Max Planck Institute for Security and Privacy, Kookmin University, Kookmin University

Multiplying Polynomials without Powerful Multiplication Instructions

Vincent Hwang, YoungBeom Kim, and Seog Chung Seo

September 15

Polynomial Multiplications

- NTT-friendly rings.
  - Prime moduli.
  - Composite moduli.
- NTT-unfriendly rings.
  - Over $\mathbb{Z}_{2^k}$.
  - Multiple NTTs + CRT.
  - NTT over large NTT-friendly rings.
  - Addition-only NTT (Schönhage, Nussbaumer).

Optimize on platforms with **expensive** high/long multiplications.

- NTT-friendly rings with prime moduli:
  - Generalized Barrett modular multiplication.
- NTT-unfriendly rings
  - Nussbaumer over $\mathbb{Z}_{2^k}$.
  - Revise cost analysis.

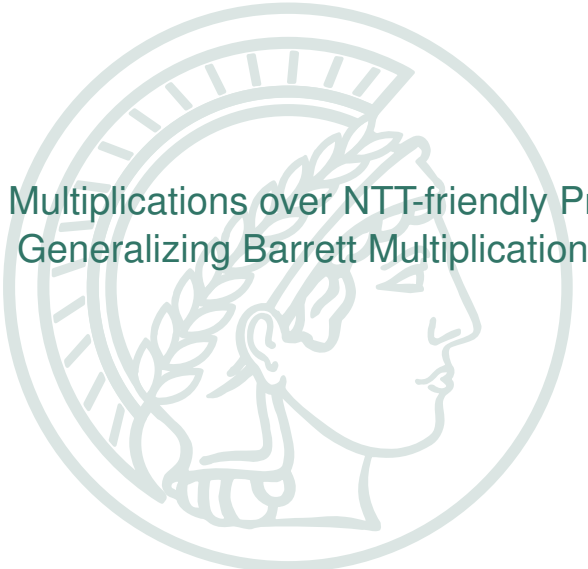
Dilithium and Cortex-M3

Dilithium (ML-DSA now)

- NIST post-quantum cryptography standard (FIPS 204).
- Dilithium NTT: Modular arithmetic with the prime modulus $q = 2^{23} - 2^{13} + 1$.

Cortex-M3

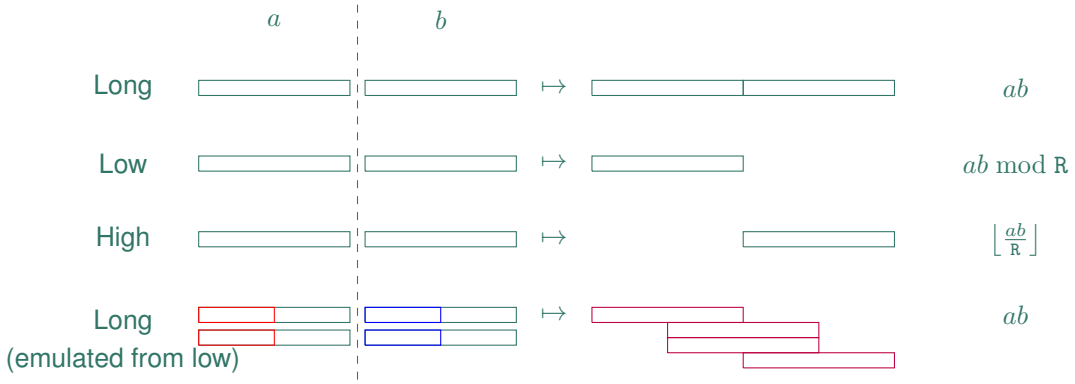
- Input-dependent-time long multiplication instructions.
- Emulate long/high multiplications with
  - low multiplication instructions, and
  - addition/logical instructions for merging and carrying.



Polynomial Multiplications over NTT-friendly Prime Moduli: Generalizing Barrett Multiplication

Multiplication Instructions

R is a power of two, 2^{32} on Cortex-M3.



Barrett Multiplication

Modulus $q < \mathtt{R}$, compute $c \equiv ab \bmod q$ with a signed modular multiplication.

