Spectral Modular Arithmetic

Çetin Kaya Koç

Oregon State University
&
Istanbul Commerce University
Information Security Research Center

koc@krip.to
http://cryptocode.net

IPAM Workshop
December 5, 2006

Outline

- Exponentiation and Multiplication
- Montgomery Multiplication
- Spectral Arithmetic
- Spectral Modular Multiplication
- Mersenne and Fermat Rings
- Architectures and Performance Analysis
- Conclusions and Future Work

Exponentiation

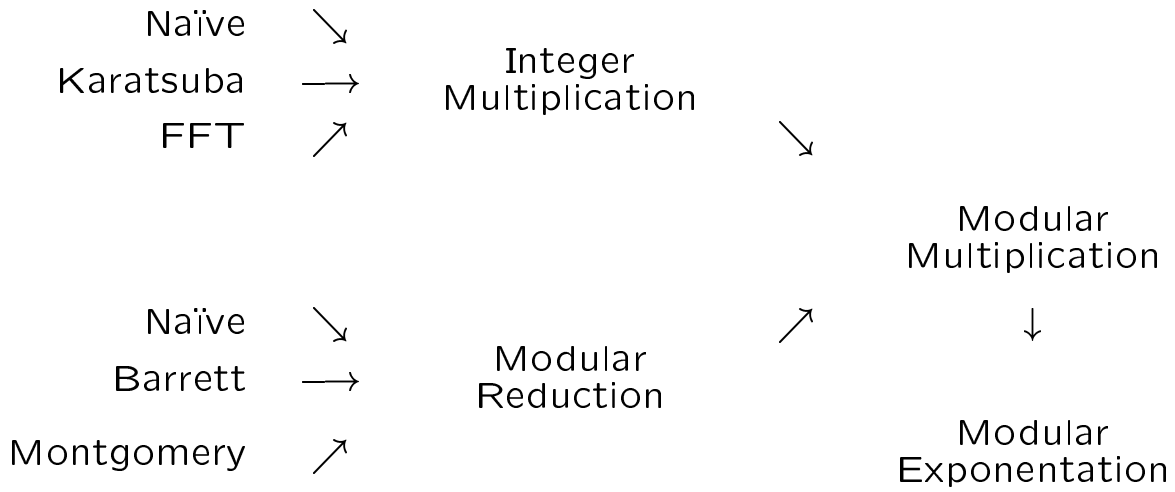
- The exponentiation operation is the most common operation in public-key cryptography:

$$c = m^e \pmod{n}$$

- Use multiply-reduce method: break the exponentiation into modular multiplications by scanning the exponent

- Optimizations:
  - Reduce the number of multiplications
  - Reduce multiplication complexity

Multiplication



Montgomery Multiplication

- Best algorithm so far (since 1985)

- Replaces division by n operation with division by $r = 2^k$

- Assuming n is a k-bit odd integer, i.e., $2^{k-1} < n < 2^k$; we assign $r = 2^k$ and map integers $a \in [0, n-1]$ to integers $\bar{a} \in [0, n-1]$ using the one-to-one mapping

$$\bar{a} = a \cdot r \pmod{n}$$

  $\bar{a}$ is called the n-residue of a

Montgomery Multiplication

- The Montgomery product of two n-residues is defined as

$$\text{MonPro}(\bar{a}, \bar{b}) = \bar{a} \cdot \bar{b} \cdot r^{-1} \pmod{n}$$

where r^{-1} is the inverse of r modulo n

- If $c = a \cdot b \pmod{n}$, then $\bar{c} = \text{MonPro}(\bar{a}, \bar{b})$

$$\begin{aligned} \bar{c} &= a \cdot b \cdot r^{-1} \pmod{n} \\ &= (a \cdot r) \cdot (b \cdot r) \cdot r^{-1} \pmod{n} \\ &= \text{MonPro}(\bar{a}, \bar{b}) \end{aligned}$$

Montgomery Multiplication

- For any a, we have $\bar{a} = \mathsf{MonPro}(a, r^2)$

$$\begin{aligned} \bar{a} &= a \cdot r \pmod{n} \\ &= a \cdot r^2 \cdot r^{-1} \pmod{n} \\ &= \mathsf{MonPro}(a, r^2) \end{aligned}$$

- For any $\bar{c}$, we have $c = \mathsf{MonPro}(\bar{c}, 1)$

$$\begin{aligned} c &= (c \cdot r) \cdot 1 \cdot r^{-1} \pmod{n} \\ &= \mathsf{MonPro}(\bar{c}, 1) \end{aligned}$$

Montgomery Product

- The pre-processing operations
  - computation of n' and $r^2 \pmod{n}$
  - conversion from ordinary to n-residue
  - conversion from n-residue to ordinary

  are time-consuming

- Montgomery's method is not suitable for a single modular multiplication

- However, it is very suitable for modular exponentiation

Montgomery Exponentiation

function ModExp(m, e, n) { n is odd }
1: Compute n' and $r^2 \pmod{n}$
2: $\bar{m} :=$ MonPro(m, r^2)
3: $\bar{c} :=$ MonPro$(1, r^2)$
4: **for** $i = h - 1$ **down to** 0 **do**
5: $\bar{c} :=$ MonPro$(\bar{c}, \bar{c})$
6: **if** $e_i = 1$ **then** $\bar{c} :=$ MonPro$(\bar{c}, \bar{m})$
7: $c :=$ MonPro$(\bar{c}, 1)$
8: **return** c

Here we are using the binary method of exponentiation, however, any exponentiation algorithm can be used

Original Montgomery Algorithm

To compute MonPro$(\bar{a}, \bar{b}) = \bar{a} \cdot \bar{b} \cdot r^{-1} \pmod{n}$, we need an additional quantity n' such that $r \cdot r^{-1} - n \cdot n' = 1$ which is computed by the extended Euclidean algorithm

function MonPro$(\bar{a}, \bar{b})$
1: $t := \bar{a} \cdot \bar{b}$
2: $m := t \cdot n' \bmod r$
3: $u := (t + m \cdot n)/r$
4: **if** $u \geqslant n$ **then return** $u - n$
5: **else return** u

It requires only mod r arithmetic ($r = 2^k$)

Binary CIOS Algorithm

$$
\begin{aligned}
u &= \text{MonPro}(a, b) \\
&= a \cdot b \cdot r^{-1} \pmod{n} \\
&= a \cdot b \cdot 2^{-k} \pmod{n} \\
&= (\sum_{i=0}^{k-1} a_i 2^i) \cdot b \cdot 2^{-k} \pmod{n} \\
&= (\sum_{i=0}^{k-1} a_i 2^{i-k}) \cdot b \pmod{n} \\
&= (a_0 2^{-k} + a_1 2^{-k+1} + \cdots + a_{k-1} 2^{-1}) \cdot b \bmod n
\end{aligned}
$$

1: $u := 0$
2: **for** $i = 0$ **to** $k - 1$
3: $\quad u := u + a_i b$
4: $\quad u := u + u_0 n$
5: $\quad u := u/2$

Word-Level CIOS Algorithm

1: $u := 0$
2: **for** $i = 0$ **to** $s - 1$
3: $\quad u := u + a_i \cdot b$
4: $\quad m := u_0 \cdot (-n_0^{-1}) \pmod{2^w}$
5: $\quad u := u + m \cdot n$
6: $\quad u := u/2^w$

Here $k = sw$ and w is the wordsize

Since $r \cdot r^{-1} - n' \cdot n = 1$ and $r = 2^k$, we have

$$\begin{aligned} 2^{sw} \cdot 2^{sw} - n' \cdot n &= 1 \pmod{2^w} \\ -n'_0 \cdot n_0 &= 1 \pmod{2^w} \end{aligned}$$

Therefore $(-n_0^{-1}) = n'_0$

We only need to compute the least significant word of n', not the whole number

Our Motivation

Invent better modular multiplication algorithms

- Utilize finite ring and field arithmetic (avoid real or complex arithmetic, i.e., floating-point)

