TU Graz

SCIENCE
PASSION
TECHNOLOGY

OpenNTT

An Automated Toolchain for Compiling High-Performance NTT and FFT Accelerators

Florian Krieger Florian Hirner Ahmet Can Mert Sujoy Sinha Roy

Optimist Workshop 2025

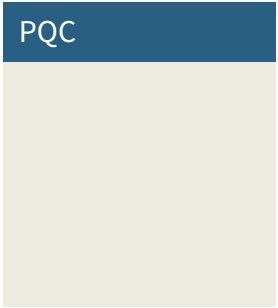
Outline

The NTT in Modern Cryptography

- Cornerstone of modern cryptography

The NTT in Modern Cryptography

- Cornerstone of modern cryptography



The NTT in Modern Cryptography

- Cornerstone of modern cryptography

PQC

- **NIST:** 3

The NTT in Modern Cryptography

- Cornerstone of modern cryptography

PQC

- **NIST:** 3
- **KPQC:** 3

The NTT in Modern Cryptography

- Cornerstone of modern cryptography

PQC

- **NIST:** 3
- **KPQC:** 3
- **ICCS:** ?

The NTT in Modern Cryptography

- Cornerstone of modern cryptography

PQC

- **NIST:** 3
- **KPQC:** 3
- **ICCS:** ?

FHE

The NTT in Modern Cryptography

- Cornerstone of modern cryptography

PQC

- **NIST:** 3
- **KPQC:** 3
- **ICCS:** ?

FHE

- **BFV:** NTT

The NTT in Modern Cryptography

- Cornerstone of modern cryptography

PQC

- **NIST:** 3
- **KPQC:** 3
- **ICCS:** ?

FHE

- **BFV:** NTT
- **BGV:** NTT

The NTT in Modern Cryptography

- Cornerstone of modern cryptography

PQC

- **NIST:** 3
- **KPQC:** 3
- **ICCS:** ?

FHE

- **BFV:** NTT
- **BGV:** NTT
- **CKKS:** NTT+FFT

The NTT in Modern Cryptography

- Cornerstone of modern cryptography

PQC	FHE	ZKP
NIST: 3	**BFV:** NTT	
KPQC: 3	**BGV:** NTT	
ICCS: ?	**CKKS:** NTT+FFT	

The NTT in Modern Cryptography

- Cornerstone of modern cryptography

PQC	FHE	ZKP
NIST: 3	**BFV:** NTT	**RS-Codes**
KPQC: 3	**BGV:** NTT	
ICCS: ?	**CKKS:** NTT+FFT	

The NTT in Modern Cryptography

- Cornerstone of modern cryptography

PQC	FHE	ZKP
NIST: 3	**BFV:** NTT	**RS-Codes**
KPQC: 3	**BGV:** NTT	**KZG**
ICCS: ?	**CKKS:** NTT+FFT	

The NTT in Modern Cryptography

- Cornerstone of modern cryptography

PQC

- **NIST:** 3
- **KPQC:** 3
- **ICCS:** ?

$N \approx 2^8$
$\log Q < 32$ bits

FHE

- **BFV:** NTT
- **BGV:** NTT
- **CKKS:** NTT+FFT

ZKP

- **RS-Codes**
- **KZG**

The NTT in Modern Cryptography

- Cornerstone of modern cryptography

PQC

- **NIST:** 3
- **KPQC:** 3
- **ICCS:** ?

$N \approx 2^8$
$\log Q < 32$ bits

FHE

- **BFV:** NTT
- **BGV:** NTT
- **CKKS:** NTT+FFT

$N \approx 2^{16}$
$\log Q \approx 64$ bits

ZKP

- **RS-Codes**
- **KZG**

The NTT in Modern Cryptography

- Cornerstone of modern cryptography

PQC

- **NIST:** 3
- **KPQC:** 3
- **ICCS:** ?