- Long-multiply and then reduce: Montgomery, and more.
- Barrett: Approximate with a q-multiple with equal high halve and then subtract.
  - High part of the long-products can be skipped.

Approximating with a q-multiple.

$$ab \bmod q = ab - \left\lfloor \frac{ab}{q} \right\rceil q.$$

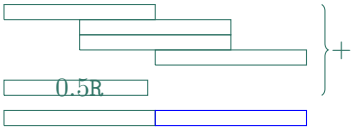
For an $z \in \mathbb{Z}$, $\left| z - \left\lfloor \frac{ab}{q} \right\rceil \right| \leq \delta \longrightarrow |(ab - zq) - ab \bmod q| \leq \delta q$.

Let $b' = \left\lfloor \frac{b\mathtt{R}}{q} \right\rceil$ be a precomputed constant.

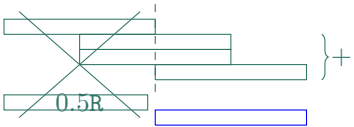
- Barrett: $z = \left\lfloor \frac{ab'}{\mathtt{R}} \right\rceil \longrightarrow \delta \leq 1$.
- This work: Relax δ for efficiency.

Generalized Barrett Multiplication (2-Limb)

Originally, compute $ab - \left\lfloor \frac{ab'}{\mathrm{R}} \right\rfloor q$ with $\left\lfloor \frac{ab'}{\mathrm{R}} \right\rfloor$ as:



Instead, compute $ab - \left[\!\left[\frac{ab'}{\mathrm{R}} \right]\!\right] q$ with $\left[\!\left[\frac{ab'}{\mathrm{R}} \right]\!\right]$ as:



A Bit More Math

- $b' = \left\lfloor \frac{b\mathtt{R}}{q} \right\rceil$.
- For a $\delta > 0$, δ-integer approximation $[\![]\!]$: $\forall r \in \mathbb{R}, |r - [\![r]\!]| \leq \delta$.
- For a b, write $b_l + b_h\sqrt{\mathtt{R}} = \left\lfloor \frac{b\mathtt{R}}{q} \right\rceil$. Define $[\![]\!]_b$ as

$$\forall r \in \mathbb{R}, [\![r]\!]_b \coloneqq \left\lfloor \frac{a_l b_h}{\sqrt{\mathtt{R}}} \right\rceil + \left\lfloor \frac{a_h b_l}{\sqrt{\mathtt{R}}} \right\rceil + a_h b_h$$

  where $a_l + a_h\sqrt{\mathtt{R}} = \frac{r\mathtt{R}}{\left\lfloor \frac{b\mathtt{R}}{q} \right\rceil}$, $b_l + b_h\sqrt{\mathtt{R}} = \left\lfloor \frac{b\mathtt{R}}{q} \right\rceil$.
- Obviously, $|[\![r]\!]_b - r| < 3$ (see previous slide).
- $|[\![r]\!]_b - \lfloor r \rceil| < 3$ (see paper).
- $\left| \left(ab - \left[\!\!\left[\frac{ab'}{\mathtt{R}} \right]\!\!\right]_b q \right) - ab \bmod q \right| \leq 3q$.

Results of Modular Multiplications

Plain multiplication		
Multiplication operation	Work	Cycle
Long (variable-time)	[ARM10]	$3-7$
Long (constant-time, non-generic)	[GKS20]	11
Long (constant-time)	[GKS20]	12
Modular multiplication (constant-time)		
Montgomery multiplication	[GKS20]	23
Barrett multiplication (approximate)	This work	12 (1.92)
Modular multiplication (variable-time)		
Montgomery multiplication	[GKS20]	$9-16$
Barrett multiplication (floor)	This work	$6-8$ $(1.13-2.67)$

Table: Overview of multiplication operations with 32-bit input values on Cortex-M3. The cycles are obtained by summing up the instruction timings from the manual [ARM10].

