Modular Multiplication in the AMNS representation : Hardware Implementation

Selected Areas in Cryptography - 2024

Louis NOYEZ[1]
Nadia EL MRABET[1]
Olivier POTIN[1]
Pascal VERON[2]

[1]Mines Saint-Etienne, CEA, Leti, Centre CMP, F - 13541 Gardanne France

[2]Laboratoire IMath, Université de Toulon, Toulon France

29 August 2024

Outline

1. Cryptography Context and AMNS algorithms

2. Hardware Implementation

3. Results and Conclusions

Cryptography Context and AMNS algorithms

Modern Cryptography

Modular Multiplication $a \cdot b[p]$

Contemporary protocols

- RSA (2048-4096 bits) Modular Exponentiation $C = m^e[M]$
- ECDSA (256-512 bits) ECC Point Scalar Multiplication $C = k.P$

Post-Quantum Cryptography

Isogeny-based Protocols

- NIST standardisation candidate SQIsign (256-512 bits)
- CSIDH (512 bits)

AMNS

Adapted Modular Number System (2006 Bajard et al. [2])

Polynomial representation of large integers modulo p odd.

$$a = 1079783763680532908 8$$

One representative of a in an AMNS is

$$A = 83086 + 7554 \cdot X + 34715 \cdot X^2 - 4780 \cdot X^3$$

AMNS

Adapted Modular Number System (2006 Bajard et al. [2])

Polynomial representation of large integers modulo p odd.

$$a = 10797837636805329088$$

One representative of a in an AMNS is

$$A = 83086 + 7554 \cdot X + 34715 \cdot X^2 - 4780 \cdot X^3$$

AMNS parameters :

- p odd such that $a \in \mathbb{Z}/p\mathbb{Z}$
- N number of coefficients of AMNS elements ($\deg(A) = N - 1$)
- ρ such that $||A||_\infty < \rho$
- $E = X^N - \lambda$ the external reduction polynomial. λ small power of two
- γ such that $E(\gamma)[p] = 0$ ($A(\gamma) = a$)

<u>Example AMNS parameters:</u> $p = 13157208063559315537$

$N = 4$ $\rho = 262144$ $E = X^4 - 2$ $\gamma = 13020125524669010305$

AMNS multiplication

Let $\mathcal{B}(p, N = 5, \rho, E = X^4 - 2)$ be an AMNS.

$$A = 83086 + 7554 \cdot X + 34715 \cdot X^2 - 4780 \cdot X^3$$

$$B = 80081 + 33377 \cdot X - 3680 \cdot X^2 + 25843 \cdot X^3$$

$$A \cdot B = 6653609966 + 3378093296 \cdot X + 2726385293 \cdot X^2 + 2895288153 \cdot X^3$$
$$- 92075238 \cdot X^4 + 914730145 \cdot X^5 - 123529540 \cdot X^6$$

$$\deg(A \cdot B) > N - 1, \qquad ||A \cdot B||_\infty > \rho \qquad \Longrightarrow A \cdot B \notin \mathcal{B}$$

AMNS multiplication

Let $\mathcal{B}(p, N = 5, \rho, E = X^4 - 2)$ be an AMNS.

$$A = 83086 + \ \ 7554 \cdot X + 34715 \cdot X^2 - \ \ 4780 \cdot X^3$$

$$B = 80081 + 33377 \cdot X - \ \ 3680 \cdot X^2 + 25843 \cdot X^3$$

$$\begin{aligned} A \cdot B = \quad & 6653609966 \quad + 3378093296 \cdot X \ + 2726385293 \cdot X^2 + 2895288153 \cdot X^3 \\ - \quad & 92075238 \cdot X^4 + \ 914730145 \cdot X^5 - \ \ 123529540 \cdot X^6 \end{aligned}$$

$$\deg(A \cdot B) > N - 1, \qquad ||A \cdot B||_\infty > \rho \qquad \implies A \cdot B \notin \mathcal{B}$$

EXTERNAL REDUCTION:

$$A \cdot B[E] = 6469459490 + 5207553586 \cdot X + 2479326213 \cdot X^2 + 2895288153 \cdot X^3$$

$$\deg(A \cdot B[E]) \leq N - 1, \quad ||A \cdot B[E]||_\infty > \rho \qquad \implies A \cdot B[E] \notin \mathcal{B}$$

AMNS multiplication

Let $\mathcal{B}(p, N = 5, \rho, E = X^4 - 2)$ be an AMNS.

$$A = 83086 + 7554 \cdot X + 34715 \cdot X^2 - 4780 \cdot X^3$$

$$B = 80081 + 33377 \cdot X - 3680 \cdot X^2 + 25843 \cdot X^3$$

$$A \cdot B = 6653609966 + 3378093296 \cdot X + 2726385293 \cdot X^2 + 2895288153 \cdot X^3 - 92075238 \cdot X^4 + 914730145 \cdot X^5 - 123529540 \cdot X^6$$

$$\deg(A \cdot B) > N - 1, \quad ||A \cdot B||_\infty > \rho \quad \implies A \cdot B \notin \mathcal{B}$$

EXTERNAL REDUCTION:

$$A \cdot B[E] = 6469459490 + 5207553586 \cdot X + 2479326213 \cdot X^2 + 2895288153 \cdot X^3$$

$$\deg(A \cdot B[E]) \leq N - 1, \quad ||A \cdot B[E]||_\infty > \rho \quad \implies A \cdot B[E] \notin \mathcal{B}$$

INTERNAL REDUCTION:

$$C = \text{RedInt}(A \cdot B[E]) = 5419 + 19939 \cdot X + 12918 \cdot X^2 + 17941 \cdot X^3$$

$$\deg(C) \leq N - 1, \quad ||C||_\infty < \rho \implies C \in \mathcal{B}, \; C(\gamma)[p] = A \cdot B(\gamma)[p]$$

The **Montgomery-like** algorithm is used for internal reduction: analogous to the Montgomery multiplication algorithm in classical representation.

Montgomery Multiplication algorithm

Speeds up repeated modular multiplications with modulus p fixed.
Replaces divisions with binary right shifts.

