CryptoBlaze

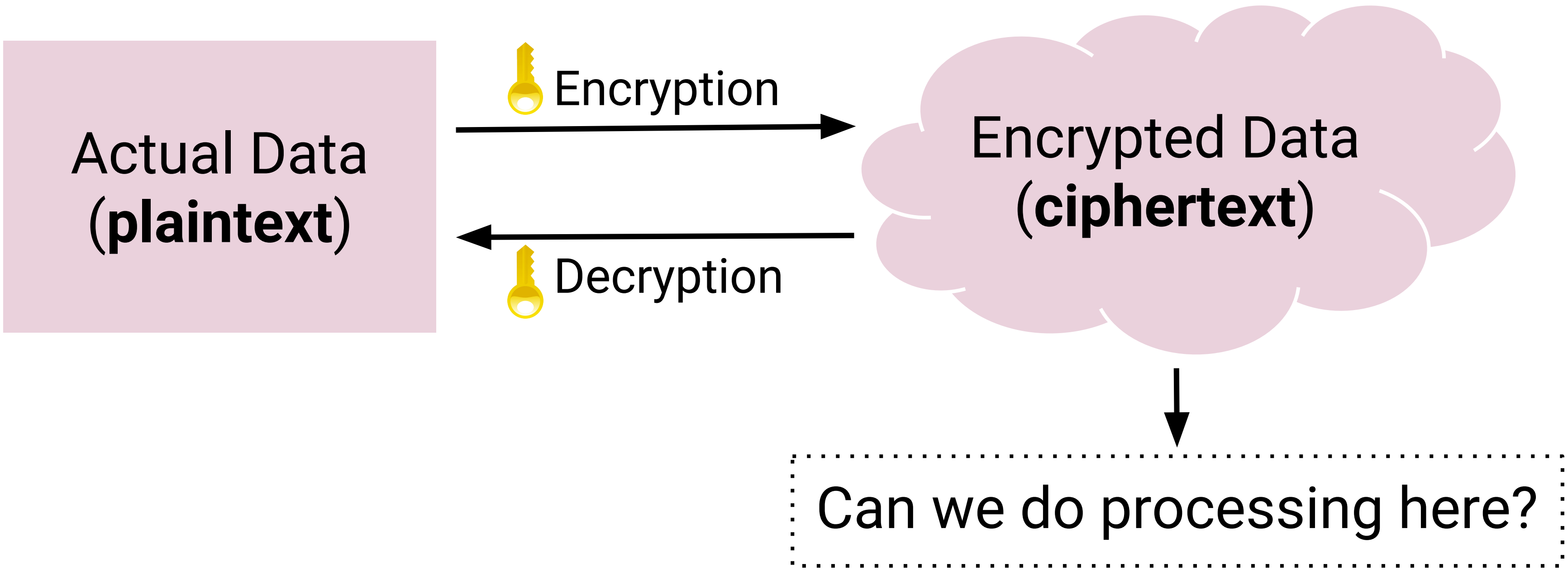
A Partially Homomorphic Processor with Multiple Instructions and Non-Deterministic Encryption Support

Florencia Irena Daniel Murphy Sri Parameswaran

The University of New South Wales, Australia

What is the motivation?

Cloud storage and confidentiality



What is the purpose of CryptoBlaze?

To enable encrypted data processing without exposing the actual data (no decryption key needed)

Encrypted Data
“x” = Enc(x)

No decryption
No knowledge of unencrypted data

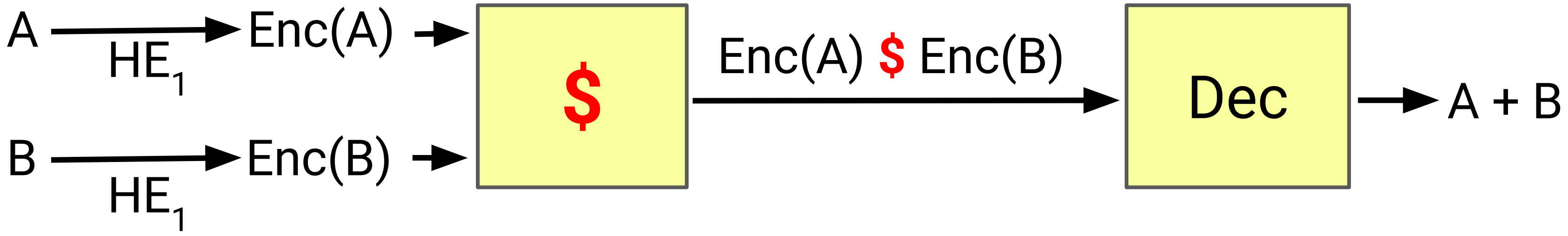
Sorted Encrypted Data
“x” = Enc(x)

Ideal for the cloud: keep actual data secret,
But **HOW**?