$N \approx 2^8$
$\log Q < 32$ bits

FHE

- **BFV:** NTT
- **BGV:** NTT
- **CKKS:** NTT+FFT

$N \approx 2^{16}$
$\log Q \approx 64$ bits

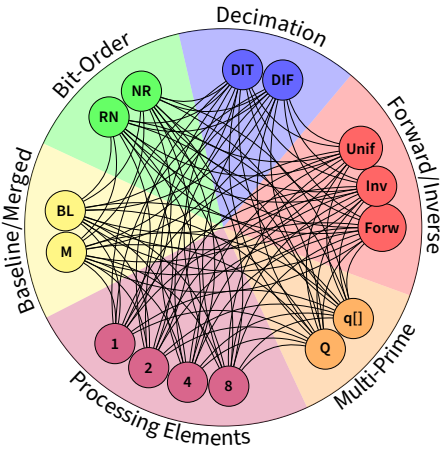
ZKP

- **RS-Codes**
- **KZG**

$N \approx 2^{30}$
$\log Q \approx 384$ bits

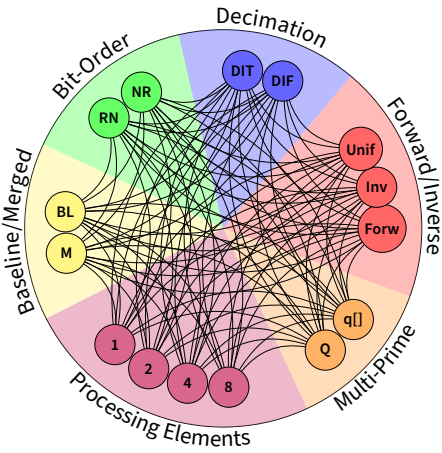
Motivation

- Many different configurations



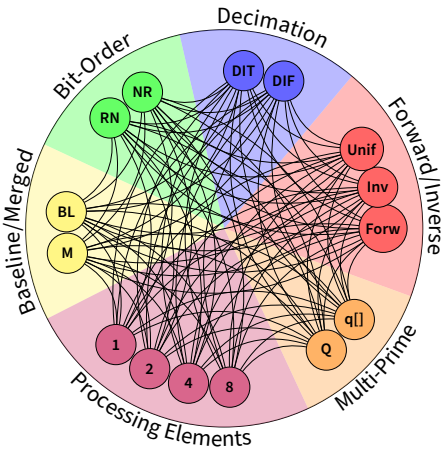
Motivation

- Many different configurations
- Performance bottleneck
  - ➔ Hardware acceleration



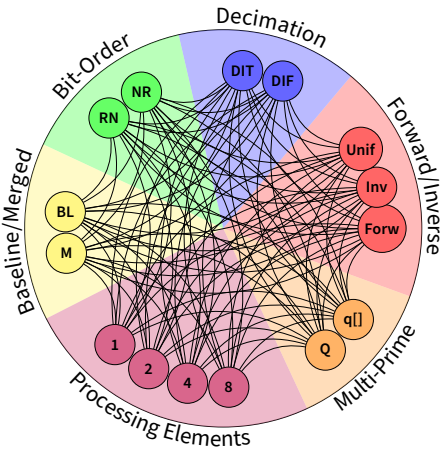
Motivation

- Many different configurations
- Performance bottleneck
  - ➔ Hardware acceleration
- Often processing secret data
  - ➔ Side-channel countermeasures



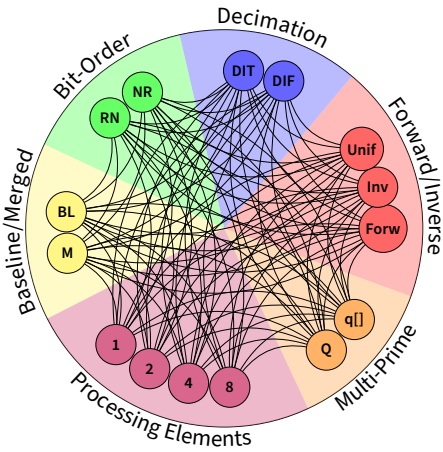
Designing Hardware Accelerators is Challenging

- High design effort for each design...



