# 第3章 抗量子密码芯片架构

“硬件，是密码学必不可缺的伙伴。”

“Hardware: an essential partner to cryptography.”

——IEEE Fellow、IACR Fellow、比利时皇家科学院院士 Ingrid Verbauwhede

芯片架构，即芯片内部的电路功能实现与组织方式，直接决定了芯片在不同技术维度上所能达到的极限，如计算速度、延迟、功耗、功能灵活性等。与其他领域的数字电路芯片不同，密码芯片作为对密码算法进行硬件加速的物理载体，在其设计过程中不仅要考虑性能、功耗以及面积（成本）等通用技术指标，还需考虑其针对侧信道攻击、故障注入等物理攻击的防御能力，即物理安全性。对于抗量子密码芯片而言，抗量子密码算法的多样性与动态演进性进一步强化了功能灵活性方面的要求。因此，如何在能量效率（综合衡量计算速度与功耗的指标）、功能灵活性与物理安全性这 3 个核心技术指标构造的设计空间内探索理想的芯片设计方案是抗量子攻击密码芯片设计的核心技术难题。

本章首先对密码芯片的设计空间进行讨论，然后在此基础上对面向抗量子密码算法的领域定制芯片架构研究进行深入分析。

## 3.1 抗量子密码芯片设计空间

密码芯片的软硬件系统自顶向下可依次分解为应用/协议、密码算法、芯片架构、电路设计、器件结构与材料等层次，如图 3-1 所示。其中，密码协议与算法属于软件范围，其他则属于硬件设计的范畴。在 Communications of the ACM 2021 的一篇文章中提出了硬件彩票的概念[1]，指出某些软件算法或者模型取得的成功并非来自其本身，而是恰好适配了特定的硬件架构。因此，在密码应用与算法确定的情况下，通过芯片架构与电路层次的定制优化可以获得最大的性能提升。因此，本书重点讨论从抗量子密码算法分析出发，如何通过芯片架构与电路设计，实现对多种不同数学困难问题的抗量子密码算法的高能效支持。在当前通过集成电路制造工艺尺寸缩小来保持性能持续增长难以为继的背景下，集合应用需求和算法特征属性，充分挖掘架构和电路的潜在空间来优化整体性能成为当前的主流趋势。首先，在架构层面的技术优化可以获得最大的功耗优化效果。MIT 的研究人员以矩阵乘法实现为例（矩阵规模为

4096×4096)，评估了不同优化技术对计算速度的影响[2]。使用高级编程语言 Python 实现版本作为计算速度的对比基准，用 C 语言实现相同功能后，性能可提升 47 倍。继续采用循环展开、向量化并行计算等架构优化技术，可以将性能最高提升 6.3 万倍。正如图灵奖得主 John Hennessy 和 David Patterson 在 2018 年的计算机体系架构顶级学术会议 ISCA 的特邀报告中强调，应用驱动的体系结构与领域定制语言的开发，对于提升系统性能、能量效率和开发效率至关重要。领域定制的计算架构研究正迎来其黄金时代[3]。

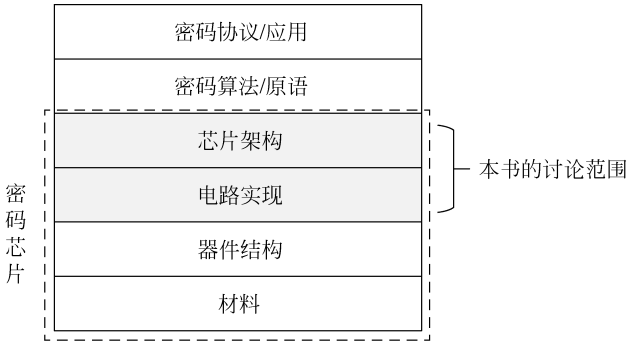


图 3-1　密码芯片的系统层次与本书的讨论范围

谈到芯片架构，可能会立即想到 x86、ARM、RISC-V 和 MIPS 等不同指令集的处理器架构。而本书的抗量子密码芯片架构主要指对算法进行硬件加速的具体实现方式。从系统设计人员角度出发，通常希望能够将密码相关的计算任务从 CPU 负载中卸载出来，将 CPU 更多地投入核心业务相关的计算与调度中。因此，根据 SoC 系统中 CPU 是否直接参与抗量子密码计算以及 CPU 与抗量子密码加速模块之间的数据通信开销，将领域定制抗量子密码芯片分为松耦合抗量子密码芯片与紧耦合抗量子密码芯片两种类型。在松耦合设计方案中，与 SoC 中其他的加速 IP 一样，抗量子密码芯片作为挂在总线上的加速引擎，独立完成抗量子密码算法的所有操作。紧耦合设计方案则是在深入分析抗量子密码算法计算特征的基础上，提取底层核心运算模块并嵌入已有的 CPU 计算通路中，通过相应的指令集扩展实现对目标算法的支持。由于松耦合设计采用全硬件加速，相对而言可以获得更高的性能与能量效率，而紧耦合设计则具有相对更高的灵活性。为了改善传统松耦合设计的功能灵活性限制，通过对目标算法的领域完备算子提取和配置机制优化，面向领域定制的粗粒度可重构计算架构可实现对功能灵活性与能量效率的动态折中。如图 3-2 所示，书中将这种粗粒度可重构抗量子密码芯片与面向特定算法的全定制加速芯片都归类为松耦合抗量子密码芯片。

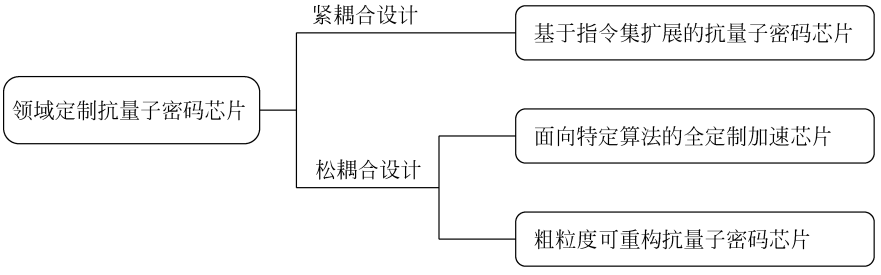


图 3-2　领域定制抗量子密码芯片的主要类型

无论哪种类型的抗量子密码芯片设计，首先都需要明确一个由技术指标约束的统一设计空间。对于抗量子密码算法和芯片设计而言，二者的设计目标存在着一定差异。如图 3-3 所

示，对于抗量子密码算法设计而言，算法的抗量子攻击属性是算法方案设计最重要的指标，然后才会继续考虑算法是否具有足够高效的计算效率或与传统公钥相比类似的存储需求等。密码芯片则是由能量效率、功能灵活性和物理安全性 3 个核心技术指标来评估[4,5]。其中，能量效率作为一个综合性技术指标，是芯片计算速度与功耗之间的比值。在任意应用场景下，芯片设计人员均期待实现比较高的能量效率。对于高性能应用场景，芯片计算速度（高吞吐、低延迟）成为芯片设计的最高优先级；而在边缘端或物联网应用中，则对芯片的整体功耗开销提出了更高的设计要求。第二个重要技术指标是功能灵活性，指的是密码芯片所能支持的密码算法数量。对于抗量子密码芯片而言，在功能上首先需同时支持密钥封装、公钥加密和数字签名等密码原语。其次，要能够同时支持基于不同基础数学困难问题的算法。最后，要兼容密码算法的不同安全等级以满足不同安全强度的应用需求。抗量子密码芯片的第三个技术指标是物理安全性，指的是密码芯片针对功耗/电磁攻击、错误注入攻击等侧信道攻击的防御能力。密码芯片的这 3 个技术指标是相互矛盾的，难以同时兼顾。提升能量效率一定是基于算法计算与存储模式的深度定制，那么必然会限制芯片的功能灵活性，并且与算法高度相关的硬件侧信道信息会造成抗物理攻击能力不足；物理安全性的提高由于额外的防护开销会降低芯片的能量效率，而且针对特定敏感路径的方法具有很强的算法差异性，会限制物理安全防护方式的灵活性。

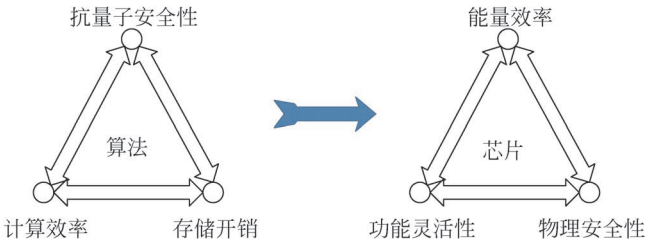


图 3-3 抗量子密码算法到芯片实现的设计空间转变

数字电路芯片主要由数据通路和控制通路两部分组成。其中，数据通路决定了芯片执行的具体功能，而控制通路通过对电路的功能配置和调度决定执行特定算法时的性能优劣。基于指令集扩展的设计方案是一种与处理器紧密耦合的软硬件协同设计方案。其设计方法是从所支持的算法出发，提取可以支撑所有算法的底层算子并嵌入通用处理器的算术逻辑单元（Arithmetic Logic Unit，ALU）中，进一步基于已有的指令集格式进行指令扩展，从而实现与通用处理器生态相兼容、能量效率进一步得到提高的抗量子密码芯片解决方案。

全定制硬件实现的设计方案是一种松耦合的算法加速 IP 设计方案。这种方案仅针对某个特定算法或者计算特征高度趋同的某一类数学困难问题算法，开展面向完整算法的全定制硬件设计。显然，这种解决方案可获得最高的能量效率，但难以保证功能灵活性。同时，由于硬件功能固定且可编程性不足，难以抵御后续可能出现的侧信道攻击威胁。

从计算单元粒度和功能灵活性角度来看，粗粒度可重构设计方案处于上述两种设计方案之间。它具有一定的硬件可编程性，可以通过芯片的编译系统实现对芯片配置信息的动态修改，从而提高芯片的物理安全防护能力。图 3-4 对 3 种实现方案对能量效率、功能灵活性和物理安全性这 3 个技术指标进行了对比。

接下来将分别介绍基于指令集扩展、全定制硬件实现以及粗粒度可重构抗量子密码芯片设计的代表性工作。

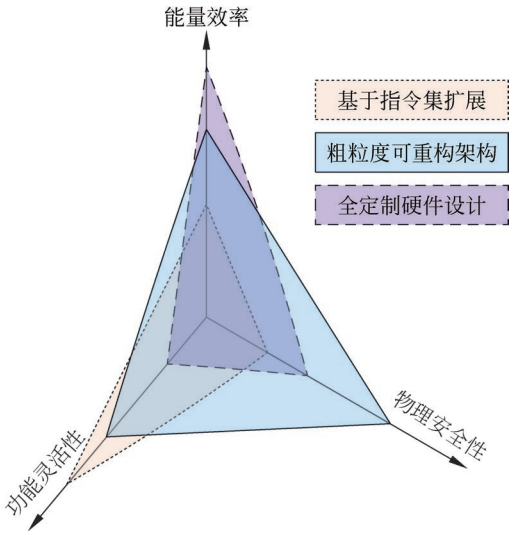


图 3-4　不同领域定制抗量子密码芯片的技术指标对比

## 3.2　基于指令集扩展的抗量子密码芯片架构

基于指令集扩展的抗量子密码芯片作为一种与 CPU 紧密耦合的软硬件协同设计方案，通过对计算通路进行细粒度定制优化与专用指令集扩展，实现对抗量子密码算法的支持。在对抗量子密码算法实现性能评估过程中，也出现了大量采用特定向量化指令集（如 AVX2、NEON 等）对不同算法进行软件加速的研究工作。随着国际抗量子密码算法标准的确定，产业界也开始针对抗量子密码标准算法展开专用指令集的研发工作。RISC-V 在 2023 年底成立了抗量子密码工作组，重点针对标准化的 Kyber、Dilithium 和“SPHINCS＋”算法展开指令集扩展工作[6]。得益于 RISC-V 指令集的开源属性及其掀起的开源芯片浪潮，基于 RSIC-V 指令集扩展的抗量子攻击密码芯片研究已成为近期这一领域的主流方案。本节将重点介绍分别来自美国 MIT 高能效电路与系统研究团队和德国慕尼黑工业大学信息安全团队的代表性研究成果。

