SCIENCE
PASSION
TECHNOLOGY

Exploring Large Integer Multiplication for Cryptography Targeting In-Memory Computing

Florian Krieger, Florian Hirner, Sujoy Sinha Roy
Institute of Information Security, TU Graz

Outline

Table of Contents

FHE and ZKP: Promising but Challenging

- FHE and ZKP offer novel opportunities
- But there are limitations:
  - Huge computational overhead ($10^5\times$ or more) [1]
  - Lots of data involved ($\sim$ 10GB) [2]

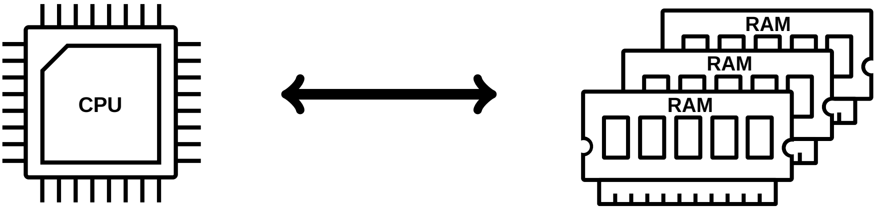
FHE and ZKP: Promising but Challenging

- FHE and ZKP offer novel opportunities
- But there are limitations:
  - Huge computational overhead ($10^5\times$ or more) [1]
  - Lots of data involved ($\sim$ 10GB) [2]

FHE and ZKP: Promising but Challenging

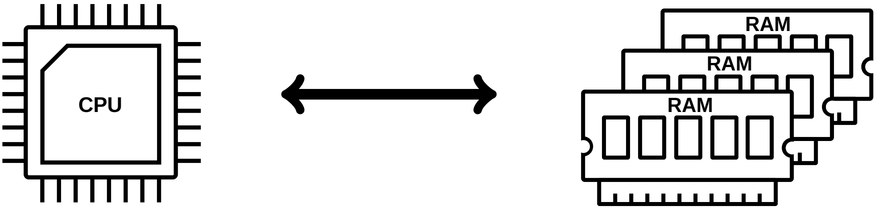
- FHE and ZKP offer novel opportunities
- But there are limitations:
  - Huge computational overhead ($10^5\times$ or more) [1]
  - Lots of data involved ($\sim$ 10GB) [2]

The von Neumann Bottleneck



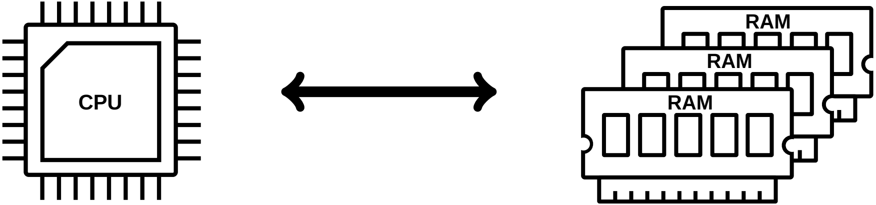
- Substantial data streams
- 100× energy overhead [3]
- Latency bottleneck

The von Neumann Bottleneck



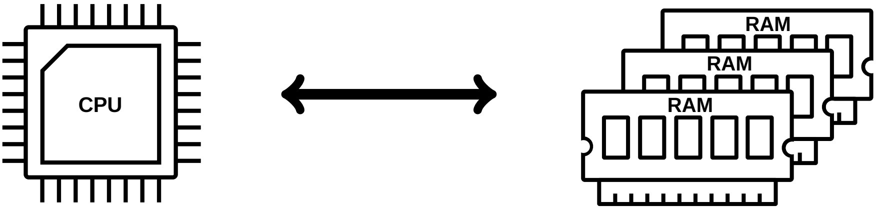
- Substantial data streams
- 100× energy overhead [3]
- Latency bottleneck