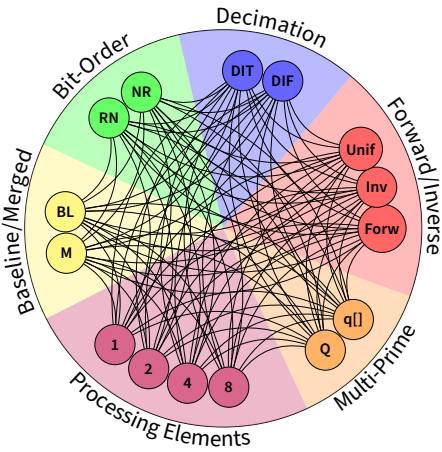
Designing Hardware Accelerators is Challenging

- High design effort for each design...
- ...to establish SCA security
  performance overhead



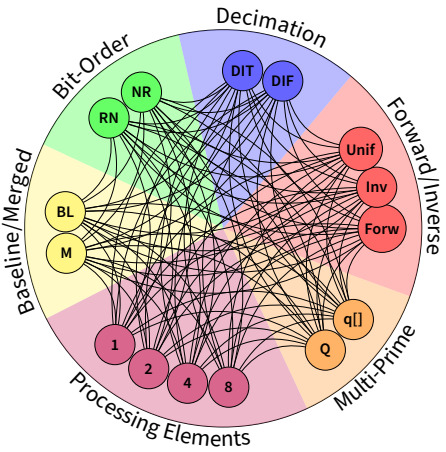
Designing Hardware Accelerators is Challenging

- High design effort for each design...
- ...to establish SCA security
  performance overhead
- ...to reach high performance



Designing Hardware Accelerators is Challenging

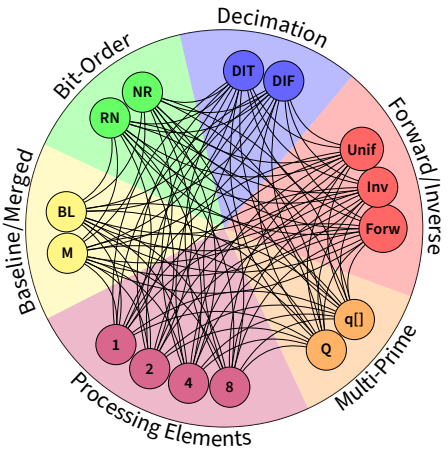
- High design effort for each design...
- ...to establish SCA security
  performance overhead
- ...to reach high performance
- ...to adapt to target platform



Designing Hardware Accelerators is Challenging

- High design effort for each design...
- ...to establish SCA security
  performance overhead
- ...to reach high performance
- ...to adapt to target platform

Hardware design as simple as running a Python script?



OpenNTT

OpenNTT: What is it?

- **Open-source hardware design tool for NTT and FFT**

OpenNTT: What is it?

- **Open-source hardware design tool for NTT and FFT**
- Easy to use

OpenNTT: What is it?

- **Open-source hardware design tool for NTT and FFT**
- Easy to use
- First tool with on-the-fly twiddle factor generation

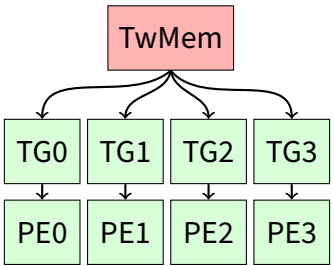
OpenNTT: What is it?

- **Open-source hardware design tool for NTT and FFT**
- Easy to use
- First tool with on-the-fly twiddle factor generation
- Highly performant

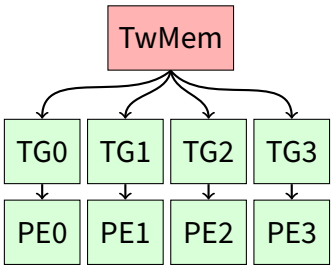
OpenNTT: What is it?