### 3.2.1　MIT Sapphire

MIT 高能效电路与系统研究团队在 2019 年的国际固态电路会议（International Solid-State Circuits Conference，ISSCC）和密码硬件与嵌入式系统会议（Conference on Cryptographic Hardware and Embedded Systems，CHES）上发表了面向物联网应用的低功耗基于格的密码处理器 Sapphire[7,8]。Sapphire 是公开发表的第一款经过流片验证的可配置抗量子密码芯片。该芯片基于 RISC-V 架构并通过软硬件协同设计方法实现对主流基于格的抗量子密码算法的支持。同时，针对功耗攻击等侧信道攻击，提出了基于软件方式的防护方法。如图 3-5 所示，Sapphire 包括一个 32 位的 RISC-V 核和密码协处理器 Sapphire 密码核。

Sapphire 支持 Frodo、Newhope 和 Kyber 等密钥封装算法以及 Dilithium 和 qTesla 等数字签名算法。作为一种软硬件协同设计方案，Sapphire 密码核仅负责算法中的哈希函数、采样及多项式乘法等核心运算，其他部分辅助功能需要通过 RISC-V 核实现。表 3-1 列出了不同算法计算过程在 RISC-V 核与 Sapphire 密码核之间的任务分配情况。

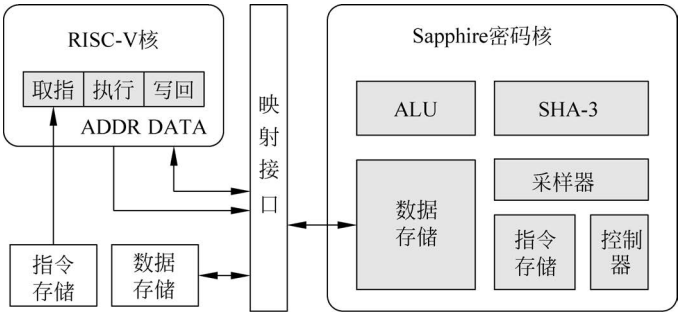


图 3-5 Sapphire 硬件架构

表 3-1 KEM 和 DS 在两核之间的任务分配

| 条　　件 | 密钥封装 KEM | 数字签名 DS |
|---|---|---|
| 支持的算法 | Frodo、NewHope、Kyber | Dilithium、qTesla |
| RISC-V 核 | 编码/压缩、密钥封装 | 编码/压缩 |
| Sapphire 密码核 | 公钥加密 | 签名 |

### 1. Sapphire 芯片设计

Sapphire 芯片设计的主要目标是物联网应用，因此其对于功耗及物理安全性的设计优先级更高。其主要创新点包括：①采用可配置的模乘法器；②高面积效率的快速 NTT 模块；③高能效的采样器设计。

该芯片的 Keccak 模块实现的是 f-函数一个周期执行完整一次的实现方式，根据 Keccak 模块的计算特点，执行 24 个周期完成一轮的 f-函数，然后根据算法要求的特点，按照每个系数的宽度输出到采样器。相对于其他的伪随机数生成器（如 AES、CHACHA20），平均生成一位的能量效率 Keccak(SHAKE-128)是最高的。

为了支持多种抗量子密码算法的采样器，对于常见的均匀采样、二项采样、高斯采样和三项采样，都分别设计了对应的采样器。对于拒绝采样来说，拒绝的阈值是可以配置的，一个时钟周期进行一次比较，如果输入元素不满足拒绝条件就进行有效输出，单周期最多输出一个元素。对于二项采样来说，随机位首先进行称重，然后彼此之间做模减运算并输出，实现方式也是串行实现，每周期输出一个有效系数。从硬件结果来看，二项采样器相比于之前实现的 Knuth-Yao 的高斯采样器，能量效率提高了 16 倍，原因是二项采样的算法复杂度和实现复杂度远小于高斯采样。

在计算通路上，NTT 算法的拓扑架构采用的是 out-of-place 的实现方式，输入 RAM 和输出 RAM 构成 Ping-Pong 形式的缓存，SRAM 因此都是单口的。相对于双口 RAM，单口 RAM 即使利用了双倍的存储空间，但是存储效率是提升的。并且对于蝶形运算而言，蝶形单元结合了 CT 和 GS 两种模式，其中模乘法器的模硬件单元为了支持可重构的模数，采用的是列举的形式，列举所有支持算法中的模数并实现其对应的移位和加减法电路。这种实现方式的缺陷是列举模数的实现方式支持的模数是有限的。从结果上来看，相对于 Cortex-M4 的软件实现，NTT 的能量效率有 7～11 倍的提升。

### 2. 指令集扩展设计

为了实现对密码计算模块的高效控制，Sapphire 在增加种子寄存器等 5 个自定义寄存器

的基础上增加了 34 条指令。如表 3-2 所示，这些指令对抗量子密码计算中的 NTT、模运算、哈希和采样等运算提供直接支持。由于每条指令对应的计算在延迟方面存在很大差异，使芯片的最高运行频率受限，在算法计算的绝对时间上存在一定损失。

表 3-2　Sapphire 自定义的指令

| 指令类型 | 数　量 | 主要功能 |
|---|---|---|
| 配置类 | 2 | 参数配置与时钟门控控制 |
| 寄存器操作 | 4 | 寄存器访问与运算 |
| 寄存器与多项式交互 | 4 | 多项式寄存器的访问操作 |
| 变换 | 2 | 主要用来实现 NTT 操作 |
| 采样 | 7 | 支持各种分布的采样操作 |
| 多项式计算 | 4 | 多项式初始化及计算 |
| 对比与分支 | 4 | 条件分支指令 |
| SHA-3 计算 | 7 | 哈希计算 |

### 3. 抗侧信道攻击分析

在 Sapphire 芯片设计中，敏感信息相关的 NTT、多项式计算以及采样模块都是恒定时间执行的，可以保证该设计不受计时攻击和简单功耗攻击（Simple Power Attack，SPA）的影响。为了应对差分功耗攻击，Sapphire 芯片进一步引入了通过扩展指令集实现带掩码防护抗量子密码算法的映射机制。评估结果表明 Sapphire 芯片可以在 3 倍开销情况下实现抗 DPA 攻击防护。

### 4. 芯片实现与性能评估

Sapphire 采用 40nm LPCMOS 工艺进行流片。芯片的工作频率为 12～72MHz，工作电压为 0.68～1.1V。Sapphire 密码核的面积为 0.28mm$^2$，包括 40.25kB SRAM 和 10.6 万等效门。在性能方面，Sapphire 芯片在执行 Kyber 和 NewHope 算法时，与在 Cortex-M4 微处理器上运行的软件实现相比，计算速度分别提高了 37 倍和 50 倍，同时计算能耗分别改善了 28 倍和 37 倍。同时，研究人员将 NTT 模块和中心二项分布采样模块的性能与相关工作进行比较。表 3-3 列出了 Sapphire 密码核与相关工作的技术指标对比。

表 3-3　Sapphire 密码核与相关工作的性能对比

| 性　能 | Cortex-M4 | ISSCC'15 [9] | CICC'18 [10] | Sapphire 密码核 |
|---|---|---|---|---|
| 工艺/nm | — | 130 | 40 | 40 |
| 源电压/V | 3.0 | 1.2 | 0.9 | 0.68～1.1 |
| 频率/MHz | 100 | 500 | 300 | 12～72 |
| 面积/mm$^2$ | — | — | 2.05 | 0.28 |
| 逻辑门 | — | — | — | 106k |
| 格密码类型 | 所有（软件） | Ring-LWE | Ring-LWE | Ring-LWE，Module-LWE |
| 支持的参数 | 所有 | $N$：256　$q$：7681 | $N$：64～2048　$q$：32 | $N$：64～2048　$q$：24 |
| NTT 性能（$N=256, q=7681$） | | | | |
| NTT 周期数 | 22031 | 1700 | 160 | 1288 |
| NTT 能量/nJ | $13.5\times10^3$ | — | 31 | 63.4 |
| 二项采样性能（$N=256, q=12289$） | | | | |
| 采样周期数 | 155872 | — | 3704 | 1009 |
| 采样能量/nJ | $95.8\times10^3$ | — | 1250 | 44.4 |

### 3.2.2 TUM RISQ-V

慕尼黑工业大学研究团队在基于 RISC-V 指令集扩展的抗量子密码芯片方向开展了多年持续研究，包括基于格的 LAC 算法和 FrodoKEM 算法实现[11,12]、基于超奇异同源的 SIKE 算法实现[13]、面向格密码算法的实现[14]等。本节将介绍该团队在密码硬件会议 CHES 2020 发表的 RISQ-V[15]和在 CHES 2022 上发表的支持掩码防护的指令集扩展工作[16]。基于 RISC-V 的面向抗量子密码的扩展硬件实现如图 3-6 所示。

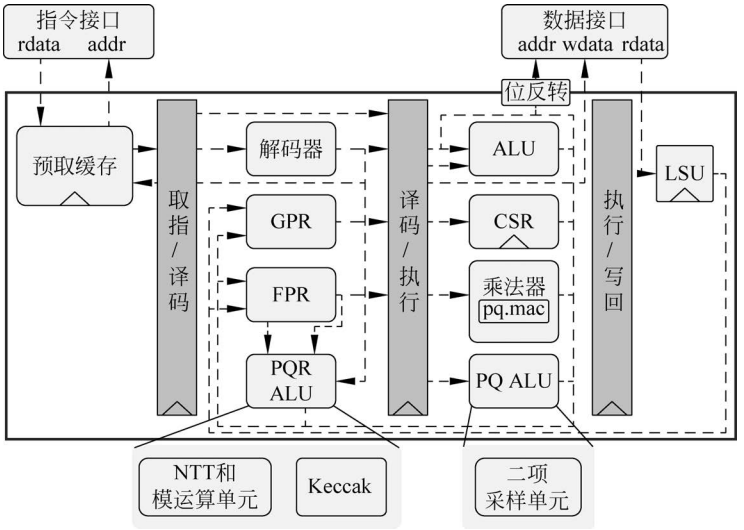


图 3-6 基于 RISC-V 的面向抗量子密码的扩展硬件实现

与该团队其他针对特定算法进行指令扩展优化的工作不同，RISQ-V 旨在最大化利用已有资源并减低数据访存的条件下通过指令集扩展来实现支持基于格的抗量子密码算法，提高能量效率的同时保持架构对演进算法的兼容性。为了实现这一目标，团队提出了一系列可以嵌入 RISC-V 流水线中的硬件加速器，并在此基础上提出了 29 条额外的基于格的密码计算指令。

该工作是基于传统 RISC-V 流水线上的扩展实现。在传统的流水线上，添加了面向抗量子密码的定制模块。相对于之前结构来说，RISQ-V 工作和传统通用处理器的耦合更加紧密，是在原有的 RISC-V 流水线添加的定制硬件模块。其优点是硬件开销小，开发成本低，灵活性更好，更容易和已有的软件栈匹配，对编译器等软件系统改动少；缺点是并行度上可能会有欠缺，在指令的控制上也会有软件必要的开销。因为通用处理器整个流水线的控制逻辑并不是针对一个特定领域单独设计，所以如果面向一个领域，就会存在一定的开销。