The von Neumann Bottleneck



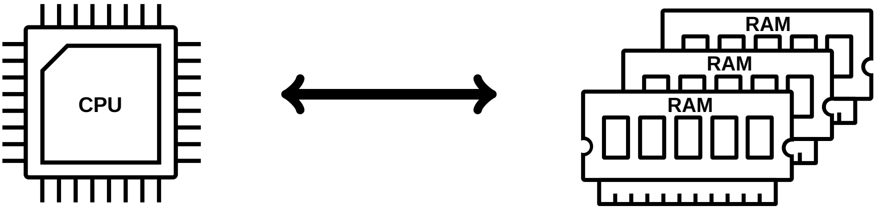
- Substantial data streams
- 100× energy overhead [3]
- Latency bottleneck

The von Neumann Bottleneck



- Substantial data streams
- $100\times$ energy overhead [3]
- Latency bottleneck

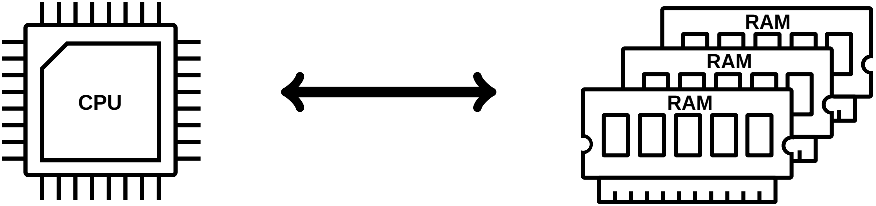
The von Neumann Bottleneck



- Substantial data streams
- 100$\times$ energy overhead [3]
- Latency bottleneck

Can we do better?

The von Neumann Bottleneck

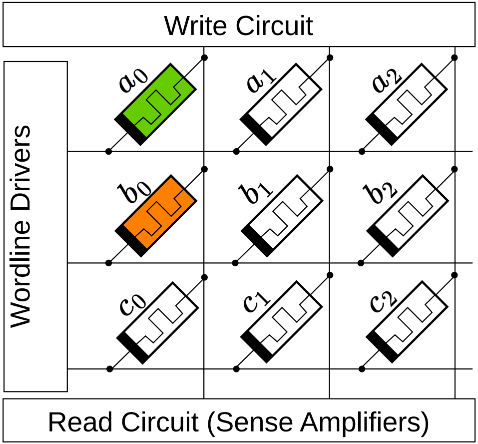


- Substantial data streams
- 100$\times$ energy overhead [3]
- Latency bottleneck

Can we do better?

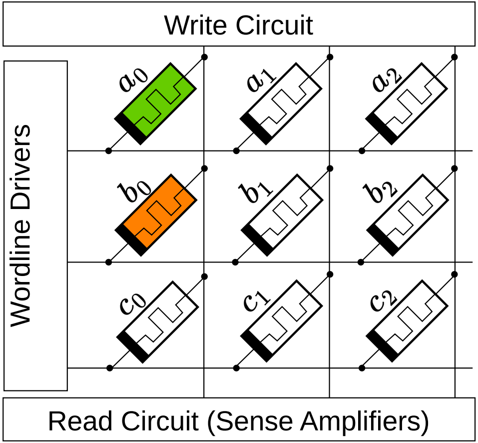
In-Memory Computing!

Resistive Memory: Basic Functionality



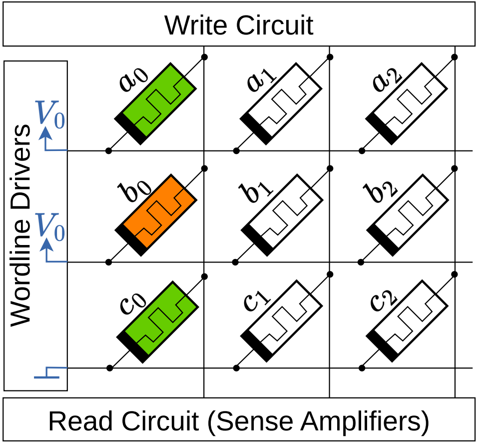
Logic 0 $\Leftrightarrow$ HR
Logic 1 $\Leftrightarrow$ LR

Resistive Memory: Basic Functionality [4]