Internal reduction parameters:

- ϕ power of 2 such that $\phi > p$
- p
- p' such that $p.p'[\phi] = -1[\phi]$

$a = \texttt{95d9a01f0c2000c0}_h \qquad b = \texttt{89b77ad47c7b16de}_h$

$p = \texttt{b697cb06205bb051}_h \qquad p' = \texttt{c44eb4055684674f}_h \qquad \phi = 2^{64}$

$$
\begin{aligned}
1 &: c = a \cdot b && = \texttt{509cdd282b13a7463361f8b4a0112680}_h \\
2 &: q = c \cdot p'[\phi] && = \texttt{1b9c786b25c86180}_h \\
3 &: t = c + q \cdot p && = \texttt{644e79f51bc8a1f20000000000000000}_h \\
4 &: \text{res} = \frac{t}{\phi} && = \texttt{644e79f51bc8a1f2}_h = a \cdot b \cdot \phi^{-1}[p]
\end{aligned}
$$

Montgomery-like algorithm

AMNS ⟷	Classical representation
ϕ power of 2 such that $\phi > 2.(1 + \lvert\lambda\rvert \cdot (N-1)).\rho$	ϕ power of 2 such that $\phi > p$
M such that $M(\gamma)[p] = 0$.	p
M' such that $M.M'[E, \phi] = -1[\phi]$	p' such that $p.p'[\phi] = -1[\phi]$

Montgomery-like algorithm

AMNS $\longleftrightarrow$ Classical representation

- ϕ power of 2 such that $\phi > 2.(1 + |\lambda| \cdot (N-1)).\rho$
- M such that $M(\gamma)[p] = 0$.
- M' such that $M.M'[E, \phi] = -1[\phi]$

- ϕ power of 2 such that $\phi > p$
- p
- p' such that $p.p'[\phi] = -1[\phi]$

$$\phi = 2^{24} \qquad A = \texttt{1a62c}_h + \texttt{1489d}_h \cdot X + \texttt{10b53}_h \cdot X^2 + \texttt{f26c}_h \cdot X^3$$

$$B = \texttt{22de4}_h + \texttt{148e0}_h \cdot X + \texttt{1065}_h \cdot X^2 + \texttt{f41e}_h \cdot X^3$$

$$M = -\texttt{3d41}_h + \texttt{ca97}_h \cdot X + \texttt{1a0}_h \cdot X^2 - \texttt{17a6}_h \cdot X^3$$

$$M' = \texttt{569fa7}_h + \texttt{2062d}_h \cdot X + \texttt{e97097}_h \cdot X^2 + \texttt{cabc27}_h \cdot X^3$$

$$1: \quad C = A \cdot B[E] = \texttt{89bd8547a}_h + \texttt{7075db400}_h \cdot X + \texttt{5d61a5ef8}_h \cdot X^2 + \texttt{50f5803e9}_h \cdot X^3$$

$$2: \quad Q = C \cdot M'[E, \phi] = \texttt{e90e10}_h + \texttt{29cde0}_h \cdot X + \texttt{d3cbc}_h \cdot X^2 + \texttt{32682d}_h \cdot X^3$$

$$3: \quad T = C + Q \cdot M[E] = \texttt{1912000000}_h + \texttt{b3a7000000}_h \cdot X + \texttt{1beb000000}_h \cdot X^2 - \texttt{11ca000000}_h \cdot X^3$$

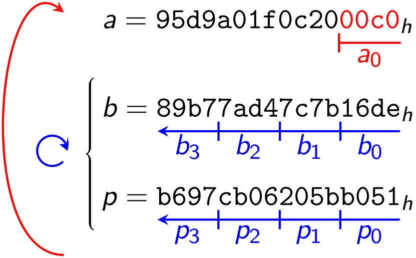
$$4: \text{RES} = \frac{T}{\phi} = \texttt{1912}_h + \texttt{b3a7}_h \cdot X + \texttt{1beb}_h \cdot X^2 - \texttt{11ca}_h \cdot X^3$$

Block Montgomery Multiplication (classical representation)

Block descriptions of the Montgomery Multiplication were classified by Koç et al. [7] for implementations on general purpose processors.

- w: bit width of words
- s: number of blocks required to slice operands ($\phi = 2^{sw}$)
- $a = \sum_{i=0}^{s-1} a_i \cdot (2^w)^i$ Same for b, p and p'

Example: $w = 16, s = 4$, FIOS (Finely Integrated Operand Scanning). One outer loop, one inner loop. Outer loop iterations can be parallelized



$a = \texttt{95d9a01f0c2000c0}_h$

$b = \texttt{89b77ad47c7b16de}_h$

$p = \texttt{b697cb06205bb051}_h$

Montgomery-like block variants (AMNS)

- No block variants of the Montgomery-like algorithm until now.
- Software implementations [5, 4][a][b] use many AMNS coefficients.
- FIOS hardware implementations from [8][c] as the basis for the methodology of hardware implementation of AMNS Montgomery-like multiplications.

[a] Fangan Yssouf Dosso et al. "PMNS for Efficient Arithmetic and Small Memory Cost". *IEEE Transactions on Emerging Topics in Computing* (2022).

[b] Titouan Coladon et al. "MPHELL: A fast and robust library with unified and versatile arithmetics for elliptic curves cryptography". *2021 IEEE 28th Symposium on Computer Arithmetic (ARITH).* 2021.

[c] Louis Noyez et al. "Montgomery Multiplication Scalable Systolic Designs Optimized for DSP48E2". (2024). ACM Transactions on Reconfigurable Technology and Systems.

- How to develop block descriptions of the Montgomery-like algorithm ?