该工作在流水线上添加了两套定制的寄存器(32 个 32 位寄存器)，并且在执行阶段和写回阶段中间有面向抗量子密码的 MULT、抗量子 ALU、NTT 以及模算数单元、Keccak 单元等，这些硬件单元都是抗量子密码在执行过程中用到的硬件功能密码。对于该结构，计算单元执行完后就被写入专门的寄存器，进入新流水线的计算模式中。

从性能上看，该工作在 FPGA 和 ASIC 上都进行了综合。对 FPGA 来说，会比原来的通用处理器多了一万多个显示查找表(Look-Up-Table，LUT)，十几个 DSP。从时钟周期上看，比没有优化过的软件实现减少到原来的 1/3。对 ASIC 来说，在 UMC 65nm 工艺下，执行一个完整的方案，功耗约为 2mW，能量消耗为 100～200μJ。因为是流水线的扩展实现，因此在能

量消耗的减少上，完整硬件方案优化会更彻底。

RISQ-V 采用具有 4 级流水、顺序执行的 32 位 RISC-V 核 CV32E40P，并在 PULPion 平台上进行集成。处理器核主要包括预取缓存、指令译码器、通用寄存器组、浮点寄存器组、算术逻辑单元、乘法单元、状态控制寄存器和访存单元。为了支持额外的抗量子密码计算，增加了两个新的硬件组件 PQR-ALU 和 PQ-ALU。PQR-ALU 包括 NTT、模计算单元、Keccak 加速器。这两个模块需要对寄存器组进行并行访问。因此，PQR-ALU 直接放在解码阶段，从而避免在执行阶段与寄存器的路由信号连接。PQ-ALU 包括二项分布采样模块，由于与 MULT 单元和 ALU 单元具有类似的结构，即需要两个输入寄存器和一个输出寄存器。为了实现对已有硬件资源的复用，将 pq. mac 操作直接集成到 MULT 单元中。由于 MULT 单元已经实现了标准的乘法操作，因此 pq. mac 的功能增强仅需要增加一个多路选择器和 2 个加法操作。

新增加的抗量子密码指令包括 7 类，分别是 NTT 配置、NTT 操作、模计算模块、位翻转、PQ-MAC、Keccak 操作和中心二项分布采样。除了其中的 NTT 操作指令需要 83 个时钟周期外（执行功能需要 80 个周期，地址单元 3 个时钟周期延迟），其他都是单周期指令。

NTT 配置类指令有如下几种。

（1）算法及安全等级选择，如 Newhope 算法、Kyber 算法等；

（2）选择是 NTT 还是 INTT；

（3）选择是 NTT 的第一轮还是最后一轮。

考虑到抗量子密码芯片的抗物理攻击需求，慕尼黑工业大学研究团队进一步针对 Kyber 算法和 Saber 算法提出了基于 RISC-V 指令集、支持掩码防护的软硬件协同设计方案。严格来讲，这种方案属于基于指令集扩展和专用硬件相结合的混合设计。在该架构中，集成了两种类型的加速器。一种是松耦合的 NTT 加速器，与系统 AXI 总线连接。而其他加速模块，如 Keccak、安全加法器等则直接嵌入 RISC-V 处理器中，对应专用的扩展指令。

为了对线性操作进行加速，提出了一种同时适用于 2 的幂次方模数的通用 NTT 乘法器。对于非线性操作设计了掩码加速器，并开发了安全执行的 RISC-V 指令集。对 Kyber 算法和 Saber 算法的一阶安全防护实现开销分别是未防护实现的 4.48 倍和 2.6 倍。

通过对目前基于指令集扩展实现的抗量子密码芯片的分析，可以发现除了沿用一致定义的指令集格式外，扩展指令集的数量与类型严重依赖芯片支持的算法需求。同时，根据对技术指标优先级的不同，即使针对相同算法，设计方案也具有很强的差异性，呈现出显著的碎片化特征。因此如何提高这种解决方案对基础指令集的依赖性，提高通用性是在未来的抗量子密码迁移过程中需要重点考虑的问题，这个问题也是目前计算架构领域的一个热点方向。

## 3.3　面向特定算法的全定制硬件加速架构

全定制硬件实现的抗量子密码芯片，可以独立实现某个或者某一类数学困难问题密码算法的完整计算过程。这种实现方式的优点是能够充分挖掘算法潜在的优化空间，通过深度定制实现更高的计算性能和更低的能耗。但其缺点也十分明显，就是功能相对单一，灵活性不足。此外，当前阶段的定制硬件设计往往没有考虑对侧信道攻击的防护实现，难以应对后续可能出现的侧信道攻击威胁。这种实现方式主要适用于算法确定，同时对能量效率（计算速度或

者功耗)具有严格要求的应用场景。虽然基于格的密码算法的加速器可以支持主流的基于格的密码算法,具有一定的可配置性,但仍难以满足应用侧对多种不同数学困难问题支持的需求。接下来本节将分别介绍针对基于格、基于编码和基于哈希的全定制抗量子密码芯片研究进展。由于在 NIST 抗量子密码算法标准化进程中,基于多变量的 Rainbow 和基于超奇异同源的 SIKE 算法先后被破解,针对多变量与超奇异同源抗量子密码算法全定制硬件研究相对较少,本书不做详细讨论。

### 3.3.1 面向基于格的密码算法的全定制硬件设计

由于基于格的密码算法安全性研究相对成熟、计算效率与存储开销相对均衡、具有可同时实现密钥封装和数字签名的功能多样性,一直是抗量子密码算法标准化过程中各阶段的主流候选算法。因此,无论是针对基于格的密码算法中的核心计算模块(如多项式乘法、采样等),还是完整算法的专用硬件设计研究都获得了更多的持续关注。本节将重点讨论经过流片验证、实现完整算法功能的代表性工作,包括针对 Saber、Kyber 和 Dilihtium 算法的 ASIC 设计与可配置的基于格的密码处理器研究。

Saber 算法是由比利时鲁汶大学 COSIC 团队设计、基于 MLWR 的密钥封装算法。虽然该算法并没有被选为标准,但由于其采用 2 的幂次方模数对于模运算的硬件实现更加友好,因而获得了比较多的硬件设计研究[17-20]。著者团队针对 Saber 算法提出了支持其 3 种安全强度的可配置计算架构[17]。普渡大学的研究人员联合算法设计团队一起针对 Saber 算法开展了深入的定制硬件设计[19,20],并取得了最高的能量效率。其主要创新点包括乘法及实现、存储管理以及与协处理器的接口。从计算的角度而言,维度为 256 的多项式乘法是 Saber 算法中计算最为复杂的部分。在这种类型的工作中,多项式乘法的模数选择是不适合 NTT 的类型,所以这些工作还讨论如何设计非 NTT 类型的高效多项式乘法结构,可能会用到 Karatsuba 算法、TOOM-COOK 算法降低硬件设计的复杂度。同时,通过微指令控制实现功能重构支持密钥产生、封装和解封装等不同功能。在实现过程中,为了进一步降低资源开销,相应的系数存储与乘法器均根据算法涉及的数据宽度进行定制,例如多项式系数为 13 位,采样的噪声宽度为 4 位。因此,深度定制的设计虽然针对 Saber 算法计算获得最高效的能效提升,但无灵活支持其他算法。

除此之外,文献[21]中还提出了同时支持 Kyber 算法和 Dilithium 算法的统一加速硬件。

### 3.3.2 面向基于编码的密码算法的全定制硬件设计

在基于编码的抗量子密码算法中,Classic McEliece、HQC 和 BIKE 等算法目前是 NIST 标准化竞赛中的第四轮候选算法。在这 3 个算法中,Classic McEliece 算法基于 Goppa 码,安全性分析相对充分,但其主要问题是公钥尺寸比较大进而造成密钥生成较慢,这也限制了 Classic McEliece 算法的应用场景。BIKE 和 HQC 算法都是基于 QC-MDPC 编码。二者相比,BIKE 算法的公钥和密文尺寸更小,HQC 算法的密钥生成及解封装速度更快。表 3-4 给出了这 3 个算法在最低安全等级的存储开销对比和计算时间对比。目前已经有针对 Classic McEliece 算法的专用硬件设计方案。目前大部分的硬件优化工作都是针对时间占比更高的高斯消元模块进行优化设计。

表 3-4　基于编码的抗量子密码算法的存储与计算开销对比(最低安全等级)

| 算　法 | 存储开销/字节 | | | 计算开销/kcycles | | |
|---|---|---|---|---|---|---|
| | 公钥 | 私钥 | 密文 | 密钥生成 | 封装 | 解封装 |
| Classic McEliece | 261120 | 6492 | 96 | 56706 | 36 | 127 |
| HQC | 2249 | 56 | 4497 | 87 | 204 | 362 |
| BIKE | 1541 | 281 | 1573 | 589 | 97 | 1135 |

目前大部分 Classic McEliece 算法的硬件优化工作都是针对时间占比更高的高斯消元模块进行优化。针对 Classic McEliece 的密钥生成时间开销过大的问题,目前已提出了高吞吐的密钥生成加速器 McKeycutter[22]。图 3-7 所示的是 McKeycutter 的整体架构。整个密钥生成加速器可分为有系统控制调度的两个并行部分:A 部分和 B 部分。在计算 B 部分执行 GF(2)消除任务时,A 部分的任务能够重新启动从而实现与 B 部分的并行执行,进而节省周期开销。一般有如下两种情况,第一,如果矩阵被检测为奇异矩阵,提前启动 A 部分可以减少重试的周期;第二,如果稍后检测到矩阵是可逆的,则提前启动可以减少下一个密钥生成的 A 部分周期。McKeycutter 有两种工作模式。第一种模式(批处理模式)执行多个密钥生成作业,最大限度地提高了隐藏的效果。第二种模式(单一模式)是只执行一个密钥生成。在每个密钥生成中,并行执行仍然可以减少延迟,因为失败尝试后的 A 部分是隐藏的。

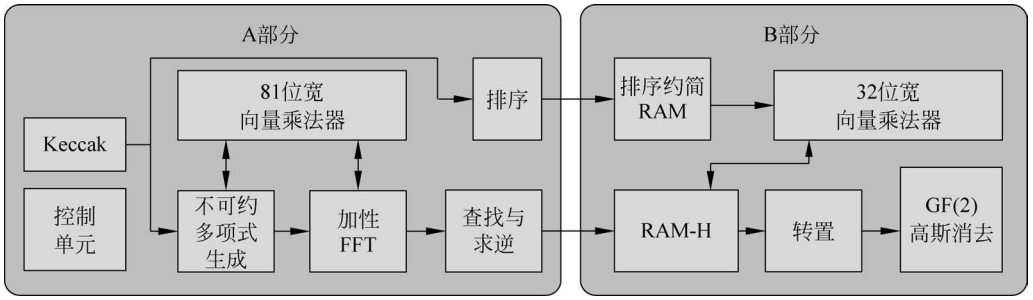


图 3-7　高吞吐密钥生成加速器 McKeycutter 架构

从 Xilinx Ultrascale+ FPGA 平台的实现结果看,CHES 2022[20]是 McKeycutter 消耗的 BRAM 的 1.7～2.7 倍。McKeycutter 在降低 BRAM 使用的同时,也将密钥生成的计算速度提升了 4 倍以上。

