Optimizing multiplications with vector instructions

Chitchanok Chuengsatiansup

INRIA and ENS de Lyon

4 June 2018

Introduction

- Current position:
  - Postdoc (INRIA and ENS de Lyon)
  - Supervisor: Damien Stehlé

Introduction

- Current position:
  - Postdoc (INRIA and ENS de Lyon)
  - Supervisor: Damien Stehlé
- Previous position:
  - PhD student at TU/Eindhoven, The Netherlands
    Cryptographic Implementations group
  - Thesis: *"Optimizing Curve-Based Cryptography"*
  - Supervisors: Daniel J. Bernstein and Tanja Lange

Introduction

- Current position:
  - Postdoc (INRIA and ENS de Lyon)
  - Supervisor: Damien Stehlé
- Previous position:
  - PhD student at TU/Eindhoven, The Netherlands
    Cryptographic Implementations group
  - Thesis: *"Optimizing Curve-Based Cryptography"*
  - Supervisors: Daniel J. Bernstein and Tanja Lange
- Experience
  - Software implementations
  - Optimizing cryptographic software and algorithms

Vectorization speedups

without vector

a

$+$

b

$=$

$a + b$

Vectorization speedups

without vector

a
$+$
b
$=$
$a+b$

with vector

a_0	a_1	a_2	a_3
$+$	$+$	$+$	$+$
b_0	b_1	b_2	b_3
$=$	$=$	$=$	$=$
a_0+b_0	a_1+b_1	a_2+b_2	a_3+b_3

Vectorization speedups

without vector

a
$+$
b
$=$
$a + b$

with vector

a_0	a_1	a_2	a_3
$+$	$+$	$+$	$+$
b_0	b_1	b_2	b_3
$=$	$=$	$=$	$=$
$a_0 + b_0$	$a_1 + b_1$	$a_2 + b_2$	$a_3 + b_3$

- **single** instruction performing n **independent** operations on **aligned** inputs

Side-channel attacks

- Prevent software side-channel attacks:
  - constant-time
  - no input-dependent branch
  - no input-dependent array index

Side-channel attacks

- Prevent software side-channel attacks:
  - constant-time
  - no input-dependent branch
  - no input-dependent array index

- Constant-time table-lookup:
  - read entire table
  - select via arithmetic
    if c is 1, select tbl[i]
    if c is 0, ignore tbl[i]
    $t = (t \cdot (1 - c)) + (tbl[i] \cdot (c))$
    $t = (t \wedge (c - 1)) \vee (tbl[i] \wedge (-c))$

Curve41417

Design of Curve41417

- High-security elliptic curve (security level above 2^{200})
- Defined over prime field $\mathbb{F}_p$ where $p = 2^{414} - 17$
- In Edwards curve form

$$x^2 + y^2 = 1 + 3617x^2y^2$$

Design of Curve41417

- High-security elliptic curve (security level above 2^{200})
- Defined over prime field $\mathbb{F}_p$ where $p = 2^{414} - 17$
- In Edwards curve form

$$x^2 + y^2 = 1 + 3617x^2y^2$$

- Large prime-order subgroup (cofactor 8)
- IEEE P1363 criteria (large embedding degree, etc.)
- Twist secure, i.e., twist of Curve41417 also secure

ECC arithmetic

- Mixed-coordinate systems:
  - doubling: projective X, Y, Z
  - addition: extended X, Y, Z, T

(See https://hyperelliptic.org/EFD/)

ECC arithmetic

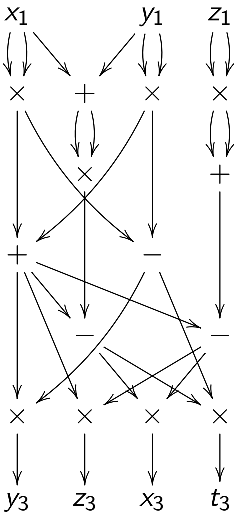
- Mixed-coordinate systems:
  - doubling: projective X, Y, Z
  - addition: extended X, Y, Z, T

(See https://hyperelliptic.org/EFD/)

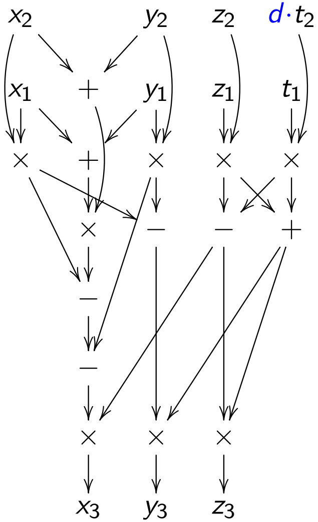
- Scalar multiplication:
  - signed fixed windows of width $w = 5$
  - precompute $0P, 1P, 2P, \dots, 16P$
    also multiply $d = 3617$ to T coordinate
  - special first doubling
  - compute T only before addition

Point operations

Point doubling



Point addition



ARM Cortex-A8 vector unit

- 128-bit vector registers
- Arithmetic and load/store unit can perform in parallel
- Operate in parallel on vectors of four 32-bit integers or two 64-bit integers

ARM Cortex-A8 vector unit

- 128-bit vector registers
- Arithmetic and load/store unit can perform in parallel
- Operate in parallel on vectors of four 32-bit integers or two 64-bit integers
- Each cycle produces:

  four 32-bit integer additions: $a_0+b_0, a_1+b_1, a_2+b_2, a_3+b_3$

  or

  two 64-bit integer additions: c_0+d_0, c_1+d_1

  or

  one multiply-add instruction: $a_0 b_0 + c_0$

  where a_i, b_i are 32- and c_i, d_i are 64-bit integers

Redundant representation

- Use **non-integer** radix $2^{414/16} = 2^{25.875}$
- Decompose integer f modulo $2^{414} - 17$ into 16 integer pieces
- Write f as

$$
\begin{array}{llll}
f_0 + & 2^{26} f_1 + & 2^{52} f_2 + & 2^{78} f_3 + \\
2^{104} f_4 + & 2^{130} f_5 + & 2^{156} f_6 + & 2^{182} f_7 + \\
2^{207} f_8 + & 2^{233} f_9 + & 2^{259} f_{10} + & 2^{285} f_{11} + \\
2^{311} f_{12} + & 2^{337} f_{13} + & 2^{363} f_{14} + & 2^{389} f_{15}
\end{array}
$$

Carries

- Goal: Bring each limb down to 26 or 25 bits

Carries

- Goal: Bring each limb down to 26 or 25 bits
- Typical carry chain:
  $m_0 \to m_1 \to m_2 \to \cdots \to m_{14} \to m_{15} \to m_0 \to m_1$