Block slicing of AMNS elements

$$A = 59 - 13 \cdot X + 3 \cdot X^2 + 52 \cdot X^3$$

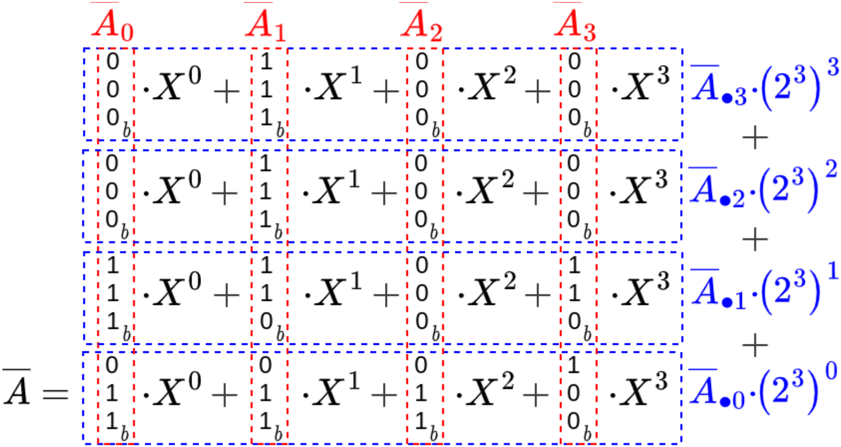
Block slicing of AMNS elements

$$A = 59 - 13 \cdot X + 3 \cdot X^2 + 52 \cdot X^3$$

$$\overline{A} = \overset{\overline{A}_0}{\boxed{\begin{array}{c}0\\0\\0\\0\\0\\0\\1\\1\\1\\0\\1\\1_b\end{array}}} \cdot X^0 + \overset{\overline{A}_1}{\boxed{\begin{array}{c}1\\1\\1\\1\\1\\1\\1\\1\\0\\0\\1\\1_b\end{array}}} \cdot X^1 + \overset{\overline{A}_2}{\boxed{\begin{array}{c}0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\1\\1_b\end{array}}} \cdot X^2 + \overset{\overline{A}_3}{\boxed{\begin{array}{c}0\\0\\0\\0\\0\\0\\1\\1\\0\\1\\0\\0_b\end{array}}} \cdot X^3$$

Block slicing of AMNS elements

$$A = 59 - 13 \cdot X + 3 \cdot X^2 + 52 \cdot X^3$$
$$w = 3, s = 4, \overline{A} = \sum_{i=0}^{s-1} \overline{A}_{\bullet i} \cdot (2^w)^i, \phi = 2^{sw}$$



$$\begin{array}{rcccccccl}
 & \overline{A}_0 & & \overline{A}_1 & & \overline{A}_2 & & \overline{A}_3 & \\
 & \begin{matrix}0\\0\\0\end{matrix}_b & \cdot X^0 + & \begin{matrix}1\\1\\1\end{matrix}_b & \cdot X^1 + & \begin{matrix}0\\0\\0\end{matrix}_b & \cdot X^2 + & \begin{matrix}0\\0\\0\end{matrix}_b \cdot X^3 & \overline{A}_{\bullet 3} \cdot \left(2^3\right)^3 \\
 & & & & & & & & + \\
 & \begin{matrix}0\\0\\0\end{matrix}_b & \cdot X^0 + & \begin{matrix}1\\1\\1\end{matrix}_b & \cdot X^1 + & \begin{matrix}0\\0\\0\end{matrix}_b & \cdot X^2 + & \begin{matrix}0\\0\\0\end{matrix}_b \cdot X^3 & \overline{A}_{\bullet 2} \cdot \left(2^3\right)^2 \\
 & & & & & & & & + \\
 & \begin{matrix}1\\1\\1\end{matrix}_b & \cdot X^0 + & \begin{matrix}1\\1\\0\end{matrix}_b & \cdot X^1 + & \begin{matrix}0\\0\\0\end{matrix}_b & \cdot X^2 + & \begin{matrix}1\\1\\0\end{matrix}_b \cdot X^3 & \overline{A}_{\bullet 1} \cdot \left(2^3\right)^1 \\
 & & & & & & & & + \\
\overline{A} = & \begin{matrix}0\\1\\1\end{matrix}_b & \cdot X^0 + & \begin{matrix}0\\1\\1\end{matrix}_b & \cdot X^1 + & \begin{matrix}0\\1\\1\end{matrix}_b & \cdot X^2 + & \begin{matrix}1\\0\\0\end{matrix}_b \cdot X^3 & \overline{A}_{\bullet 0} \cdot \left(2^3\right)^0
\end{array}$$

Hardware Implementation

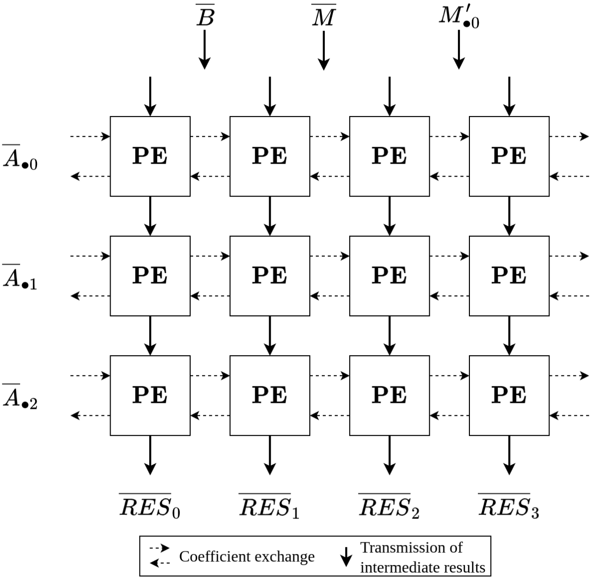
Hardware target

FPGAs devices : quick prototyping and design space exploration
Modern Xilinx Ultrascale FPGA family used in [8]
DSP48E2 arithmetic accelerator components feature:

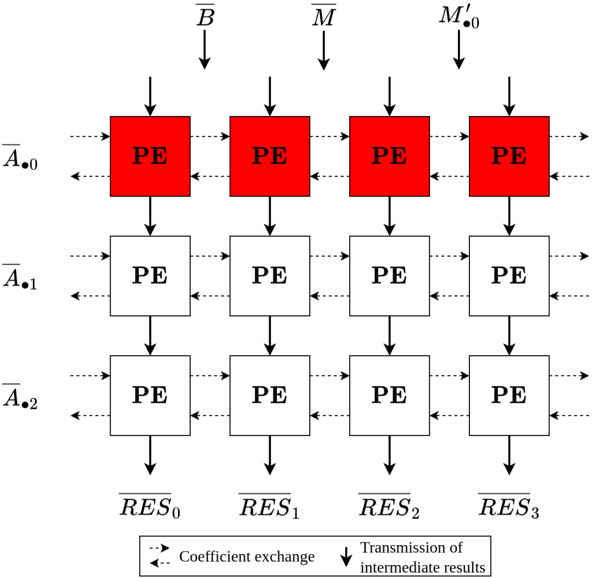
- A 17x17 bits multiplier.
- A 3-input 48-bit adder which can be used to add the result of a multiplication, accumulate data (possibly right shifted by 17 bits) and add external data in a single clock cycle.

w = 17 bits slicing of operands.
Primary goals: performance and scalability to any number of coefficients and size of coefficients.