除了针对 Classic McEliece 算法的专用硬件设计外,也有一些针对 HQC 和 BIKE 算法的硬件优化设计成果[23-26]。耶鲁大学的研究[23]第一次提出了专门针对 HQC 算法进行全定制人工设计优化,并支持密钥产生、封装和解封装的硬件设计。首先在原有 Keccak 模块基础上改善了 SHAKE256 设计,使得计算速率提升 2 倍,同时改善了面积延时积。

来自波鸿鲁尔大学的研究人员针对 BIKE 算法的硬件加速设计开展了长期研究,分别提出了 Folding BIKE[24]和 Racing BIKE[25]两项成果。在 Folding BIKE 中通过高效多项式求逆、BGF 解码器实现等实现了面向 BIKE 参数集的可扩展设计。Folding BIKE 处理延迟密钥产生用时 2.69ms,密钥封装用时 0.1ms,解封装用时 1.89ms。在基于编码的多项式乘法器里,Folding BIKE 利用 QC-MDPC 编码中的多项式稀疏性对乘法进行优化,还设计了一个密钥产生模块。Racing BIKE 主要对积稀疏多项式乘法和多项式求逆两个计算模块进行了优化,提升了密钥产生速度。除了算术部件的优化外,这项工作提出了可在不同密码原语间可资源服用的统一硬件设计,可以在 1672μs、132μs 和 1892μs 内完成密钥产生、密钥封装和密钥解封装工作。

### 3.3.3 面向基于哈希算法的全定制硬件设计

基于哈希的抗量子密码算法安全性建立在底层哈希函数的抗碰撞性。理论上讲，只要其使用的哈希函数是安全的，这类算法就是抗量子计算机攻击的。IETF 和 NIST 先后将基于有状态的哈希算法 XMSS 和 LMS 确定为数字签名算法标准，并推荐优先应用于具有长生命周期且固件升级不便的设备。在目前 NIST 的抗量子密码标准化中，基于无状态哈希的“SPHINCS+”算法也已经被选为数字签名标准，并已发布了标准草案。相比于其他类型算法，基于哈希的抗量子密码算法的主要问题是签名过程较慢、签名尺寸比较大。有状态和无状态哈希签名算法的主要区别在于签名过程中对状态信息的处理方式。有状态的签名算法在进行签名和验证操作时需要维护和更新一个状态信息。这个状态通常是由于一次性签名算法演变而来，每次签名操作都会产生一个一次性的认证密钥。这些密钥在使用后就会失效，因此需要在系统中记录哪些密钥已经被使用过，以防止重复使用。这种状态信息的维护可能会增加算法的复杂性和资源消耗。无状态签名算法在签名和验证过程中不需要维护任何状态信息，可以在没有任何先前信息的情况下独立地对每个消息进行签名和验证。因此，无状态签名算法由于其简单性和高效性，在实际应用中更为常见。针对哈希的抗量子密码加速芯片[27-29]大多基于更为成熟的 XMSS 和 LMS 算法，最近芬兰 Tampere 大学的研究人员针对 NIST 将标准化的“SPHINCS+”算法也提出了高效定制硬件设计 SLotH[30]。

基于哈希算法 XMSS 的设计主要针对计算复杂度最高的叶子节点产生操作进行流水线化硬件加速。图 3-8 所示的是该设计的系统架构，除了系统的有限状态机控制器与存储外，还包括 3 个功能模块，分别是 WOTS 模块、L-Tree 模块和 SHA256XMSS 模块。SHA256XMSS 通过四级流水线接收实现 SHA-256 计算。

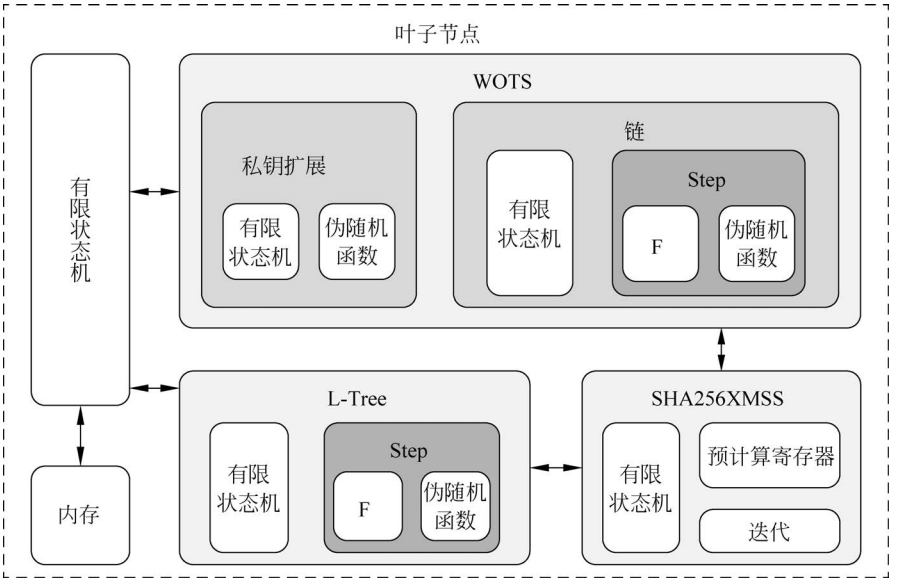


图 3-8 基于哈希算法 XMSS 加速器的硬件架构

该设计最终在 28nm 工艺下对采用和未采用流水操作的两种设计都进行了流片验证。采用流水设计的实现部分面积相比于未采用流水的方案面积开销增加了 44%，但由于关键路径缩短，芯片的最高运行频率由 823MHz 提高到了 1011MHz。

SLotH 通过对填充格式和迭代哈希计算的优化，在首次实现“SPHINCS+”算法 12 个参数集支持的同时，与相关工作相比将验签操作加速 5 倍以上。

总而言之，全定制硬件设计针对特定算法或者具有相似计算属性的某类算法，展开从数据位宽、存储模式、计算电路功能的深度定制，从而实现了最高的能量效率。但其缺点同样十分突出，即以牺牲功能灵活性为代价。这种实现方式对于算法结构仍在持续动态演进，对面临多种标准体系的抗量子密码应用而言并非是理想选择。

## 3.4　粗粒度可重构抗量子密码芯片架构

粗粒度可重构计算架构可同时兼顾密码芯片对功能灵活性、能量效率和物理安全性方面的需求。根据作者团队在可重构对称密码处理器方面的研究成果[31-33]，该架构可以通过空间数据流并行实现性能提升，配置流驱动实现功能动态改变，时空域随机重构提高芯片对侧信道攻击的防御能力。如图 3-9 所示，粗粒度可重构密码芯片由编译系统和硬件电路两部分组成。其中硬件电路由功能可动态配置的处理单元阵列组成。硬件电路的规模和互连方式决定了芯片的理论峰值性能。编译系统则实现由 C/C++ 等高级语言编程的算法程序向硬件电路的映射，直接决定了算法的运行时性能和硬件利用。本节主要对可重构抗量子攻击密码芯片的硬件架构进行展开论述，关于电路实现与编译系统的内容将分别在第 4 章和第 5 章展开。

为了在保证能量效率的前提下，进一步提高抗量子密码芯片的功能灵活性，作者研究团队研发了两款可重构密码芯片架构。这两款芯片 RePQC(Reconfigurable Post-Quantum Crypto-processor)和 PQPU(Public-Quantum Processing Unit)分别发表在国际固态电路会议 ISSCC 2022[31]和 ISSCC 2024[33]上。这两款芯片遵循共同的设计思想，只是根据 NIST 抗量子密码标准化活动不同阶段的算法遴选情况不同在算法支持上有所区别。另外在具体的数据通路与配置机制上进行迭代优化。RePQC 主要支持 NIST 在 2020 年 7 月公布的最终算法。PQPU 主要支持 NIST 在 2022 年 7 月公布的最终标准算法与仍然安全的第四轮候选算法。二者支持的具体算法类型如表 3-5 所示。

表 3-5　RePQC 和 PQPU 支持的抗量子密码算法

| 芯片架构 | 数学困难问题 | 密钥封装 KEM | 数字签名 DS |
|---|---|---|---|
| RePQC | 格 | Kyber、Saber、NTRU | Dilithium |
| | 编码 | Classic McEliece | — |
| | 多变量 | — | Rainbow |
| PQPU | 格 | Kyber、LAC | Dilithium、Falcon、Aigis |
| | 哈希 | — | SPHINCS+ |
| | 编码 | Classic McEliece、HQC、BIKE | — |

这两款可重构抗量子密码芯片的设计目标可满足以下特性。

(1) 高性能计算：满足服务器端对于公钥密码计算的高吞吐需求。

(2) 高功能灵活性：支持主流抗量子密码算法和其中所有的密码原语(包括密钥封装、公钥加密和数字签名)，兼容算法的多个安全等级。

(3) 高能量效率：在追求高计算性能的前提下，降低芯片功耗开销，提升能量效率。

受传统公钥处理器[30,31]设计启发，在对所支持的抗量子密码算法计算流程和数据模式进

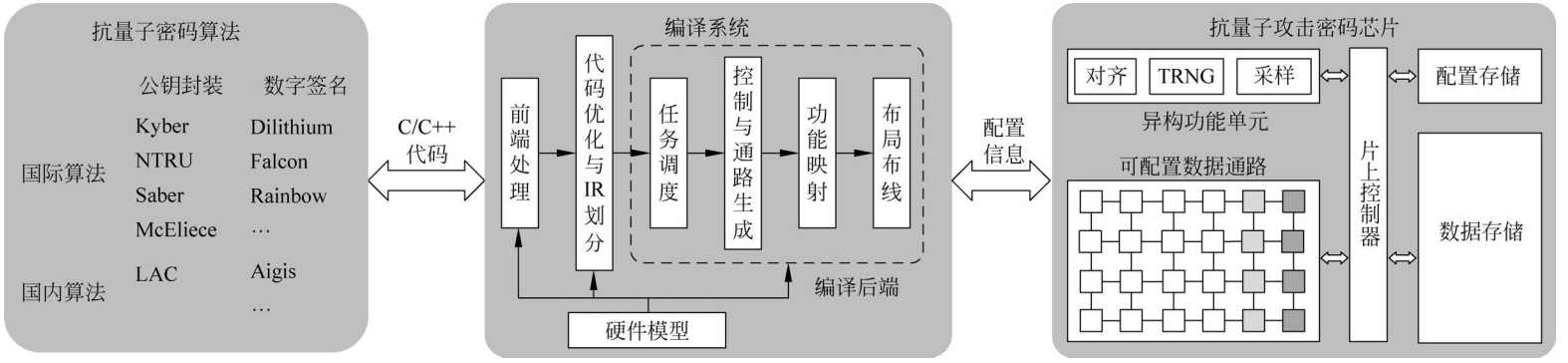


图 3-9　可重构抗量子密码芯片的通用模型

行分析的基础上，提出了一种兼顾能量效率和灵活性的计算框架。该框架通过将不同层次计算负载映射到相应粒度的功能单元上实现高能效支持。如表 3-6 所示，从算法级计算、任务级计算到系数级计算，计算粒度逐渐变小，功能灵活性逐渐提高。

表 3-6　层次化计算框架

| 层次化定义 | 对 应 计 算 |
| --- | --- |
| 算法级计算 | 完整的密码原语，如 Kyber、Classic McEliece、Dilihtium 等算法 |
| 任务级计算 | 矩阵/多项式级计算，如多项式乘法、矩阵操作、快速 NTT、多项式求逆等 |
| 系数级计算 | 针对矩阵/多项式系数的计算，如模乘、模加、蝶形运算等 |

