Hardware Acceleration Efforts for Homomorphic Encryption

Medha: Microcoded Hardware Accelerator for computing on Encrypted Data

Ahmet Can Mert

2023-06-20

IAIK – Graz University of Technology

Outline

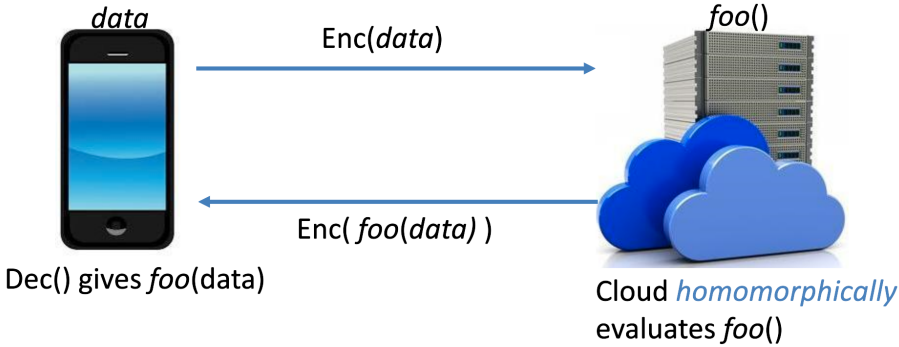
1. Homomorphic Encryption
    - Challenges in accelerating HE
2. Medha: Microcoded Hardware Accelerator for computing on Encrypted Data
    - Architecture of Homomorphic Processor
    - Customized on-chip memory design
    - Placement-friendly Layout
    - Design Methodology for Flexible Polynomial Degree
    - Evaluation Results
3. Conclusion

Ahmet Can Mert — IAIK – Graz University of Technology

Homomorphic Encryption

Homomorphic Encryption

- Enables computation on encrypted data.



Homomorphic Encryption

- Holy grail of cryptography: Fully homomorphic encryption.
- RSA cryptosystem (1978) is homomorphic...
  - with respect to multiplication.

$$m_a = a^e \pmod{n}, m_b = b^e \pmod{n}$$

$$m_a.m_b = a^e.b^e = (a.b)^e \pmod{n}$$

$$(m_a.m_b)^d \pmod{n} = ((a.b)^e)^d \pmod{n} = a.b$$

- Paillier (1999) cryptosystem is homomorphic with respect to addition.
- A *fully* homomorphic scheme for both addition and multiplication?

Homomorphic Encryption

- Craig Gentry proposed the first fully homomorphic scheme in 2009.

- Achieved it using a special operation called *boostrapping*.

First image source: `https://www.technologyreview.com/technology/homomorphic-encryption/`
Second image source: M. Joye, *SoK: Fully omomorphic Encryption over the [Discretized] Torus*, CHES, 2022.

Homomorphic Encryption

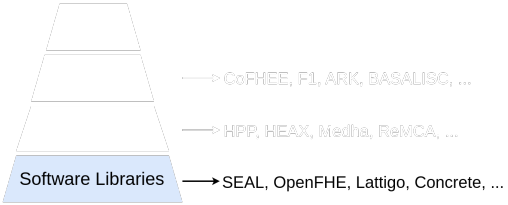
- Homomorphic encryption use cases
  - Privacy-preserving machine learning
  - Secure computation in cloud
  - Financial services, healthcare, government etc.

Homomorphic Encryption

- Homomorphic encryption research
  - New schemes with better performance/less complexity/different applications
    - 1st generation schemes: First schemes, very inefficient
    - 2nd generation schemes: Enables faster integer/fixed-pt arithmetic (e.g., BGV, CKKS)
    - 3rd generation schemes: Enables efficient Boolean algebra (e.g., FHEW)
  - Development of homomorphic application/compilers
  - Acceleration of HE

Challenges in accelerating Homomorphic Encryption

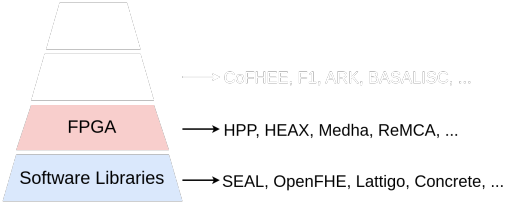
- A brief summary of acceleration efforts. Two main tracks:
  - Real accelerator prototypes vs. simulation-based modelling of accelerators.



[CoFHEE] Mohammed Nabeel et al. CoFHEE: A Co-processor for Fully Homomorphic Encryption Execution. DATE 2023.
[F1] Axel Feldmann et al. F1: A fast and programmable accelerator for fully homomorphic encryption. MICRO 2021.
[BTS] Sangpyo Kim et al. BTS: An Accelerator for Bootstrappable Fully Homomorphic Encryption. ISCA 2022.
[BASALISC] Sangpyo Kim et al. BTS: An Accelerator for Bootstrappable Fully Homomorphic Encryption. ISCA 2022.
[HPP] S. S. Roy et al. Fpga-based high-performance parallel architecture for homomorphic computing on encrypted data. HPCA, 2019.
[HEAX] M. Sadegh Riazi et al. HEAX: an architecture for computing on encrypted data. ASPLOS 2020.
[ReMCA] Y. Su et al. Remca: A reconfigurable multi-core architecture for full rns variant of bfv homomorphic evaluation. IEEE TCAS I, 2022

Challenges in accelerating Homomorphic Encryption

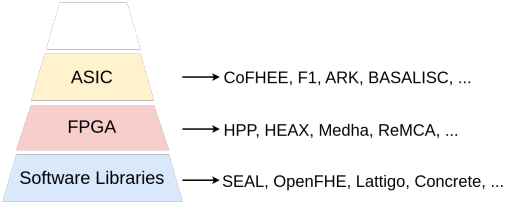
- A brief summary of acceleration efforts. Two main tracks:
  - Real accelerator prototypes vs. simulation-based modelling of accelerators.