Carries

- Goal: Bring each limb down to 26 or 25 bits
- Typical carry chain:
  $m_0 \to m_1 \to m_2 \to \cdots \to m_{14} \to m_{15} \to m_0 \to m_1$
- Increase throughput:

Carries

- Goal: Bring each limb down to 26 or 25 bits
- Typical carry chain:
  $m_0 \to m_1 \to m_2 \to \cdots \to m_{14} \to m_{15} \to m_0 \to m_1$
- Increase throughput:
  $m_0 \to m_1$
  $m_8 \to m_9$

Carries

- Goal: Bring each limb down to 26 or 25 bits
- Typical carry chain:
  $m_0 \rightarrow m_1 \rightarrow m_2 \rightarrow \cdots \rightarrow m_{14} \rightarrow m_{15} \rightarrow m_0 \rightarrow m_1$
- Increase throughput:
  $m_0 \rightarrow m_1 \ \rightarrow m_2$
  $m_8 \rightarrow m_9 \ \rightarrow m_{10}$

Carries

- Goal: Bring each limb down to 26 or 25 bits
- Typical carry chain:
  $m_0 \to m_1 \to m_2 \to \cdots \to m_{14} \to m_{15} \to m_0 \to m_1$
- Increase throughput:
  $m_0 \to m_1 \to m_2 \to m_3 \to m_4 \to m_5 \to m_6 \to m_7 \to m_8 \to m_9$
  $m_8 \to m_9 \to m_{10} \to m_{11} \to m_{12} \to m_{13} \to m_{14} \to m_{15} \to m_0 \to m_1$

Carries

- Goal: Bring each limb down to 26 or 25 bits
- Typical carry chain:
  $m_0 \to m_1 \to m_2 \to \cdots \to m_{14} \to m_{15} \to m_0 \to m_1$
- Increase throughput:
  $m_0 \to m_1 \to m_2 \to m_3 \to m_4 \to m_5 \to m_6 \to m_7 \to m_8 \to m_9$
  $m_8 \to m_9 \to m_{10} \to m_{11} \to m_{12} \to m_{13} \to m_{14} \to m_{15} \to m_0 \to m_1$
- Decrease latency:

Carries

- Goal: Bring each limb down to 26 or 25 bits
- Typical carry chain:
  $m_0 \to m_1 \to m_2 \to \cdots \to m_{14} \to m_{15} \to m_0 \to m_1$
- Increase throughput:
  $m_0 \to m_1 \to m_2 \to m_3 \to m_4 \to m_5 \to m_6 \to m_7 \to m_8 \to m_9$
  $m_8 \to m_9 \to m_{10} \to m_{11} \to m_{12} \to m_{13} \to m_{14} \to m_{15} \to m_0 \to m_1$
- Decrease latency:
  $m_0 \to m_1$
  $m_8 \to m_9$

Carries

- Goal: Bring each limb down to 26 or 25 bits
- Typical carry chain:
  $m_0 \to m_1 \to m_2 \to \cdots \to m_{14} \to m_{15} \to m_0 \to m_1$
- Increase throughput:
  $m_0 \to m_1 \to m_2 \to m_3 \to m_4 \to m_5 \to m_6 \to m_7 \to m_8 \to m_9$
  $m_8 \to m_9 \to m_{10} \to m_{11} \to m_{12} \to m_{13} \to m_{14} \to m_{15} \to m_0 \to m_1$
- Decrease latency:
  $m_0 \to m_1$
  $m_8 \to m_9$
  $m_4 \to m_5$
  $m_{12} \to m_{13}$

Carries

- Goal: Bring each limb down to 26 or 25 bits
- Typical carry chain:
  $m_0 \to m_1 \to m_2 \to \cdots \to m_{14} \to m_{15} \to m_0 \to m_1$
- Increase throughput:
  $m_0 \to m_1 \to m_2 \to m_3 \to m_4 \to m_5 \to m_6 \to m_7 \to m_8 \to m_9$
  $m_8 \to m_9 \to m_{10} \to m_{11} \to m_{12} \to m_{13} \to m_{14} \to m_{15} \to m_0 \to m_1$
- Decrease latency:
  $m_0 \to m_1 \to m_2$
  $m_8 \to m_9 \to m_{10}$
  $m_4 \to m_5$
  $m_{12} \to m_{13}$

Carries

- Goal: Bring each limb down to 26 or 25 bits
- Typical carry chain:
  $m_0 \to m_1 \to m_2 \to \cdots \to m_{14} \to m_{15} \to m_0 \to m_1$
- Increase throughput:
  $m_0 \to m_1 \to m_2 \to m_3 \to m_4 \to m_5 \to m_6 \to m_7 \to m_8 \to m_9$
  $m_8 \to m_9 \to m_{10} \to m_{11} \to m_{12} \to m_{13} \to m_{14} \to m_{15} \to m_0 \to m_1$
- Decrease latency:
  $m_0 \to m_1 \to m_2$
  $m_8 \to m_9 \to m_{10}$
  $m_4 \to m_5 \to m_6$
  $m_{12} \to m_{13} \to m_{14}$

Carries

- Goal: Bring each limb down to 26 or 25 bits
- Typical carry chain:
  $m_0 \to m_1 \to m_2 \to \cdots \to m_{14} \to m_{15} \to m_0 \to m_1$
- Increase throughput:
  $m_0 \to m_1 \to m_2 \to m_3 \to m_4 \to m_5 \to m_6 \to m_7 \to m_8 \to m_9$
  $m_8 \to m_9 \to m_{10} \to m_{11} \to m_{12} \to m_{13} \to m_{14} \to m_{15} \to m_0 \to m_1$
- Decrease latency:
  $m_0 \to m_1 \to m_2 \to m_3 \to m_4 \to m_5$
  $m_8 \to m_9 \to m_{10} \to m_{11} \to m_{12} \to m_{13}$
  $m_4 \to m_5 \to m_6 \to m_7 \to m_8 \to m_9$
  $m_{12} \to m_{13} \to m_{14} \to m_{15} \to m_0 \to m_1$