How to process ciphertext without decryption?

Homomorphic Encryption (HE)

"Encryption scheme that allows computations on ciphertext, while preserving the correct plaintext result"



- $ = the equivalent of + based on the HE scheme used
- HE_1 is *additively homomorphic*
- CryptoBlaze: homomorphic processor

CryptoBlaze: Threat Model

- Cloud service provider that:
  - **Tries to pry** inside the data memory
  - **Does not tamper** with data/program
- Denial of Service attacks are never carried out

Related Works and CryptoBlaze

	HEROIC[1]	FURISC[2]	[3]	CryptoBlaze
Deterministic	Det	Non-Det	Det	Non-Det
Single/Multi inst.	Single	Single	Multi	Multi
Encrypted prog	Encrypted	Encrypted	Unencrypted	Unencrypted
Hardware/Sim	Hardware	Simulation	Simulation	Hardware
Fully/Partial HE	Partial	Fully	Partial	Partial (+)

[1] N. G. Tsoutsos and M. Maniatakos. Heroic: Homomorphically encrypted one instruction computer. In Proceedings of the Conference on Design, Automation & Test in Europe, DATE '14, pages 246:1–246:6, 3001 Leuven, Belgium, Belgium, 2014. European Design and Automation Association.
[2] A. Chatterjee and I. Sengupta. FURISC: FHE encrypted URISC design. IACR Cryptology ePrint Archive, 2015:699, 2015.
[3] P. T. Breuer, J. P. Bowen, E. Palomar, and Z. Liu. A practical encrypted microprocessor. In Proceedings of the 13th International Joint Conference on e-Business and Telecommunications (ICETE 2016) - Volume 4: SECRYPT, Lisbon, Portugal, July 26-28, 2016., pages 239–250, 2016.

CryptoBlaze HE Scheme

Asymmetric
Encryption (public) key = (g, **n**)
Decryption (private) key = (λ, μ)

Encryption Size
$c = g^m r^n \bmod \mathbf{n^2}$

n = security parameter

Non-Deterministic Paillier Cryptosystem

For **n = b** bits, ciphertext = **2b** bits

Additively homomorphic
Dec(**c1c2 mod** $\mathbf{n^2}$) = m1 + m2

Design

CryptoBlaze ISA

- Extends a variant of MicroBlaze ISA
- 8 additional instructions for encrypted data processings

EADD ESUB	ELD EST	EMOV N2MOV	EBRZPOS EBRNEG
Homomorphic Operations	Memory load/store	Register transfer	Branching

CryptoBlaze: How big is an encrypted data?

n = security parameter

- Recall: for **n = b bits**, ciphertext = **2b bits**
- Supporting subtraction? **Pairwise storing (negation pair)**

$$c_{4b-1}c_{4b-2}\dots c_{2b}\,c_{2b-1}c_{2b-2}\dots c_0$$

Enc(A) 2b bits

Enc(-A) 2b bits

Encrypted Data Size = 4b bits

CryptoBlaze: eRegister and keyRegister

Recall: Paillier addition ('+')
c1c2 mod n^2

eRegister (ER)

32 x 4b-bit register
Ciphertext processing
EMOV: Copy between ERs

keyRegister (KR)