[CoFHEE] Mohammed Nabeel et al. CoFHEE: A Co-processor for Fully Homomorphic Encryption Execution. DATE 2023.
[F1] Axel Feldmann et al. F1: A fast and programmable accelerator for fully homomorphic encryption. MICRO 2021.
[BTS] Sangpyo Kim et al. BTS: An Accelerator for Bootstrappable Fully Homomorphic Encryption. ISCA 2022.
[BASALISC] Sangpyo Kim et al. BTS: An Accelerator for Bootstrappable Fully Homomorphic Encryption. ISCA 2022.
[HPP] S. S. Roy et al. Fpga-based high-performance parallel architecture for homomorphic computing on encrypted data. HPCA, 2019.
[HEAX] M. Sadegh Riazi et al. HEAX: an architecture for computing on encrypted data. ASPLOS 2020.
[ReMCA] Y. Su et al. Remca: A reconfigurable multi-core architecture for full rns variant of bfv homomorphic evaluation. IEEE TCAS I, 2022

Challenges in accelerating Homomorphic Encryption

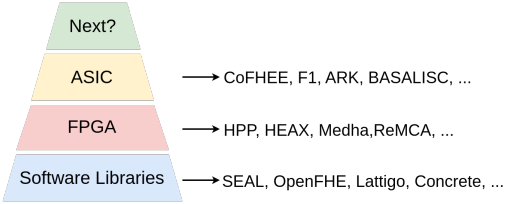
- A brief summary of acceleration efforts. Two main tracks:
  - Real accelerator prototypes vs. simulation-based modelling of accelerators.



[CoFHEE] Mohammed Nabeel et al. CoFHEE: A Co-processor for Fully Homomorphic Encryption Execution. DATE 2023.
[F1] Axel Feldmann et al. F1: A fast and programmable accelerator for fully homomorphic encryption. MICRO 2021.
[BTS] Sangpyo Kim et al. BTS: An Accelerator for Bootstrappable Fully Homomorphic Encryption. ISCA 2022.
[BASALISC] Sangpyo Kim et al. BTS: An Accelerator for Bootstrappable Fully Homomorphic Encryption. ISCA 2022.
[HPP] S. S. Roy et al. Fpga-based high-performance parallel architecture for homomorphic computing on encrypted data. HPCA, 2019.
[HEAX] M. Sadegh Riazi et al. HEAX: an architecture for computing on encrypted data. ASPLOS 2020.
[ReMCA] Y. Su et al. Remca: A reconfigurable multi-core architecture for full rns variant of bfv homomorphic evaluation. IEEE TCAS I, 2022

Challenges in accelerating Homomorphic Encryption

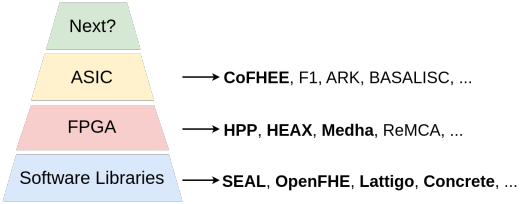
- A brief summary of acceleration efforts. Two main tracks:
  - Real accelerator prototypes vs. simulation-based modelling of accelerators.



[CoFHEE] Mohammed Nabeel et al. CoFHEE: A Co-processor for Fully Homomorphic Encryption Execution. DATE 2023.
[F1] Axel Feldmann et al. F1: A fast and programmable accelerator for fully homomorphic encryption. MICRO 2021.
[BTS] Sangpyo Kim et al. BTS: An Accelerator for Bootstrappable Fully Homomorphic Encryption. ISCA 2022.
[BASALISC] Sangpyo Kim et al. BTS: An Accelerator for Bootstrappable Fully Homomorphic Encryption. ISCA 2022.
[HPP] S. S. Roy et al. Fpga-based high-performance parallel architecture for homomorphic computing on encrypted data. HPCA, 2019.
[HEAX] M. Sadegh Riazi et al. HEAX: an architecture for computing on encrypted data. ASPLOS 2020.
[ReMCA] Y. Su et al. Remca: A reconfigurable multi-core architecture for full rns variant of bfv homomorphic evaluation. IEEE TCAS I, 2022

Challenges in accelerating Homomorphic Encryption

- A brief summary of acceleration efforts. Two main tracks:
  - Real accelerator prototypes vs. simulation-based modelling of accelerators.



[CoFHEE] Mohammed Nabeel et al. CoFHEE: A Co-processor for Fully Homomorphic Encryption Execution. DATE 2023.
[F1] Axel Feldmann et al. F1: A fast and programmable accelerator for fully homomorphic encryption. MICRO 2021.
[BTS] Sangpyo Kim et al. BTS: An Accelerator for Bootstrappable Fully Homomorphic Encryption. ISCA 2022.
[BASALISC] Sangpyo Kim et al. BTS: An Accelerator for Bootstrappable Fully Homomorphic Encryption. ISCA 2022.
[HPP] S. S. Roy et al. Fpga-based high-performance parallel architecture for homomorphic computing on encrypted data. HPCA, 2019.
[HEAX] M. Sadegh Riazi et al. HEAX: an architecture for computing on encrypted data. ASPLOS 2020.
[ReMCA] Y. Su et al. Remca: A reconfigurable multi-core architecture for full rns variant of bfv homomorphic evaluation. IEEE TCAS I, 2022

Challenges in accelerating Homomorphic Encryption

Why do we need HW acceleration?