Polynomial multiplication

- Goal: Compute $P = AB$
  given $A = a_0 + a_1 t^n$ and $B = b_0 + b_1 t^n$
- Method 1: schoolbook
  $P = a_0 b_0 + (a_0 b_1 + a_1 b_0) t^n + a_1 b_1 t^{2n}$
- Method 2: Karatsuba ($8n-4$ additions)
  $P = a_0 b_0 + ((a_0 + a_1)(b_0 + b_1) - a_0 b_0 - a_1 b_1) t^n + a_1 b_1 t^{2n}$
- Method 3: refined Karatsuba ($7n-3$ additions)
  $P = (a_0 b_0 - a_1 b_1 t^n)(1 - t^n) + (a_0 + a_1)(b_0 + b_1) t^n$

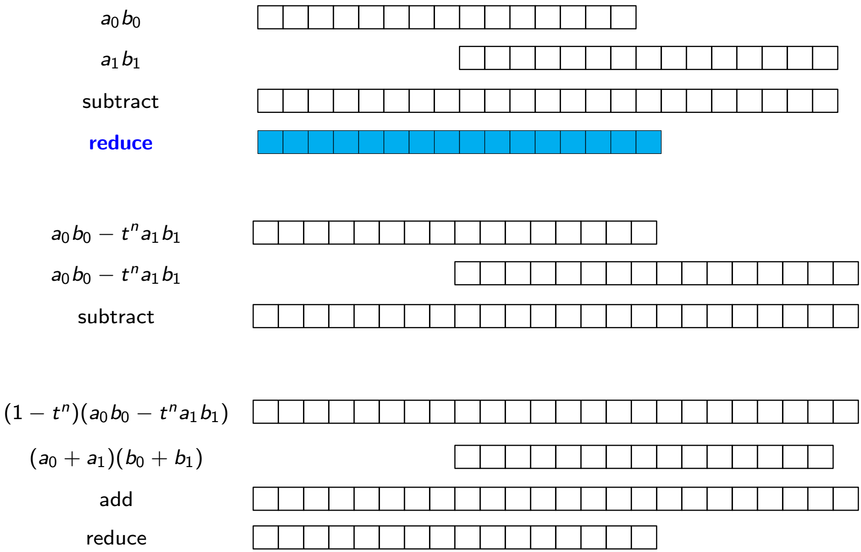
Polynomial multiplication $\bmod Q$

- Goal: Compute $P = AB \bmod Q$
  given $A = a_0 + a_1 t^n$ and $B = b_0 + b_1 t^n$
- Method 1: schoolbook
  $P = a_0 b_0 + (a_0 b_1 + a_1 b_0) t^n + a_1 b_1 t^{2n} \bmod Q$
- Method 2: Karatsuba ($8n-4$ additions)
  $P = a_0 b_0 + ((a_0 + a_1)(b_0 + b_1) - a_0 b_0 - a_1 b_1) t^n + a_1 b_1 t^{2n} \bmod Q$
- Method 3: refined Karatsuba ($7n-3$ additions)
  $P = (a_0 b_0 - a_1 b_1 t^n)(1 - t^n) + (a_0 + a_1)(b_0 + b_1) t^n \bmod Q$

Polynomial multiplication $\bmod Q$

- Goal: Compute $P = AB \bmod Q$
  given $A = a_0 + a_1 t^n$ and $B = b_0 + b_1 t^n$
- Method 1: schoolbook
  $P = a_0 b_0 + (a_0 b_1 + a_1 b_0) t^n + a_1 b_1 t^{2n} \bmod Q$
- Method 2: Karatsuba ($8n-4$ additions)
  $P = a_0 b_0 + ((a_0 + a_1)(b_0 + b_1) - a_0 b_0 - a_1 b_1) t^n + a_1 b_1 t^{2n} \bmod Q$
- Method 3: refined Karatsuba ($7n-3$ additions)
  $P = (a_0 b_0 - a_1 b_1 t^n)(1 - t^n) + (a_0 + a_1)(b_0 + b_1) t^n \bmod Q$
- Method 4: reduced refined Karatsuba ($6n-2$ additions) (new)
  $P = (a_0 b_0 - a_1 b_1 t^n \bmod Q)(1 - t^n) + (a_0 + a_1)(b_0 + b_1) t^n \bmod Q$

Reduced refined Karatsuba



Level of Karatsuba

- Karatsuba splits 1 $(2n \times 2n)$ into 3 $(n \times n)$

Level of Karatsuba

- Karatsuba splits 1 $(2n \times 2n)$ into 3 $(n \times n)$
- Zero-level Karatsuba (Schoolbook)
  e.g. for 16 limbs: $16 \times 16 = 256$

Level of Karatsuba

- Karatsuba splits 1 ($2n \times 2n$) into 3 ($n \times n$)
- Zero-level Karatsuba (Schoolbook)
  e.g. for 16 limbs: $16 \times 16 = 256$
- One-level Karatsuba
  e.g.: $16 \times 16 \rightarrow 3 \cdot (8 \times 8) +$ some additions
  $= 192 +$ some additions

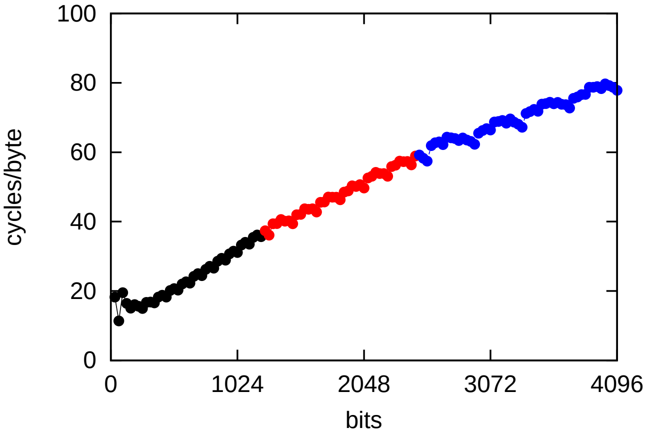
Level of Karatsuba

- Karatsuba splits 1 $(2n \times 2n)$ into 3 $(n \times n)$
- Zero-level Karatsuba (Schoolbook)
  e.g. for 16 limbs: $16 \times 16 = 256$
- One-level Karatsuba
  e.g.: $16 \times 16 \rightarrow 3 \cdot (8 \times 8) +$ some additions
  $= 192 +$ some additions
- Two-level Karatsuba
  e.g.: $3 \cdot (8 \times 8) \rightarrow 3 \cdot (3 \cdot (4 \times 4)) +$ even more additions
  $= 144 +$ even more additions