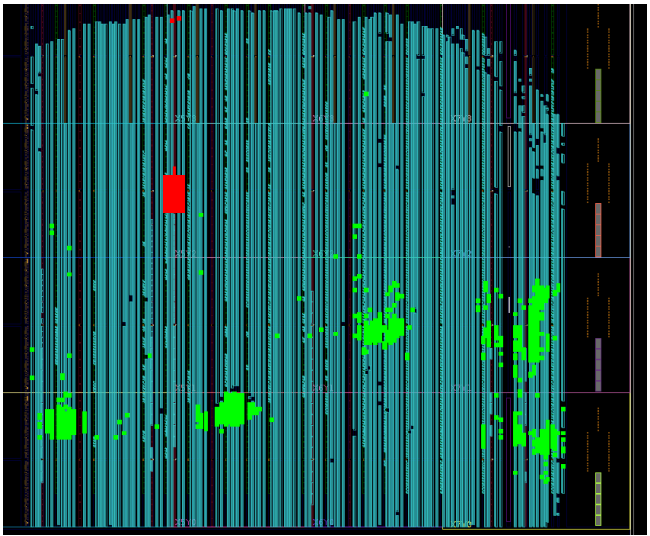
- **Open-source hardware design tool for NTT and FFT**
- Easy to use
- First tool with on-the-fly twiddle factor generation
- Highly performant
  - Platform-aware optimization techniques