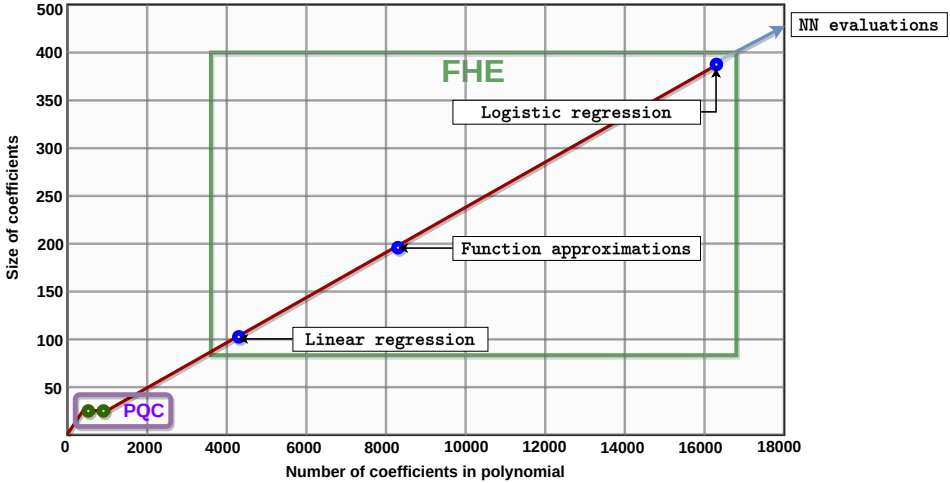
1. Computationally intensive:

Challenges in accelerating Homomorphic Encryption

Why do we need HW acceleration?

1. Computationally intensive: 100,000× plain computation

Challenges in accelerating Homomorphic Encryption



Challenges in accelerating Homomorphic Encryption

1. Computationally intensive
2. Lots of polynomial arithmetic operations
    - Large degree polynomial arithmetic
    - Long integer arithmetic

Challenges in accelerating Homomorphic Encryption

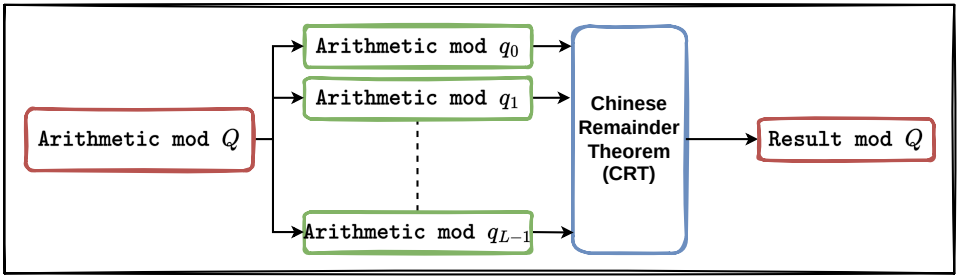
1. Computationally intensive
2. Lots of polynomial arithmetic operations
   - Large degree polynomial arithmetic
   - Long integer arithmetic
3. Memory management
   - Ciphertexts could be several MBs
   - On-Chip memory is limited
   - Off-Chip data transfer is very slow

Challenges in accelerating Homomorphic Encryption

1. Computationally intensive
2. Lots of polynomial arithmetic operations
   - Large degree polynomial arithmetic
   - **Long integer arithmetic** ← This problem is solved using RNS
3. Memory management
   - Ciphertexts could be several MBs
   - On-Chip memory is limited
   - Off-Chip data transfer is very slow

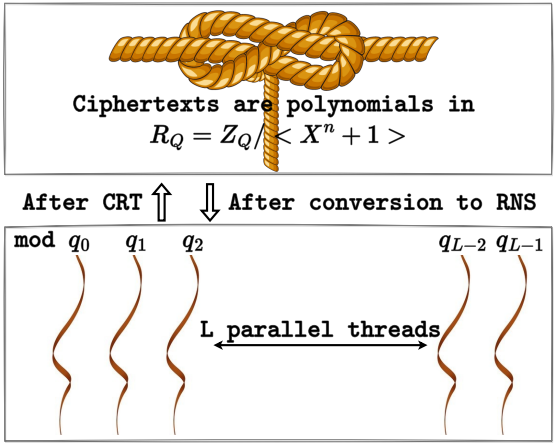
Application of Residue Number System (RNS)

1. Take a modulus $Q = \prod_0^{L-1} q_i$ where q_i are coprime.
2. Process residues independently (to some extent)

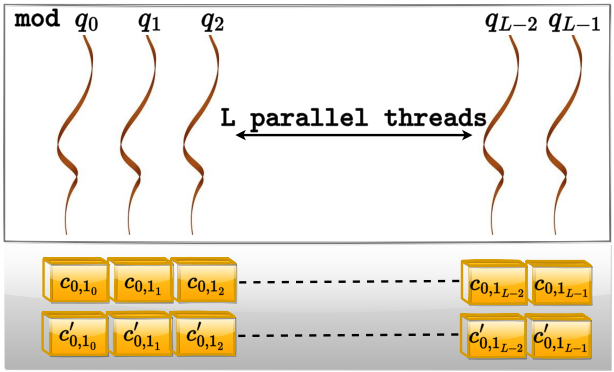


Small coefficients and Parallel computation

Parallel computation flow example



Scheme operations



Multiplication:

$c = [c_{0_i}, c_{1_i}], c' = [c'_{0_i}, c'_{1_i}]$

$d_{mult} = [d_{0_i}, d_{1_i}, d_{2_i}]$

$d_{0_i} = c_{0_i} \times c'_{0_i}$

$d_{1_i} = c_{0_i} \times c'_{1_i} + c_{1_i} \times c'_{0_i}$

$d_{2_i} = c_{1_i} \times c'_{1_i}$

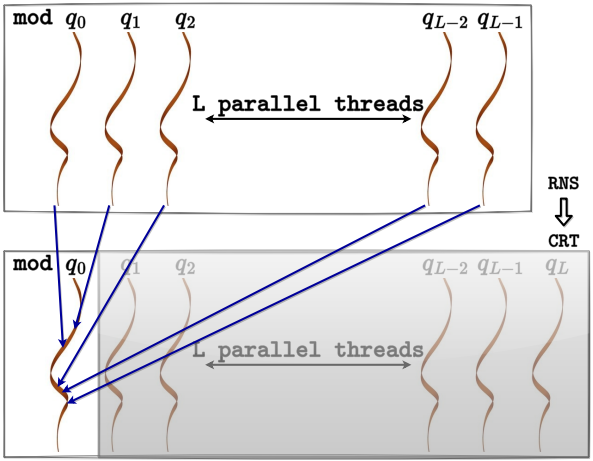
$\forall i \in [0, L-1]$

Scheme operations

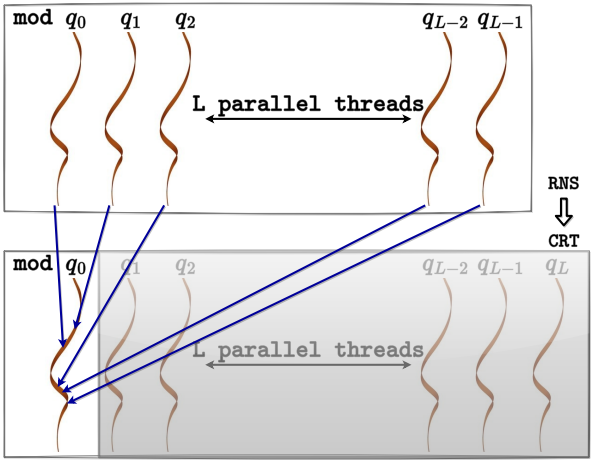
Relinearization:

$d_{mult} = [d_{0_i}, d_{1_i}, d_{2_i}]$
$\rightarrow c''_{mult} = [c''_{0_i}, c''_{1_i}]$

$c''_{0j} = \sum_{i=0}^{L-1} d_{2_i}.KSK_{0_{i,j}} \ \forall j \in [0, L]$
$c''_{1j} = \sum_{i=0}^{L-1} d_{2_i}.KSK_{1_{i,j}} \ \forall j \in [0, L]$

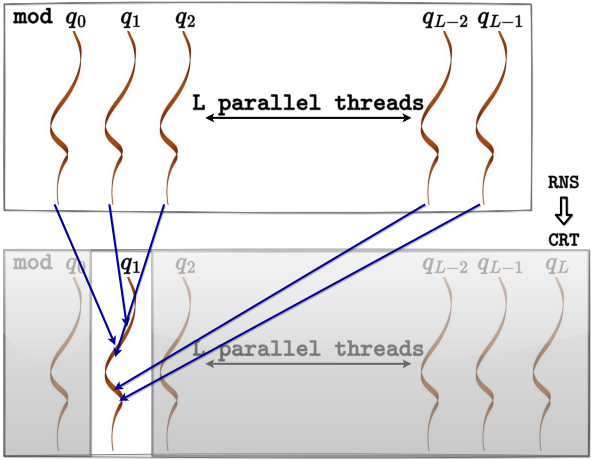


HW design challenge!



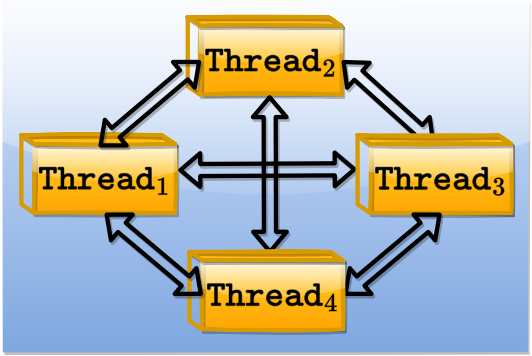
Threads need to exchange data.

HW design challenge!



Threads need to exchange data.

HW design challenge!



Threads need to exchange data.

Medha: Microcoded Hardware Accelerator for computing on Encrypted Data

Medha: *M*icrocoded *H*ardware *A*ccelerator for computing on *E*ncrypted *D*ata

Ahmet Can Mert[1], Aikata[1], Sunmin Kwon[2], Youngsam Shin[2], Donghoon Yoo[2], Yongwoo Lee[2] and Sujoy Sinha Roy[1]

[1] IAIK, Graz University of Technology, Austria,
{sujoy.sinharoy,ahmet.mert,aikata}@iaik.tugraz.at
[2] Samsung Advanced Institute of Technology, Suwon, Republic of Korea,
{sunmin7.kwon,youngsam.shin,say.yoo,yw0803.lee}@samsung.com