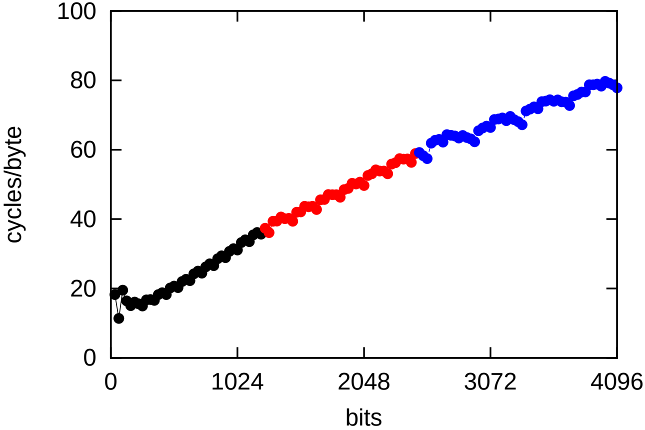
Level of Karatsuba

- Karatsuba splits 1 $(2n \times 2n)$ into 3 $(n \times n)$
- Zero-level Karatsuba (Schoolbook)
  e.g. for 16 limbs: $16 \times 16 = 256$
- One-level Karatsuba
  e.g.: $16 \times 16 \to 3 \cdot (8 \times 8)$ + some additions
  $= 192$ + some additions
- Two-level Karatsuba
  e.g.: $3 \cdot (8 \times 8) \to 3 \cdot (3 \cdot (4 \times 4))$ + even more additions
  $= 144$ + even more additions
- What is the zero-level/one-level cutoff for number of limbs?

GMP's cutoffs for Karatsuba

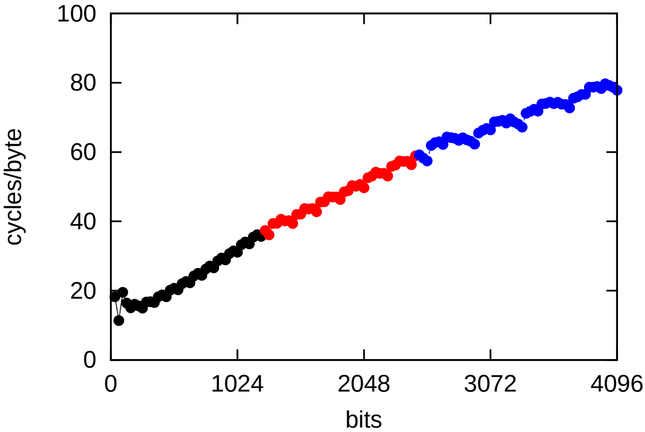


GMP's cutoffs for Karatsuba



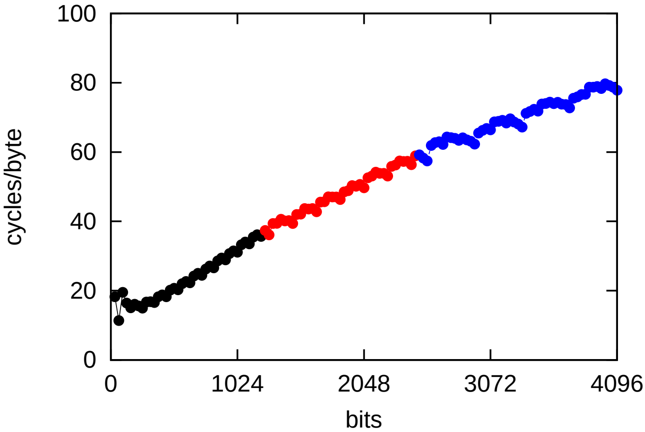
- GMP 6.0.0a library chooses 1248 bits on ARM Cortex-A8

GMP's cutoffs for Karatsuba



- GMP 6.0.0a library chooses 1248 bits on ARM Cortex-A8
- We reduce cutoff via improvements to Karatsuba

GMP's cutoffs for Karatsuba



- GMP 6.0.0a library chooses 1248 bits on ARM Cortex-A8
- We reduce cutoff via improvements to Karatsuba
- We reduce cutoff via redundant representation

Cost comparison (Karatsuba)

Level	Mult.	Add		Cost
		64-bit	32-bit	
0-level	256	15	0	256+ 8+ 0 = 264
1-level	192	59	16	192+30+ 4 = 226
2-level	144	119	40	144+60+10 = 214
3-level	108	191	76	108+96+19 = 223

Note: use multiply-add instructions

Recall:

1 cycle per multiplication

0.5 cycle per 64-bit addition

0.25 cycle per 32-bit addition

Cost comparison (refined Karatsuba)

Level	Mult.	Add		Cost
		64-bit	32-bit	
0-level	256	15	0	$256+\ 8+\ 0 = 264$
1-level	192	52	16	$192+26+\ 4 = 222$
2-level	144	103	40	$144+52+10 = 206$
3-level	108	166	76	$108+83+19 = 210$

Note: use multiply-add instructions

Recall:

1 cycle per multiplication

0.5 cycle per 64-bit addition

0.25 cycle per 32-bit addition

Cost comparison (reduced refined Karatsuba)

Level	Mult.	Add		Cost
		64-bit	32-bit	
0-level	256	15	0	$256 + 8 + 0 = 264$
1-level	192	45	16	$192 + 23 + 4 = 219$
2-level	144	96	40	$144 + 48 + 10 = 202$
3-level	108	159	76	$108 + 80 + 19 = 207$

Note: use multiply-add instructions

Recall:

1 cycle per multiplication

0.5 cycle per 64-bit addition

0.25 cycle per 32-bit addition

Performance comparison

- OpenSSL

curve	# cycle on i.MX515	# cycle on Sitara
`secp160r1`	$\approx$ 2.1 million	$\approx$ 2.1 million
`nistp192`	$\approx$ 2.9 million	$\approx$ 2.8 million
`nistp224`	$\approx$ 4.0 million	$\approx$ 3.9 million
`nistp256`	$\approx$ 4.0 million	$\approx$ 3.9 million
`nistp384`	$\approx$ 13.3 million	$\approx$ 13.2 million
`nistp521`	$\approx$ 29.7 million	$\approx$ 29.7 million

- Curve41417 (security level above 2^{200})
  - $\approx$ 1.6 million cycles on FreeScale i.MX515
  - $\approx$ 1.8 million cycles on TI Sitara

NTRU Prime

NTRU Prime

- High-security prime-degree large-Galois-group inert-modulus ideal-lattice-based cryptography
- System parameters (p, q, t)
  - p, q are prime
  - $p \geq \max\{2t, 3\}$
  - $q \geq 32t + 1$
  - $x^p - x - 1$ is irreducible in polynomial ring $(\mathbb{Z}/q)[x]$
- Fields of the form $(\mathbb{Z}/q)[x]/(x^p - x - 1)$