The MemOpt Technique

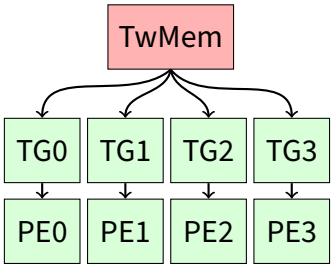


The MemOpt Technique

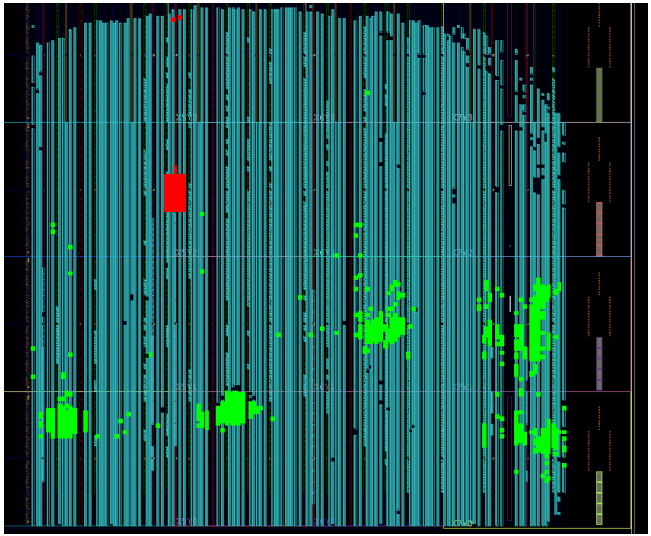




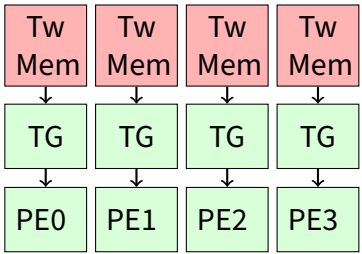
The MemOpt Technique



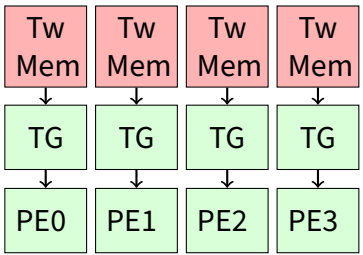
→ Minimizes memory usage

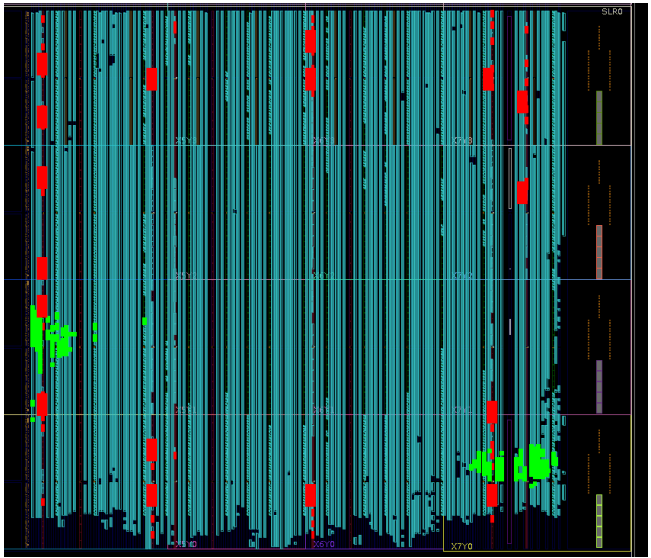


The RoutOpt Technique

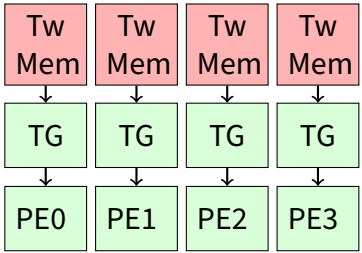


The RoutOpt Technique

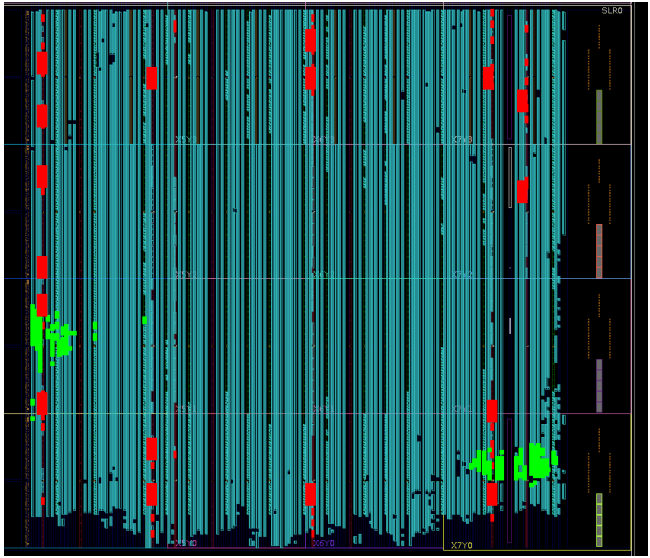




The RoutOpt Technique

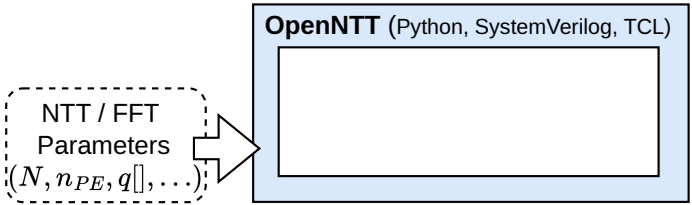


➔ Improves frequency by up to 20%



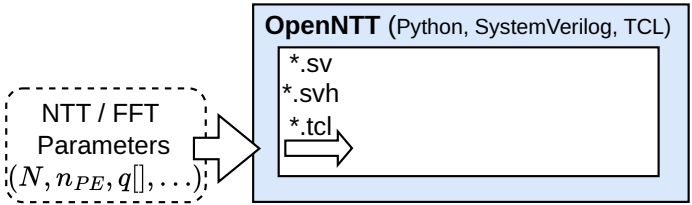
Applications of OpenNTT

The OpenNTT Workflow



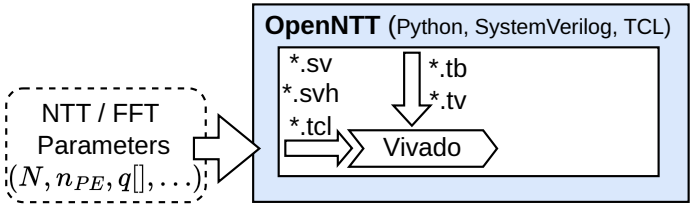

```
python3 openntt.py -transf_type=NTT -memory_opt=0 -ntt_type=mfntt_dit_nr
-q_count=1 -q_list=32 -n=4096 -io_band=8 -mem_depth=2 -coeff_arith=1
```