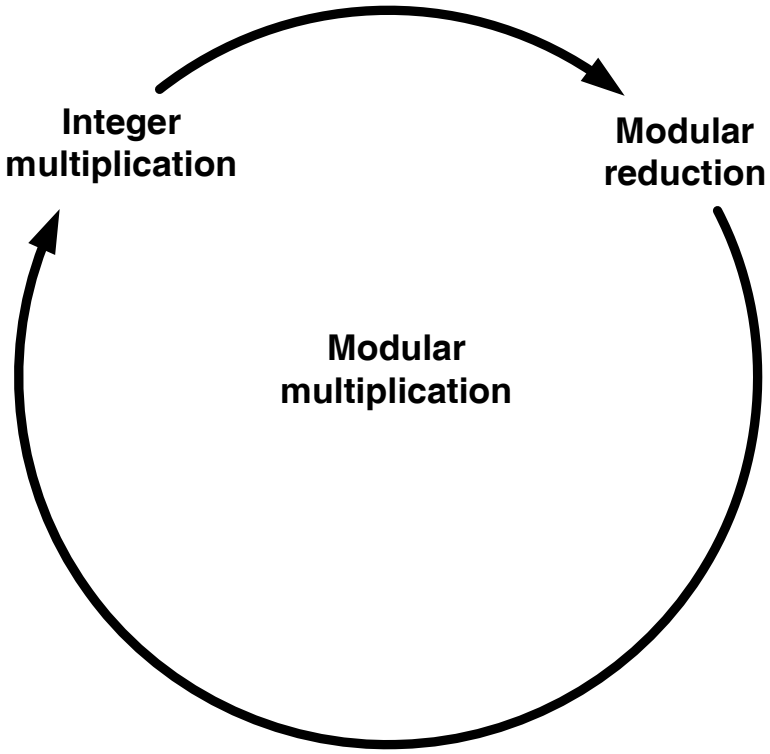
- Bring down the break-even point of efficiency for FFT-based multiplication

- Utilize the DFT and Montgomery properties to perform modular reduction

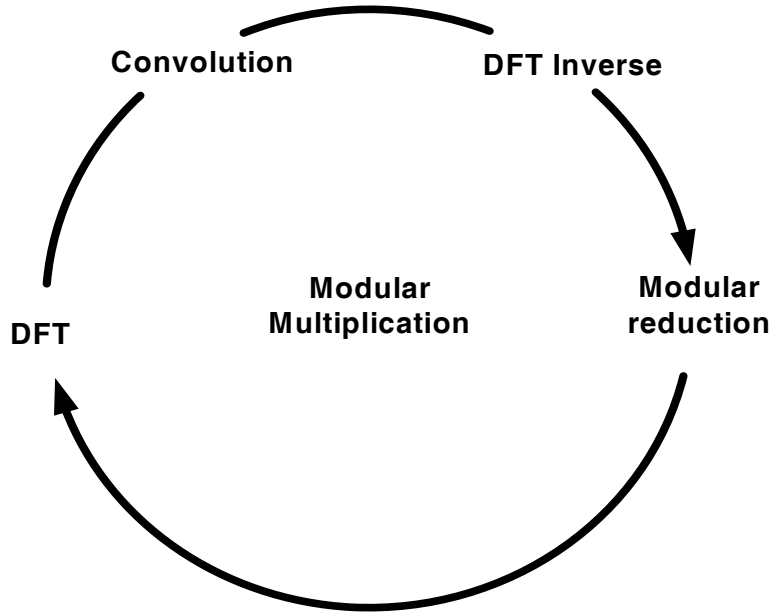
We have been thinking about this (on and off) since 1994 (*RSA Implementation*, Technical Report, RSA Labs)

We have alsoe been following other people's work (RNS, CRT, Polynomial arithmetic, etc.)

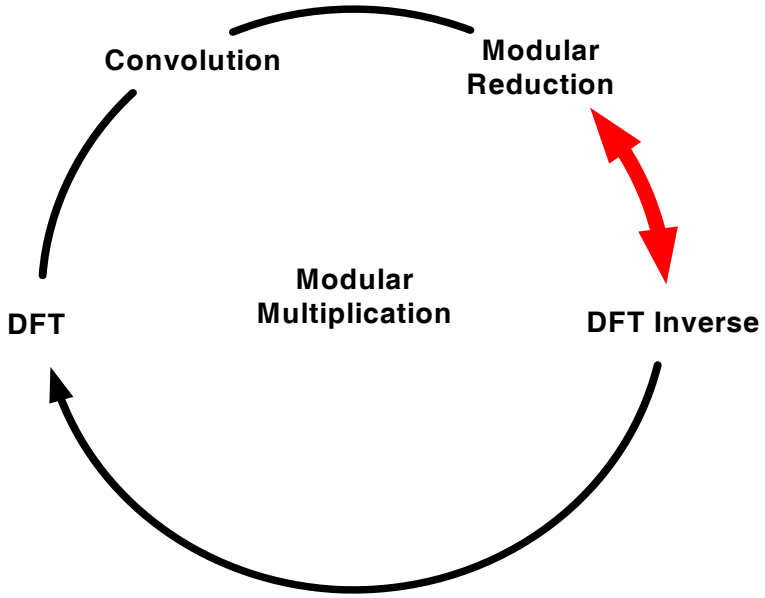
Our Motivation



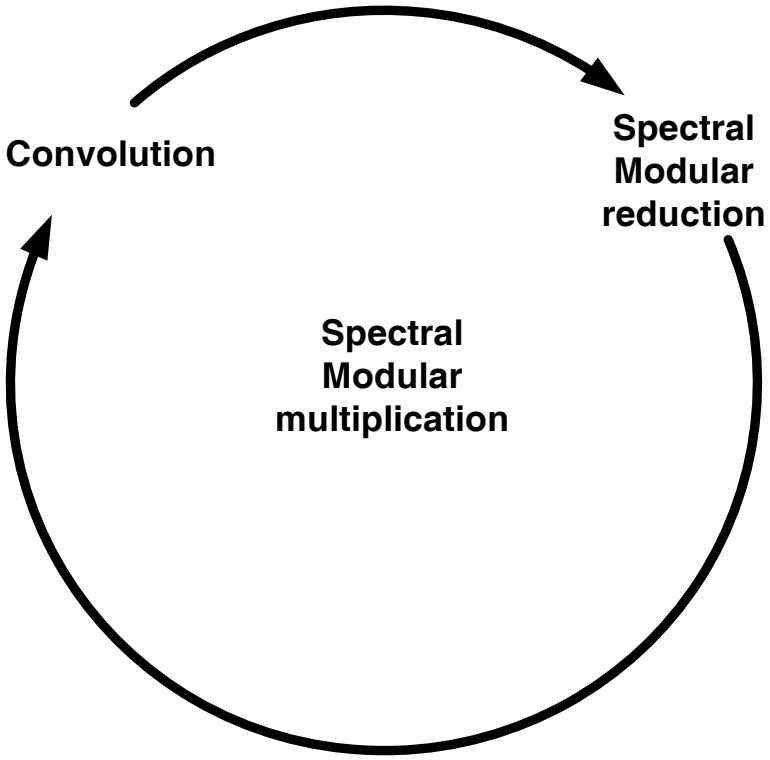
Our Motivation



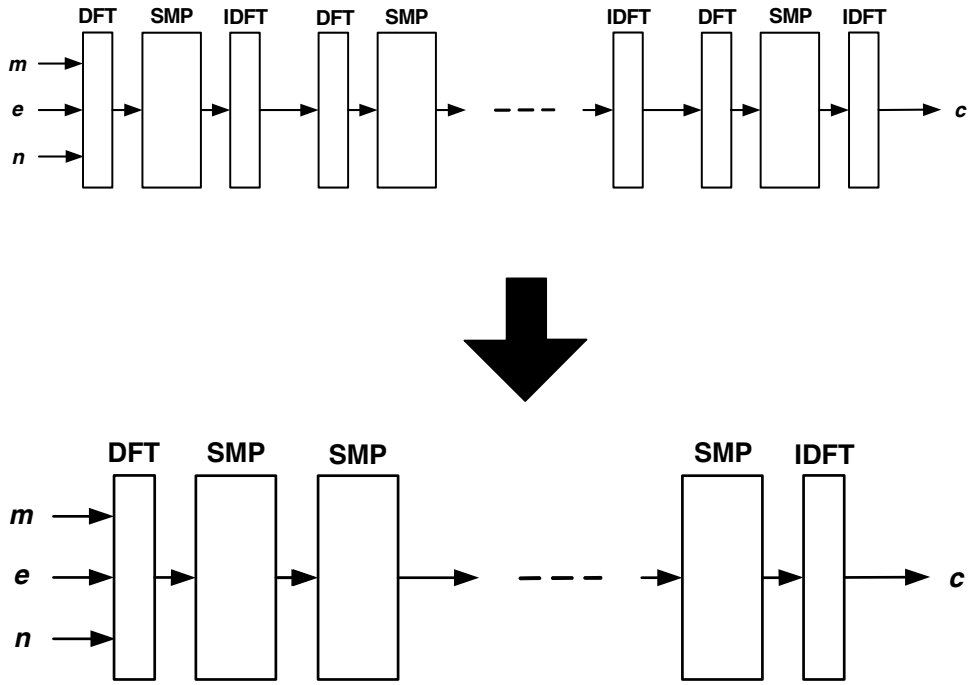
Our Motivation



Our Motivation



Another Look at the Simplification



DFT over a Finite Ring

Let ω be a primitive d-th root of unity in $\mathbb{Z}_q$ and, let $x(t)$ and $X(t)$ be polynomials of degree $d-1$ having entries in $\mathbb{Z}_q$. The DFT map over $\mathbb{Z}_q$ is an invertible set map sending $x(t)$ to $X(t)$ given by

$$X_i = DFT_d^{\omega}(x(t)) := \sum_{j=0}^{d-1} x_j \omega^{ij} \bmod q,$$

with the inverse

$$x_i = IDFT_d^{\omega}(X(t)) := d^{-1} \cdot \sum_{j=0}^{d-1} X_j \omega^{-ij} \bmod q,$$

for $i, j = 0, 1, \ldots, d-1$. Moreover, we write

$$x(t) \overset{\text{DFT}}{\longleftrightarrow} X(t)$$

and say $x(t)$ and $X(t)$ are transform pairs; $x(t)$ is called a **time polynomial** and sometimes $X(t)$ is named as the **spectrum** of $x(t)$.