Outline

1. Motivation and Background
   - Challenges in accelerating HE
   - RNS-HEAAN
2. Medha: Microcoded Hardware Accelerator for computing on Encrypted Data
   - Architecture of Homomorphic Processor
   - Customized on-chip memory design
   - Placement-friendly Layout
   - Design Methodology for Flexible Polynomial Degree
   - Evaluation Results
3. Conclusion

Architecture of the Homomorphic Processor

Design goals:

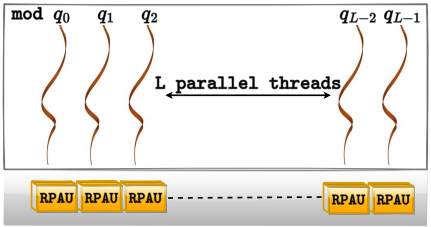
- Implementation and verification on real FPGA (Xilinx Alveo U250 card)
- Eliminating off-chip memory communication
- Supporting at least $n = 2^{14}$ with RNS moduli sizes 54/60-bit for RNS-CKKS

Architecture of the Homomorphic Processor

Design goals:

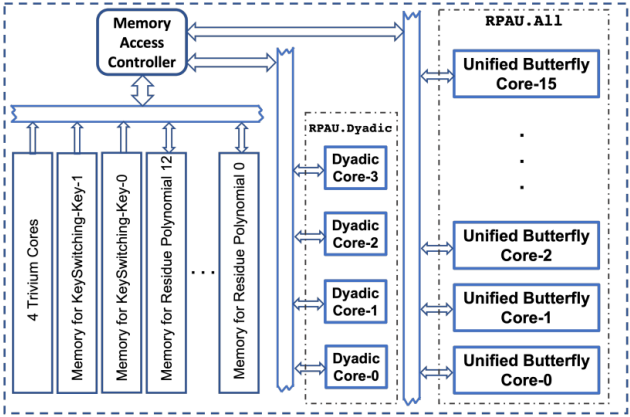
- Implementation and verification on real FPGA (Xilinx Alveo U250 card)
- Eliminating off-chip memory communication
- Supporting at least $n = 2^{14}$ with RNS moduli sizes 54/60-bit for RNS-CKKS

Thread $\rightarrow$ One residue polynomial arthmetic unit (RPAU)



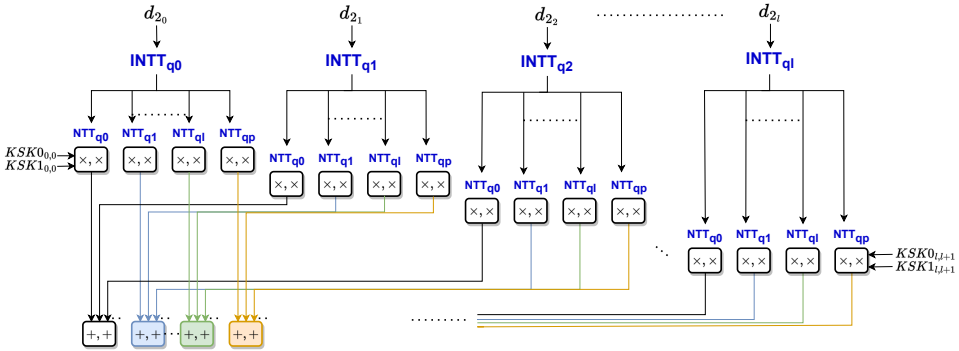
Architecture of the Homomorphic Processor

The overall architecture of one RPAU.

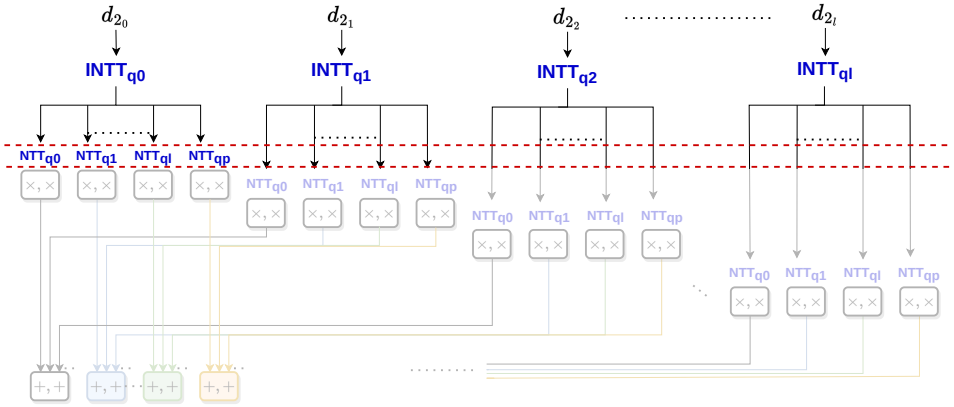


- It has two main cores
  - 16-butterfly NTT unit
  - 4-core Dyadic unit
- Memory blocks

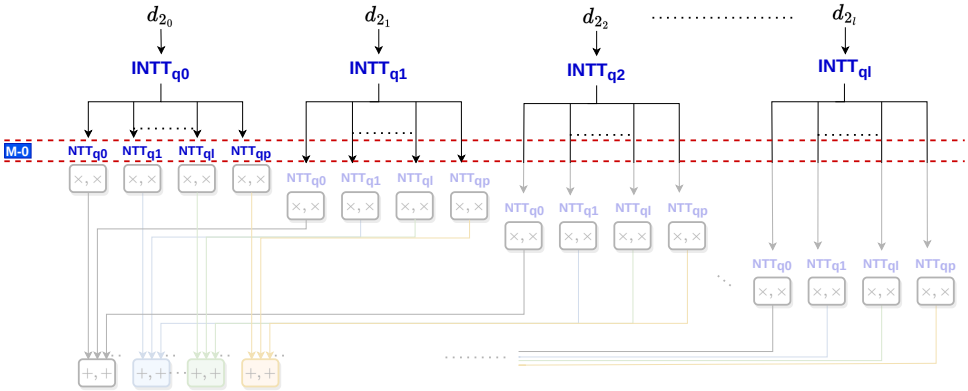
Why do we use two separate cores?