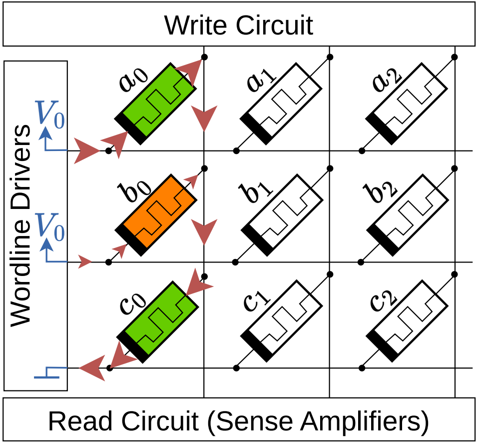
a	b	$c = \mathrm{NOR}(a, b)$
0 (HR)	0 (HR)	1 (LR)
0 (HR)	1 (LR)	0 (HR)
1 (LR)	0 (HR)	0 (HR)
1 (LR)	1 (LR)	0 (HR)

Resistive Memory: Basic Functionality [4]



a	b	$c = \text{NOR}(a, b)$
0 (HR)	0 (HR)	1 (LR)
0 (HR)	1 (LR)	0 (HR)
1 (LR)	0 (HR)	0 (HR)
1 (LR)	1 (LR)	0 (HR)

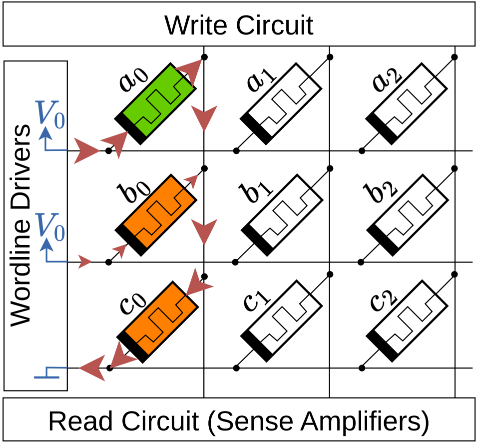
Resistive Memory: Basic Functionality [4]



a	b	$c =$ NOR(a, b)
0 (HR)	0 (HR)	1 (LR)
0 (HR)	1 (LR)	0 (HR)
1 (LR)	0 (HR)	0 (HR)
1 (LR)	1 (LR)	0 (HR)

Ohm's Law:
$I = V / R$

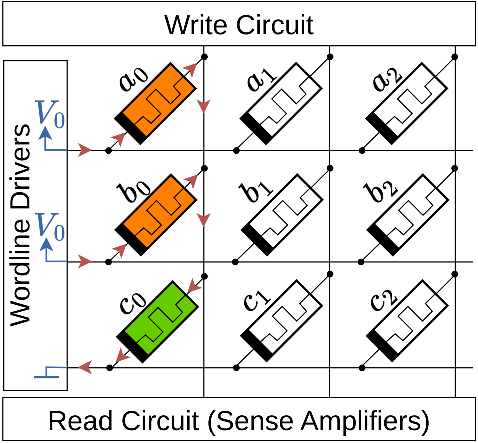
Resistive Memory: Basic Functionality [4]



a	b	$c = \textsc{Nor}(a, b)$
0 (HR)	0 (HR)	1 (LR)
0 (HR)	1 (LR)	0 (HR)
1 (LR)	0 (HR)	0 (HR)
1 (LR)	1 (LR)	0 (HR)

Ohm's Law:
$I = V/R$

Resistive Memory: Basic Functionality [4]



a	b	$c = \text{NOR}(a, b)$
0 (HR)	0 (HR)	1 (LR)
0 (HR)	1 (LR)	0 (HR)
1 (LR)	0 (HR)	0 (HR)
1 (LR)	1 (LR)	0 (HR)

Ohm's Law:
$I = V/R$

In-Memory Computing for FHE and ZKP

Challenges:

- High-level arithmetic from low-level logic gates
  - ➔ Hundreds of bits per operand
- No FHE/ZKP friendly multiplier presented
  - ➔ Typically relying on schoolbook multiplication: $O(n^2)$
- Endurance of cells
- Performance

In-Memory Computing for FHE and ZKP

Challenges:

- High-level arithmetic from low-level logic gates
  - → Hundreds of bits per operand
- No FHE/ZKP friendly multiplier presented
  - → Typically relying on schoolbook multiplication: $O(n^2)$
- Endurance of cells
- Performance