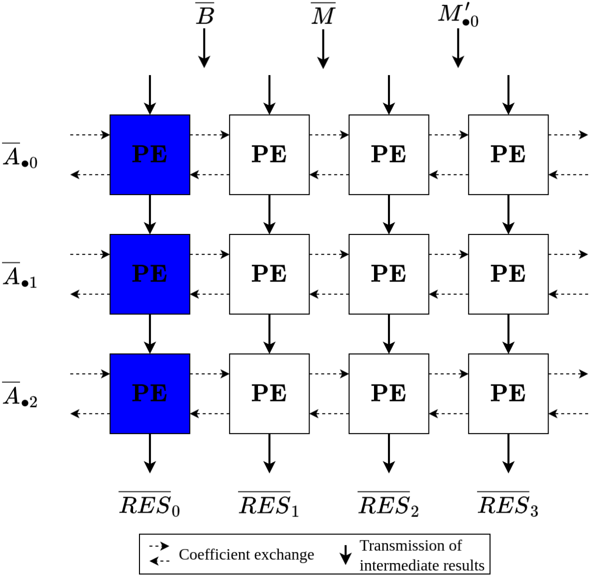
Systolic Array



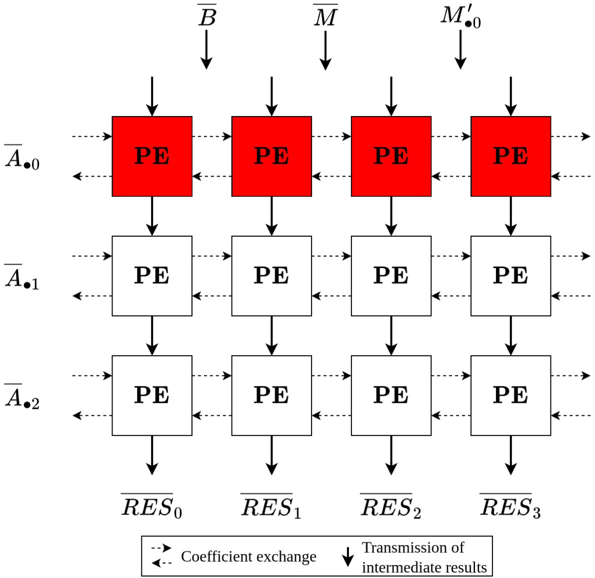
Systolic Array



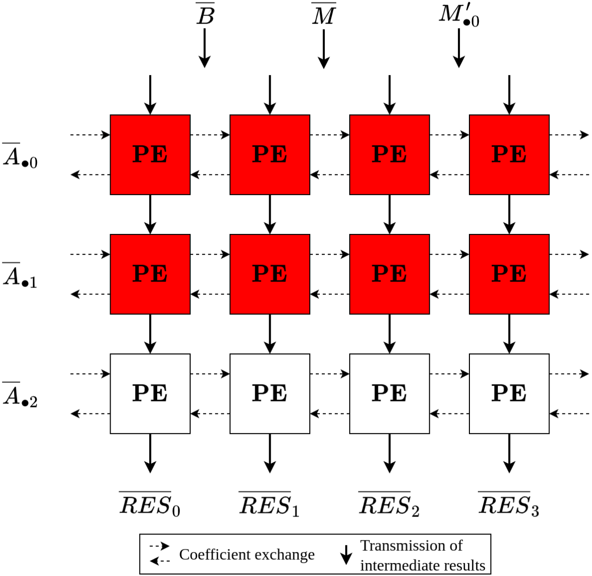
Systolic Array



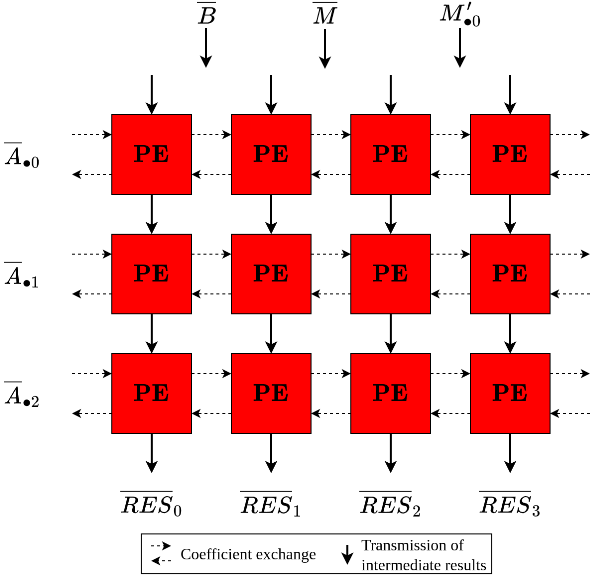
Systolic Array



Systolic Array



Systolic Array



Polynomial blocks multiplication schedulings

$\tilde{A}, \tilde{B}$: polynomial blocks such that $||A||_\infty < 2^{17}$ and $||B||_\infty < 2^{17}$, $N = 5$

Target	Cycle	Coefficients				
		X^0	X^1	X^2	X^3	X^4
$\tilde{A}$		$\tilde{A}_0$	$\tilde{A}_1$	$\tilde{A}_2$	$\tilde{A}_3$	$\tilde{A}_4$
$\tilde{B}$		$\tilde{B}_0$	$\tilde{B}_1$	$\tilde{B}_2$	$\tilde{B}_3$	$\tilde{B}_4$

