# Intel© Homomorphic Encryption Acceleration Library for FPGAs

Yan Meng,[1*] Fillipe D. M. de Souza,[1] Hubert de Lassus,[1] Brad Smith,[1] Shahzad Butt,[1] Tomás González Aragón,[2] Yongfa Zhou,[3] Yong Wang,[3] Jingyi Jin,[1*] Flávio Bergamaschi,[4*] Paky Abu-Alam,[1] Anil Goteli,[1] Nir Peled,[1]

Intel Corporation, [1]CA, USA [2]Costa Rica [3]China [4]UK

[*]To whom correspondence should be addressed; E-mail: he_fpga_support@intel.com.

**Abstract**

**Fully Homomorphic Encryption (FHE) is an emerging cryptographic technique that allows users to perform computations on encrypted data without having access to or knowing the contents of the decrypted data. It opens a multitude of practical and real-world applications without compromising data integrity, privacy, and confidentiality, from analyzing sensitive financial and medical data to enabling private query search. The benefits of FHE come at the cost of high computational demand. We have designed high speed kernels running on Intel FPGAs to obtain throughput performance gains in critical Homomorphic Encryption (HE) operations, and release our work as open-source with the hope of contributing to communities to enable the future of privacy-preserving technology.**

## Privacy-Preserving Computing

The concept of privacy-preserving computing arises when two or more organizations wish to benefit from a collaboration that involves sharing and processing of private data whose ownership do not intersect. Fraud detection and more accurate medical diagnosis are common applications that could benefit from collaborative work involving private data. In practice, this is often impeded for competitive reasons and legal regulation requirements. Homomorphic Encryption (HE) is an encryption technique that can address this problem.

## Homomorphic Encryption

Homomorphic Encryption (HE) is a type of encryption technique that enables computation whilst data remains encrypted (*5*). This capability has the potential to unlock a plethora of use cases to run on untrusted cloud computing infrastructures without violating data privacy and confidentiality requirements. Over the years, many types of HE schemes and HE software libraries implementing these schemes have been proposed. The HE software libraries provide APIs for application developers building privacy-preserving workloads. Microsoft SEAL (*7*), HELib (*6*) and PALISADE (*3*) are among the most popular open-source HE software libraries. They differ in implementation, performance and support of the types of HE schemes. However, they all carry something in common. Under the hood, their code implements algorithms that perform heavy modular polynomial arithmetic operations on large amounts of data. These operations and the size of the encrypted data (ciphertexts) form the basis of the compute bottlenecks for HE-based workloads.

## Vision Statement

HE technology has the potential to transform the way protection of data privacy and confidentiality is handled in the digital data domain. Intel wishes to contribute to the development of a hardware and software ecosystem that supports deployment of HE-driven applications at large scale, while welcoming collaborative efforts across the academic and industry communities. To this end, we publish a preliminary work, under an open-source license, that attempts to bring up FPGAs as co-processors to improve throughput of HE operations. With this initiative, we invite the community to design and co-engineer the future of privacy-preserving technology using FPGAs with us.

# Intel HEXL for FPGA Open-Source Release (Version 1.0)

Similarly to the work in (*4*), we introduce Intel© Homomorphic Encryption Acceleration Library for FPGAs (Intel HEXL for FPGA), an open-source collection of FPGA kernels for common computing building blocks of Homomorphic Encryption (HE). Those kernels are provided under an open-source license and accessible at runtime through an open-source C/C++ FPGA

Runtime API. The kernels are written in OpenCL 2.0 and allow HE kernel developers to offload homomorphic encryption operations to Intel FPGAs; in particular, the Stratix 10 GX 2800 (available as the Intel PAC D5005 product). These operations include the forward and inverse number-theoretic transform (NTT) and NTT-based modular polynomial multiplication (as a ciphertext multiplication). Even though FPGAs as accelerators typically constrain those operations to an I/O-bound regime, we believe these low-level primitives allow throughput performance gains via batched lazy executions.

### Release Contents

- OpenCL-based FPGA processing elements architectures (kernels) for computationally costly building blocks of homomorphic encryption, including (modular) polynomial multiplication, forward and inverse number-theoretic transform (NTT) algorithms;
- C/C++ API for host runtime access of the FPGA processing elements that implement HE computing building blocks;
- Source codes under an open-source license to encourage contributions from both the academic and industry communities.

To learn more, refer to paper release at `https://github.com/intel/hexl-fpga` (in the `documentation` branch).

## Contributions and Future Work

Future directions include the distribution of all the currently provided kernels ported to Intel oneAPI (*1, 2*), a unified programming paradigm for heterogeneous computing, and additional functionalities to support more homomorphic encryption operations, such as relinearization and rotation. The community of HE practinioners and researchers, from academia to industry, is invited and welcomed to contribute to this effort with new optimizations, additional FPGA kernels, and extending the C/C++ FPGA Runtime API. Learn more on how to contribute at CONTRIBUTING.md.

# References


1. Intel oneAPI a new era of heterogeneous computing. https://www.intel.com/content/www/us/en/developer/tools/oneapi/overview.html. Accessed: 2021-11-28.

2. Intel oneAPI toolkits. https://www.intel.com/content/www/us/en/developer/tools/oneapi/toolkits.html. Accessed: 2021-11-28.

3. PALISADE Lattice Cryptography Library (release 1.11.2). https://palisade-crypto.org/, May 2021.

4. Fabian Boemer, Sejun Kim, Gelila Seifu, Fillipe DM de Souza, Vinodh Gopal, et al. Intel HEXL (release 1.2). https://github.com/intel/hexl, 2021.

5. Craig Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 169–178, 2009.

6. Shai Halevi and Victor Shoup. Design and implementation of helib: a homomorphic encryption library. Cryptology ePrint Archive, Report 2020/1481, 2020. https://ia.cr/2020/1481.

7. Microsoft SEAL (release 3.7). https://github.com/Microsoft/SEAL, September 2021. Microsoft Research, Redmond, WA.