DFT over a Finite Ring

- (Convention) In the literature, DFT over a finite ring spectrum is also called as Number Theoretical Transform (NTT)

- (Existence) In order to have a DFT map over $\mathbb{Z}_q$:
  - The multiplicative inverse of DFT length d must exist in $\mathbb{Z}_q$ which requires that $\gcd(d, q) = 1$.
  - d has to divide $p - 1$ for every prime p divisor of q

DFT over a Finite Ring

- **Theorem 1:** An NTT exist in R if and only if each quotient field R/M (where M is maximal ideal) possesses a primitive root of unity

- (Efficiency) In order to have simple arithmetic

  - q should be chosen as
    a Mersenne ($q = 2^v - 1$) or
    a Fermat number $q = 2^v + 1$

  - The principal root of unity ω should be selected as a power of 2 to simplify the multiplications with roots of unity

Properties of DFT

- Under certain conditions, the Fourier transform preserves some properties of the time sequences, e.g., linearity and convolution.

- The existence conditions of these properties differ when working in finite ring spectrums

- Let ϕ and Φ be operations on time and spectral domains respectively. We write

$$\phi \overset{\text{DFT}}{\longleftrightarrow} \Phi$$

and say ϕ and Φ are transform pairs on $x(t)$ and sometimes declare that **the map** DFT_d^ω **respects the operation** ϕ on point $x(t)$ if following equation is satisfied

$$\phi(x(t)) = IDFT_d^\omega \circ \Phi \circ DFT_d^\omega(x(t))$$

Time-Frequency Dictionary

- **Time and frequency shifts** correspond to circular shifts Let

$$x(t) = x_0 + x_1 t + \ldots + x_{d-1} t^{d-1}$$

and

$$X(t) = X_0 + X_1 t + \ldots + X_{d-1} t^{d-1}$$

be a transform pair.

The *one-term right circular shift* is defined as $x(t) \circlearrowleft 1$

$$\begin{array}{c} x_1 + x_2 t + \ldots + x_{d-2} t^{d-1} + x_0 t^{d-1} \\ \updownarrow \text{DFT} \\ X(t) \odot \Gamma(t) \end{array}$$

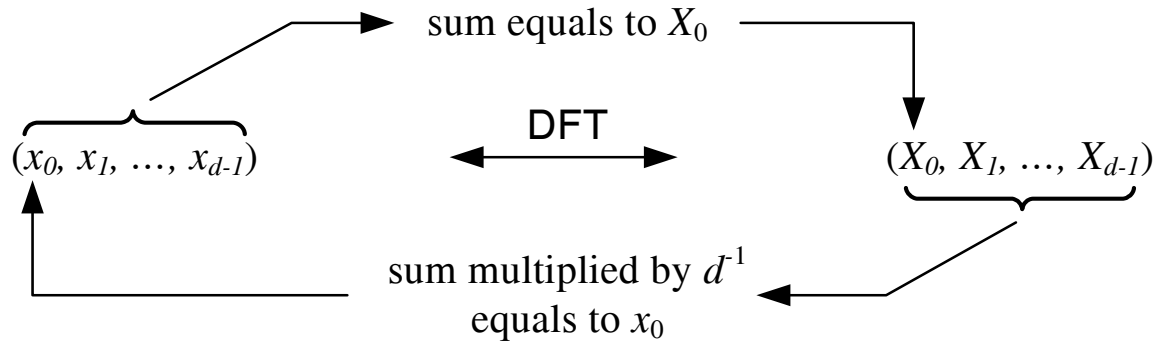
where $\odot$ stands for component-wise multiplication and

$$\Gamma(t) = 1 + \omega^{-1} t + \ldots + \omega^{-(d-1)} t^{d-1}$$

Time-Frequency Dictionary

- **Sum of sequence and first value:** The sum of the coefficients of a time polynomial equals to the zeroth coefficient of its spectral polynomial. Conversely the sum of the spectrum coefficients equals to d^{-1} times the zeroth coefficient of the time polynomial

$$x_0 = d^{-1} \cdot \sum_{i=0}^{d-1} X_i \omega^{-i} \quad \text{and} \quad X_0 = \sum_{i=0}^{d-1} x_i \omega^i$$



Time-frequency dictionary

- **Left and right logical shifts:** By using the previous properties, it is possible to perform logical left and right digit shifts $x(t) \ll 1$ as follows:

$$\begin{array}{c} (x(t) - x_0)/t = x_1 + \ldots + x_{d-1} t^{d-2} \\ \updownarrow \text{DFT} \\ (X(t) - x_0(t)) \odot \Gamma(t) \end{array}$$

  where

$$x_0(t) = x_0 + x_0 t + x_0 t^2 + \ldots + x_0 t^{d-1}$$

- The right shifts are similar, where one then uses the

$$\Omega(t) = 1 + \omega^1 t + \ldots + \omega^{(d-1)} t^{d-1}$$

  polynomial instead of $\Gamma(t)$

Time Simulations and Spectral Algorithms

We define our algorithms as the images of some time algorithms:

TIME DOMAIN	**FREQENCY DOMAIN**
Algorithm 1	**Algorithm 2**
$x(t), y(t) \in \mathbb{Z}[t]/(t^d - 1)$	$X(t), Y(t) \in \mathbb{Z}[t]/(t^d - 1)$
$z(t) = x(t)(5y(t) + x(t)) - 3x(t)$	$Z(t) = X(t) \odot (5Y(t) + X(t)) - 3X(t)$
1: $z(t) := x(t) + 5y(t)$	1: $Z(t) := X(t) + 5Y(t)$
2: $z(t) := x(t) \cdot z(t)$	2: $Z(t) := X(t) \odot Z(t)$
3: $z(t) := z(t) - 3x(t)$	3: $Z(t) := Z(t) - 3X(t)$
4: **return** $z(t)$	4: **return** $z(t)$

If these two algorithms produce agreeing results we write

$$\text{Algorithm 1} \overset{\text{DFT}}{\longleftrightarrow} \text{Algorithm 2}$$

On Translation to Spectrum

- Time simulations can be translated into spectrum using the properties of DFT

- In order to have valid a spectral algorithm, algorithms in two domains must agree, which is denoted by

  $$\text{Time Simulation} \overset{\text{DFT}}{\longleftrightarrow} \text{Spectral Algorithm}$$

On Translation to Spectrum

- In other words
  - the inputs of time simulation and spectral algorithm must agree
  - intermediate results must agree
  - actual outputs must agree

- In a finite ring spectrum, if the magnitudes of the time coefficients exceed q, an overflow occurs, which prdoduces misleading outputs