NTRU Prime

- High-security prime-degree large-Galois-group inert-modulus ideal-lattice-based cryptography
- System parameters (p, q, t)
  - p, q are prime
  - $p \geq \max\{2t, 3\}$
  - $q \geq 32t + 1$
  - $x^p - x - 1$ is irreducible in polynomial ring $(\mathbb{Z}/q)[x]$
- Fields of the form $(\mathbb{Z}/q)[x]/(x^p - x - 1)$
- Abbreviation:
  - ring $\mathbb{Z}[x]/(x^p - x - 1)$ as $\mathcal{R}$
  - ring $(\mathbb{Z}/3)[x]/(x^p - x - 1)$ as $\mathcal{R}/3$
  - field $(\mathbb{Z}/q)[x]/(x^p - x - 1)$ as $\mathcal{R}/q$

- Pick $g \in \mathcal{R}$

$$g = g_0 + \cdots + g_{p-1}x^{p-1} \text{ with } g_i \in \{-1, 0, 1\}$$

  g is required to be invertible in $\mathcal{R}/3$

- Pick $f \in \mathcal{R}$

$$f = f_0 + \cdots + f_{p-1}x^{p-1} \text{ with } f_i \in \{-1, 0, 1\} \text{ and } \sum |f_i| = 2t$$

  f is nonzero and hence invertible in $\mathcal{R}/q$

- Public key: $h = g/(3f)$ in $\mathcal{R}/q$
- Private keys: f in $\mathcal{R}$ and $1/g$ in $\mathcal{R}/3$

Streamlined NTRU Prime: KEM/DEM

- Use Key Encapsulation Mechanism (KEM) combined with Data Encapsulation Mechanism (DEM)
- KEM:
  - look up public key h
  - pick $r \in \mathcal{R}$ (i.e., $r_i \in \{-1, 0, 1\}, \sum |r_i| = 2t$)
  - compute hr in $\mathcal{R}/q$
  - round each coefficient (viewed as $\mathbb{Z} \cap [-(q-1)/2, (q-1)/2]$) to the nearest multiple of 3 to get c
  - compute $\mathrm{Hash}(r) = (C|K)$
  - send $(C|c)$, use session key K for DEM

Streamlined NTRU Prime: decapsulation

- To decrypt $(C|c)$
  - (reminder: $h = g/(3f)$ in $\mathcal{R}/q$)
  - compute $3fc = 3f(hr + m) = gr + 3fm$ in $\mathcal{R}/q$
  - reduce the coefficients modulo 3 to get $a = gr \in \mathcal{R}/3$
  - compute $r' = a/g \in \mathcal{R}/3$, lift r' to $\mathcal{R}$
  - compute $\mathrm{Hash}(r') = (C'|K')$ and c' as rounding of hr'
  - verify that $c' = c$ and $C' = C$
- If all verifications are ok, then $K = K'$ is the session key

Streamlined NTRU Prime 4591^{761}

- Field $(\mathbb{Z}/4591)[x]/(x^{761} - x - 1)$
- Parameters:
  - $p = 761$
  - $q = 4591$
  - $t = 143$
- Security: 2^{248} (pre-quantum)
  - considered hybrid lattice-reduction and meet-in-the-middle attack

Polynomial multiplication

- Main bottleneck is polynomial multiplication
- Multiplication algorithms considered:
  - Toom (3–6)
  - refined Karatsuba
  - arbitrary degree variant of Karatsuba (3–6)

Polynomial multiplication

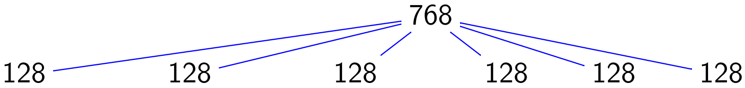
- Main bottleneck is polynomial multiplication
- Multiplication algorithms considered:
  - Toom (3–6)
  - refined Karatsuba
  - arbitrary degree variant of Karatsuba (3–6)
- Best operation count found so far for 768×768:
  - 5-level refined Karatsuba up to 128×128
  - Toom6: evaluated at $0, \pm 1, \pm 2, \pm 3, \pm 4, 5, \infty$

Combination of Toom and Karatsuba

768

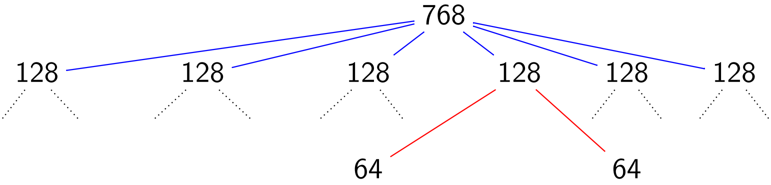
Blue = Toom
Red = Karatsuba
Green = Schoolbook

Combination of Toom and Karatsuba



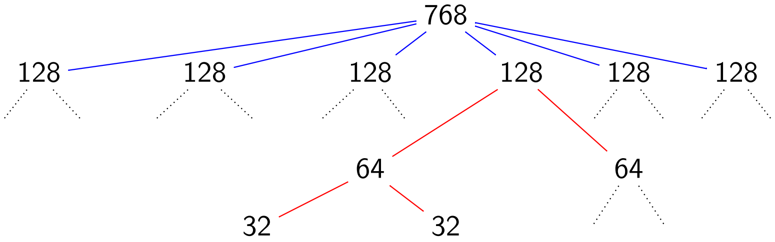
Blue = Toom
Red = Karatsuba
Green = Schoolbook

Combination of Toom and Karatsuba



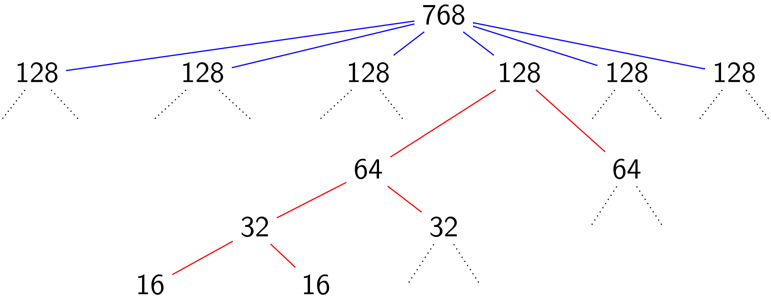
Blue = Toom
Red = Karatsuba
Green = Schoolbook