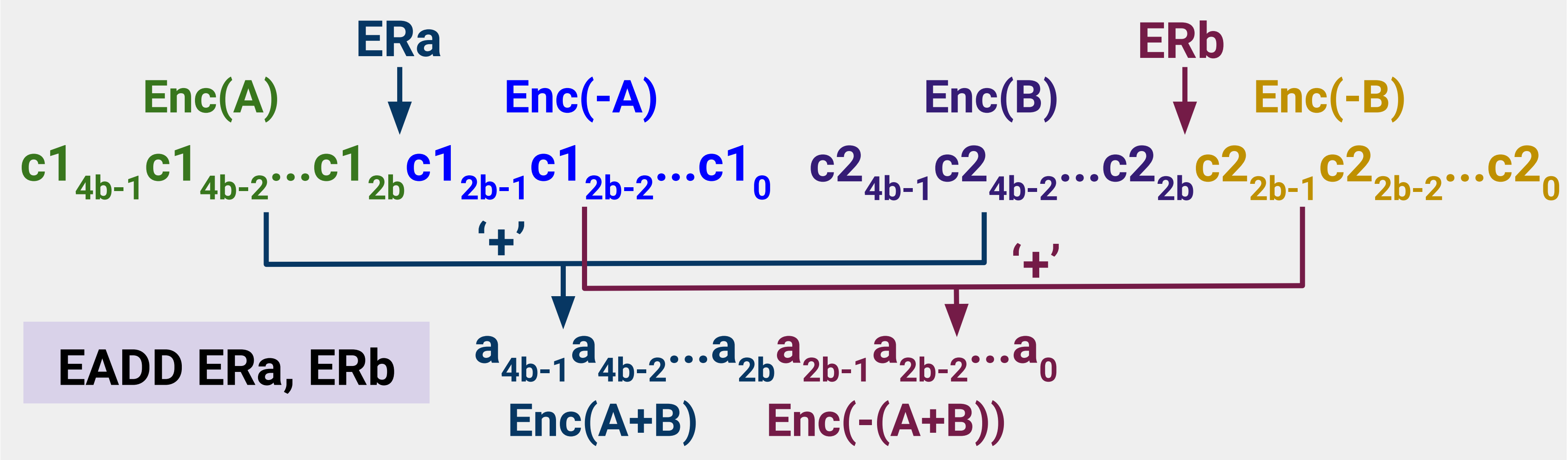
1 special register
Store n^2 (for '+')
N2MOV: ER → KR

CryptoBlaze: Memory Space and Transfer

- Shared between normal and encrypted data
- Byte-addressed
- ***ELD*** (load) and ***EST*** (store) to/from ER
- **Program: unencrypted** (multi instructions)
- Communicates via 32-bit AXI bus
  - For n = b bits, ELD/EST = $\frac{4b}{32}$ **cycles**

CryptoBlaze: *EADD*

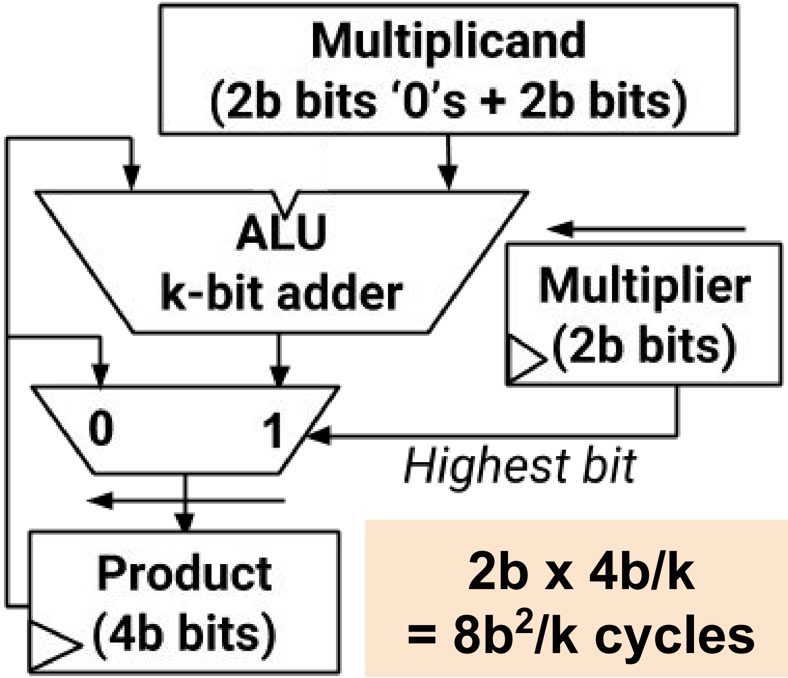
- Recall Paillier addition '+' = **$c1c2 \bmod n^2$**, for n = b bits:
  - 2b-bits c1 x 2b-bits c2 multiplication, followed by
  - 4b-bits c1c2 / 2b-bits n^2 division → remainder taken
- Negation pair: higher & lower bits separately at the same time