- A boundary (overflow) analysis is needed for finite ring spectrum

Time Simulation of SMP

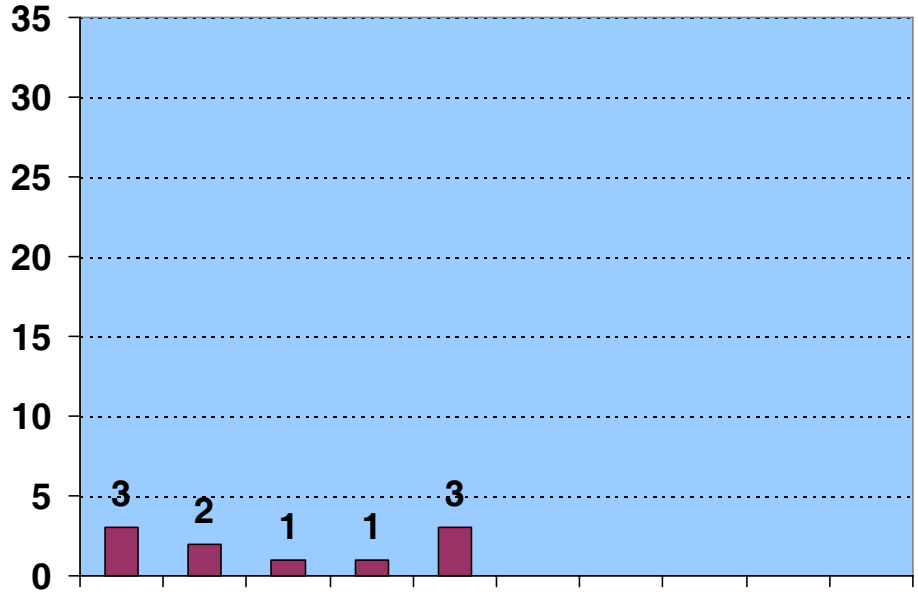
Algorithm 1.
Time Simulation of Spectral Modular Product

Suppose x, y, n *are positive integers such that* $x, y < n$. *Let* $x(t), y(t)$ *and* $n(t)$ *be polynomials having the radix-*b *representation of integers* x, y *and a multiple of modulus* n *with* $n_0 = 1$ *respectively.*

Input: $x(t), y(t)$ *and* $n(t)$.
Output: $z(t) \equiv y(t) \cdot x(t) \cdot t^{-d} \bmod n(t)$.

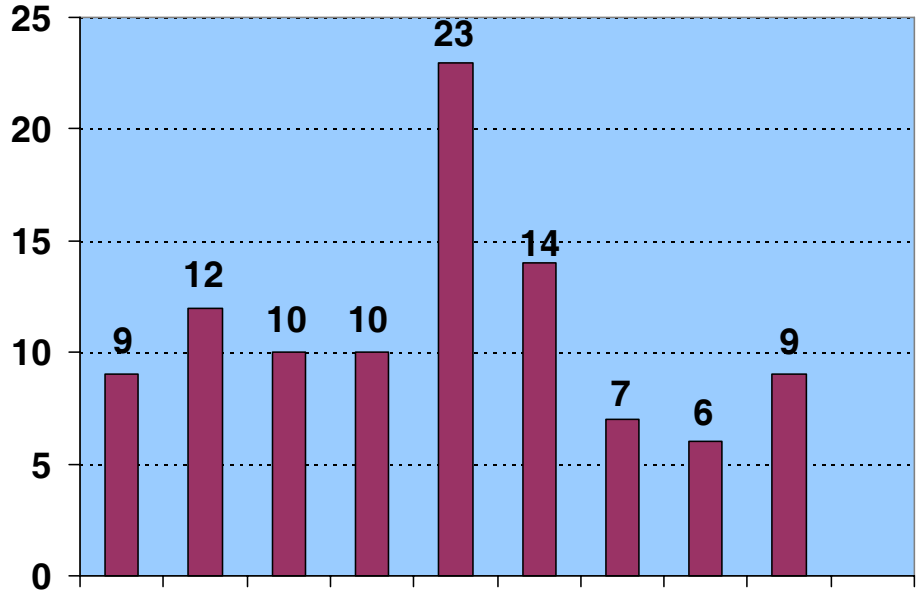
1: $z(t) := x(t)y(t)$
2: $\alpha := 0$
3: **for** $i = 0$ **to** $d - 1$
4: $\quad \beta := -(z_0 + \alpha) \bmod b$
5: $\quad \alpha := (z_0 + \alpha + \beta)$ **div** b
6: $\quad z(t) := z(t) + \beta \cdot n(t) \bmod q$
7: $\quad z(t) := (z(t) - z_0)/t$
8: **end for**
9: $z(t) := z(t) + \alpha(t)$
10: **return** $z(t)$

A Time Simulation for SMP



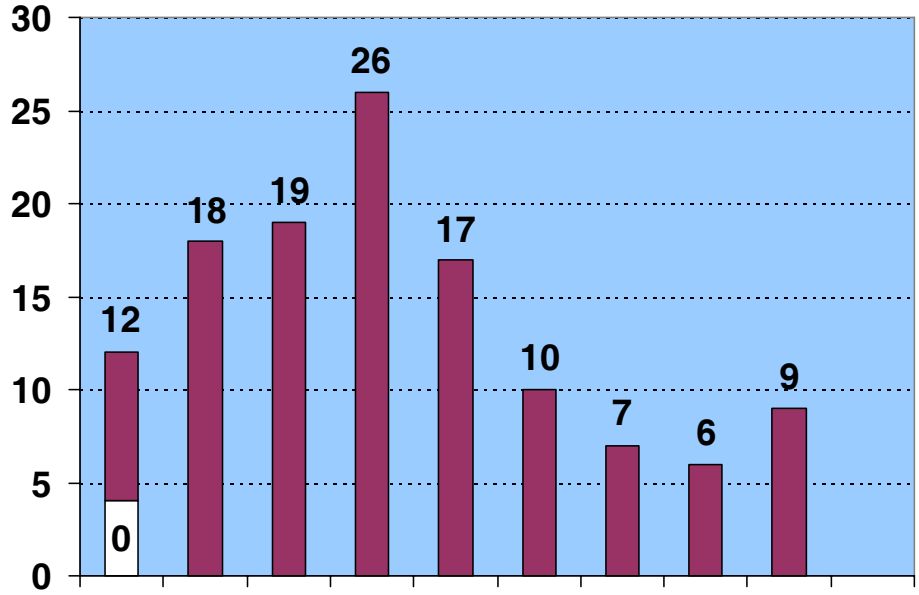
Signal $x(t)$ representing $859 = x(4)$ in base 4

A Time Simulation for SMP



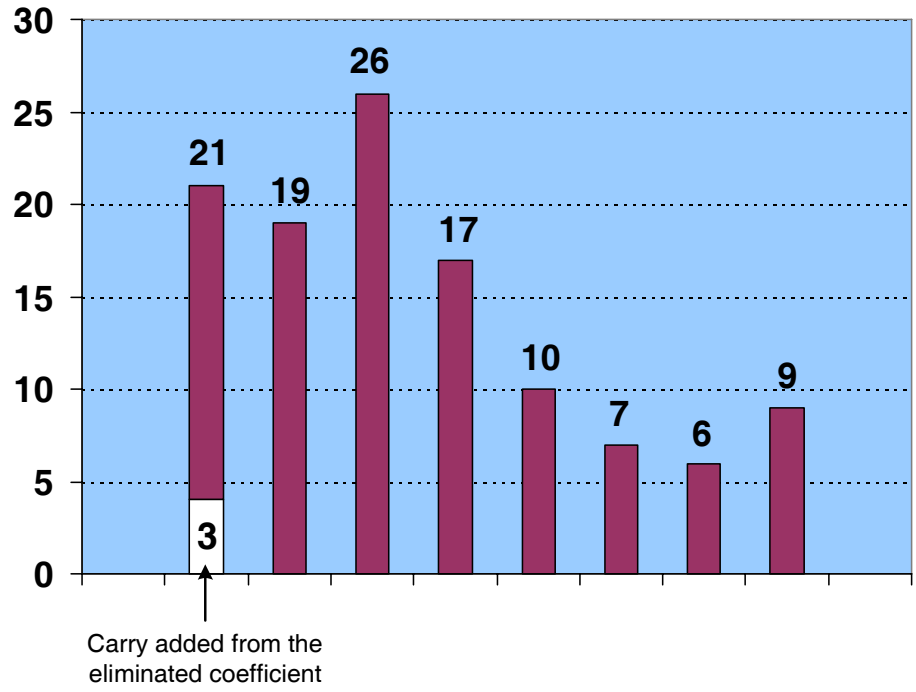
Convolving $x(t)$ with itself, we find $x^2(4) = 859^2 = 737881$