Combination of Toom and Karatsuba



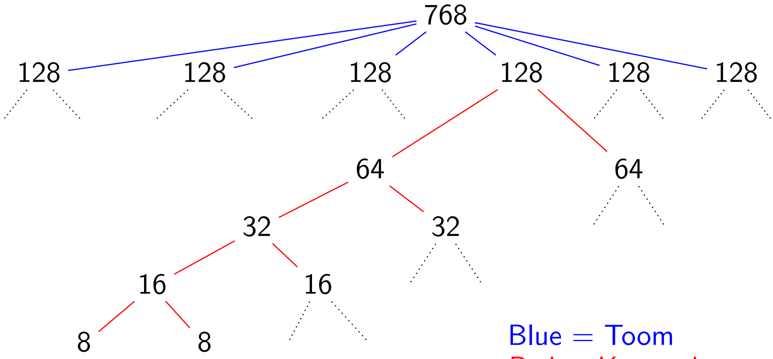
Blue = Toom
Red = Karatsuba
Green = Schoolbook

Combination of Toom and Karatsuba



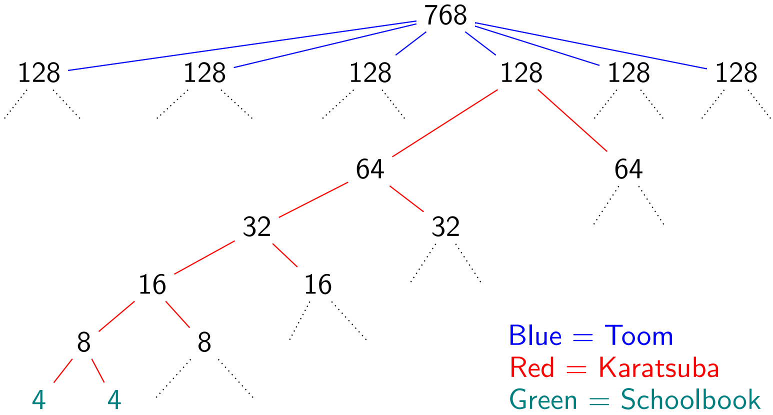
Blue = Toom
Red = Karatsuba
Green = Schoolbook

Combination of Toom and Karatsuba



Blue = Toom
Red = Karatsuba
Green = Schoolbook

Combination of Toom and Karatsuba



Toom: decomposition

- Decompose $a(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{767} x^{767}$ into

  $a(x, y) = A_0(x) + A_1(x)y + A_2(x)y^2 + A_3(x)y^3 + A_4(x)y^4 + A_5(x)y^5$

  where $y = x^{128}$ and

$$
\begin{aligned}
A_0(x) &= a_0 + a_1 x + a_2 x^2 + \cdots + a_{127} x^{127} \\
A_1(x) &= a_{128} + a_{129} x + a_{130} x^2 + \cdots + a_{255} x^{127} \\
A_2(x) &= a_{256} + a_{257} x + a_{258} x^2 + \cdots + a_{383} x^{127} \\
A_3(x) &= a_{384} + a_{385} x + a_{386} x^2 + \cdots + a_{511} x^{127} \\
A_4(x) &= a_{512} + a_{513} x + a_{514} x^2 + \cdots + a_{639} x^{127} \\
A_5(x) &= a_{640} + a_{641} x + a_{642} x^2 + \cdots + a_{767} x^{127}
\end{aligned}
$$

Toom: decomposition

- Decompose $a(x) = a_0 + a_1x + a_2x^2 + \cdots + a_{767}x^{767}$ into
  $a(x,y) = A_0(x) + A_1(x)y + A_2(x)y^2 + A_3(x)y^3 + A_4(x)y^4 + A_5(x)y^5$
  where $y = x^{128}$ and

$$A_0(x) = a_0 \quad + a_1 \quad x + a_2 \quad x^2 + \cdots + a_{127}x^{127}$$
$$A_1(x) = a_{128} + a_{129}x + a_{130}x^2 + \cdots + a_{255}x^{127}$$
$$A_2(x) = a_{256} + a_{257}x + a_{258}x^2 + \cdots + a_{383}x^{127}$$
$$A_3(x) = a_{384} + a_{385}x + a_{386}x^2 + \cdots + a_{511}x^{127}$$
$$A_4(x) = a_{512} + a_{513}x + a_{514}x^2 + \cdots + a_{639}x^{127}$$
$$A_5(x) = a_{640} + a_{641}x + a_{642}x^2 + \cdots + a_{767}x^{127}$$

- Similarly for $b(x)$, then

$$ab = C_0 + C_1y + C_2y^2 + C_3y^3 + C_4y^4 + C_5y^5$$
$$C_6y^6 + C_7y^7 + C_8y^8 + C_9y^9 + C_{10}y^{10}$$

Toom: evaluation

$$
\begin{array}{rl}
0: & A_0 \cdot B_0 \\
1: & (A_0+A_1+A_2+A_3+A_4+A_5)\cdot(B_0+B_1+B_2+B_3+B_4+B_5) \\
-1: & (A_0-A_1+A_2-A_3+A_4-A_5)\cdot(B_0-B_1+B_2-B_3+B_4-B_5) \\
2: & (A_0+2A_1+2^2A_2+2^3A_3+2^4A_4+2^5A_5)\cdot(B_0+2B_1+2^2B_2+2^3B_3+2^4B_4+2^5B_5) \\
-2: & (A_0-2A_1+2^2A_2-2^3A_3+2^4A_4-2^5A_5)\cdot(B_0-2B_1+2^2B_2-2^3B_3+2^4B_4-2^5B_5) \\
3: & (A_0+3A_1+3^2A_2+3^3A_3+3^4A_4+3^5A_5)\cdot(B_0+3B_1+3^2B_2+3^3B_3+3^4B_4+3^5B_5) \\
-3: & (A_0-3A_1+3^2A_2-3^3A_3+3^4A_4-3^5A_5)\cdot(B_0-3B_1+3^2B_2-3^3B_3+3^4B_4-3^5B_5) \\
4: & (A_0+4A_1+4^2A_2+4^3A_3+4^4A_4+4^5A_5)\cdot(B_0+4B_1+4^2B_2+4^3B_3+4^4B_4+4^5B_5) \\
-4: & (A_0-4A_1+4^2A_2-4^3A_3+4^4A_4-4^5A_5)\cdot(B_0-4B_1+4^2B_2-4^3B_3+4^4B_4-4^5B_5) \\
5: & (A_0+5A_1+5^2A_2+5^3A_3+5^4A_4+5^5A_5)\cdot(B_0+5B_1+5^2B_2+5^3B_3+5^4B_4+5^5B_5) \\
\infty: & A_5 \cdot B_5
\end{array}
$$