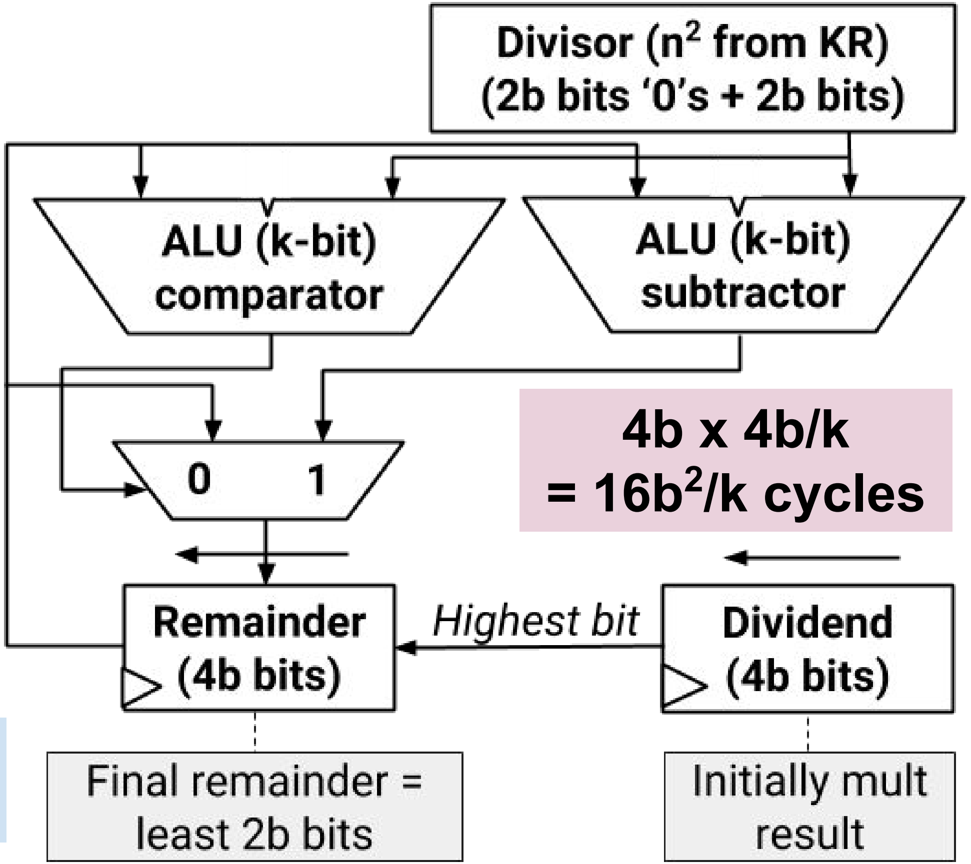
eALU: Multiplication and Division Block

- Multiplication: addition and shift
- Division: subtraction and shift
- Each multicycle, depends on:
  - **b** (the bits size of n)
  - **k** (the size of the ALU unit: addition/subtraction/comparison)
- *Stall* signal to processor