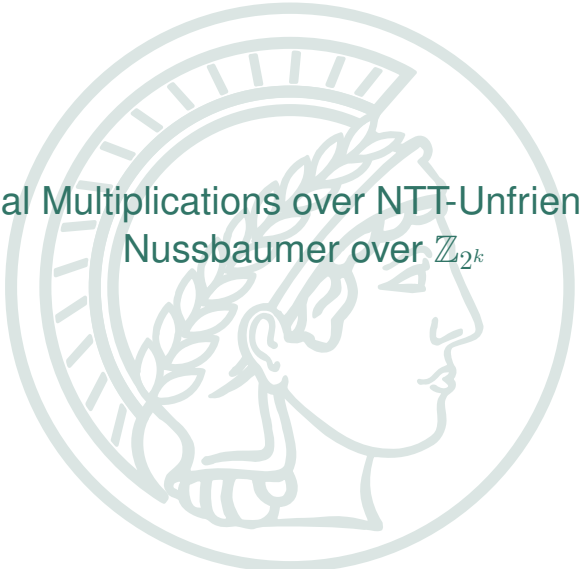
Results of NTTs/iNTTs and Matrix-Vector Multiplications

Table: Performance Cycles of Dilithium NTT/iNTT on Cortex-M3.

	Constant-time		Variable-time	
	[GKS20]	This work	[GKS20]	This work
NTT	33 025	21 876 (1.51)	19 347	15 985 (1.21)
iNTT	36 609	26 524 (1.38)	21 006	19 067 (1.10)

Table: Performance Cycles of the Matrix-Vector Multiplications for Dilithium on Cortex-M3.

Security level	Constant-time		Variable-time	
	[GKS20]	This work	[GKS20]	This work
II	414k	242k (1.71)	240k	176k (1.36)
III	639k	371k (1.72)	370k	267k (1.39)
V	999k	566k (1.77)	578k	411k (1.41)



Polynomial Multiplications over NTT-Unfriendly Moduli: Nussbaumer over $\mathbb{Z}_{2^k}$

Cooley–Tukey and Nussbaumer

$\lg = \log_2$. Consider $R[x]/\langle x^n + 1\rangle$, $n = 2^{2^k}$ for $k \in \mathbb{Z}_{\geq 0}$.

- Cooley–Tukey
  - R: NTT-friendly ring containing n^{-1}.
  - Multiplication-based.
  - R^n.
  - $O(n \lg n)$ multiplications, $O(n \lg n)$ additions.
- Nussbaumer
  - R: arbitrary ring containing n^{-1}.
  - Addition-based.
  - $R^{O(n \lg n)}$.
  - 0 multiplications, $O(n \lg n \lg \lg n)$ additions.

A More Practical Cost Analysis

1. Transform until dimension $\leq t$, a platform-dependent constant.
2. Multiply with t^α multiplications for an $1 \leq \alpha \leq 2$.

Table: Arithmetic cost of Cooley–Tukey and Nussbaumer FFTs for multiplying two size-n polynomials with the threshold t.

	Cooley-Tukey	Nussbaumer
R	An NTT-friendly ring	A ring
Transformation		
# of mul.	$\frac{1}{2 \lg t} \cdot n \lg n$	0
# of add./sub.	$\frac{1}{\lg t} \cdot n \lg n$	$\Theta\left(n \lg n \max\left(\lg \log_t n, 1\right)\right)$
# of small dim. polymul.	$\frac{n}{t}$	$\frac{1}{t \lg t} \cdot n \lg n$
Polynomial multiplication		
# of mul.	$\frac{3}{2 \lg t} \cdot n \lg n + n t^{\alpha-1}$	$\frac{t^{\alpha-1}}{\lg t} \cdot n \lg n$

The Power-of-Two Case

- Cooley–Tukey.
  - No twiddle factors in $\mathbb{Z}_{2^l}$ other than ± 1.
  - Need multiple NTTs over NTT-friendly moduli.
  - Cost of polymul.: $c_1 \cdot \frac{3}{2 \lg t} \cdot n \lg n + c_2 \cdot n t^{\alpha - 1}$ for large constants c_1, c_2.
  - On Cortex-M3:
    - 32-bit Montgomery: $c_1 = 23$.
    - 32-bit Barrett: $c_1 = 12$.
    - 16-bit Montgomery: $c_1 = 2 \cdot 3 \sim 3 \cdot 3$.
  - c_2 is closely related to c_1.
- Nussbaumer:
  - Still operate modulo a power of two.
  - Cost of polymul.: $\frac{t^{\alpha - 1}}{\lg t} \cdot n \lg n$.
  - Toeplitz matrix-vector multiplication for dimension $< t$ (see paper on why).