In-Memory Computing for FHE and ZKP

Challenges:

- High-level arithmetic from low-level logic gates
  - → Hundreds of bits per operand
- No FHE/ZKP friendly multiplier presented
  - → Typically relying on schoolbook multiplication: $O(n^2)$
- Endurance of cells
- Performance

In-Memory Computing for FHE and ZKP

Challenges:

- High-level arithmetic from low-level logic gates
  - ➔ Hundreds of bits per operand
- No FHE/ZKP friendly multiplier presented
  - ➔ Typically relying on schoolbook multiplication: $O(n^2)$
- Endurance of cells
- Performance

In-Memory Computing for FHE and ZKP

Challenges:

- High-level arithmetic from low-level logic gates
  - ➔ Hundreds of bits per operand
- No FHE/ZKP friendly multiplier presented
  - ➔ Typically relying on schoolbook multiplication: $O(n^2)$
- Endurance of cells
- Performance

In-Memory Computing for FHE and ZKP

Goal: Close research gap for large integer multipliers

Contributions:

- Algorithmic exploration
- Large integer multiplier design for IMC
  - Algorithmic optimizations
  - Implementation-specific optimizations

In-Memory Computing for FHE and ZKP

Goal: Close research gap for large integer multipliers

Contributions:

- Algorithmic exploration
- Large integer multiplier design for IMC
  - Algorithmic optimizations
  - Implementation-specific optimizations

In-Memory Computing for FHE and ZKP

Goal: Close research gap for large integer multipliers

Contributions:

- ➔ Algorithmic exploration
- ➔ Large integer multiplier design for IMC
  - Algorithmic optimizations
  - Implementation-specific optimizations

Table of Contents

Algorithmic Exploration: Schoolbook Multiplication

- Simple to implement
- $O(n^2)$ complexity
  - Prohibitive for large n
  - Large area consumption
  - High runtime

Algorithmic Exploration: Schoolbook Multiplication

- (+) Simple to implement
- (−) $O(n^2)$ complexity
  - Prohibitive for large n
  - Large area consumption
  - High runtime

Algorithmic Exploration: Toom-Cook Multiplication

- Split operands into k chunks
- Evaluate $\rightarrow$ Multiply $\rightarrow$ Interpolate

- ⊕ Good asymptotic complexity
- ⊖ Concrete efficiency is difficult
  - Vandermonde matrix: $2k\text{-}1 \times 2k\text{-}1$
- ⊖ Hard operations for IMC
  - Integer divisions

Algorithmic Exploration: Toom-Cook Multiplication

- Split operands into k chunks
- Evaluate $\rightarrow$ Multiply $\rightarrow$ Interpolate

- Good asymptotic complexity
- Concrete efficiency is difficult
  - Vandermonde matrix: $2k\text{-}1 \times 2k\text{-}1$
- Hard operations for IMC
  - Integer divisions

Algorithmic Exploration: Toom-Cook Multiplication

- Split operands into k chunks
- Evaluate $\rightarrow$ Multiply $\rightarrow$ Interpolate

- Good asymptotic complexity
- Concrete efficiency is difficult
  - Vandermonde matrix: $2k$-$1 \times 2k$-1
- Hard operations for IMC
  - Integer divisions

Algorithmic Exploration: Toom-Cook Multiplication

- Split operands into k chunks
- Evaluate $\rightarrow$ Multiply $\rightarrow$ Interpolate

- Good asymptotic complexity
- Concrete efficiency is difficult
  - Vandermonde matrix: $2k$-$1 \times 2k$-1
- Hard operations for IMC
  - Integer divisions

Algorithmic Exploration: Toom-Cook Multiplication

- Split operands into k chunks
- Evaluate $\rightarrow$ Multiply $\rightarrow$ Interpolate

- Good asymptotic complexity
- Concrete efficiency is difficult
  - Vandermonde matrix: $2k$-$1 \times 2k$-1
- Hard operations for IMC
  - Integer divisions

Algorithmic Exploration: Toom-Cook Multiplication

- Split operands into k chunks
- Evaluate $\rightarrow$ Multiply $\rightarrow$ Interpolate