在上述设计思想指导下，先后针对 NIST 抗量子密码标准化不同阶段的候选算法研制的可重构密码芯片 RePQC 和 PQPU 两款芯片。

### 3.4.1　可重构抗量子密码芯片 RePQC

RePQC 芯片是针对 NIST 抗量子密码算法标准化的第二轮选出的算法研制。在 2020 年 7 月，NIST 公布了通过第二轮评估的 15 项算法提案。这 15 项算法被分为两类，其中 7 项算法被确定为最终算法，其余 8 项算法被归类为备选算法。这两类算法的差别在于，最终算法在安全性评估方面较为深入，并且在灵活性和计算效率方面相对更具优势。而备选算法则仍需要更深入的评估分析。由于最终算法中的 Falcon 算法涉及大量浮点操作，考虑到集成浮点计算阵列对芯片设计复杂度与硬件利用率的不利影响，RePQC 支持除 Falcon 算法外的所有最终算法。

RePQC 在芯片架构方面的优化技术包括：①提出了一种混合处理单元阵列结构，通过功能重构实现算法底层的逻辑与算术运算；②提出了一种高效的任务调度机制，实现任务级算子与混合处理单元阵列间的高效协同计算；③挖掘算法潜在的并行空间，通过提高硬件利用率进一步提升计算性能。

#### 1. RePQC 芯片架构

如图 3-10 所示，RePQC 主要由四大功能部分组成：数据生成引擎（Data Generation Engine，DGE）、数据存储系统（Data Storage System，DSS）、任务级调度器（Task Level Scheduler，TLS）和混合处理单元阵列（Hybrid Processing Element Array，HPEA）。其中，DSS 是整个密码计算硬件的数据存储模块，主要完成与总线的数据交互与中间计算数据的片上存储。DGE 主要通过内部的哈希函数与采样器逻辑产生不同算法需要的随机数，并将生成结果存储在 DSS 中。TLS 对应不同算法的任务级计算操作，可独立完成某些特定计算，也可以与 HPEA 协同执行计算。HPEA 包括逻辑计算（Logical Element，LE）阵列和算术计算（Arithmetic Element，AE）阵列，分别用来实现不同逻辑与算术计算功能。

HPEA 支持包括算术和逻辑计算在内的多种向量化运算。HPEA 的 AE 阵列和 LE 阵列分别由 32 个算术计算单元 AE 和 16 个逻辑计算单元 LE 组成。细粒度的系数级算子保证 RePQC 对不同抗量子算法的兼容性，并对后续持续改进的算法提供支持。该工作没有采取[6,32]使用的 Barrett 归约方法，而是使用 Montgomery 归约方法实现模块化运算。这是因为 Montgomery 方法更适用于二项域中的模规约运算。该工作实现了混合约简，包括整数模运算和二项域模运算。并且对 Montgomery 方法的硬件实现采取了优化措施，通过将 Montgomery 模乘过程的最后累加计算减少到一半宽度来优化 Montgomery 缩减方法，从而

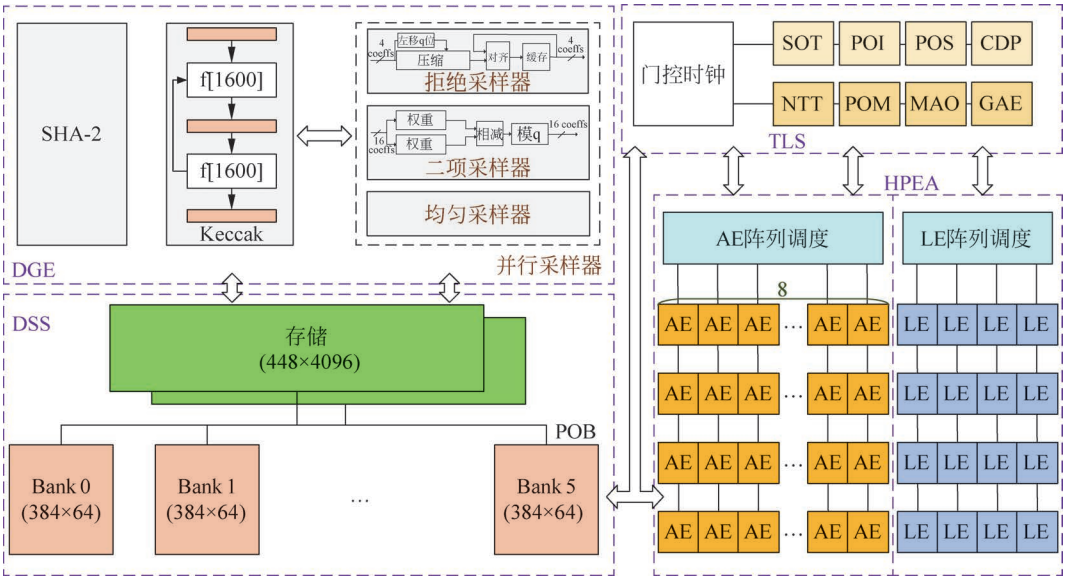


图 3-10 RePQC 的系统架构

减少了资源使用。此外，多个可控制加法器(CA)、可控制乘法器(CM)和多路复用器为 AE 提供了各种工作模式。CT-BFU 和 GS-BFU 模式下的 AE 支持 NTT 计算中的蝶形单元计算操作。在 Tri-Mul 模式下，AE 通过控制多路复用器执行 3 个整数乘法。Mul-Add 模式和 Add-Mul 模式下的 AE 支持线性算术运算。算术运算包括素数域或二进制域中的加法、减法、乘法、乘法和加法以及可配置的模数或模多项式。为了在必要时节省功耗，AE 中的所有寄存器都可以通过时钟门控以半字(14 位)模式配置，或者在 AE 空闲时关闭。

在任务级层次，基于算法-硬件协同设计实现了一系列针对抗量子密码算法中粗粒度功能的任务级算子。这些任务级算子并非是所有算法都需要的，呈现出一定的专用性，但是对算法整体性能具有重大影响。这些任务，如多项式乘法(Polynomial Multiplication，POM)、排序(Sorting，SOT)、多项式求逆(Polynomial Inversion，POI)和矩阵运算是在系数水平上与处理阵列合作或独立完成的。

在算法级别实现针对算法的配置信息优化，包括并行调度和高效计算转换等方法，从而充分利用任务级和系数级的硬件资源，实现更高的计算吞吐率和能量效率。图 3-11 展示了针对 Kyber 算法的加密功能和 Dilithium 算法的签名功能的调度机制优化。在算法分析基础上，通过将无数据相关算子并行化、计算与数据访问(包括数据生成、数据读取等操作)并行化来降低抗量子密码算法的计算延迟。

图 3-12 展示了 RePQC 的控制逻辑。所有计算模块和运算符(TLS、AE 或 LE)都由系统控制单元协调。各计算模块的执行由来自控制器的启动信号触发。此外，算子运行状态的相应标志由启动信号设置，并在执行完成时由对应算子释放。通过读取算子状态寄存器的值，控制器能够判断哪些算子正忙，并依此进行配置信息加载和任务调度。如果相邻算子执行彼此之间存在数据依赖性，则后一次执行将在前一次执行的数据流被释放后启动。如果有两次不相关的执行，后一次执行将在前一次执行开始后立即开始。用户可以通过选择何时或是否等待指定的有效标签来调整并行计划逻辑。此外，用户还可以离线更改命令中的参数字段、命令顺序和依赖性，以满足一定程度的灵活性。任务级运算符(NTT、POM 等)或 AE 调度模块

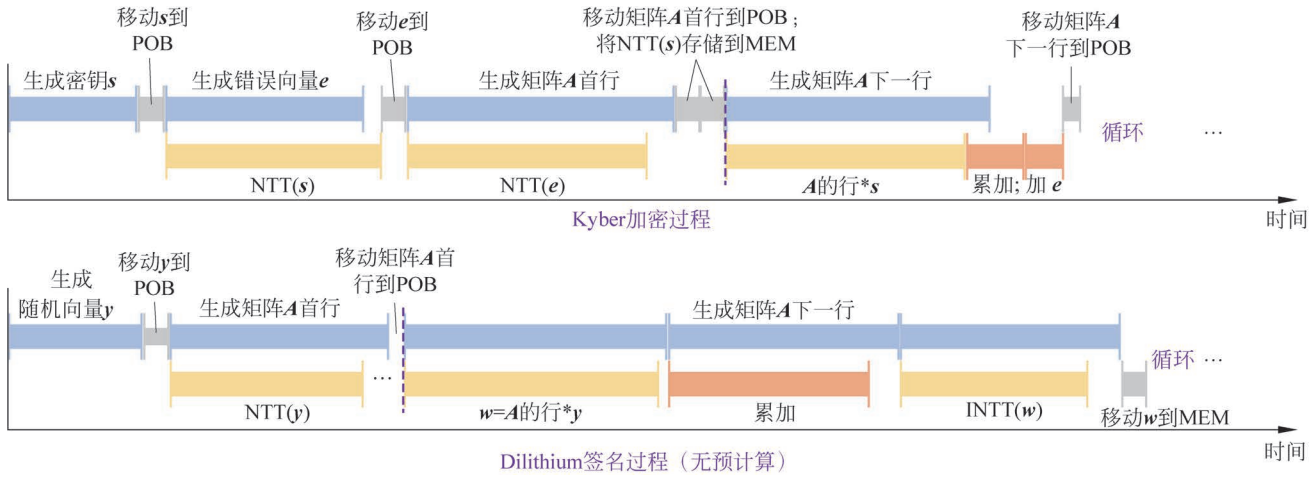


图 3-11　RePQC 在执行 Kyber 和 Dilithium 算法时的调度机制

可以使用不同的参数(如多项式的次数和数量)进行动态配置。TLS 的配置参数包括尺寸(Dims)、计算模式(mode)和起始地址(Addr)等。

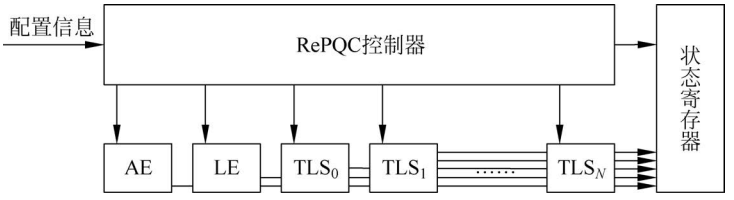


图 3-12 RePQC 的控制逻辑

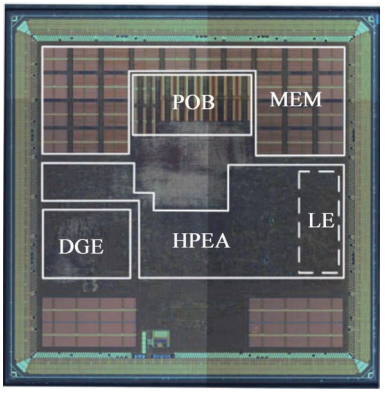


图 3-13 RePQC 的管芯照片

### 2. 芯片实现与性能评估

本节将对 RePQC 的芯片验证结果进行详细介绍,并与当时相关工作进行性能对比。

RePQC 在 TSMC 28nm HPC+工艺下完成芯片实现。如图 3-13 所示,整个管芯的面积为 6.25mm$^2$,其中 RePQC 的抗量子密码加密核面积为 3.6mm$^2$。如表 3-7 所示,在 0.9V 电压和 500MHz 工作频率的工作条件下,该芯片运行不同算法的功耗分布为 39 ~ 368mW。RePQC 的等效逻辑规模为 190 万门(等效二输入与非门),存储开销为 448KB。