Results

Table: Performance cycles of polynomial multiplications with 32-bit arithmetic precision on Cortex-M3.

Work	[ACC+21]	[HAZ+24]	This work
Coefficient ring	$\prod_{i=0,1} \mathbb{Z}_{q_i}$	$\prod_{i=0,1} \mathbb{Z}_{q_i}$	$\mathbb{Z}_{2^{\leq 24}}$
Approach	Cooley–Tukey	Cooley–Tukey	Nussbaumer
Building block			
NTT/Hom-M	16 774 (0.93)	15 626	15 820 (0.99)
NTT/Hom-V	16 774 (0.93)	15 626	8 259 (1.89)
Mul./BiHom	11 933 (0.68)	8 061	11 217 (0.72)
iNTT/Hom-I	23 721 (0.88)	20 772	10 960 (1.90)
Polynomial multiplication			
Total cycles	69 202 (0.87)	60 085	46 256 (1.30)
Ratio of mul./BiHom over total cycles	17.24%	13.42%	24.25%

Overall Performance

Table: Performance cycles of Dilithium on Cortex-M3. **K**: key generation, **S**: signature generation, and **V**: signature verification.

NIST security level	Work	Operation					
		K		**S**		**V**	
		Cycles	Hash	Cycles	Hash	Cycles	Hash
II	[HAZ+24]	1 764k	1 185k	5 617k	2 173k	1 597k	1 065k
	This work	1 540k	1 123k	4 554k	2 173k	1 508k	1 065k
III	[HAZ+24]	2 944k	2 034k	7 448k	3 399k	2 659k	1 872k
	This work	2 669k	2 034k	6 529k	3 399k	2 522k	1 872k
V	[HAZ+24]	4 923k	3 510k	20 180k	14 195k	4 525k	3 347k
	This work	4 448k	3 510k	18 383k	14 195k	4 295k	3 347k

More

- Lattice-based cryptosystem Saber.
  - Multiply in $\mathbb{Z}_{2^{13}}[x]/\langle x^{256}+1\rangle$.
- 8-bit AVR.
  - 8-bit native multiplication instructions.
  - Generalized Barrett is the fastest for Dilithium NTT.
  - For Saber: Took–Cook (16-bit) $>>$ Nussbaumer (32-bit) $>>$ others (NTT...).
    - 2^{13} is too large.
    - Nussbaumer incurs precision issues.
    - Took–Cook stays at 16-bit.

Summary

Assuming high/long multiplications are slow.

- NTT over prime moduli.
  - Generalized Barrett suitable for multi-limb arithmetic.
  - Defeating any multiply-then-reduce approaches on Cortex-M3.
    - ~~Montgomery~~
    - ~~Solinas~~
    - ~~[Super fancy reduction]~~
  - Optimizing **modular multiplication** as a whole instead of solely **modular reduction**.
  - First time seen in the literature.
- Polynomial multiplications over power-of-two moduli.
  - NTTs over odd moduli are slow: applying multiple NTTs is slow.
  - Nussbaumer over $\mathbb{Z}_{2^k}$ is the fastest if there are no precision issues.
- Paper: `https://tches.iacr.org/index.php/TCHES/article/view/11926`.
- Artifact: `https://github.com/vincentvbh/PolyMul_Without_PowerfulMul`.

Thanks for listening

Reference I

[ACC+21] Amin Abdulrahman, Jiun-Peng Chen, Yu-Jia Chen, Vincent Hwang, Matthias J. Kannwischer, and Bo-Yin Yang, *Multi-moduli NTTs for Saber on Cortex-M3 and Cortex-M4*, IACR Transactions on Cryptographic Hardware and Embedded Systems **2022** (2021), no. 1, 127–151, https://tches.iacr.org/index.php/TCHES/article/view/9292.

