	主动散热（单风扇）
工作温度	-20℃ ~ 70℃
外形尺寸	169.5mm x 69.6mm（PCIe 半高半长标准，单槽位）

由于其独特的数据流架构（详见前章节描述），CAISA 芯片的设计可以支持大多数常用的深度学习算子。因此从设计的角度来说，任何搭载该款芯片的设备均可以加速绝大多数主流的深度学习网络，且实现低延迟高芯片使用率（理论上使用率可达 95.4%甚至更高）的加速效果。X3 加速卡的 Benchmark 如表 8-2。

表 8-2 鲲云星空 X3 加速卡 Benchmark（Batch Size=4）

模型名称	网络来源	数据集	吞吐[FPS]	延时[ms]	芯片利用率
ResNet-50	TensorFlow	ImageNet	1306.93	3.06	92.3%
ResNet-152	TensorFlow	ImageNet	460.27	8.68	95.4%
YOLOv3	DarkNet	COCO	125.75	31.06	82.4%
SSD-ResNet50	NVIDIA	COCO	182.16	21.96	77.1%
U-Net Industrial	NVIDIA	COCO2017	54.01	74.07	65.0%

如 Benchmark 所示，所有在 X3 加速卡上测试运行的网络均可达到 65%以上的实测芯片使用率。该测试结果完全达到了 CAISA 3.0 AI 芯片的设计规格。另外，本次公布的 benchmark 网络涵盖了深度学习领域三个主要的应用类型（分类、检测和语义分割）。因此，这款 X3 加速卡可以为终端用户提供足够高的设计自由度，以满足不同的 AI 应用需求。

9 参考文献

[1] Victor W Lee, Changkyu Kim, Jatin Chhugani, e.t.al. Debunking the 100X GPU vs. CPU Myth: An Evaluation of Throughput Computing on CPU and GPU[C]. ISCA'10, Saint-Malo, France. 2010, pp.451-460.