$\tilde{A} \cdot \tilde{B}[E]$	1	$\tilde{A}_0 \tilde{B}_0$	$\tilde{A}_0 \tilde{B}_1$	$\tilde{A}_0 \tilde{B}_2$	$\tilde{A}_0 \tilde{B}_3$	$\tilde{A}_0 \tilde{B}_4$
	2	$+ \lambda \tilde{A}_1 \tilde{B}_4$	$+ \tilde{A}_1 \tilde{B}_0$	$+ \tilde{A}_1 \tilde{B}_1$	$+ \tilde{A}_1 \tilde{B}_2$	$+ \tilde{A}_1 \tilde{B}_3$

$\vdots$

Polynomial blocks multiplication schedulings

$\tilde{A}, \tilde{B}$: polynomial blocks such that $||A||_\infty < 2^{17}$ and $||B||_\infty < 2^{17}$, $N = 5$

Target	Cycle	Coefficients				
		X^0	X^1	X^2	X^3	X^4
$\tilde{A}$		$\tilde{A}_0$	$\tilde{A}_1$	$\tilde{A}_2$	$\tilde{A}_3$	$\tilde{A}_4$
$\tilde{B}$		$\tilde{B}_0$	$\tilde{B}_1$	$\tilde{B}_2$	$\tilde{B}_3$	$\tilde{B}_4$

		Congested Scheduling				
$\tilde{A} \cdot \tilde{B}[E]$	1	$\tilde{A}_0 \tilde{B}_0$	$\tilde{A}_0 \tilde{B}_1$	$\tilde{A}_0 \tilde{B}_2$	$\tilde{A}_0 \tilde{B}_3$	$\tilde{A}_0 \tilde{B}_4$
	2	$+ \lambda \tilde{A}_1 \tilde{B}_4$	$+ \quad \tilde{A}_1 \tilde{B}_0$	$+ \quad \tilde{A}_1 \tilde{B}_1$	$+ \quad \tilde{A}_1 \tilde{B}_2$	$+ \tilde{A}_1 \tilde{B}_3$

$\vdots$

		Relaxed Scheduling (N odd)				
$\tilde{A} \cdot \tilde{B}[E]$	1	$\tilde{A}_0 \tilde{B}_0$	$\lambda \tilde{A}_3 \tilde{B}_3$	$\tilde{A}_1 \tilde{B}_1$	$\lambda \tilde{A}_4 \tilde{B}_4$	$\tilde{A}_2 \tilde{B}_2$
	2	$+ \lambda \tilde{A}_2 \tilde{B}_3$	$+ \quad \tilde{A}_0 \tilde{B}_1$	$+ \lambda \tilde{A}_3 \tilde{B}_4$	$+ \quad \tilde{A}_1 \tilde{B}_2$	$+ \tilde{A}_4 \tilde{B}_0$

$\vdots$

Results and Conclusions

Metrics

Criteria

We use three criteria to compare against the state of the art:

- Execution time: depends on the maximum frequency and number of clock cycles (function of N and s)
- Resource cost: number of DSP/LUT/FF resources used
- Area-Time product: measure of efficiency for each type of resource

Experimental Results

Design	Freq	Latency	DSP/LUT/FF	Time	DSP/LUT/FF
parameters	(MHz)	(cc)		(μs)	AT (resource.μs)
width = 256					
CA0D2C1E [8]	625	140	16/1759/3365	0.224	3.58/394/754
AMNS [3]	200	33	120/2728/-	0.165	19.8/450/-
AMNS [3]	194	47	91/1718/-	0.242	22.0/415/-
RNS [1]	-	-	248/9450/-	0.0852	21.13/805/-
N = 3, s = 6	**625**	**111**	**18/4156/5145**	**0.178**	**3.20/738/914**
width = 512					
CA0D2C1E [8]	625	275	31/3443/6602	0.440	13.6/1510/2900
AMNS [3]	162	33	188/29985/-	0.204	38.4/6120/-
AMNS [3]	182	47	176/37138/-	0.258	45.4/9580/-
N = 7, s = 5	**550**	**199**	**35/8124/10128**	**0.362**	**12.7/2940/3660**
width = 2048					
CA0D2C1E [8]	625	1085	121/13487/22602	1.74	210/23400/39000
N = 5, s = 25	**500**	**785**	**125/29182/35008**	**1.57**	**196/45800/54900**
width = 4096					
CA0D2C1E [8]	625	2174	242/26978/44806	3.48	842/93800/156000
N = 5, s = 25	**525**	**1553**	**245/58161/66740**	**2.96**	**725/172000/197000**

Experimental Results

Design parameters	Freq (MHz)	Latency (cc)	DSP/LUT/FF	Time (μs)	DSP/LUT/FF AT (resource.μs)
width = 256					
CA0D2C1E [8]	625	140	16/1759/3365	0.224	3.58/394/754
AMNS [3]	200	33	120/2728/-	0.165	19.8/450/-
AMNS [3]	194	47	91/1718/-	0.242	22.0/415/-
RNS [1]	-	-	248/9450/-	0.0852	21.13/805/-
N = 3, s = 6	**625**	**111**	**18/4156/5145**	**0.178**	**3.20/738/914**
width = 512					
CA0D2C1E [8]	625	275	31/3443/6602	0.440	13.6/1510/2900
AMNS [3]	162	33	188/29985/-	0.204	38.4/6120/-
AMNS [3]	182	47	176/37138/-	0.258	45.4/9580/-
N = 7, s = 5	**550**	**199**	**35/8124/10128**	**0.362**	**12.7/2940/3660**
width = 2048					
CA0D2C1E [8]	625	1085	121/13487/22602	1.74	210/23400/39000
N = 5, s = 25	**500**	**785**	**125/29182/35008**	**1.57**	**196/45800/54900**
width = 4096					
CA0D2C1E [8]	625	2174	242/26978/44806	3.48	842/93800/156000
N = 5, s = 25	**525**	**1553**	**245/58161/66740**	**2.96**	**725/172000/197000**