A Time Simulation for SMP



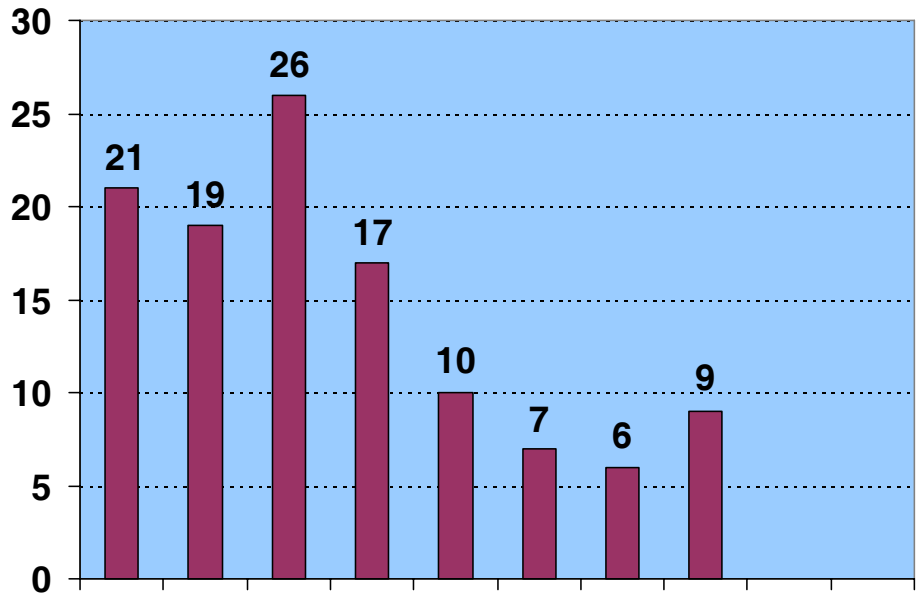
Assuming modulus $m = 1 + 2t + 3t^2 + t^4 + t^5$, we add $3m$ to the sum to anhilate the least significant b bits of the least digit

A Time Simulation for SMP



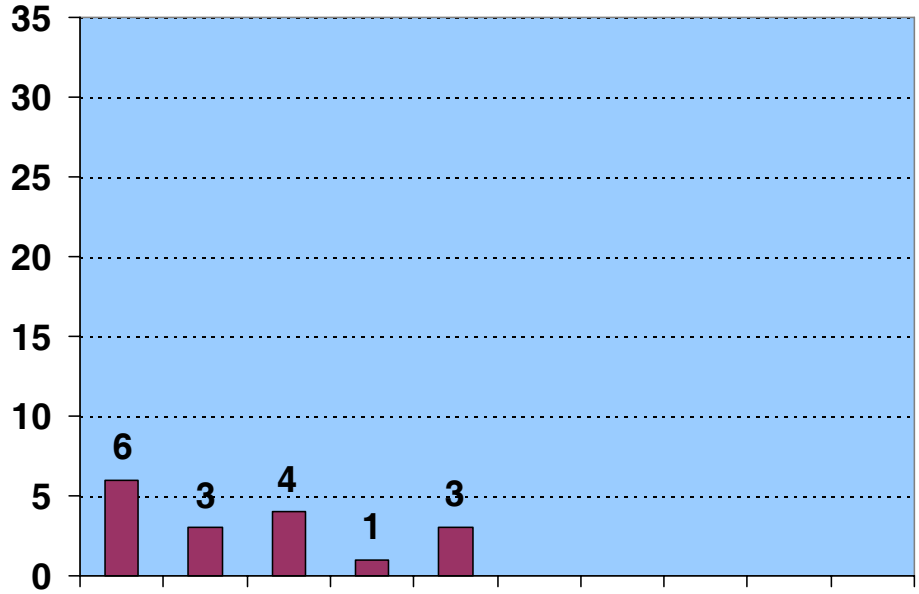
Carry goes to the next digit

A Time Simulation for SMP



and then we shift the digits

A Time Simulation for SMP



After 9 iterations, we find 914 which is

$$914 \equiv 859^2 \cdot 4^{-9} \pmod{1337}$$

SMP Algorithm

Algorithm 2.
Spectral Modular Product

Suppose that there exist a d-point DFT map for some principal root of unity ω in $\mathbb{Z}_q$, and $X(t), Y(t)$ and $N(t)$ are transform pairs of $x(t), y(t)$ and $n(t)$ respectively. We assume $x(t), y(t)$ and $n(t)$ are polynomials defined as in the time simulation.

procedure *SMP*$(X(t), Y(t), N(t))$

1: $Z(t) := X(t) \odot Y(t)$
2: $\alpha := 0$
3: **for** $i = 0$ **to** $d - 1$
4: $z_0 := d^{-1} \cdot (Z_0 + Z_1 + \ldots + Z_d) \bmod q$
5: $\beta := -(z_0 + \alpha)$ *mod* b
6: $\alpha := (z_0 + \alpha + \beta)/b$
7: $Z(t) := Z(t) + \beta \cdot N(t) \bmod q$
8: $Z(t) := Z(t) - (z_0 + \beta)(t) \bmod q$
9: $Z(t) := Z(t) \odot \Gamma(t) \bmod q$
10: **end for**
11: $Z(t) := Z(t) + \alpha(t)$
12: **return** $Z(t)$

Spectral Modular Exponentiation

Algorithm 3.
Spectral Modular Exponentiation (SME)

Suppose that there exist a d-point DFT map and $n(b)$ is a multiple of modulus n where $n_0 = 1$, deg$(n(t)) = s-1$, $s = \lceil d/2 \rceil$, gcd$(b, n) = 1$ and $b > 0$.

Input: *A modulus $n > 0$ and $m, e < n$*
Output: $c \equiv m^e \mod n$

1: Compute $\lambda(t)$ such that $\lambda = b^{2d} \bmod n$.
2: $N(t) :=$ *DFT*$(n(t))$
3: $\Lambda(t) :=$ *DFT*$(\lambda(t))$
4: $M(t) :=$ *DFT*$(m(t))$
5: $M'(t) :=$ *SMP*$(M(t), \Lambda(t), N(t))$
6: $C(t) :=$ *SMP*$(1(t), \Lambda(t), N(t))$
7: **for** $i = j - 2$ **downto** *0*
8: $\quad C(t) :=$ *SMP*$(C(t), C(t), N(t))$
9: $\quad$ **if** $e_i = 1$ **then** $C(t) :=$ *SMP*$(C(t), M'(t), N(t))$
10 $C(t) :=$ *SMP*$(C(t), 1(t), N(t))$
11 $c(t) :=$ *IDFT*$(C(t))$
12 **return** $c(b)$

An Overflow Example

Consider the integer ring $\mathbb{Z}_{31}$; the element 2 is a principal 5th root of unity and 5^{-1} exists. Observe also that there exists a 5-point NTT over ring $\mathbb{Z}_{31}$

DFT can be computed by a matrix multiplication where the transformation matrix $\mathcal{T}$ is given by

$$\mathcal{T} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 2^2 & 2^3 & 2^4 \\ 1 & 2^2 & 2^4 & 2^6 & 2^8 \\ 1 & 2^3 & 2^6 & 2^9 & 2^{12} \\ 1 & 2^4 & 2^8 & 2^{12} & 2^{16} \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 4 & 8 & 16 \\ 1 & 4 & 16 & 2 & 8 \\ 1 & 8 & 2 & 16 & 4 \\ 1 & 16 & 8 & 4 & 2 \end{bmatrix}$$

Now, consider the sequences $x = (1, 2, 3, 0, 0)$ and $y = (2, 3, 3, 0, 0)$

An Overflow Example

The transforms of x and y can be computed as

$$\begin{aligned} X &= \mathcal{T}x^T = (6, 17, 26, 23, 26) \pmod{31} \\ Y &= \mathcal{T}y^T = (8, 20, 0, 1, 12) \pmod{31} \end{aligned}$$

At this point, we perform point-wise multiplication (using convolution property), and then take the inverse transform

$$\begin{aligned} Z &= X \odot Y = (17, 30, 0, 23, 2) \pmod{31} \\ z &= \mathcal{T}^{-1}(17, 30, 0, 23, 2)^T = (2, 7, 15, 15, 9) \end{aligned}$$

which is the convolution of the signals x and y

If $x = (10, 5, 1, 0, 0)$ is taken, by the same procedure one gets $z = (20, 9, 20, 24, 9)$ which is a wrong answer caused by an overflow

Notice that z is equivalent to the correct result $(20, 40, 51, 24, 9) \pmod{31}$

Boundary analysis for SME

Let there exist a d-point DFT over $\mathbb{Z}_q$. For $s = \lceil d/2 \rceil$ and b representing the radix, we have proven the following:

Theorem 2. *SME computes a modular reduction,* $c \equiv m^e \bmod n$ *if the parameters* b, q *and* s *satisfies the following inequality*

$$(b^2 + b)^2 B(s) + b^2 s < q$$

where

$$B(s) = \frac{-2s^3}{3} - s^2 + 2s^2 r_1 - \frac{s}{3} + \frac{r_1}{3} + 2sr_1$$

for

$$r_1 = -2 + \frac{1}{3}\sqrt{3 + 18s^2 + 18s}$$

Suitable Spectrum for SME

- Complex spectrum is not appropriate
- A finite ring spectrum for which q is of the form $2^v \pm 1$ is the most suitable choice
- The rings having $2^v - 1$ elements are called the *Mersenne rings*: Arithmetic corresponds to the one's complement arithmetic
- While the rings with $q = 2^v + 1$ are called the *Fermat rings*: Arithmetic is simple if a diminished-1 representation is used.
- Principle root of unity should be taken as a power of 2
- Short transform sizes are the biggest concern for both ringss

Mersenne Number Transforms

The NTT over the rings with $q = 2^v - 1$ is called the Mersenne Number transform

Some nice MNT parameters for $2^{16} < q < 2^{80}$

ring $\mathbb{Z}_q$	prime factors	(ω, MNT length)	
$2^{17} - 1$	131071	$(2, 17)$	$(-2, 34)$
$2^{19} - 1$	524287	$(2, 19)$	$(-2, 38)$
$2^{23} - 1$	$47 \cdot 178481$	$(2, 23)$	$(-2, 46)$
$2^{29} - 1$	$233 \cdot 1103 \cdot 2089$	$(2, 29)$	$(-2, 58)$
$2^{31} - 1$	2147483647	$(2, 31)$	$(-2, 62)$
$2^{37} - 1$	$223 \cdot 616318177$	$(2, 37)$	$(-2, 74)$
$2^{41} - 1$	$13367 \cdot 164511353$	$(2, 41)$	$(-2, 82)$
$2^{43} - 1$	$431 \cdot 9719 \cdot 2099863$	$(2, 43)$	$(-2, 86)$
$2^{79} - 1$	$2687 \cdot 202029703 \cdot 1113491139767$	$(2, 79)$	$(-2, 158)$

Fermat Number Transforms

The NTT over the rings with $q = 2^v + 1$ is called the Fermat Number transform

Some nice FNT parameters for $2^{16} < q < 2^{81}$:

ring $\mathbb{Z}_q$	prime factors	(ω, MNT length)	
$2^{16}+1$	65537	$(4, 16)$	$(2, 32)$
$2^{20}+1$	$17 \cdot 61681$	$(32, 8)$	$(4100, 16)$
$2^{24}+1$	$97 \cdot 257 \cdot 673$	$(8, 16)$	$(\sqrt{8}, 32)$
$2^{32}+1$	$641 \cdot 6700417$	$(4, 32)$	$(2, 64)$
$2^{40}+1$	$257 \cdot 4278255361$	$(32, 16)$	$(\sqrt{32}, 32)$
$2^{64}+1$	$274177 \cdot 67280421310721$	$(4, 64)$	$(2, 128)$
$2^{80}+1$	$414721 \cdot 44479210368001$	$(32, 32)$	$(\sqrt{32}, 64)$

Pseudo (Mersenne or Fermat) Number Transforms

- If q is a Mersenne or a Fermat number having a small divisor p (not necessarily a prime), a transform having sufficient lenght may not exist in $\mathbb{Z}_q$

- But the arithmetic in $\mathbb{Z}_{q/p}$ can be still efficient

- Since p divides q, the arithmetic modulo (q/p) can be carried in the ring $\mathbb{Z}_q$ with a final modulo (q/p) reduction

- Such transforms are called Pseudo (Mersenne or Fermat) Number Transforms (PNT)

Pseudo (Mersenne or Fermat) Number Transforms

- PNT tailors the Mersenne or Fermat rings in a way that larger length transforms are possible

- **Example:** In $\mathbb{Z}_{2^{15}-1}$, the maximum transform length is $\gcd(6, 30, 150) = 6$, but if the PNT is employed in the ring $\mathbb{Z}_{(2^{15}-1)/7}$, we get the transform lengths up to

$$\gcd(30, 150) = 30$$

Pseudo (Mersenne or Fermat) Number Transforms

Some nice PNT parameters for $2^{16} < q < 2^{32}$:

Ring $\mathbb{Z}_q$	Prime Factors	Modulus $\mathbb{Z}_{q/p}$	ω	d	ω	d
$2^{17}+1$	$3 \cdot 43691$	$(2^{17}+1)/3$	$-2, 4$	17	2	34
$2^{19}+1$	$3 \cdot 174763$	$(2^{19}+1)/3$	$-2, 4$	19	2	38
$2^{20}+1$	$17 \cdot 61681$	$(2^{20}+1)/17$	4	20	2	40
$2^{21}+1$	$3^2 \cdot 43 \cdot 5419$	$(2^{21}+1)/9$	$-2, 4$	21	2	42
$2^{22}+1$	$5 \cdot 397 \cdot 2113$	$(2^{22}+1)/5$	4	22	2	44
$2^{23}+1$	$3 \cdot 2796203$	$(2^{23}+1)/3$	$-2, 4$	23	2	46
$2^{25}-1$	$31 \cdot 601 \cdot 1801$	$(2^{25}-1)/31$	2	25	-2	50
$2^{26}-1$	$3 \cdot 2731 \cdot 8191$	$(2^{26}-1)/3$	2	26	-2	52
$2^{27}-1$	$7 \cdot 73 \cdot 262657$	$(2^{27}-1)/511$	2	27	-2	54
$2^{28}+1$	$17 \cdot 15790321$	$(2^{28}+1)/17$	4	28	2	56
$2^{29}+1$	$3 \cdot 59 \cdot 3033169$	$(2^{29}+1)/3$	$-2, 4$	29	2	58
$2^{31}+1$	$3 \cdot 715827883$	$(2^{31}+1)/3$	$-2, 4$	31	2	62

Chinese Remainder Theorem

CRT can be employed in two different ways in spectral algorithms:

- **for degree**: follows the work of Quisquater and Couvreur trivially

- if a factorization of the modulus $n = n_1 n_2$ is known, two separate SME for modulus n_1 and n_2 are computed, and then combined.

Chinese Remainder Theorem

- **for radius**: derived from the method proposed by Schönhage and V. Strassen who further proved the well-known time-bound $O(k \log k \log\log k)$ for integer multiplication

- the congruence modulo q can be broken into small congruences and the computations can be performed in these small rings

- One has to recover the actual least significant time digit at every step of the reduction process

- Efficiency comes because of working in the small factor rings

Parameter Selection for RSA

	Bits	Ring $\mathbb{Z}_q$	d	ω	$u,\ b = 2^u$
SMP	518	$2^{73} - 1$	73	2	14
	1,185	$2^{79} - 1$	158	-2	15
	2,060	$(2^{103} + 1)/3$	206	2	20
Modified SMP	540	$2^{59} - 1$	59	2	18
	1,080	$2^{79} - 1$	79	2	40
	2,054	$2^{79} - 1$	158	2	26
	4,251	$2^{109} - 1$	218	-2	39

Parameter Selection for RSA

Bits k	Ring $\mathbb{Z}_q$	DFT d	Root w	Words s	Wordsize u
1,054	$q_1 = 2^{31} - 1$	62	-2	31	34
	$q_2 = (2^{31} + 1)/3$	62	2	31	
	$q_3 = 2^{32} + 1$	64	2	32	
2,067	$q_1 = 2^{53} - 1$	106	-2	53	39
	$q_2 = (2^{53} + 1)/3$	106	2	53	
16,428	$q_1 = (2^{111} - 1)/7$	222	-2	111	148
	$q_2 = (2^{111} + 1)/9$	222	2	111	
	$q_3 = 2^{113} - 1$	226	-2	113	