[ARM10] ARM, *Cortex-M3 Technical Reference Manual*, 2010, https://developer.arm.com/documentation/ddi0337/h.

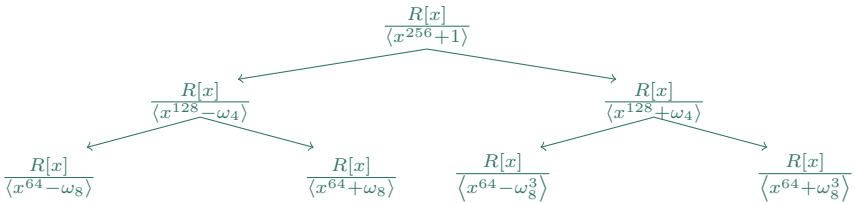
[GKS20] Denisa O. C. Greconici, Matthias J. Kannwischer, and Amber Sprenkels, *Compact Dilithium Implementations on Cortex-M3 and Cortex-M4*, IACR Transactions on Cryptographic Hardware and Embedded Systems **2021** (2020), no. 1, 1–24, https://tches.iacr.org/index.php/TCHES/article/view/8725.

Reference II

[HAZ+24] Junhao Huang, Alexandre Adomnicăi, Jipeng Zhang, Wangchen Dai, Yao Liu, Ray C. C. Cheung, Çetin Kaya Koç, and Donglong Chen, *Revisiting Keccak and Dilithium Implementations on ARMv7-M*, IACR Transactions on Cryptographic Hardware and Embedded Systems **2024** (2024), no. 2, 1–24, https://tches.iacr.org/index.php/TCHES/article/view/11419.

Cooley–Tukey

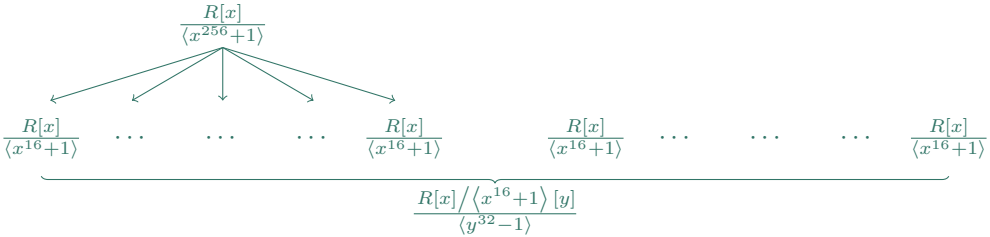
$\omega_8 \in R$ with $\omega_8^4 = -1$.



Consider $n = 2^k,\ R[x]/\langle x^n + 1\rangle$.

- $(\#\text{polynomial}, \text{dimension}) : (1, n) \to \left(2, \frac{n}{2}\right) \to \cdots \to (n, 1)$.
- Transformation cost: $O(n \lg n)$ multiplications/additions.
- Polynomial multiplication cost: $O(n \lg n)$ multiplications/additions.

Nussbaumer



$\frac{R[x]/\langle x^{16}+1\rangle[y]}{\langle y^{32}-1\rangle} \cong \left(\frac{R[x]}{\langle x^{16}+1\rangle}\right)^{32}$ with NTT in y modulo $y^{32}-1$ and x^is as twiddle factors.

Consider $n = 2^{2^k}$, $R[x]/\langle x^n+1\rangle$.

- $(\#\text{polynomial}, \text{dimension}) : (1, n) \to \left(2n^{\frac{1}{2}}, n^{\frac{1}{2}}\right) \to \cdots \to \left(2^{k-1}n, 2\right)$.
- Transformation cost: 0 multiplications, $O\left(n \lg n \lg_2 \lg_2 n\right)$ additions.
- Polynomial multiplication cost: $O\left(n \lg n\right)$ multiplications, $O\left(n \lg n \lg \lg n\right)$ additions.