Experimental Results

Design parameters	Freq (MHz)	Latency (cc)	DSP/LUT/FF	Time (μs)	DSP/LUT/FF AT (resource.μs)
width = 256					
CA0D2C1E [8]	625	140	16/1759/3365	0.224	3.58/394/754
AMNS [3]	200	33	120/2728/-	0.165	19.8/450/-
AMNS [3]	194	47	91/1718/-	0.242	22.0/415/-
RNS [1]	-	-	248/9450/-	0.0852	21.13/805/-
N = 3, s = 6	**625**	**111**	**18/4156/5145**	**0.178**	**3.20/738/914**
width = 512					
CA0D2C1E [8]	625	275	31/3443/6602	0.440	13.6/1510/2900
AMNS [3]	162	33	188/29985/-	0.204	38.4/6120/-
AMNS [3]	182	47	176/37138/-	0.258	45.4/9580/-
N = 7, s = 5	**550**	**199**	**35/8124/10128**	**0.362**	**12.7/2940/3660**
width = 2048					
CA0D2C1E [8]	625	1085	121/13487/22602	1.74	210/23400/39000
N = 5, s = 25	**500**	**785**	**125/29182/35008**	**1.57**	**196/45800/54900**
width = 4096					
CA0D2C1E [8]	625	2174	242/26978/44806	3.48	842/93800/156000
N = 5, s = 25	**525**	**1553**	**245/58161/66740**	**2.96**	**725/172000/197000**

Experimental Results

Design parameters	Freq (MHz)	Latency (cc)	DSP/LUT/FF	Time (μs)	DSP/LUT/FF AT (resource.μs)
width = 256					
CA0D2C1E [8]	625	140	16/1759/3365	0.224	3.58/394/754
AMNS [3]	200	33	120/2728/-	0.165	19.8/450/-
AMNS [3]	194	47	91/1718/-	0.242	22.0/415/-
RNS [1]	-	-	248/9450/-	0.0852	21.13/805/-
N = 3, s = 6	**625**	**111**	**18/4156/5145**	**0.178**	**3.20/738/914**
width = 512					
CA0D2C1E [8]	625	275	31/3443/6602	0.440	13.6/1510/2900
AMNS [3]	162	33	188/29985/-	0.204	38.4/6120/-
AMNS [3]	182	47	176/37138/-	0.258	45.4/9580/-
N = 7, s = 5	**550**	**199**	**35/8124/10128**	**0.362**	**12.7/2940/3660**
width = 2048					
CA0D2C1E [8]	625	1085	121/13487/22602	1.74	210/23400/39000
N = 5, s = 25	**500**	**785**	**125/29182/35008**	**1.57**	**196/45800/54900**
width = 4096					
CA0D2C1E [8]	625	2174	242/26978/44806	3.48	842/93800/156000
N = 5, s = 25	**525**	**1553**	**245/58161/66740**	**2.96**	**725/172000/197000**

Experimental Results

Design	Freq	Latency	DSP/LUT/FF	Time	DSP/LUT/FF
parameters	(MHz)	(cc)		(μs)	AT (resource.μs)
width = 256					
CA0D2C1E [8]	625	140	16/1759/3365	0.224	3.58/394/754
AMNS [3]	200	33	120/2728/-	0.165	19.8/450/-
AMNS [3]	194	47	91/1718/-	0.242	22.0/415/-
RNS [1]	-	-	248/9450/-	0.0852	21.13/805/-
N = 3, s = 6	**625**	**111**	**18/4156/5145**	**0.178**	**3.20/738/914**
width = 512					
CA0D2C1E [8]	625	275	31/3443/6602	0.440	13.6/1510/2900
AMNS [3]	162	33	188/29985/-	0.204	38.4/6120/-
AMNS [3]	182	47	176/37138/-	0.258	45.4/9580/-
N = 7, s = 5	**550**	**199**	**35/8124/10128**	**0.362**	**12.7/2940/3660**
width = 2048					
CA0D2C1E [8]	625	1085	121/13487/22602	1.74	210/23400/39000
N = 5, s = 25	**500**	**785**	**125/29182/35008**	**1.57**	**196/45800/54900**
width = 4096					
CA0D2C1E [8]	625	2174	242/26978/44806	3.48	842/93800/156000
N = 5, s = 25	**525**	**1553**	**245/58161/66740**	**2.96**	**725/172000/197000**

Experimental Results

Design	Freq	Latency	DSP/LUT/FF	Time	DSP/LUT/FF
parameters	(MHz)	(cc)		(μs)	AT (resource.μs)
width = 256					
CA0D2C1E [8]	625	140	16/1759/3365	0.224	3.58/394/754
AMNS [3]	200	33	120/2728/-	0.165	19.8/450/-
AMNS [3]	194	47	91/1718/-	0.242	22.0/415/-
RNS [1]	-	-	248/9450/-	0.0852	21.13/805/-
N = 3, s = 6	**625**	**111**	**18/4156/5145**	**0.178**	**3.20/738/914**
width = 512					
CA0D2C1E [8]	625	275	31/3443/6602	0.440	13.6/1510/2900
AMNS [3]	162	33	188/29985/-	0.204	38.4/6120/-
AMNS [3]	182	47	176/37138/-	0.258	45.4/9580/-
N = 7, s = 5	**550**	**199**	**35/8124/10128**	**0.362**	**12.7/2940/3660**
width = 2048					
CA0D2C1E [8]	625	1085	121/13487/22602	1.74	210/23400/39000
N = 5, s = 25	**500**	**785**	**125/29182/35008**	**1.57**	**196/45800/54900**
width = 4096					
CA0D2C1E [8]	625	2174	242/26978/44806	3.48	842/93800/156000
N = 5, s = 25	**525**	**1553**	**245/58161/66740**	**2.96**	**725/172000/197000**