eALU: Multiplication and Division Block



2b x 4b/k
= $8b^2/k$ cycles

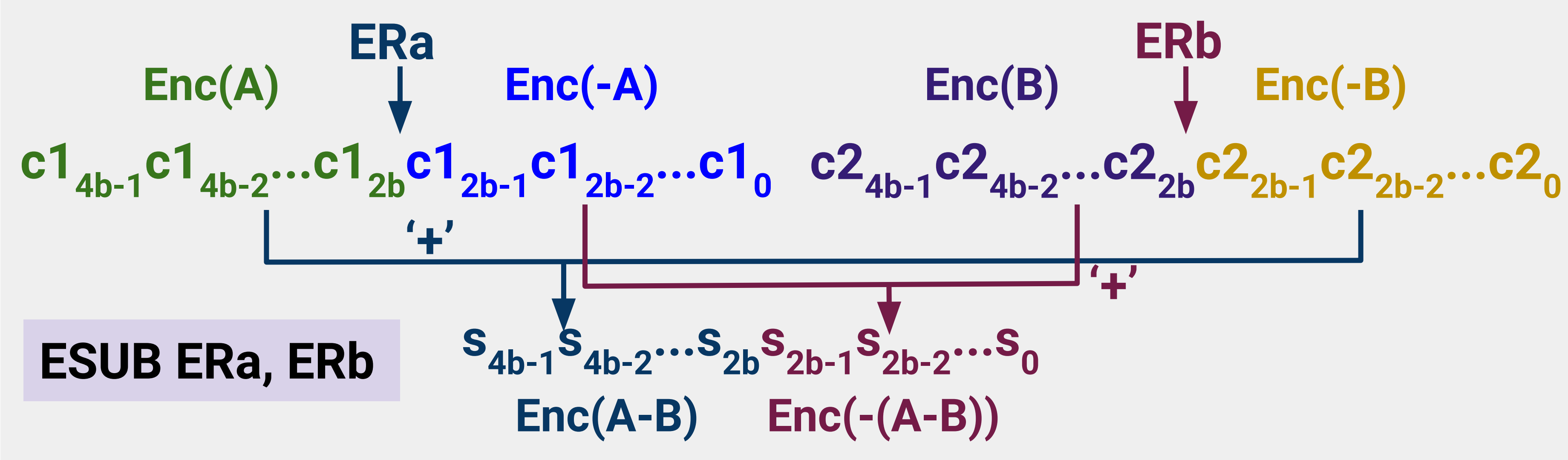


4b x 4b/k
= $16b^2/k$ cycles

TOTAL = $24b^2/k$ cycles

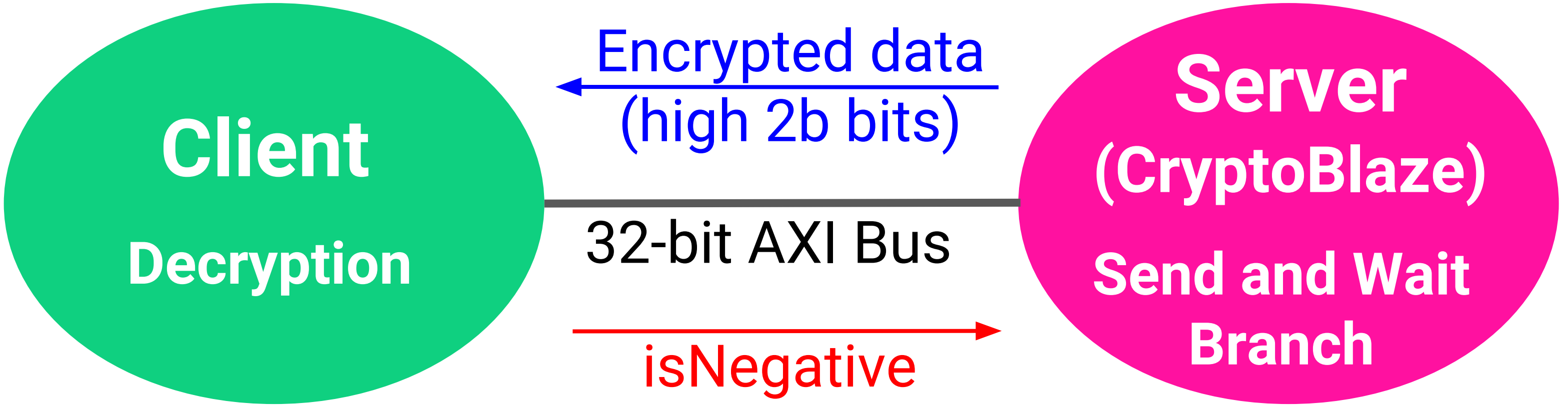
CryptoBlaze: *ESUB*

- Similar to *EADD*: uses homomorphic addition (**'+'**) = **c1c2 mod n^2**
- Negation pair
  - '+' **higher** bits of c1 **with lower** bits of c2
  - '+' **lower** bits of c1 **with higher** bits of c2