The OpenNTT Workflow



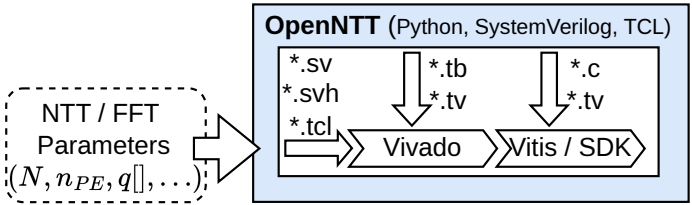

```
python3 openntt.py -transf_type=NTT -memory_opt=0 -ntt_type=mfntt_dit_nr
-q_count=1 -q_list=32 -n=4096 -io_band=8 -mem_depth=2 -coeff_arith=1
```

The OpenNTT Workflow



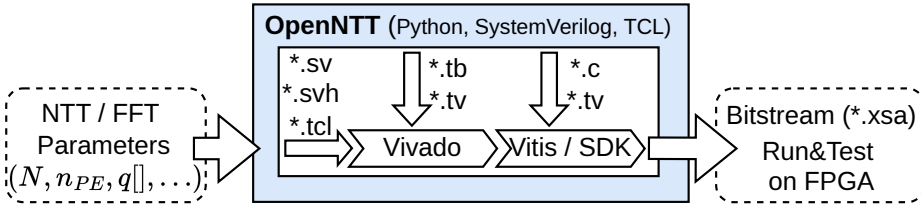

```
python3 openntt.py -transf_type=NTT -memory_opt=0 -ntt_type=mfntt_dit_nr
-q_count=1 -q_list=32 -n=4096 -io_band=8 -mem_depth=2 -coeff_arith=1
```

The OpenNTT Workflow



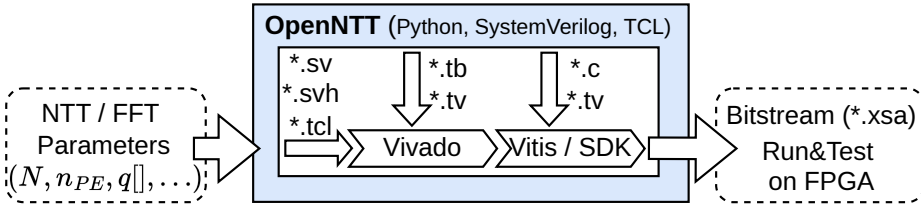

```
python3 openntt.py -transf_type=NTT -memory_opt=0 -ntt_type=mfntt_dit_nr
-q_count=1 -q_list=32 -n=4096 -io_band=8 -mem_depth=2 -coeff_arith=1
```

The OpenNTT Workflow




```
python3 openntt.py -transf_type=NTT -memory_opt=0 -ntt_type=mfntt_dit_nr
-q_count=1 -q_list=32 -n=4096 -io_band=8 -mem_depth=2 -coeff_arith=1
```