Experimental Results

Design parameters	Freq (MHz)	Latency (cc)	DSP/LUT/FF	Time (μs)	DSP/LUT/FF AT (resource.μs)
width = 256					
CA0D2C1E [8]	625	140	16/1759/3365	0.224	3.58/394/754
AMNS [3]	200	33	120/2728/-	0.165	19.8/450/-
AMNS [3]	194	47	91/1718/-	0.242	22.0/415/-
RNS [1]	-	-	248/9450/-	0.0852	21.13/805/-
N = 3, s = 6	**625**	**111**	**18/4156/5145**	**0.178**	**3.20/738/914**
width = 512					
CA0D2C1E [8]	625	275	31/3443/6602	0.440	13.6/1510/2900
AMNS [3]	162	33	188/29985/-	0.204	38.4/6120/-
AMNS [3]	182	47	176/37138/-	0.258	45.4/9580/-
N = 7, s = 5	**550**	**199**	**35/8124/10128**	**0.362**	**12.7/2940/3660**
width = 2048					
CA0D2C1E [8]	625	1085	121/13487/22602	1.74	210/23400/39000
N = 5, s = 25	**500**	**785**	**125/29182/35008**	**1.57**	**196/45800/54900**
width = 4096					
CA0D2C1E [8]	625	2174	242/26978/44806	3.48	842/93800/156000
N = 5, s = 25	**525**	**1553**	**245/58161/66740**	**2.96**	**725/172000/197000**

Conclusions and Perspectives

Conclusions

- Block descriptions of the AMNS Montgomery-like algorithm.
- Set of tools for verification and exploration using python's sagemath.
- Study of different schedulings of polynomial blocks multiplication.
- **Open source** [6] implementations of AMNS multipliers based on the modern Xilinx Ultrascale family of FPGA and DSP48E2 components. They provide **competitive performance** and **efficiency** compared to state of the art, and they are **highly flexible and scalable** to a wide range of coefficients and coefficient sizes.

Perspectives

- Study the resilience of AMNS to side-channel attacks and the security implications of its redundancy.
- Implement different block variants of the Montgomery-like algorithm besides FIOS.
- Explore additional polynomial block multiplication schedulings.

Source Code

Open-Source project

`https://github.com/LOUISNOYEZ/AMNS_MM`

bibliography I

[1] Javad Ahsan et al. "Efficient FPGA implementation of RNS Montgomery multiplication using balanced RNS bases". In: *Integration* 84 (2022), pp. 72–83. ISSN: 0167-9260. DOI: https://doi.org/10.1016/j.vlsi.2021.12.006. URL: https://www.sciencedirect.com/science/article/pii/S0167926021001322.

[2] Jean-Claude Bajard, Laurent Imbert, and Thomas Plantard. "Modular Number Systems: Beyond the Mersenne Family". In: *Selected Areas in Cryptography*. Ed. by Helena Handschuh and M. Anwar Hasan. Red. by David Hutchison et al. Vol. 3357. Series Title: Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 159–169. ISBN: 978-3-540-24327-4 978-3-540-30564-4. URL: http://link.springer.com/10.1007/978-3-540-30564-4_11 (visited on 11/28/2023).

bibliography II

[3] Asma Chaouch et al. "Two hardware implementations for modular multiplication in the AMNS: Sequential and semi-parallel". In: *Journal of Information Security and Applications* 58 (2021), p. 102770. ISSN: 2214-2126. URL: https://www.sciencedirect.com/science/article/pii/S2214212621000193.

[4] Titouan Coladon, Philippe Elbaz-Vincent, and Cyril Hugounenq. "MPHELL: A fast and robust library with unified and versatile arithmetics for elliptic curves cryptography". In: *2021 IEEE 28th Symposium on Computer Arithmetic (ARITH)*. 2021, pp. 78–85. DOI: 10.1109/ARITH51176.2021.00026.

[5] Fangan Yssouf Dosso, Jean-Marc Robert, and Pascal Véron. "PMNS for Efficient Arithmetic and Small Memory Cost". In: *IEEE Transactions on Emerging Topics in Computing* 10.3 (2022), pp. 1263–1277. DOI: 10.1109/TETC.2022.3187786.

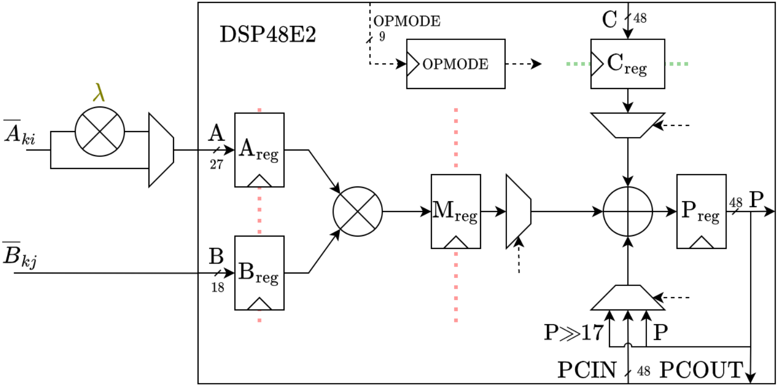
bibliography III

[6] *FIOS DSP Montgomery Multiplier*. https://github.com/LOUISNOYEZ/AMNS_MM. Version 2.0.4. 2024.

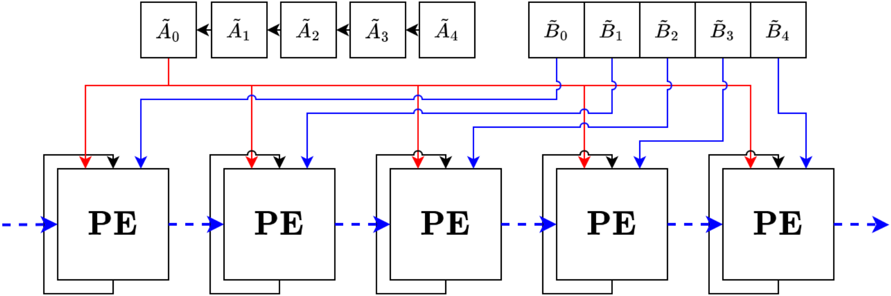
[7] C. Kaya Koc, T. Acar, and B.S. Kaliski. "Analyzing and comparing Montgomery multiplication algorithms". In: *IEEE Micro* 16.3 (1996), pp. 26–33. DOI: 10.1109/40.502403.