Karatsuba

$$(F_0 + t^n F_1)(G_0 + t^n G_1) = (1 - t^n)(F_0 G_0 - t^n F_1 G_1) + t^n(F_0 + F_1)(G_0 + G_1)$$

Karatsuba

$$(F_0 + t^n F_1)(G_0 + t^n G_1) = (1 - t^n)(F_0 G_0 - t^n F_1 G_1) + t^n(F_0 + F_1)(G_0 + G_1)$$

- Level 1:

$$F_0 = f_0 + f_1 x + f_2 x^2 + \ldots + f_{63} x^{63}; \quad F_1 = f_{64} + f_{65} x + f_{66} x^2 + \ldots + f_{127} x^{63};$$
$$G_0 = g_0 + g_1 x + g_2 x^2 + \ldots + g_{63} x^{63}; \quad G_1 = g_{64} + g_{65} x + g_{66} x^2 + \ldots + g_{127} x^{63};$$

$$fg = (1 - x^{64})(F_0 G_0 - x^{64} F_1 G_1) + x^{64}(F_0 + F_1)(G_0 + G_1)$$

Karatsuba

$$(F_0 + t^n F_1)(G_0 + t^n G_1) = (1 - t^n)(F_0 G_0 - t^n F_1 G_1) + t^n(F_0 + F_1)(G_0 + G_1)$$

- Level 1:

$$F_0 = f_0 + f_1 x + f_2 x^2 + \ldots + f_{63} x^{63}; \quad F_1 = f_{64} + f_{65} x + f_{66} x^2 + \ldots + f_{127} x^{63};$$
$$G_0 = g_0 + g_1 x + g_2 x^2 + \ldots + g_{63} x^{63}; \quad G_1 = g_{64} + g_{65} x + g_{66} x^2 + \ldots + g_{127} x^{63};$$

$$fg = (1 - x^{64})(F_0 G_0 - x^{64} F_1 G_1) + x^{64}(F_0 + F_1)(G_0 + G_1)$$

- Level 2:

$$F_{00} = f_0 + f_1 x + f_2 x^2 + \ldots + f_{31} x^{31}; \quad F_{01} = f_{32} + f_{33} x + f_{34} x^2 + \ldots + f_{63} x^{31};$$
$$F_{10} = f_{64} + f_{65} x + f_{66} x^2 + \ldots + f_{95} x^{31}; \quad F_{11} = f_{96} + f_{97} x + f_{98} x^2 + \ldots + f_{127} x^{31};$$

Karatsuba

$$(F_0 + t^n F_1)(G_0 + t^n G_1) = (1 - t^n)(F_0 G_0 - t^n F_1 G_1) + t^n (F_0 + F_1)(G_0 + G_1)$$

- Level 1:

$$F_0 = f_0 + f_1 x + f_2 x^2 + \ldots + f_{63} x^{63}; \quad F_1 = f_{64} + f_{65} x + f_{66} x^2 + \ldots + f_{127} x^{63};$$
$$G_0 = g_0 + g_1 x + g_2 x^2 + \ldots + g_{63} x^{63}; \quad G_1 = g_{64} + g_{65} x + g_{66} x^2 + \ldots + g_{127} x^{63};$$

$$fg = (1 - x^{64})(F_0 G_0 - x^{64} F_1 G_1) + x^{64}(F_0 + F_1)(G_0 + G_1)$$

- Level 2:

$$F_{00} = f_0 + f_1 x + f_2 x^2 + \ldots + f_{31} x^{31}; \quad F_{01} = f_{32} + f_{33} x + f_{34} x^2 + \ldots + f_{63} x^{31};$$
$$F_{10} = f_{64} + f_{65} x + f_{66} x^2 + \ldots + f_{95} x^{31}; \quad F_{11} = f_{96} + f_{97} x + f_{98} x^2 + \ldots + f_{127} x^{31};$$

let $F_2 = F_0 + F_1 = F_{20} + x^{32} F_{21}$

$$F_{20} = (f_0 + f_{64}) + (f_1 + f_{65})x + (f_2 + f_{66})x^2 + \ldots + (f_{31} + f_{95})x^{31};$$
$$F_{21} = (f_{32} + f_{96}) + (f_{33} + f_{97})x + (f_{34} + f_{98})x^2 + \ldots + (f_{63} + f_{127})x^{31};$$

Karatsuba

$$(F_0 + t^n F_1)(G_0 + t^n G_1) = (1 - t^n)(F_0 G_0 - t^n F_1 G_1) + t^n(F_0 + F_1)(G_0 + G_1)$$

- Level 1:

$$F_0 = f_0 + f_1 x + f_2 x^2 + \ldots + f_{63} x^{63}; \quad F_1 = f_{64} + f_{65} x + f_{66} x^2 + \ldots + f_{127} x^{63};$$
$$G_0 = g_0 + g_1 x + g_2 x^2 + \ldots + g_{63} x^{63}; \quad G_1 = g_{64} + g_{65} x + g_{66} x^2 + \ldots + g_{127} x^{63};$$

$$fg = (1 - x^{64})(F_0 G_0 - x^{64} F_1 G_1) + x^{64}(F_0 + F_1)(G_0 + G_1)$$

- Level 2:

$$F_{00} = f_0 + f_1 x + f_2 x^2 + \ldots + f_{31} x^{31}; \quad F_{01} = f_{32} + f_{33} x + f_{34} x^2 + \ldots + f_{63} x^{31};$$
$$F_{10} = f_{64} + f_{65} x + f_{66} x^2 + \ldots + f_{95} x^{31}; \quad F_{11} = f_{96} + f_{97} x + f_{98} x^2 + \ldots + f_{127} x^{31};$$

let $F_2 = F_0 + F_1 = F_{20} + x^{32} F_{21}$

$$F_{20} = (f_0 + f_{64}) + (f_1 + f_{65})x + (f_2 + f_{66})x^2 + \ldots + (f_{31} + f_{95})x^{31};$$
$$F_{21} = (f_{32} + f_{96}) + (f_{33} + f_{97})x + (f_{34} + f_{98})x^2 + \ldots + (f_{63} + f_{127})x^{31};$$

$$F_0 G_0 = (1 - x^{32})(F_{00} G_{00} - x^{32} F_{01} G_{01}) + x^{32}(F_{00} + F_{01})(G_{00} + G_{01});$$
$$F_1 G_1 = (1 - x^{32})(F_{10} G_{10} - x^{32} F_{11} G_{11}) + x^{32}(F_{10} + F_{11})(G_{10} + G_{11});$$
$$F_2 G_2 = (1 - x^{32})(F_{20} G_{20} - x^{32} F_{21} G_{21}) + x^{32}(F_{20} + F_{21})(G_{20} + G_{21});$$