RePQC 的技术性能指标如表 3-7 所示。

表 3-7 RePQC 的技术性能指标

| 芯片规格参数 | |
|---|---|
| 工艺 | 28nm HPC CMOS |
| 源电压 | 0.9V |
| 芯片尺寸 | 2.5mm×2.5mm |
| **抗量子密码加密核** | |
| 面积 | 3.6mm$^2$ |
| SRAM | 448KB |
| 逻辑门 | 190 万门(等效二输入与非门) |
| 哈希函数 | SHA-3/SHA-2 |
| 伪随机数发生器 | SHAKE-128/256 |
| 功耗 | 39~368mW |

表 3-8 列出了 RePQC 在运行不同算法时的计算性能,包括不同原语的计算延迟(时钟周期数)、吞吐率和功耗。RePQC 支持 Kyber、Saber、Dilithium 和 NTRU 算法所有安全等级的所有功能。由于 Classic McEliece 算法和 Rainbow 算法的密钥尺寸开销过大,RePQC 仅支持 McEliece 算法的密钥封装/解封装功能和 Rainbow 算法的签名/验签功能。表 3-8 中列出的吞吐率和功耗分别是 McEliece 密钥封装和 Rainbow 算法签名的性能。

以 Kyber 和 Dilithium 算法为基准对 ReQPC 与相关工作进行技术对比。密歇根大学在 CICC 2018 发表的 LEIA[10]针对基于格的抗量子密码算法中的多项式乘法进行硬件加速优化。通过表 3-9 中的对比可以看到,针对 NTT 计算,RePQC 的计算延迟相比该工作降低 80%。与 MIT 的 Sapphire[34]和复旦大学的 VPQC[35]工作相比,RePQC 除了支持基于格的

抗量子密码算法外，还支持包括基于编码和多变量的抗量子密码算法。其次无论是计算吞吐率还是能量效率，在经过归一化对比后均取得显著改善。从表 3-9 可以看到，对于 Kyber-512 算法而言，RePQC 的吞吐率分别是 Sapphire 和 VPQC 的 23 倍和 3.8 倍，能量效率则分别改善了 74.4%和 73.4%。对于 Dilithium 算法，RePQC 的吞吐率是 Sapphire 的 132 倍，能量效率改善了 52.7%。

表 3-8　不同算法在 RePQC 上的性能指标

| 算　　法 | 密钥生成/Cycle | 密钥封装/签名/Cycle | 密钥解封/验签/Cycle | 吞吐率/OPS | 功耗/mW |
|---|---|---|---|---|---|
| Kyber-512 | 2178 | 3519 | 4736 | 47925 | 163 |
| Kyber-768 | 3505 | 4914 | 6359 | 33834 | 193 |
| Kyber-1024 | 5135 | 6621 | 8177 | 25084 | 218 |
| LightSaber | 2313 | 2889 | 3783 | 55648 | 139 |
| Saber | 3917 | 4640 | 5669 | 35147 | 172 |
| FireSaber | 5873 | 6696 | 7924 | 24399 | 195 |
| Dilithium-Ⅱ | 6048 | 31215.5 | 6113 | 11527 | 237 |
| Dilithium-Ⅲ | 10889 | 51117.8 | 9118 | 7030 | 278 |
| Dilithium-Ⅴ | 14228 | 54236.8 | 13244 | 6119 | 304 |
| NTRU-509 | 22746 | 4112 | 5486 | 15459 | 308 |
| NTRU-677 | 38496 | 6446 | 8080 | 9430 | 291 |
| NTRU-821 | 54424 | 7009 | 9917 | 7008 | 369 |
| McEliece | — | 6314 | 108082 | 79189 | 103 |
| Rainbow-Ⅰ | — | 18677 | 20345 | 26771 | 54 |

表 3-9　RePQC 与相关工作技术指标对比

| 对比项 | CICC'18[10] | ISSCC'19[34] | TCAS-I'20[35] | RePQC |
|---|---|---|---|---|
| 工艺 | 40nm | 40nm | 28nm | **28nm** |
| 频率/MHz | 300 | 12-72 | 300 | **500** |
| 面积/$mm^2$ | 2.05 | 0.28 | — | **3.6** |
| 电压/V | 0.9 | 0.68～1.1 | 0.9 | **0.9** |
| 功耗/mW | 140 | 7～10 | ～30 | **39～368** |
| 数学困难问题 | — | 基于格 | 基于格 | **基于格、基于编码、基于多变量** |
| NIST 算法 | — | Kyber、Dilithium | Kyber | **Kyber、Saber、NTRU、Dilithium、McEliece、Rainbow** |
| 模数 $q$ | ＜32 位素数 | ＜24 位素数 | ＜16 位素数 | **＜24 位（素数/2 的次幂/不可约多项式）** |
| 功能 | NTT | | | |
| 周期数 | 160 | 1288 | 45 | **32** |
| 算法 | Kyber 算法 | | | |
| 吞吐率/OPS | — | 207 | 2077 | **47925** |
| 能量效率/(μJ/Op) | — | 26.6 | 12.8 | **3.4** |
| 归一能量效率/(μJ/Op) | — | 13.3 | 12.8 | **3.4** |
| 算法 | Dilithium 算法 | | | |
| 吞吐率/OPS | — | 87 | — | **11527** |
| 能量效率/(μJ/Op) | — | 87.2 | — | **20.6** |
| 归一能量效率/(μJ/Op) | — | 43.6 | — | **20.6** |

表 3-10 列出了在执行不同算法时，RePQC 中在执行算法计算时占用时间最多的 3 个功能模块。可以看到，除了 Rainbow 算法外，混合计算阵列均是其他算法中的核心运算模块。

表 3-10 不同算法运行过程中主要功能模块的利用率

| 算　法 | Saber | Kyber768 | NTRU677 | Dilithium-Ⅲ | McEliece | Rainbow |
|---|---|---|---|---|---|---|
| 占用时间最多 | POM | DGE | POM | DGE | AE | MAO |
| 占用时间次之 | DGE | AE | POI | LE | LE | GAE |
| 占用时间第三 | LE | NTT | LE | NTT | CDP | CDP |

### 3.4.2 可重构抗量子密码芯片 PQPU

2022 年 7 月，NIST 公布了第三轮抗量子密码算法评选结果。其中，Kyber、Dilithium、Falcon 和“SPHINCS+”4 项算法提案被选为标准草案，正式进入标准化流程。另外，BIKE、HQC、Classic McEliece、SIKE 4 项算法提案被选为候选算法进入第四轮评估（SIKE 算法不久后即被破解）。在抗量子安全性不变的前提下，NIST 会在第四轮评估后最终选择一项算法推荐为后续标准。PQPU 的主要设计目标是支持包括初始标准算法和第四轮有效候选算法在内的 7 项算法，同时支持全国密码算法竞赛的全部一等奖获奖算法 LAC 和 Aigis。与 RePQC 工作相比，除了支持更多的算法类型外，PQPU 在芯片架构上进行了更多尝试突破。在基于层次化计算架构基础上，针对不同算法的计算流程进行分析，从而提出了面向多种抗量子密码算法功能分簇的架构与调度机制，具体包括：①基于任务聚类可扩展并行计算架构；②基于区域的任务动态更新机制；③高能效任务算子设计。

#### 1. PQPU 芯片架构

如图 3-14 所示，依据资源复用、流水操作的设计原则，将抗量子密码算法中常见的计算模式（功能）包括输入/输出（Input/Output，I/O）处理、随机数生成、特定分布采样、数据格式转换、计算功能实现、格式化与逻辑操作等几类函数映射成不同的硬件簇，如哈希计算簇、采样逻辑簇、格式化逻辑簇、计算功能簇以及数据交互簇等。

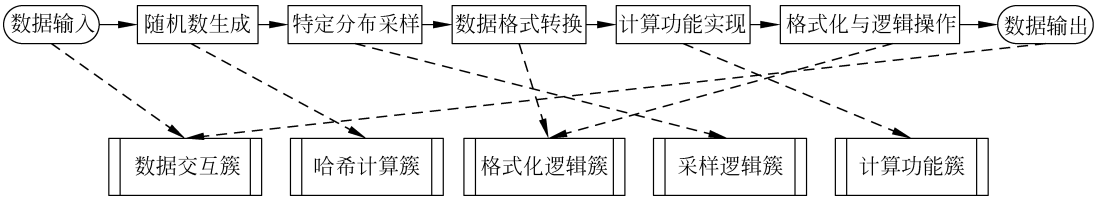


图 3-14 抗量子密码算法常用计算模式与硬件电路簇的映射关系

图 3-15 所示的是 PQPU 的数据通路，包括任务算子簇（Task Operator Cluster，TOC）与 SRAM 组成的数据存储部分。TOC 由上述 5 个任务簇组成。各任务簇之间以流水方式执行。计算功能簇中的算术单元 AE 阵列、浮点单元（Floating Element，FE）阵列和算术缓存可分别与算子协作完成计算规模更大的算术、浮点/复数运算及相关数据访问。每个任务簇的算子均复用同一个与数据缓存的交互接口。

在哈希簇中集成了 Keccak、AES 和 Chacha20 三种伪随机数发生器。其中，Keccak 模块中集成了带有自动填充与对齐逻辑的 Keccak 核。Keccak 核可以在 1 个时钟周期内执行 2 个完整的 f-函数。计算过程中的 Keccak 模式、并行度和 I/O 长度均可通过配置信息进行定义。