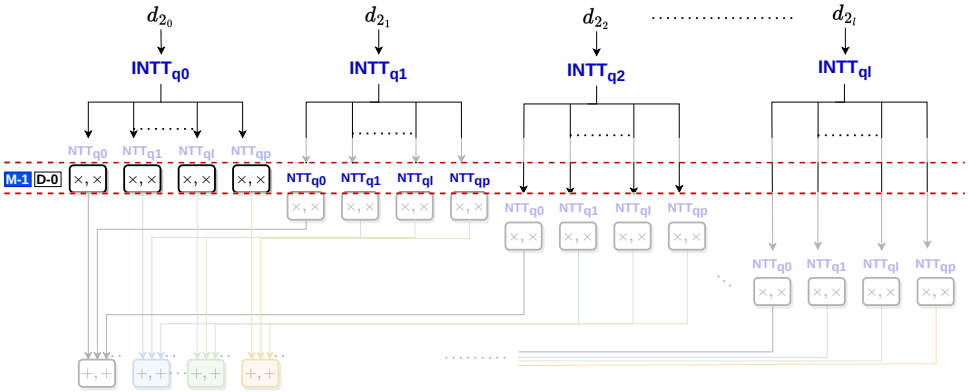
Why do we use two separate cores?



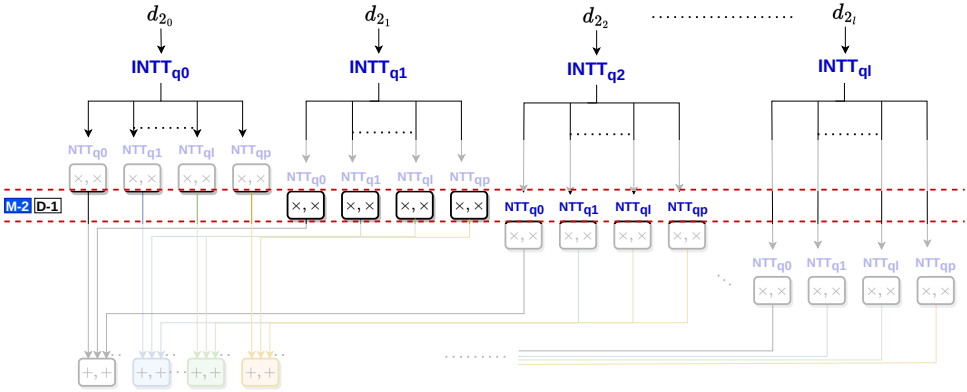
Why do we use two separate cores?



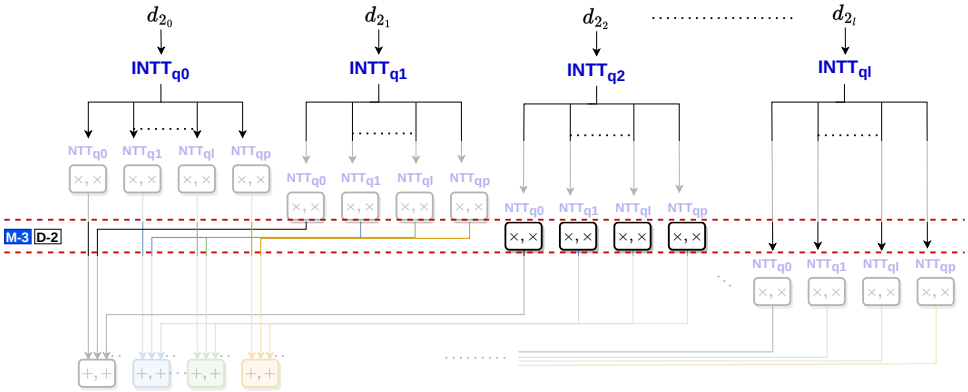
Why do we use two separate cores?



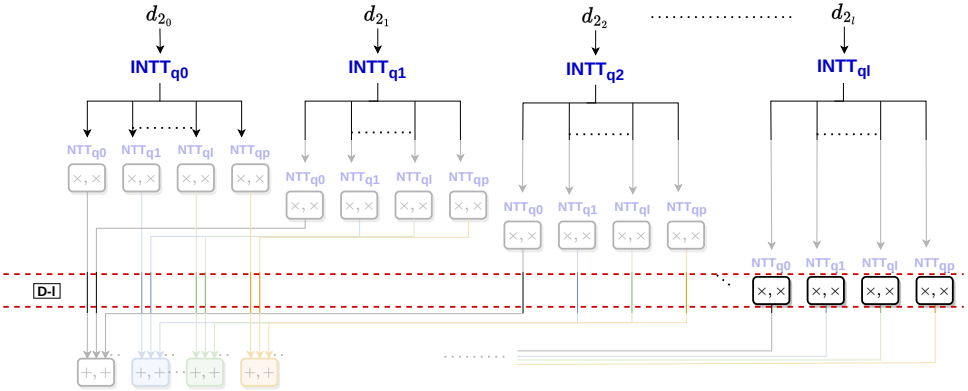
Why do we use two separate cores?