CryptoBlaze: Branching

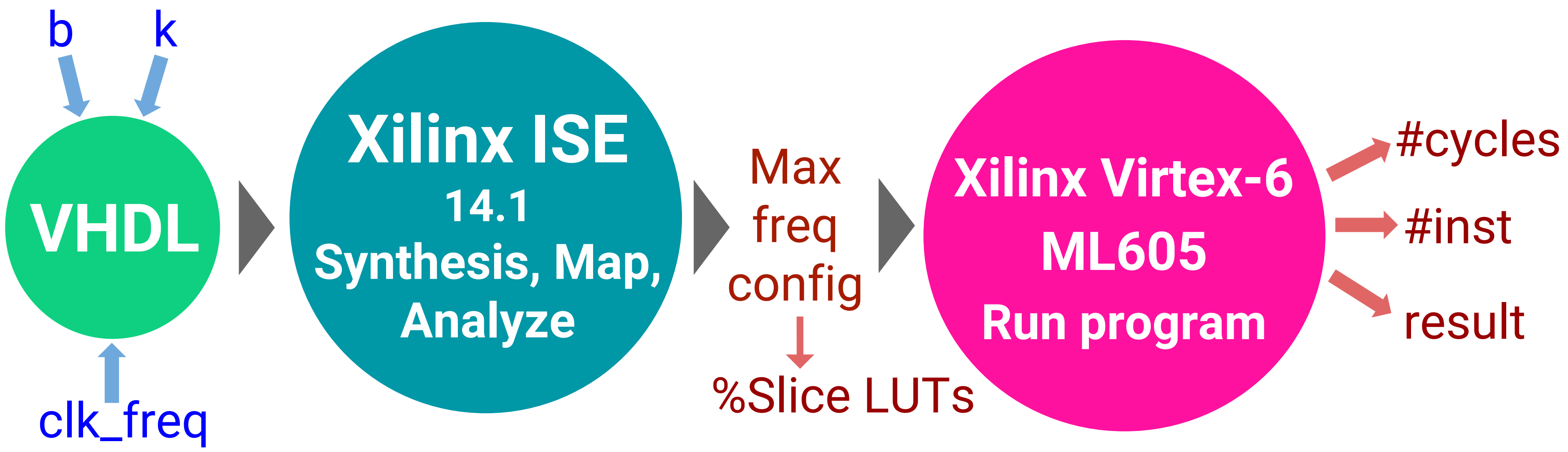
- Homomorphic addition ‘+’ only: cannot determine if ERa < 0
- Decryption is needed, but no private key
- Non-deterministic : Sign LUT is not viable
- Solution: **client-server communication → EBRNEG, EBRZPOS**



Experimental Setup

Experimental Setup

- Input parameters:
  - **DATA_WIDTH (4b)** → b = 32, 64, 128, 256, 512, 1024
  - **ALU_WIDTH (k)** → k = 32, 64, 128, 256, 512, 1024, 2048, 4096
  - **CLK_FREQ** → 5MHz to 125MHz with 5MHz increment



Experimental Setup

- **Benchmark programs**: Fibonacci, Factorial, Bubble Sort
  - Same benchmark as HEROIC and FURISC for comparison
  - Result checked for correctness
- **Client modelling** (for branching):
  - A variant of **MicroBlaze**
  - 32-bit AXI Bus connection to CryptoBlaze
  - Actually requires decryption
    - **Pre-computed sign array (O(1) operation))** → simplify and minimize dependency

Result

Result: Number of Executed Instructions

Benchmark	HEROIC	CryptoBlaze
Fibonacci	1617294	898
Factorial	1011994	5656
Bubblesort	1882234	112882

Result: Synthesis - Minimum Latency Config

	Slice LUTs%	#Cycles	Max Freq	Latency
Increase data size (4b)	increase	increase	no change*	increase
Increase ALU Size (k)	no change	decrease	decrease	decrease**

eRegisters dominate

Critical path = ALU (+/-)