[8] Louis Noyez et al. "Montgomery Multiplication Scalable Systolic Designs Optimized for DSP48E2". In: 17.1 (2024). ACM Transactions on Reconfigurable Technology and Systems, pp. 1–31. ISSN: 1936-7406, 1936-7414. URL: https://dl.acm.org/doi/10.1145/3624571.

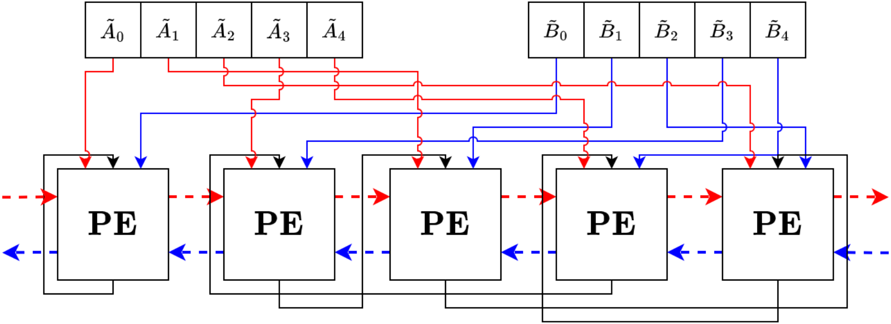
DSP48E2


DSP48E2
OPMODE
9
OPMODE
C
48
C_{reg}
λ
$\overline{A}_{ki}$
A
27
A_{reg}
$\overline{B}_{kj}$
B
18
B_{reg}
M_{reg}
P_{reg}
48
P
P≫17
P
PCIN
48
PCOUT

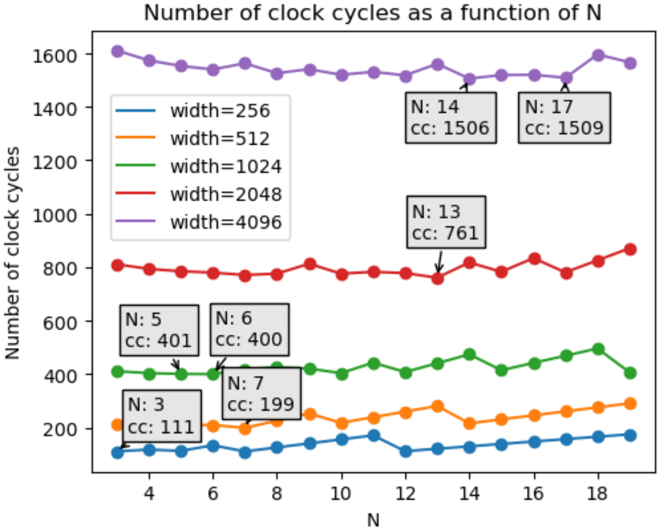
Congested Scheduling



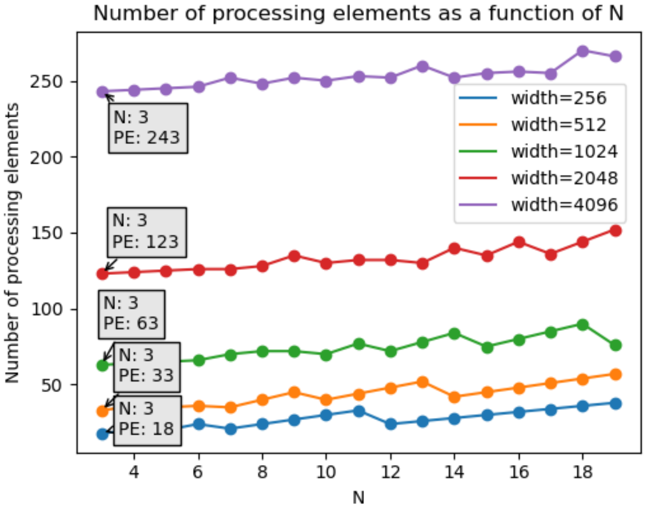
Relaxed Scheduling



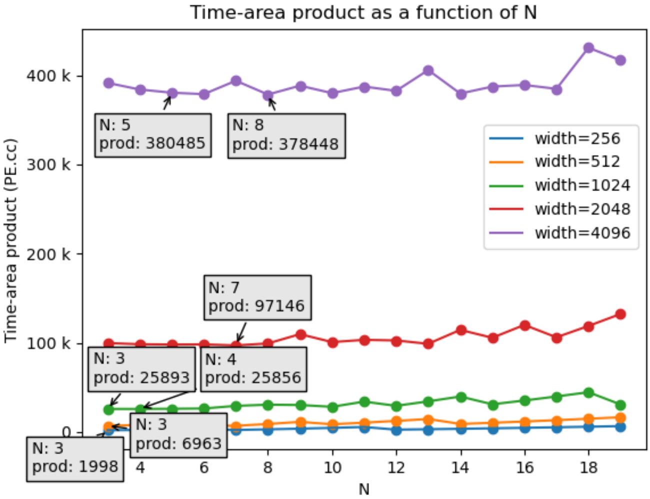
CC vs N


Number of clock cycles as a function of N
Number of clock cycles
N
1600
1400
1200
1000
800
600
400
200
4
6
8
10
12
14
16
18
width=256
width=512
width=1024
width=2048
width=4096
N: 14
cc: 1506
N: 17
cc: 1509
N: 13
cc: 761
N: 5
cc: 401
N: 6
cc: 400
N: 7
cc: 199
N: 3
cc: 111

PE vs N


Number of processing elements as a function of N
Number of processing elements
N
250
200
150
100
50
4
6
8
10
12
14
16
18
width=256
width=512
width=1024
width=2048
width=4096
N: 3
PE: 243
N: 3
PE: 123
N: 3
PE: 63
N: 3
PE: 33
N: 3
PE: 18

CCvsN



Time-area product as a function of N
Time-area product (PE.cc)
400 k
300 k
200 k
100 k
0
4
6
8
10
12
14
16
18
N
width=256
width=512
width=1024
width=2048
width=4096
N: 5
prod: 380485
N: 8
prod: 378448
N: 7
prod: 97146
N: 3
prod: 25893
N: 4
prod: 25856
N: 3
prod: 6963
N: 3
prod: 1998