Fall 2025 **CWRU Intersections**

Case Western Reserve University | Dec. 5th 2025

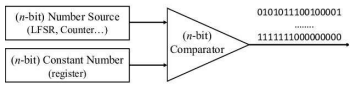
Stochastic Computing Implementation of Homomorphic Encryption Algorithms

Andrew Chen† (CE), Luke Dischiave† (CE), Evan Grover† (EE, CE), Advised by Dr. M. Hassan Najafi

Abstract– Recent advances in fields requiring massively parallel computing have accelerated the development of stochastic computing. At the cost of accuracy, stochastic computing is able to compute basic arithmetic operations for a lower hardware cost than traditional computing. Similarly, the rise of cloud computing has created a greater need for fully homomorphic encryption algorithms, which allow arithmetic operations to be performed on encrypted data without decryption by the cloud provider. Unfortunately, fully homomorphic encryption remains computationally expensive with traditional computing methodologies. This presentation proposes stochastic computing (SC) implementations of two fully homomorphic encryption algorithms, TFHE and BFV. We present our stochastic implementations, demonstrate their application, and provide information about their speed, accuracy, and hardware cost. Implementations are programmed in synthesizable SystemVerilog, while proofs and simulations are offered to back up our accuracy claims.

Introduction to SC

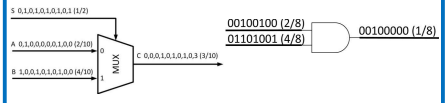
- Numbers are encoded into stochastic domain using random number generators and comparators
- Stochastic numbers are represented as the probability of 1 occurring in output bitstream



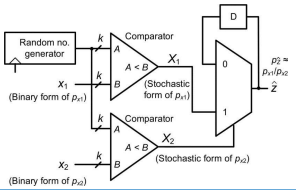
Mathematical Operations in Stochastic Domain

Addition (scaled): 2x1 multiplexer

Multiplication: AND gate



Division: Correlated division (CORDIV) circuit

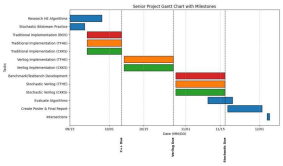


Approach & Methodology

1. Implemented algorithms programmatically
2. Implemented algorithms conventionally in SystemVerilog
3. Implemented algorithms stochastically in SystemVerilog

4 Algorithms were tried, 2 that showed the most promise were fully implemented

Project Management



Relevant Coursework

Andrew Chen: ECSE 318, ECSE 317, ECSE 315, ECSE 301

Luke Dischiave: ECSE 318, ECSE 317, ECSE 600, ECSE 315, ECSE 301

Evan Grover: ECSE 318, ECSE 315, ECSE 301

BFV HE Algorithm

Encoding:

- 8-bit grayscale values (m) are scaled into plaintext using Δ and padded into vector of polynomial coefficients

Encryption:

- Polynomial vector is encrypted into ciphertext pair ($c_0(x)$, $c_1(x)$) using error, public, and secret key polynomials

$c_0(x) = b(x)u(x) + e_1(x) + \Delta m(x)$

$c_1(x) = a(x)u(x) + e_2(x)$

Decryption:

- Recover Δm using secret key. Rounding removes injected error

$\Delta m(x) = c_0(x) + c_1(x)s(x)$

Decoding:

- Recover message by computing $\Delta m \times t/q = m$
- Hybrid BFV design replaces the large binary multiplier and divider with AND gate and CORDIV divider

TFHE Algorithm

Encryption:

- To convert a message into a ciphertext tuple (a, b) using the following formula where a is a randomly generated public key:

$b(x) = (a(x)s(x) + \Delta m(x) + e(x)) \bmod q$

- To ensure encryption and decryption go smoothly, error e(x) is bounded by O(q/2p).

Decryption:

- Once addition is performed by adding the respective elements in the tuple, decryption is possible with the equation:

$m(x) = (b(x) - a(x)s(x) \bmod q) / \Delta$

- Sometimes with error, the actual quotient m(x) is not an integer so floor division is used.

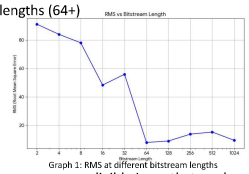
Bootstrapping:

- In applications where multiple homomorphic operations are done, bootstrapping is done to reduce errors. For our singular operations, we decided it wasn't necessary.

Latency & Accuracy vs. Orig Design

BFV:

- Bitstream length = latency in cycles
- Stochastic noise dominates inaccuracy in low bitstream lengths (2 – 32)
- RNG bias and ring wrap-around dominate inaccuracy in higher bitstream lengths (64+)

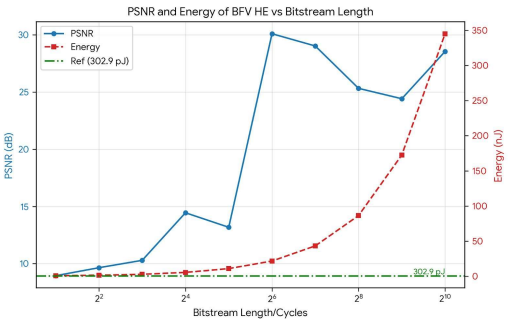


Graph 1: RMS at different bitstream lengths

TFHE:

- Inaccuracy was a non-negligible issue that made a stochastic model impractical.
- Accuracy for values ranged around ~79% (at its worst) to roughly 85% (average, best case).
- While the stochastic model provides a decrease in latency, the decrease in accuracy is too much.

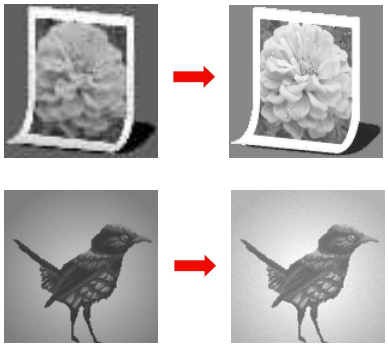
	Latency (ns)	LUTs	FFs	Power@10ns clk (W)
Trad. TFHE	6.034	338	69	0.77
Stoch. TFHE	5.587	335	67	0.076



Graph 2: BFV Image Accuracy

Image Processing (TFHE)

With TFHE, adding brightness of 40 using encrypted ciphertexts:



Evaluation/Success Criteria

Do the novel stochastic models provide enough advantages (i.e. reduced circuit complexity) to justify accuracy loss, latency increase, and energy increase? We are aiming for, at the most, 10% decrease in accuracy to see at least a 20% reduction in transistor count (circuit area).

BFV: Our Novel implementation meets success criteria by **reducing** circuit area by **32%**, with a scalable **3.2%** accuracy loss

- Massive **increase** in **energy consumption** that scales with bitstream length
- Latency scales linearly with cycles (trivial)
- Good quality/accuracy (30dB) can be achieved with **64 cycles**
- MSE@64 cycles = 65.03 → RMS = 8.06. 8.06/255 = **3.2%** accuracy loss

TFHE: For the stochastic implementation of TFHE, we observed a decrease in the critical path by roughly 7.41%. However, the cost of accuracy on the stochastic model made it unjustifiable.

Data Presented Dec 5. 2025, Intersections, Case Western Reserve University, Cleveland OH

†Presented in fulfillment of ECSE 398