Recall: '+' takes $24b^2/k$ cycles

Latency = #Cycles / MaxFreq

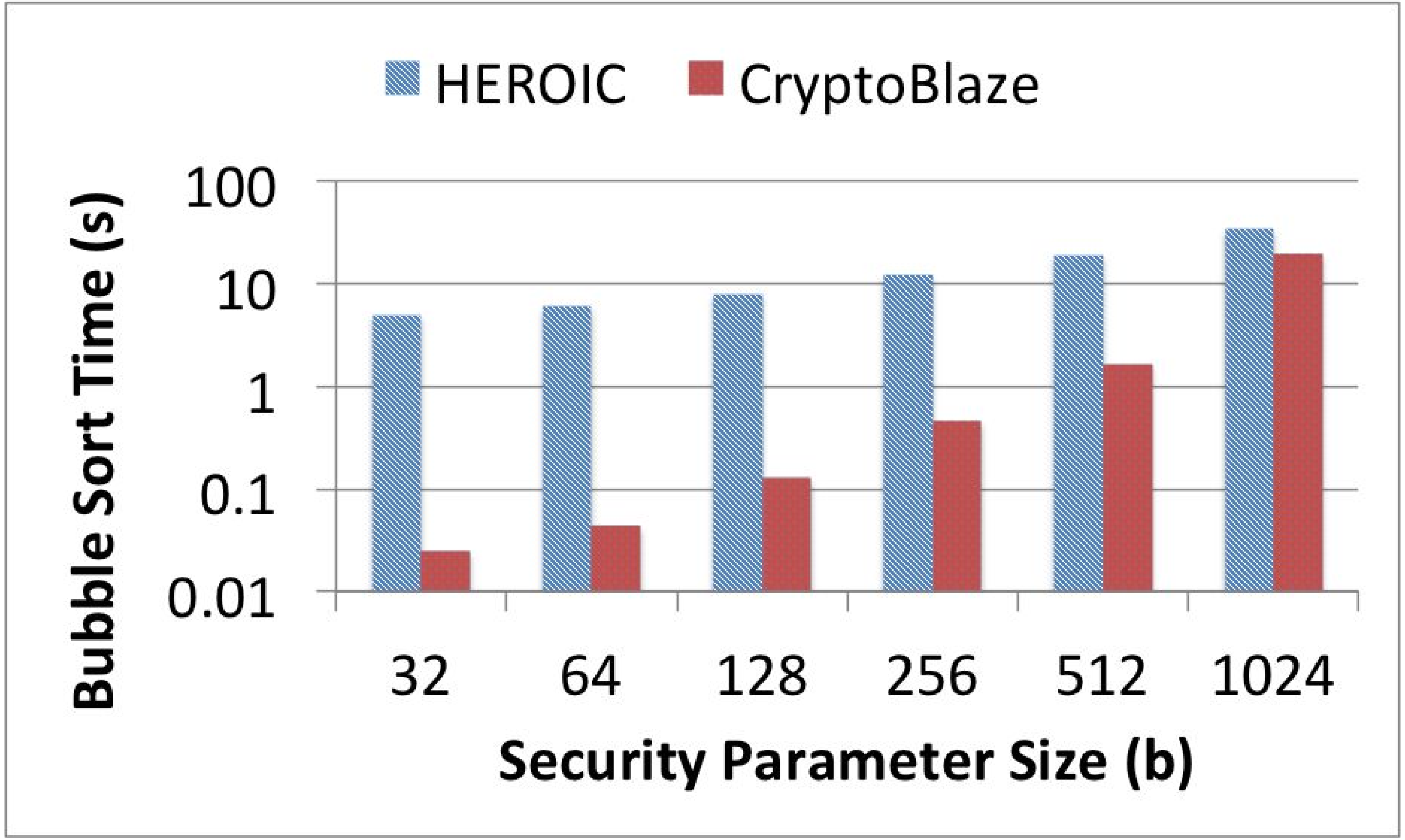
*) Exception: for k <= 256, critical path = register R/W

**) Exception: for k >= 1024, max freq too low → high latency

Result: Number of Cycles - Bubble Sort

b(bits)	HEROIC	FURISC	CryptoBlaze (min. *latency* config)	
			# cycles	k(bits)
32	$5.77 * 10^7$	-	$3.10 * 10^6$	128
64	$7.14 * 10^7$	-	$5.49 * 10^6$	256
128	$9.89 * 10^7$	-	$1.04 * 10^7$	512
256	$1.54 * 10^8$	$3.51 * 10^{11}$	$5.03 * 10^7$	256
512	$2.64 * 10^8$	-	$9.98 * 10^7$	512
1024	$4.84 * 10^8$	-	$1.19 * 10^9$	128

Result: Latency - Bubble Sort



Discussion: Performance

- Generally faster than HEROIC and FURISC with assumptions:
  - Fast decryption in client (modelled to be O(1))
  - High-speed communication channel between client and server
- Depends on the decryption and client-server communication speed

Discussion: Security

+	-
Robust against known-ciphertext attacks (e.g. frequency analysis)	Abused client-server communication → possible solution: server authorization
Robust against chosen-plaintext attacks	Unencrypted program memory exposes executed algorithm

Conclusion and Future Work

Non-deterministic Partially Homomorphic Processor

Supports multiple instructions (8 for ciphertext)

CryptoBlaze

FPGA Implementation

Faster given fast decryption and communication channel

Future researches:

- Securing client-server communication
- Optimizing implementation
- Exploring design space