- ⊕ Good asymptotic complexity
- ⊖ Concrete efficiency is difficult
  - Vandermonde matrix: $2k\text{-}1 \times 2k\text{-}1$
- ⊖ Hard operations for IMC
  - Integer divisions

e.g. $k = 3$:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1/2 & 1/3 & -1 & 1/6 & -2 \\ -1 & 1/2 & 1/2 & 0 & -1 \\ -1/2 & 1/6 & 1/2 & -1/6 & 2 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Algorithmic Exploration: Karatsuba Multiplication

- Special case of Toom-Cook: $k = 2$
- Slightly slower than Toom-Cook
- Lower complexity than schoolbook
- Suitable for IMC
  - Uses additions and multiplications
  - No divisions, no large matrices

Algorithmic Exploration: Karatsuba Multiplication

- Special case of Toom-Cook: $k = 2$
- Slightly slower than Toom-Cook
- Lower complexity than schoolbook
- Suitable for IMC
  - Uses additions and multiplications
  - No divisions, no large matrices

Algorithmic Exploration: Karatsuba Multiplication

- Special case of Toom-Cook: $k = 2$
- Slightly slower than Toom-Cook
- Lower complexity than schoolbook
- Suitable for IMC
  - Uses additions and multiplications
  - No divisions, no large matrices

Algorithmic Exploration: Karatsuba Multiplication

- Special case of Toom-Cook: $k = 2$
- ⊖ Slightly slower than Toom-Cook
- ⊕ Lower complexity than schoolbook
- ⊕ Suitable for IMC
  - Uses additions and multiplications
  - No divisions, no large matrices

Algorithmic Exploration: Karatsuba Multiplication

- Special case of Toom-Cook: $k = 2$
- Slightly slower than Toom-Cook
- Lower complexity than schoolbook
- Suitable for IMC
  - Uses additions and multiplications
  - No divisions, no large matrices

Selected Karatsuba

Table of Contents

Our Kogge-Stone Adder

$$\left.\begin{matrix} x_7 & x_6 & x_5 & x_4 & x_3 & x_2 & x_1 & x_0 \\ y_7 & y_6 & y_5 & y_4 & y_3 & y_2 & y_1 & y_0 \end{matrix}\right\} \begin{matrix}\text{compute} \\ s = x + y\end{matrix}$$

Our Kogge-Stone Adder

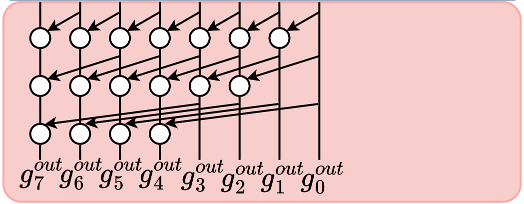
$$\left.\begin{array}{cccccccc} x_7 & x_6 & x_5 & x_4 & x_3 & x_2 & x_1 & x_0 \\ y_7 & y_6 & y_5 & y_4 & y_3 & y_2 & y_1 & y_0 \end{array}\right\} \text{compute } s = x + y$$

$$\begin{array}{cccccccc} p_7 & p_6 & p_5 & p_4 & p_3 & p_2 & p_1 & p_0 \\ g_7 & g_6 & g_5 & g_4 & g_3 & g_2 & g_1 & g_0 \end{array} \quad \begin{array}{l} p_i = x_i \oplus y_i \\ g_i = x_i \wedge y_i \end{array}$$

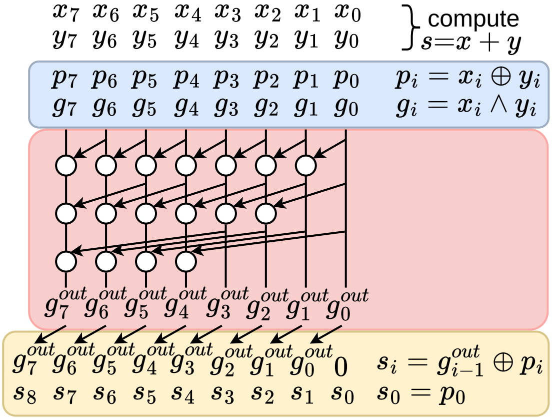
Our Kogge-Stone Adder

$$\left.\begin{matrix} x_7 & x_6 & x_5 & x_4 & x_3 & x_2 & x_1 & x_0 \\ y_7 & y_6 & y_5 & y_4 & y_3 & y_2 & y_1 & y_0 \end{matrix}\right\} \text{compute } s{=}x+y$$