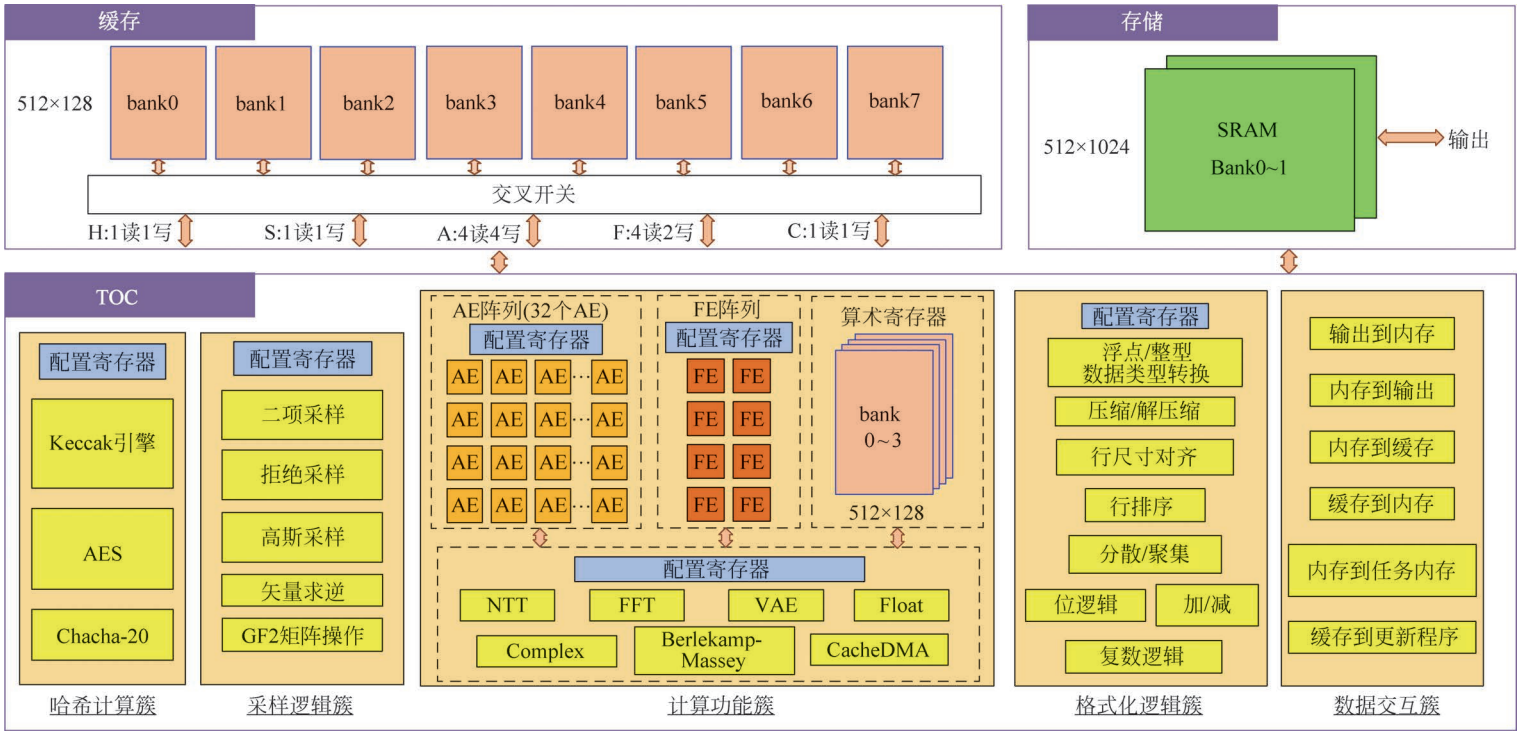


图 3-15　PQPU 架构图结构

在抗量子密码不同功能模块之间经常会用到位流-字节之间的格式转换。在格式与逻辑簇中集成了一个基于 I/O 使能与反馈机制的对齐器，通过自动删除或补充 0 值实现任意位宽向量向另外任意一种位宽的转换。在采样逻辑簇中，通过可配置的比较器、分类网络和对齐器来实现任意位宽的拒绝采样。

计算功能簇仍然沿用任务级-系数级的层次化执行架构。素数域和二项域上的系数级运算在 AE 阵列上实现，而复数域上的系数级运算在 FE 阵列上实现。通过 AE 阵列和 FE 阵列实现粗粒度任务或向量化计算中的线性操作。

如图 3-16 所示，除了上述的数据计算通路外，PQPU 的任务通路（Task Path，TP）由任务存储、任务提取、任务更新和任务调度 4 个模块组成，分别实现抗量子密码算法的任务信息存储、任务解码、任务生成与动态更新、任务依赖性查验与任务发射。任务发射前会在任务调度器中维护好相应的寄存器和状态。在计算完成后，计算模块也会发射相应的结束信号给任务通路，进行进一步的状态维护。

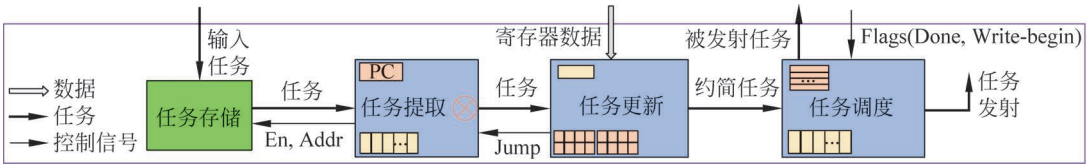


图 3-16　PQPU 的控制通路组成

对于任务更新器而言，任务会存到一个任务缓存 buffer 中。如图 3-17 所示，寄存器被划分成了静态和动态两个集合，分别对应更新操作是数据无关的还是数据有关的。对于密码算法中的循环操作，循环体中的任务需要在任务更新器中对源地址、目的地址等字段进行相应更新。是否要更新、哪些字段需要更新都是被更新前缀字段中的更新有效位使能的。除此之外，也可以更新类型的任务对更新器中的寄存器数值进行相应维护。例如，对于循环变量的递增而言，循环变量的数值就会维护在寄存器中，更新类型的任务就会对寄存器中的数值进行加减乘运算，在任务运行的同时来维护好寄存器的值。对于跳转等任务也区分了是静态还是动态，以便对不同类型任务进行分别维护和管理。对于动态任务，因为需要在等待数据 BUF 向更新器传递值过程中避免读取到过时数据，对于新到来的动态任务而言，需要等待

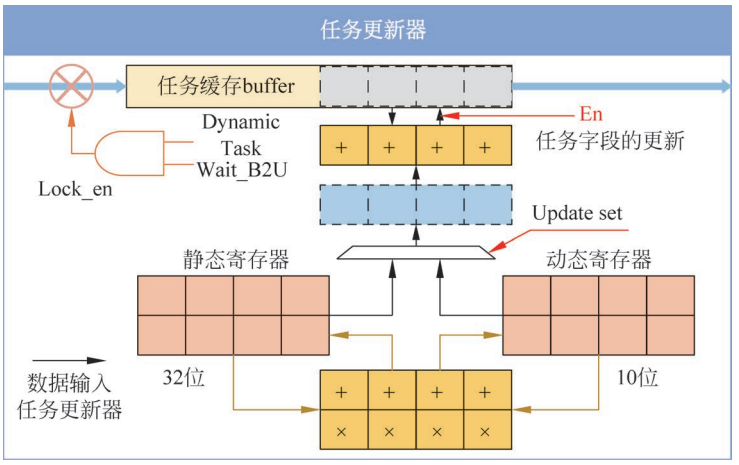


图 3-17　PQPU 的任务更新器

BUF2UPDATER(B2U)任务执行完毕，才能保护算法功能的正确实现。在任务更新器模块中，有一个锁机制可以保障语义的正确，如果目前任务缓存 buffer 中的任务是动态的且正处于等待 B2U 任务执行完毕的过程中，那么会启动相应的锁机制限制任务的进一步输入，使任务更新器处于静止状态直至 B2U 任务执行完毕。将任务划分成动态任务和静态任务，也避免了本没有依赖的静态任务所谓的启动锁机制可能导致更多的时间开销。

任务调度器负责的是更新任务的调度和相关性的维护。任务调度器包含 3 个模块。

(1) Issue FIFO。

(2) 属于发射任务的 FIFO。

(3) 区域依赖性检查，对于正在执行的任务和即将发射的任务进行区域性依赖检查以及相应的结构依赖性检查。如果有依赖，Issue FIFO 会被停滞，来等待正在执行的任务；如果没有依赖，会向相应的计算簇和计算模块发射，并且在相应的状态寄存器中维护好相应的项目状态，为后续待发射任务提供相应信息。

任务之间的相关性主要取决于两方面；一是数据依赖性，即后续任务的输入数据是否依赖之前任务的输出；另一个是结构依赖性，指两个任务是否需占用同一硬件模块，导致任务并行执行期间产生硬件资源争夺。因此在判断依赖时，也要分成两个依赖。在数据依赖上，由于 PQPU 架构的特点，每个运行任务都声明了两个源地址区域和一个目的地址区域，判断依赖时需要判别区域之间是否有交叠。

### 2. 芯片实现与性能评估

PQPU 在 TSMC 28nm HPC 工艺下完成芯片验证。管芯照片(包括金属层 M5 和顶层金属)如图 3-18 所示，芯片整体面积为 7.26mm$^2$，其中 PQPU 的抗量子密码计算部分面积为 3.2mm$^2$。芯片采用 FCBGA 封装。表 3-11 给出了 PQPU 的具体技术性能指标。PQPU 在 0.9V 工作电压、500MHz 工作频率下，功耗范围为 91～420mW。PQPU 可同时支持素数域、二项域和复数域的密码计算函数。

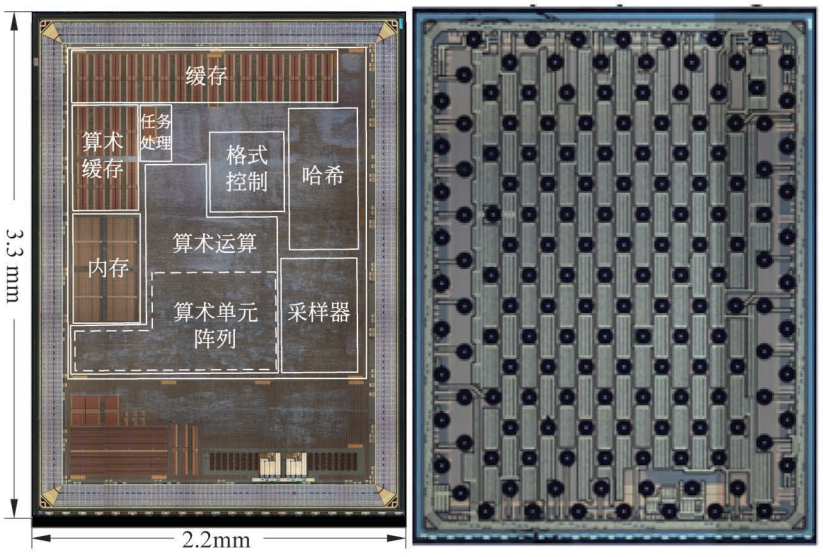


图 3-18　管芯照片和芯片参数

表 3-11 PQPU 的技术参数

| | |
|---|---|
| **芯片核心面积/$m^2$** | 3.2 |
| **存储容量/KB** | 228.5 |
| **逻辑规模(等效二输入与非门)** | $2.1\times10^6$ |
| **有限域类型** | 素数域/二项域/复数域 |
| **功耗/mW** | 91～420 |
| **工作电压/V** | 0.7～1.1 |
| **工作频率/MHz** | 275～750 |

从表 3-12 可以看到,与近期相关工作对比,PQPU 是目前公开发表的首款可同时支持格、编码和哈希等数学难题抗量子密码算法的芯片。同时,PQPU 针对 Kyber、Dilithium 和 Falcon 等算法均实现了最优的能量延迟积。

表 3-12 PQPU 与相关工作的性能对比

| 技术指标 | Sapphire[34] | JSSC'23[19] | RePQC[31] | ESSCIRC'22[36] | PQPU |
|---|---|---|---|---|---|
| 工艺 | 40nm | 65nm | 28nm HPC | 28nm LP | 28nm HPC |
| 频率/MHz | 12～72 | 40～160 | 500 | 35～190 | 275～750 |
| 面积/$mm^2$ | 0.28 | 0.158 | 3.6 | 0.18 | 3.2 |
| 电压/V | 0.67～1.1 | 0.7～1.1 | 0.9 | 0.65～1.35 | 0.7～1.1 |
| 功耗/mW | 7～10 | 0.3～10 | 39～368 | $10^+$ | 91～420 |
| 数学难题 | 格 | 格 | 格、编码 | 格 | 格、编码、哈希 |
| NIST 算法 | Kyber、Dilithium | Saber | Kyber、Dilithium、Classic McEliece | Kyber、Dilithium | Kyber、Dilithium、Falcon、"SPHINCS+"、BIKE、Classic McEliece、HQC |
| 模数 $q$ | 特定模数 $q$(24 位) | $q$：13 位 | All (24 位) | All (24 位；Zq) | All (32 位；Zq/GF($2^n$)) |
| Kyber(Saber)-512 (密钥生成+封装+解封装) | | | | | |
| 吞吐率/OPS | 207 | 2841 | 47925 | 1924 | 69473 |
| 能量效率/(μJ/Op) | 26.6 | 1.7 | 3.4 | 5.2 | 4.4 |
| 能量延迟积比率 | 2020.27 | 9.67 | 1.12 | 42.65 | 1.0 |
| Dilithium-Ⅱ (密钥生成+签名+验签) | | | | | |
| 吞吐率/OPS | 87 | — | 11527 | N/A | 15832 |
| 能量效率/(μJ/Op) | 87.2 | — | 20.6 | N/A | 22.8 |
| 能量延迟积比率 | 695.99 | — | 1.24 | N/A | 1.0 |
| Falcon-512 签名 | | | | | |
| 吞吐率/OPS | — | — | — | — | 4467 |
| 能量效率/(μJ/Op) | — | — | — | — | 47.8 |