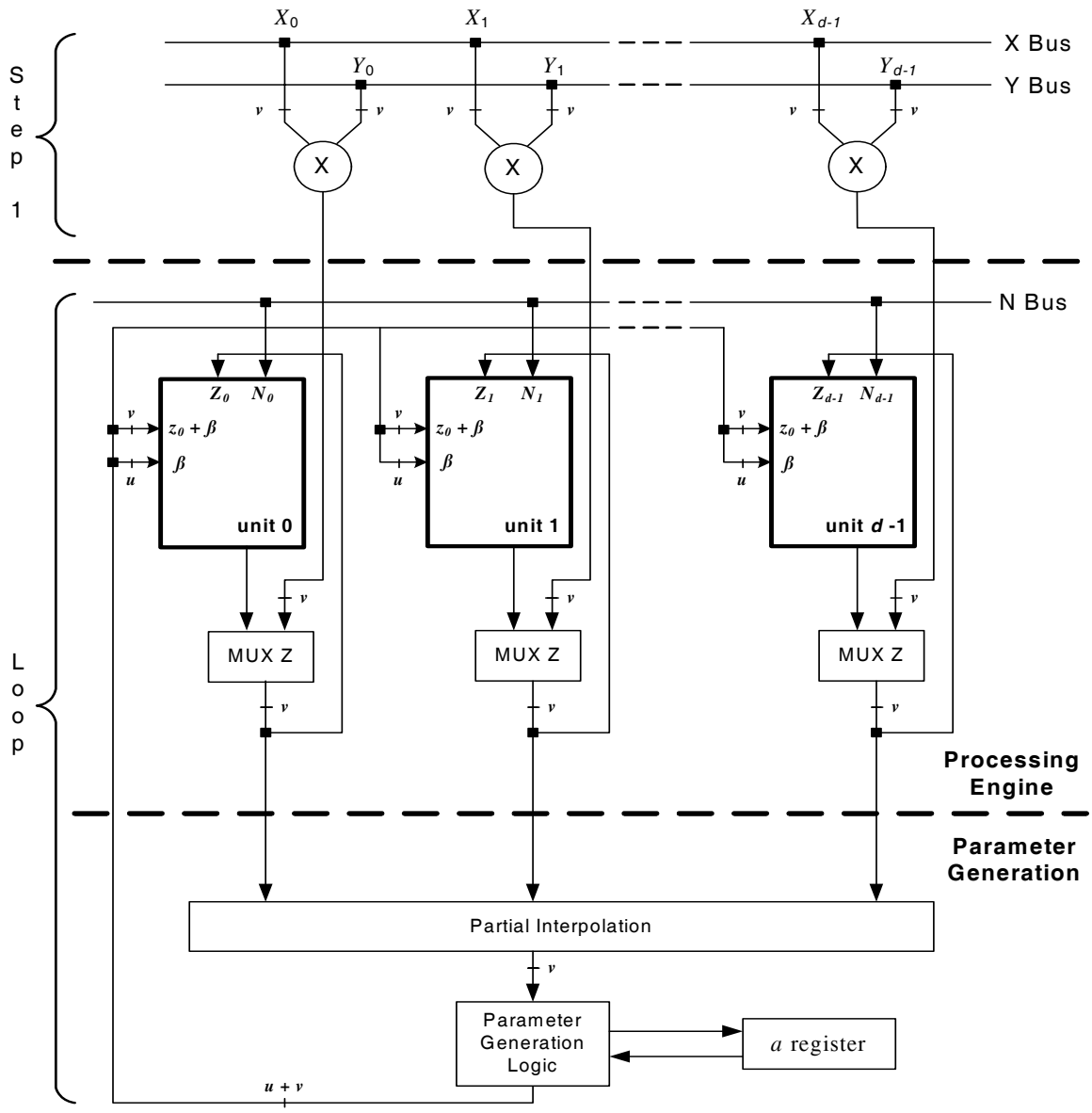
Hardware Architectures

- Spectral methods are powerful: they bring parallelism to modular exponentation computation by dividing the larger problem into smaller pieces

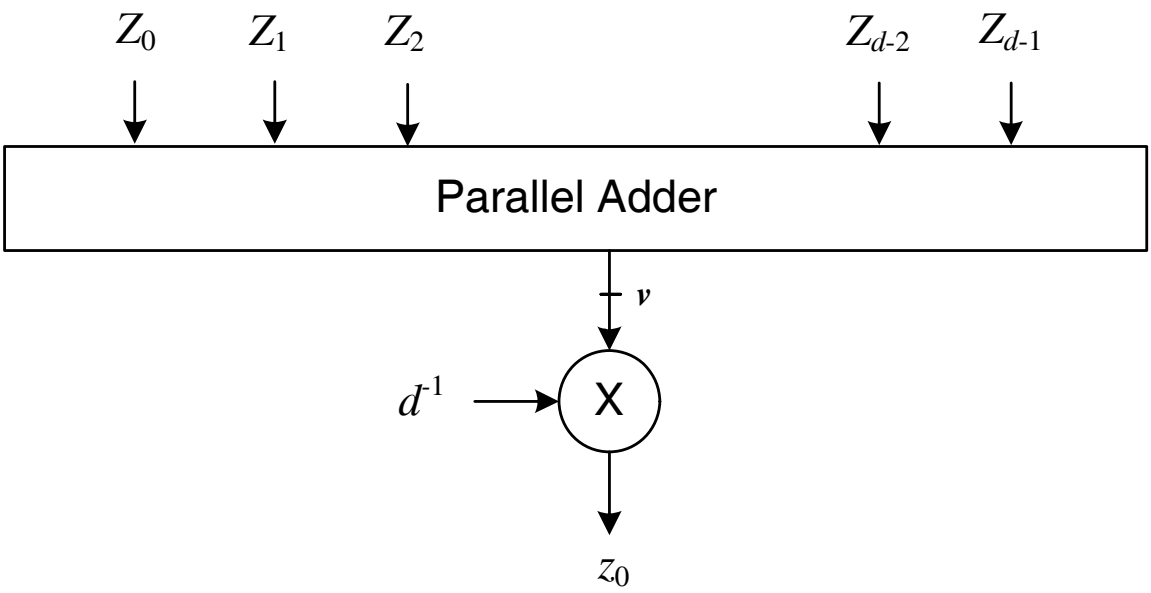
- Moreover, it is possible to employ some low level parallelism on the resulting partitions

- The main drawback is the area as in any other parallel systems

SMP Module

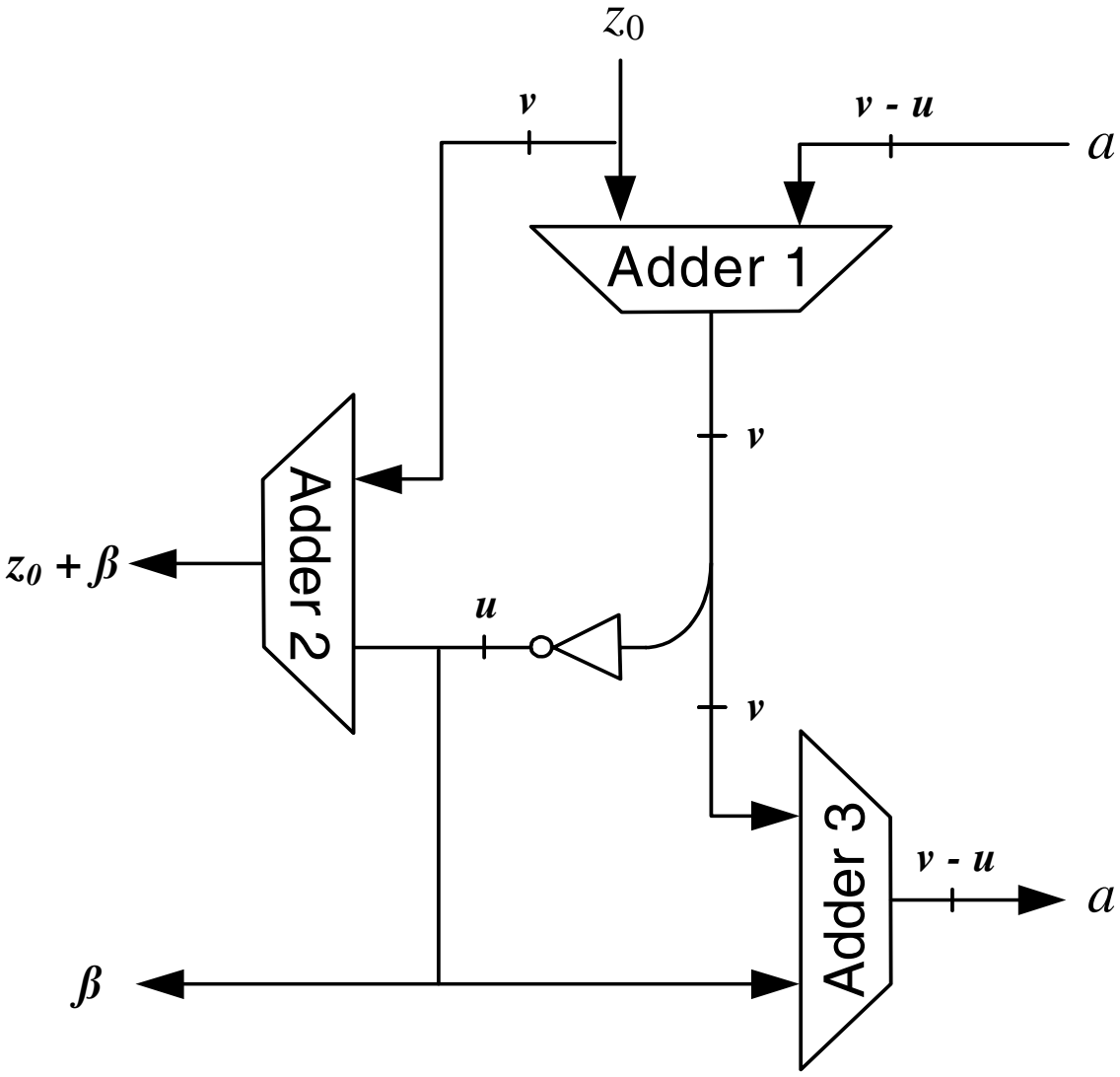

Step 1
Loop
X Bus
Y Bus
N Bus
unit 0
unit 1
unit d -1
MUX Z
Processing Engine
Parameter Generation
Partial Interpolation
Parameter Generation Logic
a register