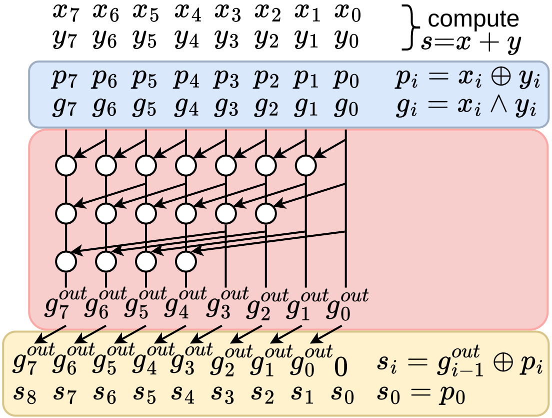
$$\begin{matrix} p_7 & p_6 & p_5 & p_4 & p_3 & p_2 & p_1 & p_0 \\ g_7 & g_6 & g_5 & g_4 & g_3 & g_2 & g_1 & g_0 \end{matrix} \qquad \begin{matrix} p_i = x_i \oplus y_i \\ g_i = x_i \wedge y_i \end{matrix}$$



Our Kogge-Stone Adder

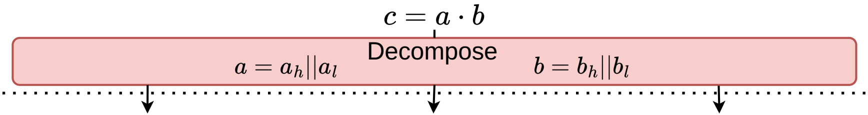


Our Kogge-Stone Adder

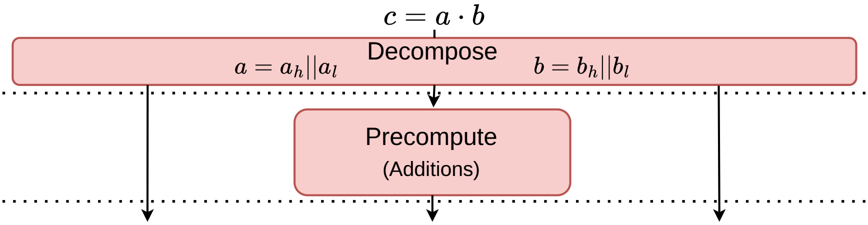


- Logarithmic depth
- Regular structure
- Good choice for IMC

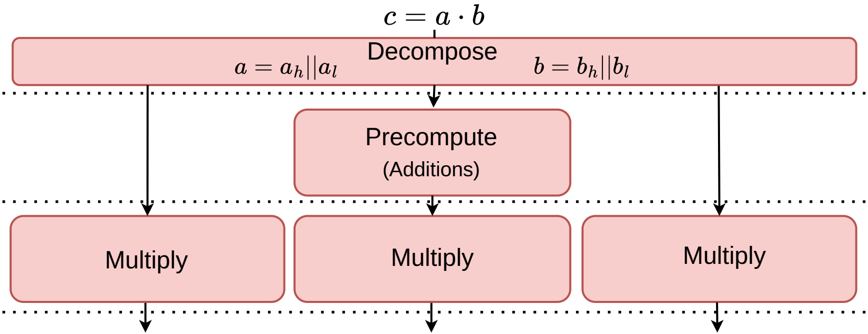
Karatsuba Multiplier Design for IMC



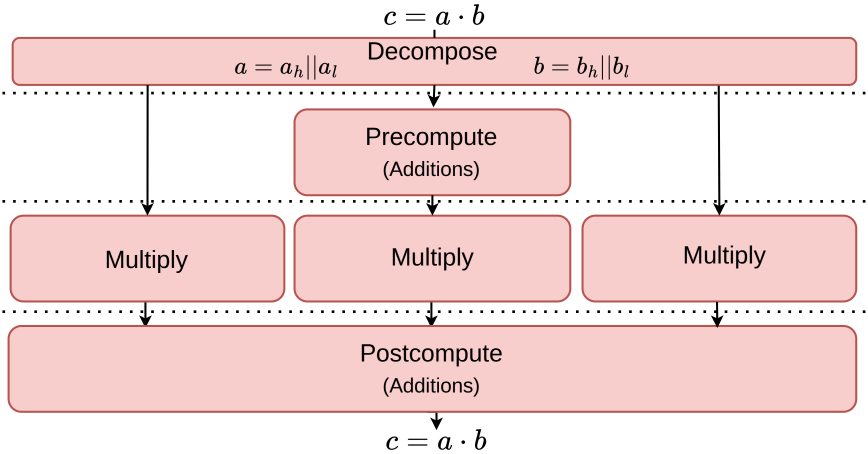
Karatsuba Multiplier Design for IMC



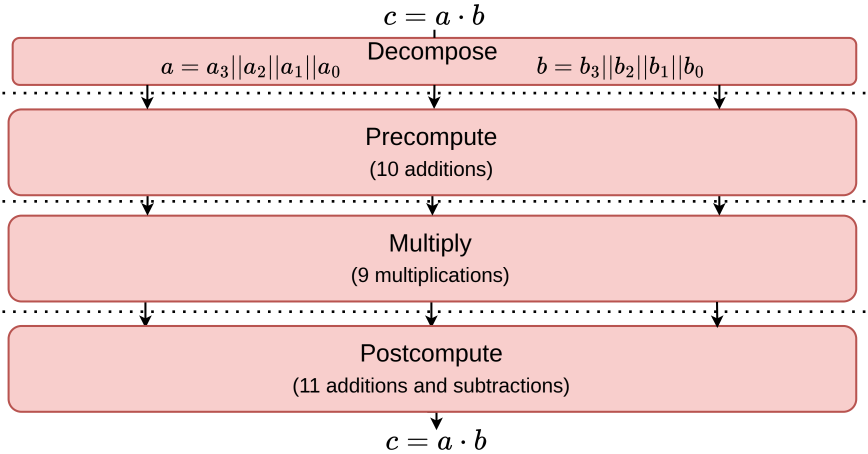
Karatsuba Multiplier Design for IMC



Karatsuba Multiplier Design for IMC



Karatsuba Multiplier Design for IMC



Karatsuba Multiplier Design for IMC

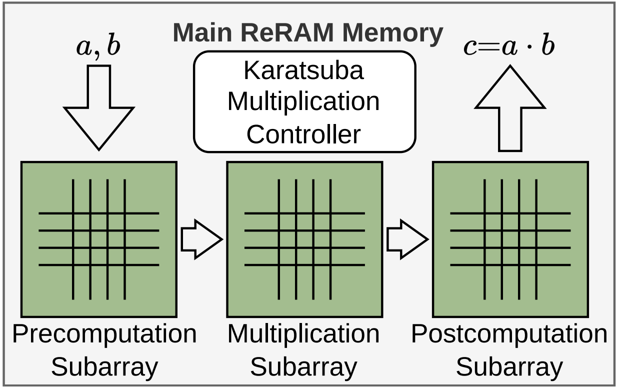
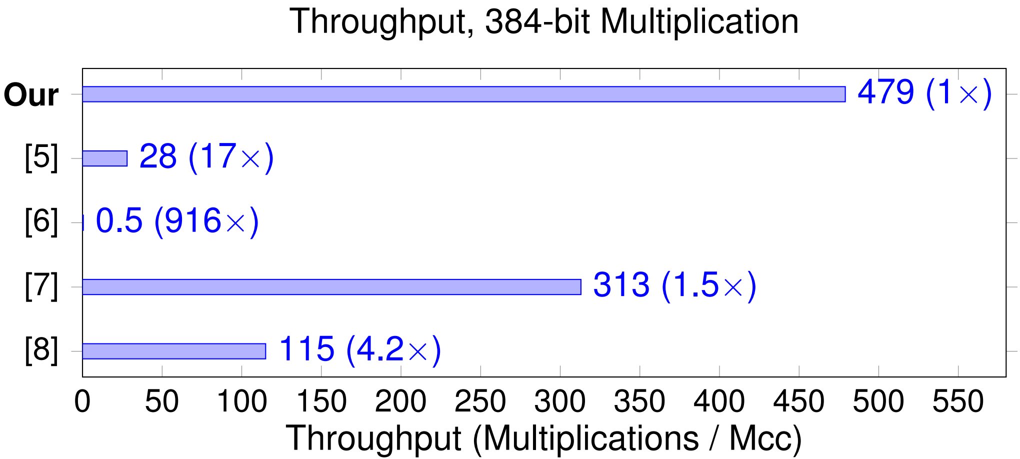
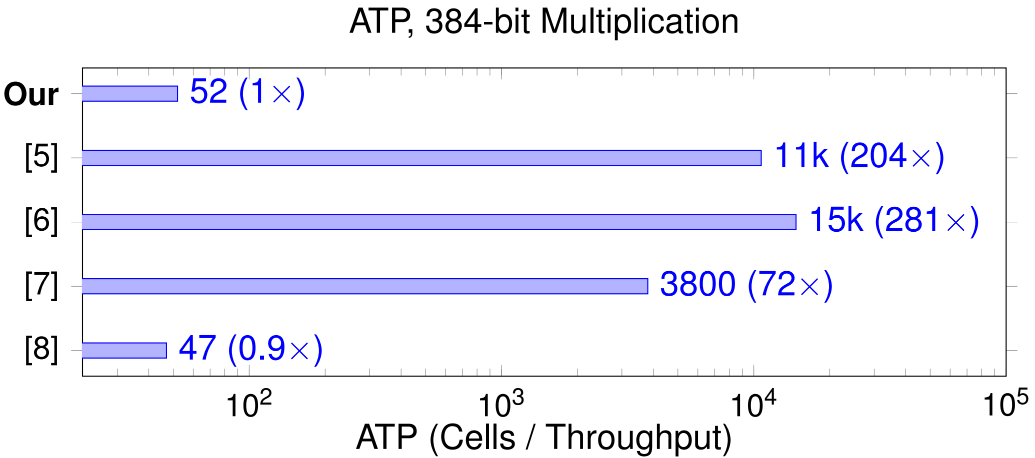