Why do we use two separate cores?



Why do we use two separate cores?



- Parallel execution of NTT & dyadic cores for key-switching procedure results in:

40% reduction in cycle!

Outline

1. Motivation and Background
   - Challenges in accelerating HE
   - RNS-HEAAN
2. Medha: Microcoded Hardware Accelerator for computing on Encrypted Data
   - Architecture of Homomorphic Processor
   - Customized on-chip memory design
   - Placement-friendly Layout
   - Design Methodology for Flexible Polynomial Degree
   - Evaluation Results
3. Conclusion

Customized on-chip memory design

Our goal: Implementing homomorphic operations using only on-chip memory.
We analyzed the peak memory requirement of one RPAU.

Example: For $n = 2^{14}$ with 10 RNS bases of 54/60-bits, each RPAU needs to store at least 49 polynomials of size 2^{14} for residue polynomials and keys.

- On-chip BRAMs or URAMs alone cannot fulfill the requirements...

Customized on-chip memory design

Our goal: Implementing homomorphic operations using only on-chip memory.
We analyzed the peak memory requirement of one RPAU.

Example: For $n = 2^{14}$ with 10 RNS bases of 54/60-bits, each RPAU needs to store at least 49 polynomials of size 2^{14} for residue polynomials and keys.

- On-chip BRAMs or URAMs alone cannot fulfill the requirements...

Optimizations to reduce on-chip memory requirements

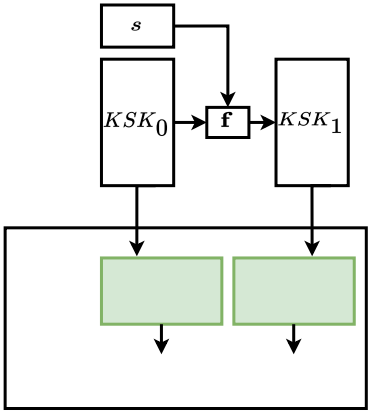
1. On the fly evaluation key generation
2. Utilizing left-over bits in BRAM/URAM

1. On-the-fly evaluation key generation

$KSK_0 \leftarrow \$$
$KSK_0 \leftarrow \mathsf{PRNG}(\mathsf{seeds})$
$KSK_1 \leftarrow \mathbf{f}(KSK_0, s)$

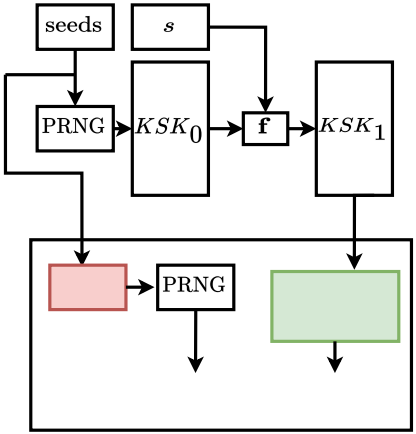
1. On-the-fly evaluation key generation

$KSK_0 \leftarrow \$$
$KSK_0 \leftarrow \text{PRNG}(\text{seeds})$
$KSK_1 \leftarrow \mathbf{f}(KSK_0, s)$



1. On-the-fly evaluation key generation

$KSK_0 \leftarrow \$$
$KSK_0 \leftarrow \text{PRNG}(\text{seeds})$
$KSK_1 \leftarrow \mathbf{f}(KSK_0, s)$



2. Utilizing left-over bits in BRAM/URAM

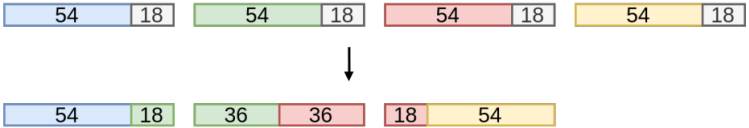
- One URAM address can store 72-bits
- One BRAM address can store 18/36/72-bits

2. Utilizing left-over bits in BRAM/URAM

- One URAM address can store 72-bits
- One BRAM address can store 18/36/72-bits

We created a virtual memory to utilize left-over bits in BRAM/URAM.

- Example: 54-bit coefficient storage in URAM



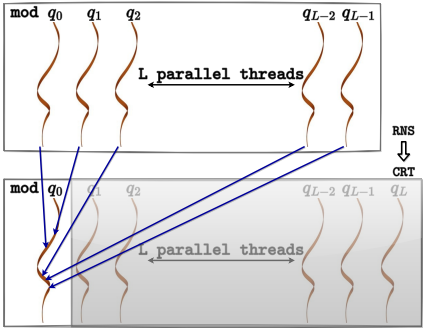
Outline

1. Motivation and Background
   - Challenges in accelerating HE
   - RNS-HEAAN
2. Medha: Microcoded Hardware Accelerator for computing on Encrypted Data
   - Architecture of Homomorphic Processor
   - Customized on-chip memory design
   - Placement-friendly Layout
   - Design Methodology for Flexible Polynomial Degree
   - Evaluation Results
3. Conclusion

Placement-friendly Layout

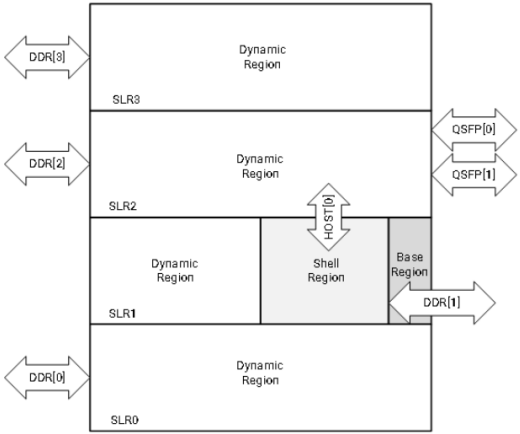
Each RNS base is implemented by one RPAU.