Karatsuba

$$(F_0 + t^n F_1)(G_0 + t^n G_1) = (1 - t^n)(F_0 G_0 - t^n F_1 G_1) + t^n(F_0 + F_1)(G_0 + G_1)$$

- Level 1:

$$F_0 = f_0 + f_1 x + f_2 x^2 + \ldots + f_{63} x^{63}; \quad F_1 = f_{64} + f_{65} x + f_{66} x^2 + \ldots + f_{127} x^{63};$$
$$G_0 = g_0 + g_1 x + g_2 x^2 + \ldots + g_{63} x^{63}; \quad G_1 = g_{64} + g_{65} x + g_{66} x^2 + \ldots + g_{127} x^{63};$$

$$fg = (1 - x^{64})(F_0 G_0 - x^{64} F_1 G_1) + x^{64}(F_0 + F_1)(G_0 + G_1)$$

- Level 2:

$$F_{00} = f_0 + f_1 x + f_2 x^2 + \ldots + f_{31} x^{31}; \quad F_{01} = f_{32} + f_{33} x + f_{34} x^2 + \ldots + f_{63} x^{31};$$
$$F_{10} = f_{64} + f_{65} x + f_{66} x^2 + \ldots + f_{95} x^{31}; \quad F_{11} = f_{96} + f_{97} x + f_{98} x^2 + \ldots + f_{127} x^{31};$$

let $F_2 = F_0 + F_1 = F_{20} + x^{32} F_{21}$

$$F_{20} = (f_0 + f_{64}) + (f_1 + f_{65})x + (f_2 + f_{66})x^2 + \ldots + (f_{31} + f_{95})x^{31};$$
$$F_{21} = (f_{32} + f_{96}) + (f_{33} + f_{97})x + (f_{34} + f_{98})x^2 + \ldots + (f_{63} + f_{127})x^{31};$$

$$F_0 G_0 = (1 - x^{32})(F_{00} G_{00} - x^{32} F_{01} G_{01}) + x^{32}(F_{00} + F_{01})(G_{00} + G_{01});$$
$$F_1 G_1 = (1 - x^{32})(F_{10} G_{10} - x^{32} F_{11} G_{11}) + x^{32}(F_{10} + F_{11})(G_{10} + G_{11});$$
$$F_2 G_2 = (1 - x^{32})(F_{20} G_{20} - x^{32} F_{21} G_{21}) + x^{32}(F_{20} + F_{21})(G_{20} + G_{21});$$

- Similarly for level 3, level 4 and level 5

Schoolbook

- Lowest-level multiplication of $4n \times 4n$
  e.g., $F_{00000} G_{00000}$

$$
\begin{aligned}
h_0 &= f_0 g_0 \\
h_1 &= f_0 g_1 + f_1 g_0 \\
h_2 &= f_0 g_2 + f_1 g_1 + f_2 g_0 \\
h_3 &= f_0 g_3 + f_1 g_2 + f_2 g_1 + f_3 g_0 \\
h_4 &= f_1 g_3 + f_2 g_2 + f_3 g_1 \\
h_5 &= f_2 g_3 + f_3 g_2 \\
h_6 &= f_3 g_3
\end{aligned}
$$

Schoolbook

- Lowest-level multiplication of $4n \times 4n$
  e.g., $F_{00000} G_{00000}$

$$
\begin{aligned}
h_0 &= f_0 g_0 \\
h_1 &= f_0 g_1 + f_1 g_0 \\
h_2 &= f_0 g_2 + f_1 g_1 + f_2 g_0 \\
h_3 &= f_0 g_3 + f_1 g_2 + f_2 g_1 + f_3 g_0 \\
h_4 &= f_1 g_3 + f_2 g_2 + f_3 g_1 \\
h_5 &= f_2 g_3 + f_3 g_2 \\
h_6 &= f_3 g_3
\end{aligned}
$$

- Using 5-level Karatsuba, there are $3^5 = 243$ of $4n \times 4n$ for one 128×128

Haswell floating-point vector unit

- 256-bit 4-way vectorization

Haswell floating-point vector unit

- 256-bit 4-way vectorization
- Two vectorized multiply-add units (port 0 and port 1)

Haswell floating-point vector unit

- 256-bit 4-way vectorization
- Two vectorized multiply-add units (port 0 and port 1)
  Each cycle produces 8 independent multiply-add $ab + c$
  for 64-bit double-precision inputs a, b, c

Haswell floating-point vector unit

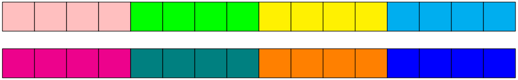
- 256-bit 4-way vectorization
- Two vectorized multiply-add units (port 0 and port 1)
  Each cycle produces 8 independent multiply-add $ab + c$
  for 64-bit double-precision inputs a, b, c
- One vectorized addition unit (port 1)

Haswell floating-point vector unit

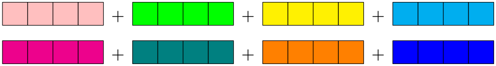
- 256-bit 4-way vectorization
- Two vectorized multiply-add units (port 0 and port 1)
  Each cycle produces 8 independent multiply-add $ab + c$
  for 64-bit double-precision inputs a, b, c
- One vectorized addition unit (port 1)
  Each cycle produces 4 independent additions $a + b$
  for 64-bit double-precision input a, b

Vectorization

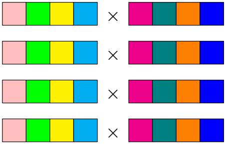
$f =$
$g =$



- Toom & Karatsuba
  - vectorize inside each limb



- Schoolbook
  - transpose inputs
  - vectorize *across* independent multiplications



Performance

- Theoretical lower bound
  - 0.125 cycles per floating-point multiplication
  - 0.250 cycles per floating-point addition and shift
  - permutation fully interleavable

	mul	con mult	add	shift	total
op.	42768	9700	98548	6385	157401
cycles	5346	1213	24637	1597	32793

- Actual implementation
  - 46784 cycles
  - possibly due to dependency, latency, scheduling issues

Current projects

- PRF from module lattices
- Module-NTRU in QROM
- Ring-signature from module lattices
- Middle product and integer LWE