Table of Contents

Comparison with Related Work



Comparison with Related Work



Conclusion

- First long integer multiplier for IMC
- Building block for modular arithmetic
- Contributes to IMC deployment of modern cryptography

Conclusion

- First long integer multiplier for IMC
- Building block for modular arithmetic
- Contributes to IMC deployment of modern cryptography

Conclusion

- First long integer multiplier for IMC
- Building block for modular arithmetic
- Contributes to IMC deployment of modern cryptography

References

[1] N. Samardzic *et al.*, F1: A Fast and Programmable Accelerator for Fully Homomorphic Encryption. in MICRO-54, 2021.

[2] A. Ray *et al.*, Hardcaml MSM: A High-Performance Split CPU-FPGA Multi-Scalar Multiplication Engine. in FPGA'24, 2024.

[3] X. Zou *et al.*, Breaking the von Neumann bottleneck: architecture-level processing-in-memory technology. in Sci. China Inf. Sci. 64, 2021.

[4] S. Kvatinsky *et al.*, MAGIC-Memristor-Aided Logic. in IEEE Transactions on Circuits and Systems II: Express Briefs, 2014.

References

[5] D. Radakovits *et al.*, A memristive multiplier using semi-serial imply-based adder, IEEE Transactions on Circuits and Systems I, 2020.

[6] A. Haj-Ali *et al.*, Imaging: In-memory algorithms for image processing, IEEE Transactions on Circuits and Systems I, 2018.

[7] V. Lakshmi *et al.*, A novel in-memory wallace tree multiplier architecture using majority logic, IEEE Transactions on Circuits and Systems I, 2022.

[8] O. Leitersdorf *et al.*, Multpim: Fast stateful multiplication for processing-in-memory, IEEE Transactions on Circuits and Systems II, 2022.

Exploring Large Integer Multiplication for Cryptography Targeting In-Memory Computing

Florian Krieger, Florian Hirner, Sujoy Sinha Roy
Institute of Information Security, TU Graz

SCIENCE
PASSION
TECHNOLOGY