- Key-switching operation requires polynomials to be exchanged between RPAUs.



Placing/Interconnecting RPAUs is a huge engineering challenge!

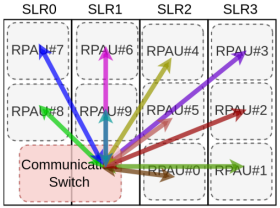
Placement-friendly Layout



- Alveo U250 platform consists of four 'semi-separated' SLR regions
- Two neighboring SLRs are connected using a limited number of wires.

Placement-friendly Layout

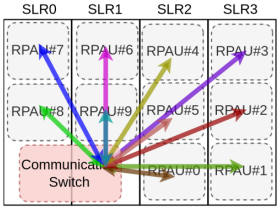
Placing/Interconnecting the RPAUs must consider SLR-to-SLR connection constraints.



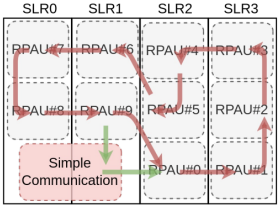
- Complicates placement and routing
- Many SLR-crossing nets
- Achieves very low clock frequency

Placement-friendly Layout

Placing/Interconnecting the RPAUs must consider SLR-to-SLR connection constraints.



- Complicates placement and routing
- Many SLR-crossing nets
- Achieves very low clock frequency



- Only neighboring RPAUs are connected
- Data is sent through a chain of RPAUs
- Achieves high clock frequency

Outline

1. Motivation and Background
    - Challenges in accelerating HE
    - RNS-HEAAN
2. Medha: Microcoded Hardware Accelerator for computing on Encrypted Data
    - Architecture of Homomorphic Processor
    - Customized on-chip memory design
    - Placement-friendly Layout
    - Design Methodology for Flexible Polynomial Degree
    - Evaluation Results
3. Conclusion

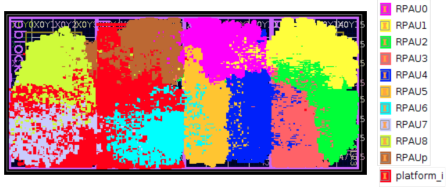
Evaluation

As a proof of concept, our implementation employs 10 RPAUs and supports two parameter sets.

- $(\log_2(pQ) = 438, n = 2^{14})$ and $(\log_2(pQ) = 546, n = 2^{15})$

Resource utilization on Alveo U250 card.

- 1.09M LUTs (55.4%), 3,607 DSPs (29.3%)
- 1,576.5 BRAMs (58.6%), 931 URAMs (72.8%)

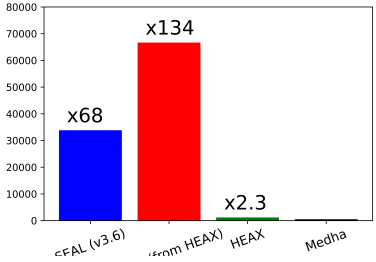


Evaluation Results

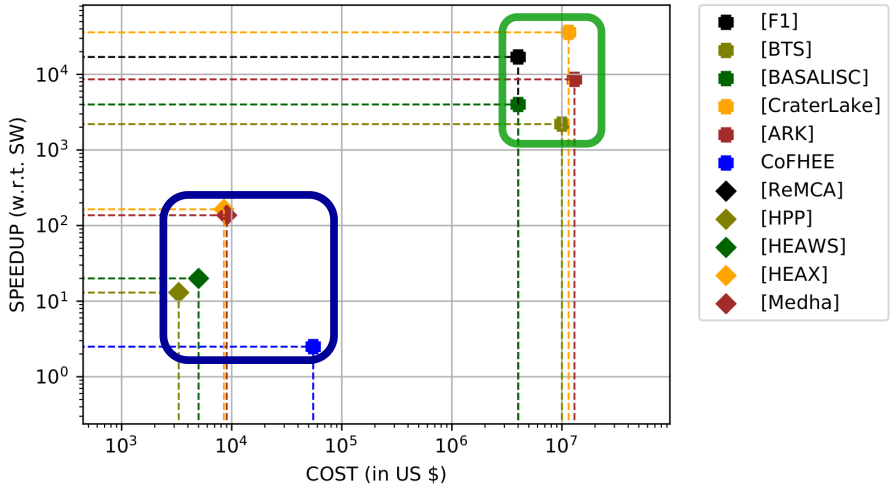
End-to-end application benchmark for the logistic regression model proposed by iDASH2017 competition winners.

- 64× *speedup compared to the implementation in SEAL*

Comparison with SEAL and HEAX:



Comparison with HW-based HE accelerators in the literature



Conclusion

Conclusion

- Accelerators show promising performance results, but ...
  - most of them are not proven in silicon
  - only few FPGA/GPU works are verified and tested in real HW
- We proposed one of *few* real hardware accelerators in literature
- HW acceleration is not enough to make HE practical, we need better schemes
- HW Benchmarking and comparison is challenging.
  - Very few open source works.

Hardware Acceleration Efforts for Homomorphic Encryption

Medha: Microcoded Hardware Accelerator for computing on Encrypted Data

Ahmet Can Mert

2023-06-20

IAIK – Graz University of Technology