Partial Interpolation



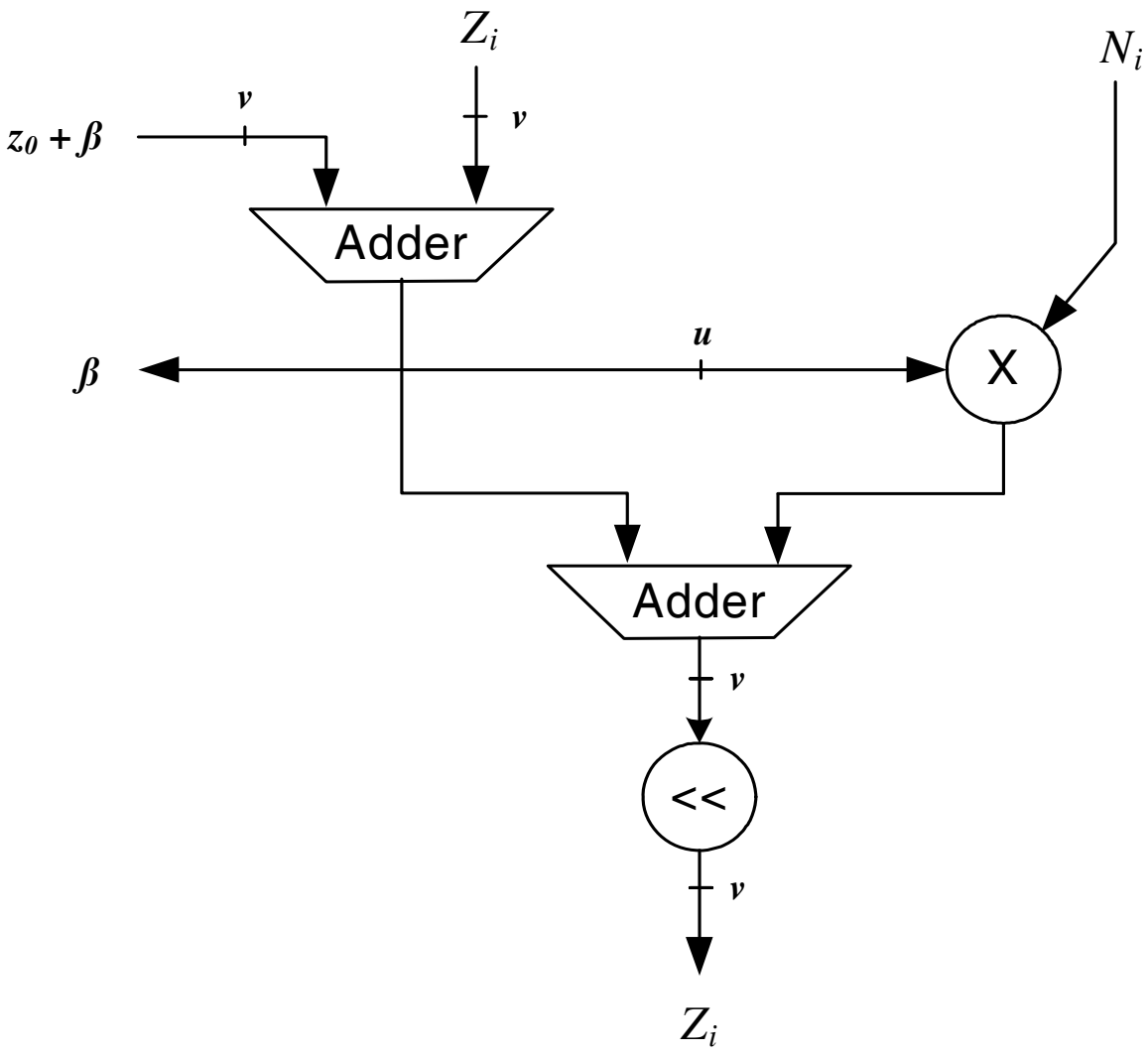
This circuit realizes step 4 of SMP, calculates the least time digit of the partial sum

Generation Logic



This circuit computes steps 5 and 6 of SMP, and generates the parameters α and β

Processing Unit



The main compuation unit corresponding to the Steps 7, 8 and 9

Performance Analysis

Under the following specific assumptions:

- Mersenne or Fermat arithmetic is used for every unit
- All adders except one are modular parallel CSA adders
- Parallel prefix adders are used for CPA
- For multi-operand addition, we use parallel Wallace tree networks

The Cost of SMP

ring	area	delay
M	$\frac{23}{2}v^2 + 7vd - 42v$ $+8uv + \frac{3}{2}u\log u + 7u$	$4\theta(v) + 1 + 2d(\lg(u) + 5) +$ $4d(\theta(d) + \theta(v/2) + \theta(u+1))$
F	$13v^2 + 8vd + 27v$ $+9uv + \frac{3}{2}u\log u + 7u$	$4\theta(v+1) + 4 + 2d(\lg(u) + 19/2) +$ $4d(\theta(d+1) + \theta(v/2+1) + \theta(u+2))$

d is the DFT length, $q = 2^v \pm 1$, $b = 2^u$ and $\theta(x)$ is as follows:

x	3	4	5–6	7–9	10–13	14–19	20–28	29–42	$\cdots$
$\theta(x)$	1	2	3	4	5	6	7	8	$\cdots$

A Sample Cost Computation

- Let's compute the complexity of SME with a modulus 704-bits by using the DFT in the ring $\mathbb{Z}_q$ with $q = 2^{64} + 1$, $\omega = 2$ and $u = 11$. Plugging these into the cost function gives

$$\begin{aligned} T_{SMP} &= 4\theta(v+1) + 4 + 2d(\lg(u)+8) + \\ &\quad 4d(\theta(d+1) + \theta(u+2)) \\ &= 4 \cdot 10 + 4 + 256(4+8) + 512(11+5) \\ &= 11308 \\ A_{SMP} &= 9v^2 + 8vd + 25v + 9uv + \\ &\quad \frac{3}{2} u \log u + 7u \\ &= 177871 \end{aligned}$$

- A realization of SMP with the above parameters allocate 177871 gates and have 11308 gate delay for an output.

- If this SMP is used in a SME, with the same area complexity, one approximately has a $11308K$ gate delays.

Conclusions

- This work is the first utilization of spectral techniques for modular arithmetic

- The asymptotic crossovers for modular exponentiation are brought to cryptographic sizes

- A boundary analysis for optimal domain and related parameter selection are given

- Highly parallel architectures for hardware realizations of the RSA, DH, DSA, and ECC over prime fields are proposed

Future Work

- The use of complex spectrum can be investigated by performing a thorough roundoff error analysis

- The use of finite rings with modulus of the form $2^u \pm 2^v \pm 1$ gives longer DFT lengths with a slightly increased complexity

- Multidimensional transform methods can be investigated

- Extensions to point multiplication on elliptic curves over $GF(2^k)$ and $GF(p^k)$ can be be analyzed

- Actual hardware implementations need to be realized

Publications

1. G. Saldamlı. *Spectral Modular Arithmetic*, Ph.D. Dissertation, Department of Electrical & Computer Engineering, Oregon State University, June 2005.

2. G. Saldamlı and Ç. K. Koç. *Spectral Modular Arithmetic Method and Apparatus*, US Patent Application, April 2005.

3. G. Saldamlı and Ç. K. Koç. Spectral modular exponentiation, submitted to *18th IEEE Symposium on Computer Arithmetic*, Montpellier, France, June 25-27, 2007.

4. G. Saldamlı and Ç. K. Koç. Spectral Modular Arithmetic, preprint, 2006.

5. G. Saldamlı and Ç. K. Koç. Spectral Modular Arithmetic for Binary Extension Fields, preprint, 2006.

6. G. Saldamlı and Ç. K. Koç. Spectral Modular Arithmetic for Extension Fields, preprint, 2006.