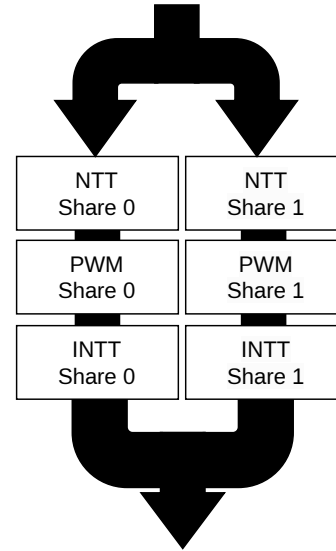
The OpenNTT Workflow




```
python3 openntt.py -transf_type=NTT -memory_opt=0 -ntt_type=mfntt_dit_nr
-q_count=1 -q_list=32 -n=4096 -io_band=8 -mem_depth=2 -coeff_arith=1
```

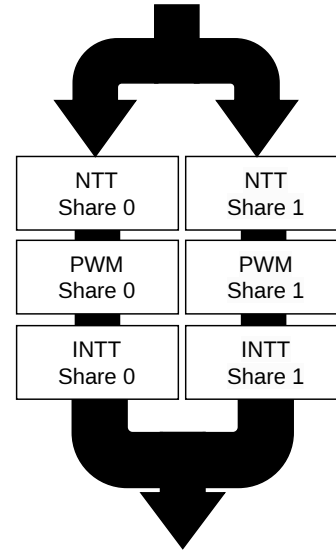
Example: Masked PQC Accelerator

- OpenNTT generates NTT module
  - ...using parameters from specification



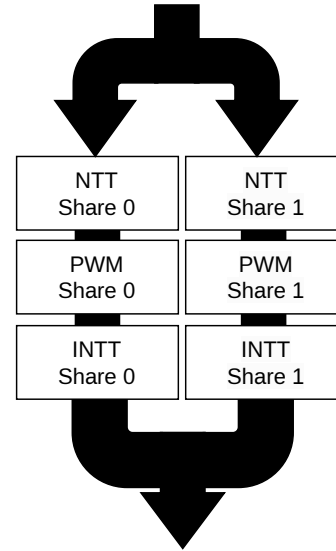
Example: Masked PQC Accelerator

- OpenNTT generates NTT module
  - ...using parameters from specification
- Instantiating the module twice
  - each module operates on one share



Example: Masked PQC Accelerator

- OpenNTT generates NTT module
  - ...using parameters from specification
- Instantiating the module twice
  - each module operates on one share
- Ongoing work:
  - Offering optimizations for masking
  - Shuffling support for hiding



Example: Client-Side CKKS Accelerator

- CKKS client needs NTT and FFT

Example: Client-Side CKKS Accelerator

- CKKS client needs NTT and FFT
- Use OpenNTT to compile NTT and FFT modules

Example: Client-Side CKKS Accelerator

- CKKS client needs NTT and FFT
- Use OpenNTT to compile NTT and FFT modules
- 1.9× more efficient than state-of-the-art designs [Kri+24]

Conclusion

- Difficult to provide generic NTT/FFT design tools

github.com

Paper

Conclusion

- Difficult to provide generic NTT/FFT design tools
- OpenNTT is very easy to use

github.com

Paper

Conclusion

- Difficult to provide generic NTT/FFT design tools
- OpenNTT is very easy to use
  - Parameter flexible
  - Platform aware
  - Applies to many different applications

github.com

Paper

TU Graz

SCIENCE
PASSION
TECHNOLOGY

OpenNTT

An Automated Toolchain for Compiling High-Performance NTT and FFT Accelerators

Florian Krieger Florian Hirner Ahmet Can Mert Sujoy Sinha Roy

Optimist Workshop 2025

Bibliography

[Kri+24] Florian Krieger et al. **Aloha-HE: A Low-Area Hardware Accelerator for Client-Side Operations in Homomorphic Encryption**. 2024 Design, Automation and Test in Europe Conference and Exhibition (DATE). 2024, pp. 1–6. DOI: 10.23919/DATE58400.2024.10546608.