表 3-13 给出了 PQPU 运行不同抗量子密码算法时达到的计算吞吐率和能量效率。可以看到,Kyber 算法和 Dilithium 算法具有更高的计算性能与能效。需要指出的是 Falcon 算法给出的是签名/验签的性能,Classic McEliece 算法给出的是加解密性能。

为了进一步验证 PQPU 相比于 RePQC 在架构优化方面取得的性能提升,针对 NIST 优先推荐的 Kyber 和 Dilithium 算法性能进行对比。性能对比情况如表 3-14 所示。首先,PQPU 支持更多的数学困难问题与算法。其次,在所有安全等级的算法参数下,PQPU 在计

算吞吐率方面均大幅提升。当然，由于支持算法种类的增加，相同工作条件下 PQPU 的功耗相比 RePQC 略有增加，但每次操作的能量延迟积都更具优势。

表 3-13　PQPU 不同算法的性能情况

| 算　　法 | 吞吐率/OPS | 能量效率 /(μJ/Op) |
|---|---|---|
| Kyber-512 | 69473 | 4.4 |
| Kyber-768 | 51674 | 6.6 |
| Kyber-1024 | 37241 | 9.3 |
| Dilithium-Ⅱ | 15832 | 26.6 |
| Dilithium-Ⅲ | 9234 | 42.8 |
| Dilithium-Ⅴ | 7717 | 55.6 |
| Falcon(签名/验签) | 4467/270005 | 47.8/1.5 |
| LAC-128 | 3924 | 44.7 |
| “SPHINCS+”-128f | 211 | 841.5 |
| “SPHINCS+”-256s | 10 | 23030 |
| Classic McEliece(加密/解密) | 77981/14311 | 2.4/30.5 |
| BIKE-128 | 13063 | 17 |
| BIKE-192 | 48 | 2011 |
| BIKE-256 | 22 | 4424 |
| HQC-128 | 1711 | 114 |
| HQC-192 | 1089 | 189 |
| HQC-256 | 551 | 374 |

表 3-14　PQPU 与 RePQC 的性能对比

| 规格参数 | RePQC | PQPU |
|---|---|---|
| 算法 | Kyber、Dilithium、Classic McEliece、Rainbow、NTRU、Saber | Kyber、Dilithium、Falcon、“SPHINCS+”、BIKE、Classic McEliece、HQC、Aigis、LAC |
| Kyber-768（密钥生成+封装+解封） | | |
| 吞吐率/OPS | 33834 | 51674 |
| 能量效率/(μJ/Op) | 5.7 | 6.6 |
| 能量延迟积比率 | 1.32 | 1.0 |
| Kyber-1024（密钥生成+封装+解封） | | |
| 吞吐率/OPS | 25084 | 37241 |
| 能量效率/(μJ/Op) | 8.7 | 9.3 |
| 能量延迟积比率 | 1.38 | 1.0 |
| Dilithium-Ⅲ（密钥生成+签名+验签） | | |
| 吞吐率/OPS | 7030 | 9234 |
| 能量效率/(μJ/Op) | 39.6 | 42.8 |
| 能量延迟积比率 | 1.22 | 1.0 |
| Dilithium-Ⅴ（密钥生成+签名+验签） | | |
| 吞吐率/OPS | 6119 | 7117 |
| 能量效率/(μJ/Op) | 49.6 | 55.6 |
| 能量延迟积比率 | 1.04 | 1.0 |

相比于基于指令集扩展和面向特定算法的全定制硬件设计方案，粗粒度可重构计算架构可以在功能灵活性与能量效率之间获得较为理想的折中。但是，如何保证其对持续演进的抗

量子密码算法仍然是有待深入研究。目前针对这一问题的解决方案包括通过灵活编译器将硬件架构本不支持的操作类型转换为可支持的等效操作[35]。

## 参考文献


[1] Hooker S. The hardware lottery[J]. Commun. ACM,2021,64(12): 58-65.

[2] Leiserson C E, Thompson N C, Emer J S, et al. There's plenty of room at the top: What will drive computer performance after Moore's law? [J]. Science,368(6495): eaam9744.

[3] Hennessy J L,Patterson D A. A new golden age for computer architecture[J]. Commun. ACM,2019,62(2): 48-60.

[4] Bossuet L,Grand M,Gaspar L,et al. Architectures of flexible symmetric key crypto engines—a survey: from hardware coprocessor to multi-crypto-processor system on chip[J]. ACM Computing Surveys, 2013,45(4).

[5] 刘雷波,王博,魏少军. 可重构计算密码处理器[M]. 北京: 科学出版社,2018.

[6] RISC-V Summit 2023: Benchmarking RISC-V Post-Quantum-Markk[EB/OL]. [2024-06-25]. https://riscvsummit2023. sched. com/event/1QUpL.

[7] Banerjee U,Wright A,Juvekar C,et al. An energy-efficient reconfigurable DTLS cryptographic engine for securing internet-of-things applications [J]. IEEE Journal of Solid-State Circuits, 2019, 54 (8): 2339-2352.

[8] Banerjee U,Ukyab T S,Chandrakasan A P. Sapphire: A configurable crypto-processor for post-quantum lattice-based protocols[J]. IACR Transactions on Cryptographic Hardware and Embedded Systems, 2019: 17-61.

[9] Verbauwhede I,Balasch J,Roy S S,et al. 24. 1 Circuit challenges from cryptography[C]//2015 IEEE International Solid-State Circuits Conference,2015.

[10] Song S,Tang W,Chen T,et al. LEIA: A 2. 05mm$^2$ 140mW lattice encryption instruction accelerator in 40nm CMOS[C]//IEEE Custom Integrated Circuits Conference,2018.

[11] Fritzmann T,Sigl G,Sep U L J. Extending the RISC-V instruction set for hardware acceleration of the post-quantum scheme LAC[Z]. 2020: 1420-1425.

[12] Karl P,Fritzmann T,Sigl G. Hardware accelerated FrodoKEM on RISC-V[C]//2022 25th International Symposium on Design and Diagnostics of Electronic Circuits and Systems,2022.

[13] Roy D B, Fritzmann T, Sigl G. Efficient hardware/software co-design for post-quantum crypto algorithm SIKE on ARM and RISC-V based microcontrollers[Z]. 2020.

[14] Karl P,Schupp J,Fritzmann T,et al. Post-quantum signatures on RISC-V with hardware acceleration [J]. ACM Trans. Embed. Comput. Syst. ,2023,23(2): 1-30.

[15] Fritzmann T, Sigl G, Sepúlveda J. RISQ-V: Tightly coupled RISC-V accelerators for post-quantum cryptography[J]. IACR Transactions on Cryptographic Hardware and Embedded Systems,2020,2020(4): 239-280.

[16] Fritzmann T,Beirendonck M V,Roy D B,et al. Masked accelerators and instruction set extensions for post-quantum cryptography [J]. IACR Transactions on Cryptographic Hardware and Embedded Systems,2022: 414-460.

[17] Zhu Y,Zhu M,Yang B,et al. LWRpro: An energy-efficient configurable crypto-processor for module-LWR[J]. IEEE Transactions on Circuits and Systems I: Regular Papers,2021,68(3): 1146-1159.

[18] Imran M,Almeida F,Basso A,et al. High-speed SABER key encapsulation mechanism in 65nm CMOS [J]. Journal of Cryptographic Engineering,2023,13(4): 461-471.

[19] Ghosh A,Mera J M B,Karmakar A,et al. A 334μW 0. 158mm$^2$ saber learning with rounding based post-quantum crypto accelerator[C]//IEEE Custom Integrated Circuits Conference,2022.

[20] Ghosh A, Mera J M B, Karmakar A, et al. A 334μW 0.158mm$^2$ ASIC for post-quantum key-encapsulation mechanism saber with low-latency striding toom-cook multiplication[J]. IEEE Journal of Solid-State Circuits, 2023, 58(8): 2383-2398.

[21] Aikata A, Mert A C, Imran M, et al. KaLi: A crystal for post-quantum security using kyber and dilithium[J]. IEEE Transactions on Circuits and Systems Ⅰ: Regular Papers, 2023, 70(2): 747-758.

[22] Yihonf Z, Wenping Z, Chen C, et al. Mckeycutter: A high-throughput key generator of classic McEliece on hardware[C]//60th ACM/IEEE Design Automation Conference (DAC), 2023.

[23] Deshpande S, Xu C, Nawan M, et al. Fast and efficient hardware implementation of HQC[Z]. 2023.

[24] Brockmann J, Monoj, Gunysu T, et al. Folding BIKE: Scalable hardware implementation for reconfigurable devices[J]. IEEE Transactions on Computers, 2022, 71(5): 1204-1215.

[25] Richter-Brockmann J, Chen M S, Ghosh S, et al. Racing BIKE: Improved polynomial multiplication and inversion in hardware[J]. IACR Transactions on Cryptographic Hardware and Embedded Systems, 2021, 2022(1): 557-588.

[26] Aguilar-Melchor C, Deneuville J C, Dion A, et al. Towards automating cryptographic hardware implementations: A case study of HQC[Z]. 2022.

[27] Thoma J P, Hartlief D, G U Neysu T. Agile acceleration of stateful hash-based signatures in hardware [J]. ACM Trans. Embed. Comput. Syst., 2024, 23(2): 1-29.

[28] Cao Y, Wu Y, Qin L, et al. Area, time and energy efficient multicore hardware accelerators for extended merkle signature scheme[J]. IEEE Transactions on Circuits and Systems Ⅰ: Regular Papers, 2022, 69(12): 4908-4918.

[29] Mohan P, Wang W, Jungk B, et al. ASIC accelerator in 28nm for the post-quantum digital signature scheme XMSS[C]//2020 IEEE 38th International Conference on Computer Design, 2020.

[30] Saarinen M J O. Accelerating SLH-DSA by two orders of magnitude with a single hash unit[A/OL]. (2024)[2024-03-11]. https://eprint.iacr.org/2024/367.

[31] Zhu Y, Zhu W, Zhu M, et al. A 28nm 48KOPS 3.4μJ/Op agile crypto-processor for post-quantum cryptography on multi-mathematical problems [C]//IEEE International Solid- State Circuits Conference, 2022.

[32] Zhu Y, Zhu W, Li C, et al. RePQC: A 3.4-μJ/Op 48-kOPS post-quantum crypto-processor for multiple-mathematical problems[J]. IEEE Journal of Solid-State Circuits, 2023, 58(1): 124-140.

[33] Zhu Y, Zhu W, Ouyang Y, et al. A 28nm 69.4kOPS 4.4μJ/Op versatile post-quantum crypto-processor across multiple mathematical problems[C]//IEEE International Solid-State Circuits Conference, 2024.

[34] Banerjee U, Pathak A, Chandrakasan A P. 2.3 An energy-efficient configurable lattice cryptography processor for the quantum-secure internet of things [C]//IEEE International Solid-State Circuits Conference, 2019.

[35] Xin G, Han J, Yin T, et al. VPQC: A domain-specific vector processor for post-quantum cryptography based on RISC-V architecture[J]. IEEE Transactions on Circuits and Systems Ⅰ: Regular Papers, 2020, 67(8): 2672-2684.

[36] Kim B, Park J, Moon S, et al. Configurable energy-efficient lattice-based post-quantum cryptography processor for IoT devices[C]//IEEE 48th European Solid State Circuits Conference, 2022.